



Low quality pictures

[juliachencoding.blogspot.ca](http://juliachencoding.blogspot.ca)



# Contents

<b>1</b>	<b>2015</b>	<b>19</b>
1.1	September . . . . .	19
	Draw a circle algorithm (2015-09-10 22:11) . . . . .	19
	Backtracking algorithm: rat in maze (2015-09-10 22:21) . . . . .	19
	Leetcode 106: construct binary tree from inorder and post order traversal (2015-09-13 00:29) . . .	20
	Java Script, C# learning - reading / videos material (2015-09-21 23:02) . . . . .	22
	Leetcode 109: convert sorted list to a binary search tree (2015-09-22 21:33) . . . . .	35
	Leetcode: Convert sorted array to binary search tree (2015-09-25 20:54) . . . . .	41
	Study time - watch videos, and read articles (2015-09-25 23:04) . . . . .	44
	Object oriented principles - S.O.L.I.D. (2015-09-26 00:10) . . . . .	46
	ASP.NET framework (2015-09-27 22:03) . . . . .	48
1.2	October . . . . .	48
	Good API video watching and notes taken (2015-10-22 00:04) . . . . .	48
	The clean code talks - unit testing (2015-10-25 12:50) . . . . .	50
	Study notes - "The clean code talks - 'Global State and Singletons' " (2015-10-25 14:07) . . . . .	52
	reading time on Sunday - housing issue, health issue, problem solving (2015-10-25 15:51) . . . . .	54
	Back to leetcode code algorithms (2015-10-26 22:44) . . . . .	56
	Leetcode question 241: Different ways to add parentheses (2015-10-28 21:40) . . . . .	57
1.3	November . . . . .	59
	testing, automation and testing patterns (2015-11-08 12:51) . . . . .	59
	OO design principles - DRY - Don't repeat yourself principle (2015-11-14 15:34) . . . . .	62
	book reading: 97 Things Every Programmer Should Know (I) (2015-11-15 13:05) . . . . .	63
	The clean code talks (2015-11-18 10:05) . . . . .	68
	Study time - OO design, principles and testability (2015-11-21 12:24) . . . . .	75
	Study time - Learn dependency injection, and others (2015-11-22 12:49) . . . . .	78
	Testing and Refactoring Legacy Code (2015-11-25 22:55) . . . . .	79
	Study time - watch CppCon videos (2015-11-29 15:46) . . . . .	80

1.4	December	82
	C++ - write quick code in C++ (2015-12-03 00:22)	82
	Coding standards - quick review (2015-12-04 23:27)	83
	C++, coding standards, and code refactoring (2015-12-05 11:54)	84
	Algorithms, performance with data structures (2015-12-05 14:36)	84
	C++ core guidelines (2015-12-07 22:57)	85
	reading book - Exceptional C++ (2015-12-08 22:12)	87
	Coding principles, good/bad design (2015-12-10 21:10)	88
	Learn C++ (2015-12-12 14:39)	89
	C++ - make simple tasks simple (2015-12-12 14:43)	89
	Book reading: a tour of C++ (2015-12-12 14:46)	89
	Book reading: Mobile First (2015-12-13 00:07)	91
	Book reading: 97 thing every programmer should know (II) (2015-12-13 23:50)	92
	OO principle - S.O.L.I.D., Single Responsibility, Open/ close principle drill (2015-12-15 20:36)	92
	Book Reading: 97 Things Every Programmer Should Know (III) (2015-12-17 19:01)	100
	Book Reading: 97 Things Programmer Should Know (IV) (2015-12-17 21:58)	101
	Book Reading: Effective Java (2015-12-18 00:19)	102
	Book reading: Head first Design pattern (2015-12-20 21:45)	103
	OO principles - The Open / Closed Principle (2015-12-22 22:58)	104
	Code School: courses study (2015-12-27 15:10)	105
	Think Fast, Talk Smart: Communication Techniques (2015-12-27 15:26)	107
	Vacation break in 2015 - from Dec. 28 to 31, 2015 (2015-12-29 23:20)	109
<b>2</b>	<b>2016</b>	<b>113</b>
2.1	January	113
	Favorites of top 10 rules for success (2016-01-01 01:18)	113
	Divide and Conquer: Preorder traversal of ternary tree (2016-01-06 21:49)	117
	Bootstrap - CSS framework (2016-01-08 15:34)	119
	Leetcode algorithm 160-170 study (2016-01-10 12:29)	120
	Leetcode 51: N - Queen problems (2016-01-13 00:45)	121
	Google hiring Best Practices - How to interview others? (2016-01-16 14:00)	123
	Leetcode 314: Binary Tree Vertical Order Traversal (2016-01-16 14:04)	124
	Cracking the Coding Interview (Part 1, 2 of 2) (2016-01-16 14:16)	125
	Leetcode 5: Longest substring palindrome (2016-01-16 14:20)	133
	pluralsight: online courses study (2016-01-16 14:24)	135
	Learn Google interview process - 25 things to learn (2016-01-17 01:12)	137

string function - strStr - BoyerMoore algorithm - needle in hay (2016-01-17 14:27) . . . . .	143
Algorithms night (1) (2016-01-17 22:57) . . . . .	144
Leetcode 317: Shortest distance from all buildings (2016-01-20 00:19) . . . . .	151
Book Reading: C# book Effective C# (2016-01-20 21:38) . . . . .	154
Leetcode questions 20: Valid Parentheses (2016-01-20 21:44) . . . . .	154
Algorithm night (2): 10 questions to read (2016-01-20 22:08) . . . . .	156
Algorithm night (3) (2016-01-21 22:54) . . . . .	159
Leetcode 17: Letter Combinations of a phone number (DFS) (2016-01-24 15:02) . . . . .	161
Leetcode 1, 15, 16: Two sum, 3 sum, 3 sum closest (2016-01-24 20:44) . . . . .	165
Sunday - Algorithm day (4) - 25 algorithms study (2016-01-31 12:07) . . . . .	166
2.2 February . . . . .	171
quicksort: a practice makes difference (2016-02-02 21:53) . . . . .	171
Algorithm: Merge two sorted singly linked list (2016-02-03 00:43) . . . . .	175
Algorithm: Count the number of palindromes in a string (2016-02-03 21:39) . . . . .	176
Algorithm: reading blog (2016-02-03 22:25) . . . . .	176
Netflix company culture - take some notes (2016-02-05 21:56) . . . . .	180
Do not complain - a tip worthy to share (2016-02-06 00:28) . . . . .	183
Sunday - Reading time - Career Advice (2016-02-07 11:15) . . . . .	184
Sunday - Reading time (2) (2016-02-07 12:11) . . . . .	186
Leetcode 318: Maximum Product of Word Length (2016-02-07 14:46) . . . . .	187
Leetcode 322: Coin Change (2016-02-08 01:10) . . . . .	189
Leetcode 319: Bulb Switch (2016-02-08 01:14) . . . . .	189
Leetcode : Wiggle Sort (2016-02-08 01:20) . . . . .	190
Leetcode 328: Odd Even Linked List (Easy) (2016-02-08 01:22) . . . . .	191
Leetcode 329: Longest increasing path in matrix (2016-02-08 01:27) . . . . .	192
Leetcode 331: Verify Preorder serialization of a Binary Tree (2016-02-08 01:39) . . . . .	193
Leetcode 295: Find medium from data stream (2016-02-09 00:22) . . . . .	194
Algorithm: Write a function to detect if the string has the unique character (2016-02-09 00:25) . . .	195
Algorithm: Possible Triangle (2016-02-09 00:29) . . . . .	201
Leetcode 310: Minimum Height Trees (2016-02-09 00:41) . . . . .	202
Leetcode 316: Remove duplicate letters (2016-02-09 00:46) . . . . .	202
Leetcode: Longest increasing subsequence (2016-02-09 00:48) . . . . .	203
Video watching: The Future of Analytics (2016-02-10 22:05) . . . . .	205
Sunday: Reading time (3) (2016-02-13 10:58) . . . . .	206
Pluralsight: Responsive In-Browser Web Page Design with HTML and CSS (2016-02-16 19:39) . . . .	209

Leetcode 312: Burst Balloons (2016-02-17 23:27) . . . . .	209
New School - HackerRank - algorithm: Beautiful Pairs (2016-02-21 13:09) . . . . .	211
HackerRank - New School - Algorithm: Fix the cycles (2016-02-21 21:37) . . . . .	212
HackerRank - New School - Nice code sprint on algorithm challenges (2016-02-21 21:39) . . . . .	213
HackerRank - New School - JavaScript (2016-02-22 00:12) . . . . .	214
Mock interview experience (I) (2016-02-24 00:06) . . . . .	214
Algorithm contests? Competitive programming? (2016-02-24 23:32) . . . . .	215
React and Flux for Angular Developers (2016-02-26 00:01) . . . . .	216
HackerEarth: first algorithm practice - Milly Chocolate (2016-02-27 01:26) . . . . .	217
HackerRank: Pangram (2016-02-28 01:49) . . . . .	218
code challenge: count of substring (2016-02-28 18:44) . . . . .	219
Blogs Reading: Algorithm problems (2016-02-28 23:42) . . . . .	220
Pluralsight: AngularUI Fundamentals (2016-02-29 18:46) . . . . .	221
Mental toughness training - small talk how to perform algorithm as well (2016-02-29 18:50) . . . . .	221
2.3 March . . . . .	222
Angular - put a web app together in short future (2016-03-01 23:26) . . . . .	222
Mock interview experience (2016-03-02 00:32) . . . . .	222
HackerRank: HourRank 6 (2016-03-03 00:36) . . . . .	224
HackerRank: Bear and Steady Gene (I) (2016-03-04 01:05) . . . . .	227
HackerRank: Bear Steady Gene (II) (2016-03-05 17:22) . . . . .	230
HackerRank: Bear and Steady Gene algorithm (III) (2016-03-05 22:28) . . . . .	233
HackerRank: Bear and Steady Gene algorithm (IV) (2016-03-05 23:22) . . . . .	235
HackerRank: Bear And Steady Gene - binary search algorithm (V) (2016-03-06 00:45) . . . . .	238
Sunday watching: Jamie Dimon to HBS MBA Class of 2009 (2016-03-06 17:02) . . . . .	240
HackerRank: Bear And Steady Gene - Binary Search (II) (2016-03-07 00:09) . . . . .	241
Pluralsight: JavaScript for C# developer (2016-03-08 00:41) . . . . .	244
Pluralsight: Structuring JavaScript (2016-03-08 20:47) . . . . .	244
CodeSchool: AngularJS Tutorial (2016-03-09 23:24) . . . . .	246
Pluralsight: Angular Fundamentals (2016-03-11 00:40) . . . . .	247
small talk: what you can control, 3 variables from Michael Bloomberg (2016-03-11 21:20) . . . . .	247
HackerRank: String algorithm - funny string (2016-03-12 19:03) . . . . .	248
HackerRank - String algorithm - Alternating Characters (2016-03-12 19:32) . . . . .	249
HackerRank: String algorithm - Game Throne. (2016-03-12 21:52) . . . . .	249
HackerRank - strings - GemStones (2016-03-12 23:26) . . . . .	250
Sunday study time: Good advice how to have good work ethnics (2016-03-13 11:51) . . . . .	251

HackerRank: string algorithm - Make it anagram (2016-03-13 14:41)	252
HackerRank: string algorithm - Anagram (2016-03-13 22:41)	254
Mock interview (4th practice): Matrix Spiral Print (2016-03-16 22:49)	256
Mock interview (practice III) - Award budget cut (2016-03-17 00:49)	259
Hacker Rank: Two Strings - thinking in C# 15+ ways (2016-03-18 21:41)	260
HackerRank: Two string - thinking in JavaScript over 10 ways (2016-03-19 22:22)	266
BootStrap CSS framework: Building Responsive UI with Bootstrap 6-hours - Full Sample code (2016-03-20 11:19)	269
Reading time: blogs (2016-03-20 22:53)	270
Leetcode 238 Product of Array except itself (2016-03-21 22:50)	270
Pluralsight: Angular Fundamentals (2016-03-22 21:29)	272
AngularJs fundamentals (2016-03-22 21:40)	272
Pluralsight: Play by Play: Modernizing C++ Code with Kate Gregory (2016-03-23 20:37)	273
Learning AngularJS - developer org document (2016-03-23 20:38)	273
Learning JavaScript (2016-03-23 20:39)	274
Pluralsight: C++ Core Guidelines and the Guideline Support Library (2016-03-23 20:40)	274
Pluralsight: C++ Advanced Topics (2016-03-23 22:14)	276
HackerRank: Two string - thinking in Cplusplus over 15 ways (2016-03-24 19:33)	276
Leetcode 33: Search in sorted rotated array (2016-03-25 11:32)	280
HackerRank: Two string - thinking in Java (2016-03-25 20:16)	281
HackerRank: String - Sherlock and anagrams (I) (2016-03-27 09:02)	282
HackerRank: Sherlocks and Anagram (III) (2016-03-27 11:26)	284
HackerRank: Sherlock and anagrams (II) (2016-03-27 13:04)	285
HackerRank: Sherlock and Anagrams IV (2016-03-27 13:19)	291
HackerRank: Sherlock and anagrams (V) (2016-03-27 14:52)	292
HackerRank: Sherlock And Anagram (VI) (2016-03-27 16:21)	296
Video watching: a guide to object-oriented practice (2016-03-27 21:25)	298
HackerRank: String algorithm - Palindrome index (2016-03-30 23:08)	299
HackerRank: string algorithm - Reverse Shuffle Merge (I) - First step: Failed twice (2016-03-31 08:47)	300
2.4 April	301
HackerRank: string algorithm - Reverse Shuffle Merge (II) - next - forming a partial correct idea (2016-04-02 13:28)	301
Practice difference: How top sports player practice? (2016-04-02 17:45)	303
10-000 hour rule (2016-04-03 08:52)	306
HackerRank: string algorithm - Reverse Shuffle Merge (III) (2016-04-03 16:06)	306
HackerRank: string algorithm - Reverse Shuffle Merge (IV) (2016-04-03 16:58)	307

Algorithm, Rotate matrix nxm clockwise 90 degree (2016-04-05 17:58) . . . . .	310
HackerRank articles - reading time (2016-04-05 21:35) . . . . .	312
Distributed System - study time (2016-04-07 21:49) . . . . .	314
JavaScript - slide show case studies (2016-04-07 21:58) . . . . .	315
Article reading: Microsoft Interview Process (2016-04-08 22:42) . . . . .	316
Article reading: How to get hired at Microsoft (2016-04-08 23:08) . . . . .	317
Article reading: Get that coveted tech interview (2016-04-08 23:19) . . . . .	318
Elizabeth Holmes's Top 10 rules for success (2016-04-09 11:05) . . . . .	318
HackerRank - String algorithm - Count Strings (I) (2016-04-09 11:15) . . . . .	319
HackerRank: Count strings (II) (2016-04-09 11:41) . . . . .	320
HackerRank: Count string (III) (2016-04-09 13:53) . . . . .	323
HackerRank: count string (IV) - JavaScript (2016-04-09 16:02) . . . . .	328
HackerRank - count string (V) - C plus plus solution (2016-04-09 16:36) . . . . .	328
Mental skills to help players (2016-04-09 22:08) . . . . .	329
HackerRank: string function calculation - string algorithm - Brute Force Solution (2016-04-10 15:35)	330
HackerRank: string function calculation (II) - string algorithm - Boyer Moore Algorithm (2016-04-10 15:44) . . . . .	331
HackerRank: String Calculate function (III) - Suffix array (2016-04-10 21:16) . . . . .	332
HackerRank: String Calculate function (III) - Suffix array (II) (2016-04-11 20:15) . . . . .	333
K Index algorithm (2016-04-12 08:45) . . . . .	337
rotate array (2016-04-12 08:48) . . . . .	338
Advice for practice on coding question - easy, medium to hard questions (2016-04-12 21:25) . . . .	341
HackerRank: String Calculate function (III) - LCP array (2016-04-13 20:34) . . . . .	342
HackerRank: Matrix Rotation (2016-04-14 18:12) . . . . .	342
HackerRank: Connected Cell in a Grid - C# solution - using DFS (2016-04-16 15:03) . . . . .	344
HackerRank - Connected Cell In A Grid (II) - C# solution (II) using Queue (2016-04-17 16:19) . . . . .	345
HackerRank: Connected Cell in a Grid (III) - C# solution (III) - using recursive function (2016-04-17 17:08)	347
reading time - reading articles (2016-04-17 18:23) . . . . .	349
HackerRank: Connected Cells in a grid (IV) - JavaScript code (2016-04-17 20:37) . . . . .	351
HackerRank: Connected Cells in a grid (V) - C++ solution (2016-04-17 20:39) . . . . .	352
K index - Algorithm (II) using Queue (2016-04-18 23:13) . . . . .	353
HackerRank: Matrix Rotation (II) (2016-04-18 23:26) . . . . .	354
HackerRank: Matrix Rotation (III) - using extra space - an array (2016-04-18 23:29) . . . . .	355
video: How to Keep Calm at Crunch Time - Ask Ian #31 (2016-04-19 21:35) . . . . .	356
HackerRank: facts to share (2016-04-19 23:28) . . . . .	357
C# tutorial - reading first, coding follows (2016-04-21 20:55) . . . . .	357



K nearest point (2016-04-21 21:04) . . . . .	358
Window minimum (2016-04-21 21:05) . . . . .	359
Matrix shortest distance set - get its Maximum value (2016-04-21 21:06) . . . . .	360
Find optimal weight (2016-04-21 21:07) . . . . .	360
Two rectangles to see if they overlap (2016-04-21 21:08) . . . . .	360
Insert value into a circle linked list (2016-04-21 21:09) . . . . .	360
Leetcode 17 Phone number combination - practice (II) (2016-04-21 21:10) . . . . .	365
Search in Matrix (2016-04-21 21:11) . . . . .	366
Two sum pair (2016-04-21 21:11) . . . . .	366
Merge two sorted linked list (2016-04-21 21:12) . . . . .	367
longest palindrome substring (2016-04-21 21:13) . . . . .	367
A binary tree is subtree of another binary tree (2016-04-21 21:14) . . . . .	367
valid parenthesis pairs (2016-04-21 21:14) . . . . .	369
gray code (2016-04-21 21:15) . . . . .	370
Remove Vowels From A String in Place (2016-04-21 21:15) . . . . .	375
Largest continuous sub array problem (2016-04-22 21:39) . . . . .	377
Fisher-Yates algorithm (2016-04-22 22:45) . . . . .	379
Tortoise-hare algorithm (2016-04-22 22:47) . . . . .	381
Find the lowest common ancestor of two nodes in a Binary Tree. (2016-04-22 22:48) . . . . .	381
Find if a Directed Acyclic Graph has a cycle. (2016-04-22 22:50) . . . . .	382
HackerRank: Even Tree - Graph Problem (I) - Just thinking (2016-04-23 14:46) . . . . .	383
HackerRank: Even Tree - Graph Problem (II) - Coding first try (2016-04-23 14:59) . . . . .	384
HackerRank: Even Tree - C# solutions to study (III) (2016-04-23 17:23) . . . . .	384
HackerRank: Even Tree (V) C# solution - use queue to help counting spanning tree's node (2016-04-23 22:05) . . . . .	385
Back tracking - N Queen problem (2016-04-24 09:50) . . . . .	388
Leetcode 314: Binary Tree Vertical Order Traversal (2016-04-24 09:57) . . . . .	388
Articles to read - big O cheat sheet (2016-04-24 21:51) . . . . .	389
Skip List (2016-04-24 22:05) . . . . .	389
Leetcode 235: Lowest common ancestor in binary search tree (2016-04-25 18:21) . . . . .	390
Leetcode 236: Lowest common ancestor in binary tree (2016-04-25 18:23) . . . . .	392
Leetcode 266: Palindrome permutation (2016-04-25 18:32) . . . . .	393
Leetcode - 300 algorithms - Learning by Reading Code First (2016-04-26 21:35) . . . . .	393
Leetcode 146: LRU - Cache (2016-04-26 21:36) . . . . .	394
HackerRank: Bear Steady Gene (II) - Better Code (2016-04-27 22:00) . . . . .	397
Clean Code - book reading (2016-04-27 22:06) . . . . .	401

The art of readable code - book review second time (2016-04-27 22:12)	401
HackerRank: Bear And Steady Gene - binary search algorithm (VI) (2016-04-28 21:10)	401
Testable JavaScript - Book Reading (2016-04-28 21:15)	404
What are the best programming blogs? (2016-04-28 22:57)	404
Leetcode 200: Number of Island (2016-04-30 11:19)	405
Leetcode 207: course schedule (2016-04-30 11:22)	405
Leetcode 210: course schedule (2016-04-30 11:28)	406
coursera course: Algorithm (2016-04-30 12:31)	410
2.5 May	410
Leetcode 133: clone graph (2016-05-01 13:56)	410
Leetcode 261: graph-valid-tree (2016-05-01 14:59)	414
Leetcode 323: Number of connected components (2016-05-01 15:16)	414
HackerRank: article reading (2016-05-03 18:32)	415
HackerRank: Delete duplicate value nodes from a sorted linked list (2016-05-06 21:47)	415
HackerRank: Linked List - Find Merge Point of two linked list (2016-05-08 11:34)	419
HackerRank: Insert a node in a double linked list (2016-05-08 15:42)	420
Reverse a linked list (2016-05-09 21:01)	423
HackerRank: Linked List - Get Node value (2016-05-09 21:01)	424
Linked List: Delete a node (2016-05-09 21:02)	424
Insert a node at tail of a linked list (2016-05-09 21:03)	424
Insert a node at the head of a linked list (2016-05-09 21:03)	425
Insert a node at a specific position in a linked list (2016-05-09 21:03)	425
Linked List: print list (2016-05-09 21:04)	426
Merge two sorted linked list (2016-05-09 21:08)	426
Reverse a doubly linked list: (2016-05-09 21:08)	426
Compare two linked list (2016-05-09 21:09)	427
Reverse print (2016-05-09 21:10)	427
Height of a tree (2016-05-09 22:01)	428
HackerRank: Algorithms > Sorting (2016-05-09 22:33)	428
HackerRank: Search algorithms (2016-05-09 22:35)	429
HackerRank: Greedy Algorithms (2016-05-09 22:37)	429
Common mistakes in code writing (2016-05-09 22:40)	429
External merge sort - Duplicate Elements of An Array (2016-05-10 20:24)	430
HackerRank - Connected Cell in a Grid - Warm up with Five Practices (2016-05-11 19:07)	431
HackerRank - Connected Cell in a Grid - Warm up with Five Practices (II) (2016-05-11 19:15)	433

HackerRank – Connected Cell in a Grid - Warm up with Five Practices (III) (2016-05-11 19:19) . . . .	433
HackerRank – Connected Cell in a Grid - Warm up with Five Practices (IV) (2016-05-11 19:23) . . . .	434
HackerRank – Connected Cell in a Grid - Warm up with Five Practices (V) (2016-05-11 19:30) . . . .	434
Leetcode 317 - shortest distance from all building - a warm up practice (2016-05-11 19:33) . . . .	435
Leetcode 239: sliding window maximum (2016-05-11 19:44) . . . . .	435
Leetcode 48: rotate image (2016-05-11 21:49) . . . . .	437
Algorithm: Rotate array by one element (2016-05-11 21:52) . . . . .	437
Leetcode 17: Phone Number - Practice using DFS/ Stack and BFS/ Queue (2016-05-12 19:07) . . . .	438
Some algorithms to study (2016-05-13 22:56) . . . . .	439
Top 10 strategies Work Hard for Entrepreneurs (2016-05-14 12:59) . . . . .	441
Les Brown's motivation talk (2016-05-14 14:08) . . . . .	442
Roger Federer's Top 10 Rules For Success (2016-05-15 00:02) . . . . .	443
Leetcode 4: Median of two sorted array - a warmup practice (2016-05-15 14:28) . . . . .	444
Leetcode 215: Find kth largest element in the array (2016-05-15 15:46) . . . . .	445
Connect nodes at the same level in a binary tree (2016-05-16 22:44) . . . . .	447
web technology - a short research in Vancouver technology industry (2016-05-16 23:21) . . . . .	448
Human resource analytical articles (2016-05-17 08:36) . . . . .	449
Leetcode 16 - 3 sum closest (2016-05-17 19:45) . . . . .	449
System Design (2016-05-17 19:46) . . . . .	451
Leetcode 15: 3 Sum (2016-05-17 22:20) . . . . .	451
Leetcode 18: 4 Sum (2016-05-17 22:22) . . . . .	453
Leetcode 236: Lowest common ancestor (2016-05-18 23:30) . . . . .	453
Leetcode 236: Lowest Common Ancestor - Warm up practice (2016-05-21 21:18) . . . . .	458
Leetcode 37: Sudoku - a warm up practice (2016-05-21 22:17) . . . . .	460
Leetcode 331: Verify Preorder serialization of a Binary Tree (2016-05-22 00:41) . . . . .	460
Radix Sort - a distribution sort (2016-05-22 17:48) . . . . .	461
Binary Tree Post Order Traversal Iterative solution (2016-05-22 20:56) . . . . .	462
Leetcode 236: Lowest Common Ancestor - practice (III) (2016-05-22 22:58) . . . . .	463
Leetcode 236: Binary Tree Lowest Common Ancestor - warm up practice (IV) (2016-05-23 09:57) . .	464
Leetcode 297: Serialize and deserialize a binary tree - 2 study cases (2016-05-23 14:31) . . . . .	465
Algorithm: check if the number is power of 2 (2016-05-24 22:43) . . . . .	468
LeetCode 159: Longest Substring with At Most Two Distinct Characters (2016-05-24 23:32) . . . . .	470
Leetcode 252: meeting room (2016-05-24 23:33) . . . . .	470
Leetcode 126: word ladder II (Hard) (2016-05-24 23:34) . . . . .	470
Leetcode 127: word ladder (Medium) (2016-05-24 23:34) . . . . .	471

Binary Tree Inorder traversal - Iterative solution - warm up practice (2016-05-25 20:55) . . . . .	472
string algorithms - C# practice (2016-05-25 21:14) . . . . .	474
Leetcode 88: Merge two sorted arrays (2016-05-25 21:16) . . . . .	475
Binary Tree Preorder Traversal - Iterative Solution - Warm up Practice (2016-05-25 23:24) . . . . .	475
HackerRank: string algorithm - Reverse Shuffle Merge - Stack, backtracking techniques (2016-05-26 21:05) . . . . .	478
Design patterns - Quick Study (2016-05-26 23:06) . . . . .	480
Leetcode 126: word ladder II - warm up practice (2016-05-28 22:18) . . . . .	481
Leetcode 126: Word Ladder II - warm up practice (II) (2016-05-29 13:49) . . . . .	482
Leetcode 126: Word Ladder II - warm up practice (III) (2016-05-29 23:34) . . . . .	484
Amazon Leadership principle study (2016-05-30 20:39) . . . . .	485
Leetcode 126: word ladder - a practice (IV) (2016-05-30 20:41) . . . . .	488
Leetcode 126: word ladder II - study C++ code using BFS - Queue (Practice V) (2016-05-30 21:13) . .	489
Leetcode 126: word ladder II - using BFS - Queue, Set Paths (Practice VI) (2016-05-30 21:36) . . . . .	491
Distributed System, noSQL, operating system Quick Review (2016-05-30 22:23) . . . . .	496
Leetcode 103: Binary Tree ZigZag traversal - a practice (2016-05-31 22:21) . . . . .	497
Leetcode 124: Binary Tree Maximum Path Sum - reasoning and analysis (I, II, III) (2016-05-31 22:40)	498
2.6 June . . . . .	502
Leetcode 146: LRU Cache - a practice makes difference (III) (2016-06-02 21:11) . . . . .	502
Leetcode 146: LRU cache - a practice makes difference (II) (2016-06-02 21:11) . . . . .	503
small talk on code interview (2016-06-02 21:28) . . . . .	505
Top 10 Ranking Algorithm problems - June 2016 (2016-06-02 23:30) . . . . .	507
Study on failing - a small research starts a learning Sunday (2016-06-05 10:18) . . . . .	509
Top 10 ranking algorithm and practice in June, 2016 (II) (2016-06-07 21:44) . . . . .	512
A small research on RestAPI, SOA, AWS, AZURE (2016-06-07 22:48) . . . . .	515
Content marketing - ideas to write blogs (2016-06-10 20:13) . . . . .	516
C# Hashtable vs Dictionary (2016-06-10 20:36) . . . . .	517
Learn to lose - my favorite verse to study (2016-06-11 11:13) . . . . .	519
Top 10 Sales techniques for Entrepreneurs (2016-06-11 11:56) . . . . .	519
Algorithms to study (2016-06-11 23:07) . . . . .	523
HackerRank: Sherlock and anagram - warmup practice after 3 months (2016-06-12 13:31) . . . . .	526
Array Class - C#, C++, JavaScript, Java (2016-06-12 20:20) . . . . .	528
Hashtable - C++, C#, Java, JavaScript (2016-06-12 20:23) . . . . .	536
Queue Class - C++, C#, Java, JavaScript (2016-06-12 20:55) . . . . .	537
Stack Class - C++, C#, Java, JavaScript (2016-06-12 20:55) . . . . .	538
Double Linked List - C++, C#, Java, JavaScript (2016-06-12 21:10) . . . . .	538

Good programmer or just good Googler - a small research (2016-06-14 20:23) . . . . .	540
Industry work vs academic research - blogs reading time (2016-06-14 20:25) . . . . .	541
The Art of Speaking: Scott Hanselman (2016-06-14 20:30) . . . . .	543
Leetcode 269: Alien Dictionary (2016-06-15 22:42) . . . . .	544
LINQ - Language-Integrated Query (2016-06-15 22:50) . . . . .	549
Leetcode 269 - Alien Dictionary - Practice 2 more times (2016-06-16 21:27) . . . . .	550
LINQ Architecture - Pluralsight.com (2016-06-17 23:28) . . . . .	555
LINQ Fundamentals - pluralsight.com (2016-06-18 11:43) . . . . .	555
.Net programming technologies (2016-06-18 16:17) . . . . .	556
Rest Fundamentals - pluralsight.com (2016-06-19 12:55) . . . . .	557
Leetcode 329: Longest increasing path in matrix (2016-06-20 23:23) . . . . .	558
Strength knowledge of data structures and algorithms (2016-06-21 00:21) . . . . .	561
ATP - Tennis Coaching - Sports coaching (2016-06-21 23:47) . . . . .	562
Leetcode 139: word break I - 3+ practices (2016-06-22 18:01) . . . . .	563
HackerRank: world code sprint #4 - minimum distance (2016-06-26 00:56) . . . . .	567
HackerRank: Equal stacks (2016-06-26 00:59) . . . . .	567
HackerRank: AorB - intense workout - 3 hours+ (2016-06-26 01:03) . . . . .	568
Remove leading 0 in HexaDecimal string (2016-06-26 10:21) . . . . .	572
Convert a binary number from 0 - 16 to a char (HexaDecimal binary string to a char) (2016-06-26 10:23)	573
Convert a HexaDecimal char to an integer value (2016-06-26 10:26) . . . . .	573
Reverse a string (2016-06-26 10:32) . . . . .	573
AorB - HackerRank workout example (2016-06-26 12:45) . . . . .	573
Leetcode solution to study (2016-06-26 13:31) . . . . .	576
Hexadecimal number bit manipulation practice (2016-06-26 22:31) . . . . .	577
A comparison of Microsoft web technology - pluralsight (2016-06-27 20:54) . . . . .	578
Hexadecimal OR calculation (2016-06-27 21:32) . . . . .	578
A small research on HackerRank world sprint #4 (2016-06-27 22:37) . . . . .	579
HackerRank: World codesprint #4 - Roads in HackerLand (2016-06-29 22:07) . . . . .	582
JavaScript Array - 2+ hour study (2016-06-30 19:35) . . . . .	584
JavaScript Array.from function - 60 minutes study (2016-06-30 22:13) . . . . .	588
JavaScript Array.of Function study (2016-06-30 22:23) . . . . .	589
2.7 July . . . . .	590
Office politics, hard conversation, company culture study (2016-07-01 10:51) . . . . .	590
Teach people how to be a manager from a programmer background (2016-07-01 13:24) . . . . .	592
Nine principles of innovation (2016-07-01 13:26) . . . . .	593

JavaScript Array developer.Mozilla.org - 3+ hours study (2016-07-02 11:02) . . . . .	594
C# questions to review (2016-07-03 09:52) . . . . .	598
C#, C++, JavaScript, Java String, StringBuilder class study (2016-07-03 16:06) . . . . .	599
HackerRank: AorB - code submission C# study (2016-07-03 16:37) . . . . .	603
HackerRank: AorB - JavaScript code study (2016-07-03 18:51) . . . . .	604
AorB - HackerRank - C++ code study (2016-07-03 19:01) . . . . .	605
Facebook code lab - daily coding practice (2016-07-05 21:35) . . . . .	606
Leetcode 130: surrounded region (2016-07-06 21:41) . . . . .	608
Tennis Legends - hingis tactics (2016-07-06 22:31) . . . . .	608
Encouraging story from tennis player - World No. 50 Elena Vesnina - Wimbledon 2016 (2016-07-06 23:02) . . . . .	609
String primitive - JavaScript study (2016-07-07 20:17) . . . . .	611
pluralsight - Building Highly Scalable Web Applications in Azure (2016-07-07 22:44) . . . . .	616
A small talk on Learning Style Change - coding, system design (2016-07-09 00:43) . . . . .	616
A small research about frugality (2016-07-09 17:55) . . . . .	617
Business intelligent study (2016-07-09 22:46) . . . . .	618
Leetcode 56: Merge Intervals (2016-07-10 00:24) . . . . .	619
CSS learning / forgetting - a better cycle (2016-07-10 15:58) . . . . .	620
Data Analysis Fundamentals with Tableau - pluralsight.com (2016-07-10 21:09) . . . . .	623
Leetcode 142: Linked List Cycle II - two examples to show the idea (2016-07-14 22:04) . . . . .	624
Leetcode 142: Linked List Cycle II - first practice (2016-07-14 23:17) . . . . .	627
Leetcode 142: Linked List Cycle II - second practice (2016-07-14 23:57) . . . . .	630
Leetcode 142: Linked List Cycle II - third practice (2016-07-14 23:58) . . . . .	632
Sorted Linked List duplicates removal - Facebook code lab (2016-07-16 10:17) . . . . .	632
Coding practice talk (2016-07-16 10:32) . . . . .	633
Bulbs - facebook code lab - five practices (2016-07-16 11:53) . . . . .	635
Flatten - facebook code lab - 2 practices (2016-07-16 13:58) . . . . .	636
Reverse unsigned 32 bit integer - facebook code lab - 5+ practice (2016-07-16 16:47) . . . . .	638
Swap odd bit and even bit of unsigned 32 bit integer (2016-07-16 17:26) . . . . .	641
Reverse unsigned 32 bit integer - facebook code lab - No.1 Java code (2016-07-16 17:26) . . . . .	643
Reverse unsigned 32 bit integer - facebook code lab - 5th practice - C++, swap bits (2016-07-16 17:32)	643
Reverse unsigned 32 bit integer - facebook code lab - how to express Math.pow using bit shift (2016-07-16 17:47) . . . . .	644
Remove duplicates from Sorted Array (2016-07-16 22:20) . . . . .	646
DiffK - facebook code lab - practices (2016-07-16 23:16) . . . . .	646
Big Data Analytics with Tableau - pluralsight.com (2016-07-17 12:24) . . . . .	647

Majority - facebook code lab - optimal solution (2016-07-18 00:08) . . . . .	648
A small research how top tennis player progresses to be top 7 from 152 ranking (2016-07-18 00:31)	649
Leetcode 173: Binary Search Tree Iterator - facebook code lab (2016-07-18 22:15) . . . . .	650
Leetcode 23: Merge K Sorted Lists (2016-07-19 22:13) . . . . .	651
HackerRank: World codesprint #4 - Roads in HackerLand - II - code study (2016-07-20 22:27) . . . .	655
Introduction to website layout - pluralsight.com (2016-07-20 23:22) . . . . .	657
CSS in-depth - pluralsight.com (2016-07-20 23:24) . . . . .	657
Typograph for web - pluralsight.com (2016-07-20 23:29) . . . . .	661
Creating Responsive Landing Pages in Photoshop and CSS - pluralsight.com (2016-07-20 23:36) . . .	662
The Art of A/B Testing for Web Design - pluralsight.com (2016-07-20 23:38) . . . . .	662
Merge N sorted Array - merge sort from bottom-up implementation (2016-07-21 23:51) . . . . .	663
AWS Developer Fundamentals - pluralsight.com (2016-07-22 20:05) . . . . .	664
GALS show - favorite time to spend 30 minutes (2016-07-23 13:46) . . . . .	664
Union-Find Algorithm - an undirected graph algorithm (2016-07-23 15:08) . . . . .	665
HackerRank - project euler (2016-07-23 15:17) . . . . .	666
Short Palindrome - HackerRank - world codesprint #5 (2016-07-25 00:03) . . . . .	666
Camel Case - HackerRank - world codesprint #5 (2016-07-25 00:03) . . . . .	669
String Construction - HackerRank - world codesprint #5 (2016-07-25 00:05) . . . . .	670
HackerRank - world codesprint #5 Longest increasing subsequence arrays (2016-07-25 00:16) . . . .	670
Balanced forest - HackerRank - world codesprint #5 (2016-07-25 00:23) . . . . .	671
Build a palindrome - HackerRank world codesprint #5 (2016-07-25 00:28) . . . . .	672
Longest common prefix - LCP - facebook code lab (2016-07-25 21:42) . . . . .	673
longest palindrome - facebook code lab (2016-07-25 21:42) . . . . .	675
substring - facebook code lab - Leetcode 30 (2016-07-25 21:43) . . . . .	676
Pascal's triangle - facebook code lab (2016-07-25 21:44) . . . . .	678
Factorial - facebook code lab (2016-07-25 21:45) . . . . .	678
Lintcode 79: longest common substring (2016-07-26 21:59) . . . . .	679
Binary Tree Path Sum - two with same value checking (2016-07-28 20:06) . . . . .	683
Leetcode 347: Find k most frequent numbers in the array - C# solution (2016-07-28 23:00) . . . . .	685
Leetcode 347 - Find k most frequent numbers in the array - C++/ Java Solutions (2016-07-29 19:30) .	686
Introduction to LINQ Queries (C#) - Language Integrate Query (LINQ) study series (I) (2016-07-29 20:43)	687
3+ tips - how to read technical articles quickly (2016-07-29 20:45) . . . . .	688
.NET collections - Language Integrate Query - LINQ study series (II) (2016-07-29 20:47) . . . . .	689
Generics in .NET framework - Language Integrate Query (LINQ) study series (III) (2016-07-29 20:49) .	691
Simplify your programs with LINQ - Language Integrate Query (LINQ) study series IV (2016-07-29 20:51)	692

	Summary of study for Leetcode 347 - K most frequent element in the array (2016-07-29 20:54) . . .	693
	K largest elements in the array - various ideas (2016-07-29 20:57) . . . . .	694
	Binary search advanced (2016-07-31 23:27) . . . . .	695
2.8	August . . . . .	695
	AWS study - pluaralsight (2016-08-01 11:43) . . . . .	695
	Productivity Tips for the Busy Tech Professional - pluralsight.com (2016-08-01 11:48) . . . . .	696
	HackerRank - Prepare to get experience on advanced level algorithms on HackerRank (2016-08-01 12:33) . . . . .	697
	Leetcode 124: Binary tree maximum path sum - a quick review (2016-08-04 21:22) . . . . .	698
	Leetcode 124: Binary Tree Maximum Path Sum - Single Responsibility Principle (SRP) (2016-08-04 21:27) . . . . .	701
	ITINT5: tree maximum path sum (II) (2016-08-04 21:54) . . . . .	704
	Count inversions - Extended merge sort - 3 Lecture Notes Study (2016-08-05 21:10) . . . . .	705
	A small research - tennis coaching vs algorithm lecturing (2016-08-07 14:30) . . . . .	711
	C# Extension Methods - pluralsight.com (2016-08-08 21:20) . . . . .	713
	Become a Full-stack .NET Developer - Architecture and Testing - pluralsight.com (2016-08-08 22:08)	713
	.NET distributed system architecture - pluralsight.com (2016-08-08 22:44) . . . . .	716
	C# Design Strategies - pluralsight.com (2016-08-08 22:55) . . . . .	716
	Mastering C# 4.0 - pluralsight.com (2016-08-08 22:58) . . . . .	716
	Designing Fluent APIs in C# - pluralsight.com (2016-08-08 23:29) . . . . .	716
	JavaScript - a programmer random thoughts (2016-08-11 23:51) . . . . .	717
	Clip: cropping an image - CSS learning and sharing on JSFiddle (2016-08-12 21:39) . . . . .	718
	Microsoft Development Service - watch and learn (2016-08-14 11:32) . . . . .	722
	Leetcode 380 - Insert, delete, getRandom() O(1) time - Practice 1 (2016-08-16 21:45) . . . . .	723
	Leetcode 380 - Insert, delete, getRandom() O(1) time - Practice 2 (2016-08-16 22:00) . . . . .	724
	Leetcode 380 - Insert, delete, getRandom() O(1) time - Practice 3 (2016-08-16 22:10) . . . . .	725
	Leetcode 380 - Insert, delete, getRandom() O(1) time - Practice 4 (2016-08-16 22:12) . . . . .	725
	JavaScript Style Guide Study - Airbnb (2016-08-17 21:05) . . . . .	727
	Become a Full-stack .NET Developer - pluralsight.com (2016-08-17 23:10) . . . . .	731
	Leetcode 278 - binary search algorithm and discussion (2016-08-19 19:24) . . . . .	733
	Leetcode 125 - valid palindrome - 5+ practice (2016-08-19 19:25) . . . . .	734
	Confidence and determination coaching - Nishikori Kei and his coach (2016-08-20 12:47) . . . . .	740
	Leetcode 125 - Valid Palindrome - Code styles study (2016-08-21 10:41) . . . . .	741
	Leetcode 125: Valid Palindrome (III) (2016-08-22 21:27) . . . . .	744
	Linked List Queue (2016-08-25 21:30) . . . . .	745
	Array Queue (2016-08-25 21:30) . . . . .	745



System Design: Design a URL shortening service (2016-08-25 22:06)	745
Leetcode 348: Design Tic-Tac-Toe (2016-08-25 22:10)	748
CSS - Responsive background image (2016-08-25 22:30)	748
Code study: Leetcode blogs (2016-08-25 22:30)	748
Programming Principles - study (2016-08-27 11:08)	750
Java Design Pattern - code study (2016-08-27 11:12)	751
Html, CSS style guideline study (2016-08-27 11:17)	751
Blog writing - aiming for good writing talent (2016-08-27 11:42)	751
Abbreviation - HackerRank world code sprint #6 (2016-08-28 12:10)	754
Bonetrroule - HackerRank world code sprint #6 - Practice 1 (2016-08-28 12:15)	755
Bonetrroule - HackerRank world code sprint #6 - Practice 2 (2016-08-28 12:22)	756
Bonetrroule - HackerRank world code sprint #6 - Practice 3 (2016-08-28 12:30)	756
Bonetrroule - HackerRank world code sprint #6 - code study (2016-08-28 12:31)	757
C# string class - API study (2016-08-28 21:37)	758
Bonetrroule - HackerRank world code sprint #6 - Time complexity (2016-08-28 22:13)	759
C# StringBuilder Class - study (2016-08-28 22:51)	760
Beautiful 3 set - code study (2016-08-29 23:48)	761
A small research on HackerRank - seek excellence (2016-08-31 00:25)	762
CSS - responsive background image (2016-08-31 23:02)	764
2.9 September	765
A drill - Leetcode solution code study (2016-09-01 23:34)	765
Book reading: competitive programming (2016-09-02 22:56)	765
System design - a new skill to acquire (2016-09-03 13:51)	766
A small research - project management (2016-09-03 14:23)	767
Leetcode 72: Edit distance - code study (2016-09-04 12:01)	767
Suffix Array and longest common prefix array (LCP array) - study (2016-09-05 10:56)	768
Sports talk: onsite coaching, long hour match, interview, double partner (2016-09-05 13:39)	770
Sports training - a programmer needs strong physical body to perform tasks - strong back muscle (2016-09-05 15:40)	772
Lecture study - Scalability Harvard Web Development (2016-09-07 20:06)	774
Longest Common Prefix - using Trie (2016-09-07 20:07)	775
10 Tips to help you perform to your highest potential in coding (2016-09-07 20:17)	775
Performance review - HackerRank world code sprint #4, #5, #6 (2016-09-07 22:14)	776



# 1. 2015

## 1.1 September

### Draw a circle algorithm (2015-09-10 22:11)

August 18, 2015

Interesting problem – draw a circle,

blogs to read:

1. [1]<http://petercai.com/the-microsoft-sde-interview/>
2. [2]<http://www.startuplessonslearned.com/2008/11/abcdefs-of-conducting-technical.html>
3. [3][https://en.wikipedia.org/wiki/Midpoint\\_circle\\_algorithm](https://en.wikipedia.org/wiki/Midpoint_circle_algorithm)

C # code:

[4]<https://github.com/jianminchen/AlgorithmsProblems/blob/master/DrawACircle.cs>

1. <http://petercai.com/the-microsoft-sde-interview/>
  2. <http://www.startuplessonslearned.com/2008/11/abcdefs-of-conducting-technical.html>
  3. [https://en.wikipedia.org/wiki/Midpoint\\_circle\\_algorithm](https://en.wikipedia.org/wiki/Midpoint_circle_algorithm)
  4. <https://github.com/jianminchen/AlgorithmsProblems/blob/master/DrawACircle.cs>
- 

### Backtracking algorithm: rat in maze (2015-09-10 22:21)

Sept. 10, 2015

Study again the back tracking algorithm using recursive solution, rat in maze, a classical problem. Made a few of mistakes through the practice, one is how to use two dimension array, another one is that "not all return path returns value", not so confident that "return false" at the end of function.

????????????, Rat in Maze, ?????? ! ??????, ?????????????, ?????????, ??????. ?????????, ?????, ??????????????.  
?????????, ??????????. ??????????, ??????????; ??, ??, "return false" ?????????????, ??????. ??, ?????????, ??????????; ?????????,  
?????????, ?????????.

Here are my favorite blogs about this problem:

1. [1]<http://www.geeksforgeeks.org/backtracking-set-2-rat-in-a-maze/>
2. [2]<http://algorithms.tutorialhorizon.com/backtracking-rat-in-a-maze-puzzle/>
3. [3]<https://www.cs.bu.edu/teaching/alg/maze/>



[3]<https://github.com/jianminchen/Leetcode>  
rderPostOrderTraversal.cs

\_C-/blob/master/106ConstructuBTreeFromIno-

In her coding practice of function build(...), she spent over 20 minutes to figure out the coding task: the code to partition the inorder traversal into two partitions, first is left subtree, second is right subtree. And then, post order traversal also can be partitioned into two intervals, first one is for left subtree, and then, second one is for right subtree.

She took more than 10-15 minutes to understand the solution. That is too long for real problem solving. Figure out that only job is to find the root node, and then, its left child and right child is also the root node of left subtree/ or right subtree. The recursive call, actually two of them, can help to do the task.

So, she needs to cut down the practice time, and make sure the calculation is correct, easy to tell/ maintainable code/ testable code. So, she decided to practice it using class Range instead of two arguments start/ end integers. Hopefully, this practice will enhance her memory about partition the array into two parts, one is dividing in mid point, another one is by length of first interval - left subtree.

```
public class Range {  
    public int start, end;  
    ...  
}
```

Also, she likes to enhance her memory about this solution, so, she writes second implementation using C #, and see if the code can pass online judge. Most important, she tried to use different solution, to improve, challenge herself. Write the code without any mistake first time, in less than 10 minutes based on previous one.

[4]<https://github.com/jianminchen/Leetcode> \_C-/blob/master/106ConstructBTreeFromInOrderPostOrderTraversal\_B.cs

Julia likes to train herself using leetcode questions, and biggest problem is to cut down the time to write a solution. She tries to cut down time from hours to 10-30 minutes.

After the code writing, she thought about more about great ideas out there, she should not miss. So, she reads the second reference, and like the most about the analysis:

"[5]Construct Binary Tree from Preorder and Inorder Traversal  
Z...A...H...".  
-

Julia , , , , 10-30 , . , , , ; , , , , .

, , , , .

, ; , 10-15-cn . , .

1. [http://siddontang.gitbooks.io/leetcode-solution/content/tree/construct\\_binary\\_tree.html](http://siddontang.gitbooks.io/leetcode-solution/content/tree/construct_binary_tree.html)
2. <http://blog.csdn.net/linhuanmars/article/details/24390157>

3. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/106ConstructuBTreeFromInorderPostOrderTraversal.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/106ConstructuBTreeFromInorderPostOrderTraversal.cs)
4. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/106ConstructBTreeFromInOrderPostOrderTraversal\\_B.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/106ConstructBTreeFromInOrderPostOrderTraversal_B.cs)
5. <http://blog.csdn.net/linhuanmars/article/details/24389549>

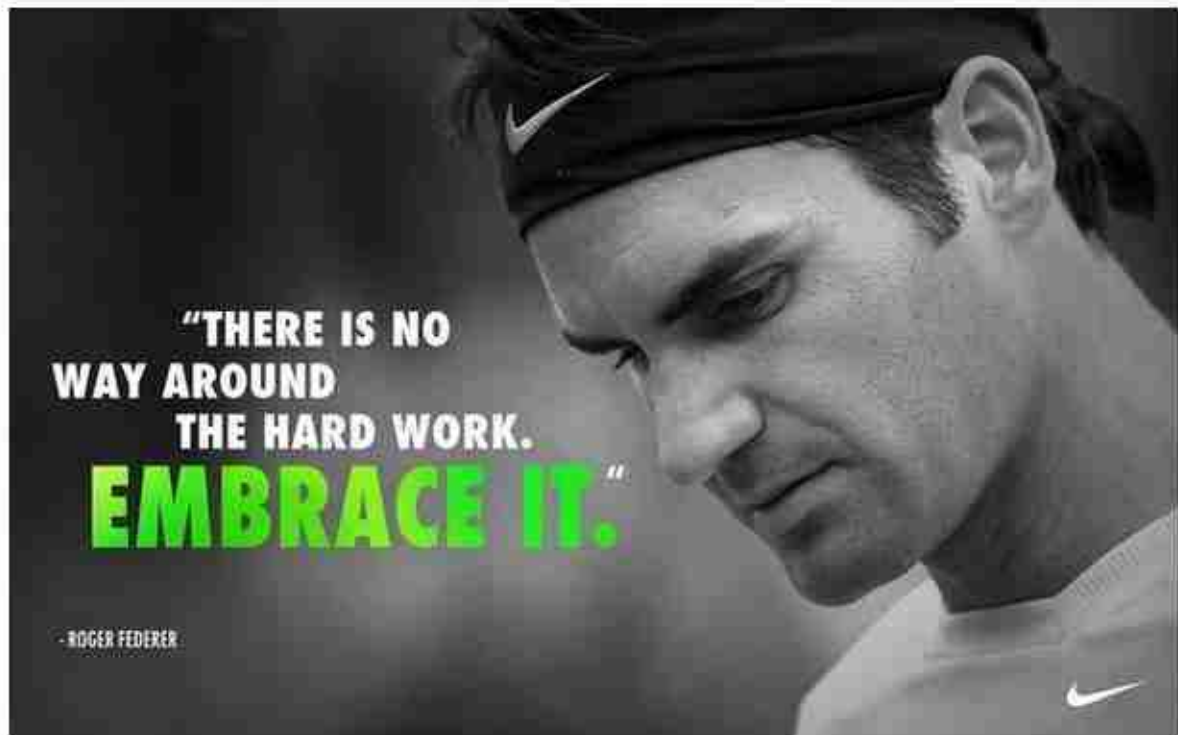
## Java Script, C# learning - reading / videos material (2015-09-21 23:02)

September 21, 2015

First, share my favorite two verses to encourage myself to work harder, and continue to pursuit efforts in JavaScript/ C # / CSS/ other technologies learning in my career. I should have more organized, I just copied all my posts from my facebook and documented in a blog in Google+. Review them, and get more reading / study done continuously while practicing Leetcode questions.



Share the verse I like as well. There is no way around the hard work. Embrace it. Repeat again, there is no way around the hard work. Embrace it. That is easy to remember.



Roger Federer, Fans from all over the world

RT de Roger

Nike Tennis @NikeTennis 19 min

Lessons From The Court 2013: @RogerFederer Share yours and tag it #CourtLessons

2014?????, ?????JavaScript???. ?????JavaScript, 6????, ?????????, ?????????, ??????. ??????, ??????, ??????.

From the beginning of 2014, I spent over 12 months to learn JavaScript; Try to explore all I can learn in JavaScript, expose myself to all the best teachers, reading /videos in the world. Learn how smart people are in this technology world, JavaScript area. First time, I did slow down everything to learn a functional programming language.

After 6 - 8 months, I found out that I start to generate ideas to rewrite the Javascript wrote before.

So, document my learning experience and help myself to move forward when I have time.

June 1, 2015

Share the video. I read chinese blog in wechat, and then, catch up this one in English.

CHM Revolutionaries: " How Google Works" Eric Schmidt & Jonathan Rosenberg

[1][https://www.youtube.com/attribution\\_link?a=s1oxvH9-h4E &u= %2Fwatch %3Fv %3D3tNpYpcU5s4 %26feature %3Dshare](https://www.youtube.com/attribution_link?a=s1oxvH9-h4E&u=%2Fwatch%3Fv%3D3tNpYpcU5s4%26feature%3Dshare)

March 15, 2015

So relax and also learn something on Sunday morning. Ajax and async.

[2][https://sec.ch9.ms/ch9/03b0/3ac5c264-043f-4ffd-b2ee-422f39ba03b0/Intro TojQueryM06 \\_mid.mp4](https://sec.ch9.ms/ch9/03b0/3ac5c264-043f-4ffd-b2ee-422f39ba03b0/Intro+TojQueryM06_mid.mp4)

February 22, 2015

Spent half hour on CSS preprocessor. Great topic, and think about benefits to use it.

[3][http://channel9.msdn.com/Series/Adding-Style-with-CSS/06?wt.mc\\_id=EntriesInArea](http://channel9.msdn.com/Series/Adding-Style-with-CSS/06?wt.mc_id=EntriesInArea)

Enjoyed one hour video about CSS transition and transformation. Sunday is such great time to learn something, and then, go out to play sports in Vancouver.

[4]<http://channel9.msdn.com/Series/Adding-Style-with-CSS/05>

Being a web developer, it is fun to learn something called transition, transformation; Great video.

[5][http://channel9.msdn.com/Series/Adding-Style-with-CSS/05?fb \\_action \\_ids=823960460986167 &fb \\_action \\_types=og.likes &fb \\_source=other \\_multiline &action \\_object \\_map= %5B771052266276726 %5D &action \\_type \\_map= %5B %22og.likes %22 %5D &action \\_ref \\_map= %5B %5D](http://channel9.msdn.com/Series/Adding-Style-with-CSS/05?fb_action_ids=823960460986167&fb_action_types=og.likes&fb_source=other_multiline&action_object_map=%5B771052266276726%5D&action_type_map=%5B%22og.likes%22%5D&action_ref_map=%5B%5D)

February 9, 2015

Cooking time - learn something.

[6][http://channel9.msdn.com/Series/Introduction-to-jQuery/02?wt.mc\\_id=player](http://channel9.msdn.com/Series/Introduction-to-jQuery/02?wt.mc_id=player)

February 7, 2015

Will have good time this weekend. Try to watch series video about single page application.

[7][http://channel9.msdn.com/Series/Single-Page-Applications-with-jQuery- or-AngularJS/02?wt.mc \\_id=EntriesInArea](http://channel9.msdn.com/Series/Single-Page-Applications-with-jQuery-or-AngularJS/02?wt.mc_id=EntriesInArea)

February 6, 2015

My favorite tip learned today. Learned a few tips, peek definition is also my favorite.

[8][http://channel9.msdn.com/Series/vstips/lazycodesnippets?wt.mc\\_id=player](http://channel9.msdn.com/Series/vstips/lazycodesnippets?wt.mc_id=player)

February 5, 2015

My Thursday night study time, Java Script is a good choice.



[9][http://channel9.msdn.com/Shows/Defrag-Tools/Defrag-Tools-37-JavaScript -Part-1?wt.mc\\_id=EntriesInArea](http://channel9.msdn.com/Shows/Defrag-Tools/Defrag-Tools-37-JavaScript-Part-1?wt.mc_id=EntriesInArea)

February 1, 2015

My favorite video this Sunday. Learn LINQ, good framework. Save time to develop.

[10][http://www.youtube.com/watch?v=tGr6xm9nUm4 &feature=share &fb\\_ref=share](http://www.youtube.com/watch?v=tGr6xm9nUm4&feature=share&fb_ref=share)

January 31, 2015

Saturday morning video. Nice to learn a few things, portable library is one.

Programming in C # ( 6 of 8) Splitting Assemblies, WINMD, Diagnostics and Instrumentation

[11]<http://youtu.be/ypdnMpR5jZ8>

January 26, 2015

Watch the video and learn something new.

Programming in C # (1/8) OOP, Managed Languages and C #

January 26, 2015

Share the video I like to watch. To test java script is new thing to learn.

[12][http://www.youtube.com/watch?v=yRMsxyGsez4 &list=PLfXiENmg6yyUpIVY9XVOkdbmBPx6PUm9 \\_ &feature=share &index=3 &fb\\_ref=share](http://www.youtube.com/watch?v=yRMsxyGsez4&list=PLfXiENmg6yyUpIVY9XVOkdbmBPx6PUm9_&feature=share&index=3&fb_ref=share)

Netflix: Stop Making Excuses and Start Testing Your JavaScript!

[13]<http://goo.gl/QBWjvA>

January 25, 2015

Sunday morning video, relax me and then learn something about Javascript.

Netflix JavaScript Talks - Async JavaScript with Reactive Extensions

[14][http://www.youtube.com/attribution\\_link?a=LtVczHAKqbM &u= %2Fwatch %3Fv %3DXRYN2xt11Ek %26feature %3Dshare &fb\\_ref=share](http://www.youtube.com/attribution_link?a=LtVczHAKqbM&u=%2Fwatch%3Fv%3DXRYN2xt11Ek%26feature%3Dshare&fb_ref=share)

January 23, 2015

Study time, good presentation!

Netflix JavaScript Talks - Version 7: The Evolution of JavaScript

[15][http://www.youtube.com/watch?v=DqMFX91ToLw &feature=share &fb\\_ref=share](http://www.youtube.com/watch?v=DqMFX91ToLw&feature=share&fb_ref=share)

January 22, 2015

So easy to find an advance topic about java script through youtube, and then enjoy the video.

Lo-Dash and JavaScript Performance Optimizations

[16]<http://www.youtube.com/watch?v=cD9utLH3QOk> &feature=share &fb\_ref=share

January 21, 2015

Douglas Crockford: Advanced JavaScript

[17]<https://www.youtube.com/watch?v=DwYPG6vreJg>

January 15, 2015

Share the video I watched tonight. Good video

The Definitive Guide to Object-Oriented JavaScript

[18]<http://www.youtube.com/watch?v=PMfcsYzj-9M> &feature=share &fb\_ref=share

January 1, 2015

Enjoy new year day off in Vancouver. Relax, read java script definitive guide, read again and again 20 pages from 163-180, and then, learn something from the video. My new year resolution is to become a stronger java script programmer in 2015.

What the heck is the event loop anyway? JSConf EU 2014

[19][http://www.youtube.com/attribution\\_link?a=Cqtja3KchLO](http://www.youtube.com/attribution_link?a=Cqtja3KchLO) &u= %2Fwatch %3Fv %3D8aGhZQkoFbQ %26feature %3Dshare &fb\_ref=share

Dec. 30, 2015

Such a good video for me to watch, and then learn the scope chains and closures.

JavaScript Scope Chains and Closures

[20]<http://www.youtube.com/watch?v=zRZNb4GDOPI> &feature=share &fb\_ref=share

Dec. 27, 2015

Share the video. The CRAP rules - contrast, repeat, alignment, proximity, very great idea for website design.

HTML and CSS Web Development - Footer and Design Theory (CRAP rules) - Part 13

[21][http://www.youtube.com/attribution\\_link?a=JeHOVs6tcZw](http://www.youtube.com/attribution_link?a=JeHOVs6tcZw) &u= %2Fwatch %3Fv %3D9lpeHgl \_g90 %26fea-

ture %3Dshare &fb\_ref=share

Dec. 14, 2014

o2 Creation Patterns Creational Patterns

[22][https://www.youtube.com/watch?v=LjIWRvOL7aM &list=PLrZrNeNx3kNHsaPfrpPo0AIW-MhJE6gOA &index=10](https://www.youtube.com/watch?v=LjIWRvOL7aM&list=PLrZrNeNx3kNHsaPfrpPo0AIW-MhJE6gOA&index=10)

Dec. 27, 2014

Make me smile, the short video, nice teaching.

HTML and CSS Web Development - How to apply transition effects - Part 14

[23][http://www.youtube.com/watch?v=WRADgQVSWG4 &feature=share &fb\\_ref=share](http://www.youtube.com/watch?v=WRADgQVSWG4&feature=share&fb_ref=share)

Dec. 25, 2015

My favorite half hour to watch this Java script video in Christmas day morning.

Exam Application Programming Tutorial JavaScript Quiz Online Test

[24][http://www.youtube.com/attribution\\_link?a=UiB8WtJBt5A &u= %2Fwatch %3Fv %3Dd \\_UuOVhuCF8 %26feature%3Dshare &fb\\_ref=share](http://www.youtube.com/attribution_link?a=UiB8WtJBt5A&u=%2Fwatch%3Fv%3Dd_UuOVhuCF8%26feature%3Dshare&fb_ref=share)

Dec. 24, 2015

Just watch, and then, learn something interesting. Good teaching

Image Cycle JavaScript HTML CSS Web Programming Tutorial

[25][http://www.youtube.com/watch?v=3QuSKsr47f0 &feature=share &list=PL00952AC35D0A4701 &index=3 &fb\\_ref=share](http://www.youtube.com/watch?v=3QuSKsr47f0&feature=share&list=PL00952AC35D0A4701&index=3&fb_ref=share)

Java Script Tutorial Videos (Object oriented)

[26]<https://www.youtube.com/playlist?list=PL00952AC35D0A4701>

Dec. 23, 2014

My favorite video this Christmas holiday. Java script, is a fun, challenging language to learn.

Douglas Crockford: Advanced JavaScript

[27][http://www.youtube.com/watch?v=DwYPG6vreJg &feature=share &fb\\_ref=share](http://www.youtube.com/watch?v=DwYPG6vreJg&feature=share&fb_ref=share)

Dec. 21, 2014

Spent time to read the book, this weekend. Just start to read creation pattern. Good book for a programmer to start to love the java script.

Book: JavaScript Patterns

[28]<http://www.amazon.ca/JavaScript-Patterns-Stoyan-Stefanov/dp/0596806752>

Dec. 15, 2014

Google I/O 2014 - Keynote

[29]<https://www.youtube.com/watch?v=wtLJPvx7-ys>

Dec. 13, 2014

Such a good teaching. Excellent for a morning hour.

AngularJS Fundamentals In 60-ish Minutes

[30][http://www.youtube.com/attribution\\_link?a=-U2QuDFQH48&u=%2Fwatch%3Fv%3Di9MHigUZKEM%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=-U2QuDFQH48&u=%2Fwatch%3Fv%3Di9MHigUZKEM%26feature%3Dshare&fb_ref=share)

[31][http://www.youtube.com/watch?v=DwYPG6vreJg&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=DwYPG6vreJg&feature=share&fb_ref=share)

Dec. 13, 2015

Like the presentation. Programming can be easy with great training. Still learn java script, having fun.

Structuring JavaScript with the Revealing Module Pattern - Video

[32][http://www.youtube.com/watch?v=V\\_X86VyrEjc&sns=fb](http://www.youtube.com/watch?v=V_X86VyrEjc&sns=fb)

My favorite learning time is insomnia, reading does help. Learning helps more.

Using the JQuery each() Function - video

[33]<http://www.youtube.com/watch?v=Fekw8FwJcOk&sns=fb>

Nov. 29, 2014

Being a big fan of the book last 2 months, motivated to rewrite the code make more readable. Coaching is such great experience, easy to read, short book, and training myself with a coach will make life much easy.

The Art of Readable Code (Theory in Practice) - Book

[34]<http://www.amazon.com/The-Readable-Code-Theory-Practice/dp/0596802293>

November 29, 2014

Great video to watch, still on the way to learn JQuery, Java script.

Fundamentals for Great jQuery Development

[35]<http://www.youtube.com/watch?v=YcyISiDoOio> &feature=share &fb\_ref=share

Nov. 26, 2014

Share the video I watched tonight.

Douglas Crockford: The Better Parts - JSConfUY 2014

[36][http://www.youtube.com/attribution\\_link?a=v2YJsPYGvZM](http://www.youtube.com/attribution_link?a=v2YJsPYGvZM) &u= %2Fwatch %3Fv %3Dbo36MrBfTk4 %26feature %3Dshare &fb\_ref=share

November 17, 2014

Learning java is so much fun. Like the video.

Google I/O 2009 - Big Modular Java with Guide

[37][http://www.youtube.com/attribution\\_link?a=GCVjQsKVR5E](http://www.youtube.com/attribution_link?a=GCVjQsKVR5E) &u= %2Fwatch %3Fv %3DhBVJbzAagfs %26feature %3Dshare &fb\_ref=share

Nov. 12, 2014

So many questions and answers. Like those answers.

Google I/O 2011: Programming Well with Others: Social Skills for Geeks

[38][http://www.youtube.com/attribution\\_link?a=NHVbdKud8r0](http://www.youtube.com/attribution_link?a=NHVbdKud8r0) &u= %2Fwatch %3Fv %3Dq-7l8cnpl4k %26feature %3Dshare &fb\_ref=share

Nov. 11, 2014

So many good ideas. Never thought about those ideas before. Be a manager seems to be like a teacher.

Google I/O 2010 - The joys of engineering leadership

[39][http://www.youtube.com/attribution\\_link?a=7r4t5Axk4pc](http://www.youtube.com/attribution_link?a=7r4t5Axk4pc) &u= %2Fwatch %3Fv %3DskD1fjxSRog %26feature %3Dshare &fb\_ref=share

Watch the video. Two experts share principles, good points: do not burn the bridge, face time, visible/invisible work etc.

Google I/O 2012 - The Art of Organizational Manipulation

[40][http://www.youtube.com/attribution\\_link?a=-05Nv51UgJU](http://www.youtube.com/attribution_link?a=-05Nv51UgJU) &u= %2Fwatch %3Fv %3DOTCuYzAw31Y %26feature %3Dshare &fb\_ref=share

Nov. 11,2014

Videos I like

Fluent 2013: Nicholas Zakas, "A " Thank You" Can Change Your life"

[41][http://www.youtube.com/attribution\\_link?a=P6SF3zGOp4U&u=%2Fwatch%3Fv%3D78eexsReL9M%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=P6SF3zGOp4U&u=%2Fwatch%3Fv%3D78eexsReL9M%26feature%3Dshare&fb_ref=share)

Nov. 9, 2014

Really a struggle to read a few pages Java script definitive guide; and then, find the companion of reading, piano music, players. Learning a programming language is much easy to compare to piano.

Music video

[42][http://www.youtube.com/attribution\\_link?a=pxl27DzEb5I&u=%2Fwatch%3Fv%3DAkCpX2Pnj0w%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=pxl27DzEb5I&u=%2Fwatch%3Fv%3DAkCpX2Pnj0w%26feature%3Dshare&fb_ref=share)

Nov. 2, 2014

Reading "definitive Java Script guide" is still slow, 1 hour a time, still on the page 38 out of 1098 pages. Learning a programming language needs me to be more patient.

Oct. 19, 2014

Good idea to learn a few things to write a language. It took 10 days for some one to design Java script.

JavaScript: Choose Your Own Adventure!

[43][http://www.youtube.com/attribution\\_link?a=uklLhqKvXYA&u=%2Fwatch%3Fv%3D\\_\\_8eIX0QFXU%26feature%3Dshare%26list%3DPLndbWGuLoHeZVgHEBjGGtCqQaXpZJfv51%26index%3D4&fb\\_ref=share](http://www.youtube.com/attribution_link?a=uklLhqKvXYA&u=%2Fwatch%3Fv%3D__8eIX0QFXU%26feature%3Dshare%26list%3DPLndbWGuLoHeZVgHEBjGGtCqQaXpZJfv51%26index%3D4&fb_ref=share)

Learn java script programming language, warm up before I read through java script definitive guide book.

Context in JavaScript - 1/4 - Purpose and Problems with JavaScript's "This"

[44][http://www.youtube.com/attribution\\_link?a=387pTjKZkJg&u=%2Fwatch%3Fv%3Dsu-SdgebJCE%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=387pTjKZkJg&u=%2Fwatch%3Fv%3Dsu-SdgebJCE%26feature%3Dshare&fb_ref=share)

Oct. 16, 2014

Good video to spend 1 hour.

JavaScript Essentials - video

[45][http://www.youtube.com/attribution\\_link?a=9hzmX-0p57Q&u=%2Fwatch%3Fv%3D03EQu\\_2K2cs%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=9hzmX-0p57Q&u=%2Fwatch%3Fv%3D03EQu_2K2cs%26feature%3Dshare&fb_ref=share)

Oct. 13, 2014

Good topic. Testable design, is a good start to think about testing.

GTAC 2010: Flexible Design? Testable Design? You Don't Have To Choose!

[46][http://www.youtube.com/watch?v=K3q\\_y8H1ZTo](http://www.youtube.com/watch?v=K3q_y8H1ZTo) &feature=share &fb\_ref=share

Oct. 13, 2014

Watch the video, website administration work is fun and challenge sometimes.

[47][https://www.youtube.com/watch?v=5\\_-YukDEDBE](https://www.youtube.com/watch?v=5_-YukDEDBE) &list=UUtXKDgv1AVoG88PLI8nGXmw &index=427

Thanksgiving day is so much fun, and with a lot of talks and laughing and singing around the house.

Game On: 16 Design Patterns for User Engagement - Google Tech Talk

[48]<http://www.youtube.com/watch?v=YZCX-izCYMk> &list=UUtXKDgv1AVoG88PLI8nGXmw &feature=share &index=21 &fb\_ref=share

Oct. 13, 2014

My Canadian thanksgiving morning time video - great time to relax and enjoy some talks

NYC Tech Talk Seris: Javascript Testing at Google Scale

[49]<http://www.youtube.com/watch?v=draY63DasmQ>

Oct. 12, 2014

watch the video. Testability is a good topic.

Design Tech Talk Series Presents: OO Design for Testability - Google Tech Talk

[50]<http://www.youtube.com/watch?v=acjvKJiOvXw> &feature=share &fb\_ref=share

Oct. 11, 2014

Like this talk. Angular JS, 1 hour learning. Good time to spend at home to learn something other people working on.

Google I/O 2013 - Design Decisions in AngularJS

[51]<http://www.youtube.com/watch?v=HCR7i5F5L8c> &feature=share &fb\_ref=share

Like this talk. It is good for Saturday morning, organizing house while watching the video.

Google I/O 2009 - The Myth of the Genius Programmer

[52][http://www.youtube.com/watch?v=OSARbwvvhupQ&list=PLC0C39A4F2D9580F4&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=OSARbwvvhupQ&list=PLC0C39A4F2D9580F4&feature=share&fb_ref=share)

Oct. 10, 2015

Learning to love Javascript. Who else is learning?

Google I/O 2011: Learning to Love JavaScript

[53][http://www.youtube.com/attribution\\_link?a=ml4cQMSJT4g&u=%2Fwatch%3Fv%3DseX7jYI96GE%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=ml4cQMSJT4g&u=%2Fwatch%3Fv%3DseX7jYI96GE%26feature%3Dshare&fb_ref=share)

Start to watch more videos at home, learn from experts. Java script, will be my favorite language.

Speed Up Your JavaScript

[54][http://www.youtube.com/attribution\\_link?a=EqQZpKBee6c&u=%2Fwatch%3Fv%3DmHtdZgou0qU%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=EqQZpKBee6c&u=%2Fwatch%3Fv%3DmHtdZgou0qU%26feature%3Dshare&fb_ref=share)

Oct. 7, 2014

Cannot believe that today I read this book first time. What a great book.

Functional Programming in Java

[55]<http://shop.oreilly.com/product/9781937785468.do>

Oct. 5, 2014

Spent Sunday morning to watch the video. Learn Java Script language knowledge from the expert. Learning is so much fun, depth of the knowledge is out of my imagination.

Crockford on JavaScript - Chapter 2: And Then There Was JavaScript

[56][http://www.youtube.com/attribution\\_link?a=AmQCwXmbO60&u=%2Fwatch%3Fv%3DRO1Wnu-xKoY%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=AmQCwXmbO60&u=%2Fwatch%3Fv%3DRO1Wnu-xKoY%26feature%3Dshare&fb_ref=share)

Oct. 2, 2014

Enjoy the talks in the evening, Model this  $1+2*3$ , good question.

"The Clean Code Talks – Inheritance, Polymorphism, & Testing"

[57]<https://www.youtube.com/watch?v=4F72VULWFvc>

Sept. 28, 2014

Learning Java Script is so much fun. Got lost in the video first, and then, read the book to try to understand, and then, play with the sample code.

Douglas Crockford: Advanced JavaScript



[58][http://www.youtube.com/attribution\\_link?a=m9mNGZOAGLM &u= %2Fwatch %3Fv %3DDwYPG6vreJg %26feature %3Dshare &fb\\_ref=share](http://www.youtube.com/attribution_link?a=m9mNGZOAGLM&u=%2Fwatch%3Fv%3DDwYPG6vreJg%26feature%3Dshare&fb_ref=share)

Sept. 26, 2014

So great to have a video to watch, and then, try to make sense from my own experience.

How to Write Clean, Testable Code

[59][http://www.youtube.com/watch?v=XcT4yYu\\_TTs &feature=share &fb\\_ref=share](http://www.youtube.com/watch?v=XcT4yYu_TTs&feature=share&fb_ref=share)

Sept. 25, 2014

?????, ??????. ??????, ?????, ?????!

Douglas Crockford: An Inconvenient API - The Theory of the DOM

[60][http://www.youtube.com/watch?v=Y2Y0U-2qJMs &feature=share &fb\\_ref=share](http://www.youtube.com/watch?v=Y2Y0U-2qJMs&feature=share&fb_ref=share)

Like the book "testable java script"; and then, have chance to know the expert, and then, watch videos to learn Java script.

Douglas Crockford: The JavaScript Programming Language

[61][http://www.youtube.com/attribution\\_link?a=6fbrVqm1kJ0 &u= %2Fwatch %3Fv %3Dv2ifWcnQs6M %26feature %3Dshare &fb\\_ref=share](http://www.youtube.com/attribution_link?a=6fbrVqm1kJ0&u=%2Fwatch%3Fv%3Dv2ifWcnQs6M%26feature%3Dshare&fb_ref=share)

Sept. 21, 2014

So great to spend 3-4 hours in the Sunday morning to go through a book, for 30-40 pages, comparing to read all weChat blogs. Be able to read through the book and read other people's source code, really feels good.

Book: High Performance JavaScript

1. [https://www.youtube.com/attribution\\_link?a=s1oxvH9-h4E&u=%2Fwatch%3Fv%3D3tNpYpcU5s4%26feature%3Dshare](https://www.youtube.com/attribution_link?a=s1oxvH9-h4E&u=%2Fwatch%3Fv%3D3tNpYpcU5s4%26feature%3Dshare)
2. [https://sec.ch9.ms/ch9/03b0/3ac5c264-043f-4ffd-b2ee-422f39ba03b0/IntroTojQueryM06\\_mid.mp4](https://sec.ch9.ms/ch9/03b0/3ac5c264-043f-4ffd-b2ee-422f39ba03b0/IntroTojQueryM06_mid.mp4)
3. [http://channel9.msdn.com/Series/Adding-Style-with-CSS/06?wt.mc\\_id=EntriesInArea](http://channel9.msdn.com/Series/Adding-Style-with-CSS/06?wt.mc_id=EntriesInArea)
4. <http://channel9.msdn.com/Series/Adding-Style-with-CSS/05>
5. [http://channel9.msdn.com/Series/Adding-Style-with-CSS/05?fb\\_action\\_ids=823960460986167&fb\\_action\\_types=og.likes&fb\\_source=other\\_multiline&action\\_object\\_map=%5B771052266276726%5D&action\\_type\\_map=%5B%22og.likes%22%5D&action\\_ref\\_map=%5B%5D](http://channel9.msdn.com/Series/Adding-Style-with-CSS/05?fb_action_ids=823960460986167&fb_action_types=og.likes&fb_source=other_multiline&action_object_map=%5B771052266276726%5D&action_type_map=%5B%22og.likes%22%5D&action_ref_map=%5B%5D)
6. [http://channel9.msdn.com/Series/Introduction-to-jQuery/02?wt.mc\\_id=player](http://channel9.msdn.com/Series/Introduction-to-jQuery/02?wt.mc_id=player)
7. [http://channel9.msdn.com/Series/Single-Page-Applications-with-jQuery-or-AngularJS/02?wt.mc\\_id=EntriesInArea](http://channel9.msdn.com/Series/Single-Page-Applications-with-jQuery-or-AngularJS/02?wt.mc_id=EntriesInArea)
8. [http://channel9.msdn.com/Series/vstips/lazycodesnippets?wt.mc\\_id=player](http://channel9.msdn.com/Series/vstips/lazycodesnippets?wt.mc_id=player)
9. [http://channel9.msdn.com/Shows/Defrag-Tools/Defrag-Tools-37-JavaScript-Part-1?wt.mc\\_id=EntriesInArea](http://channel9.msdn.com/Shows/Defrag-Tools/Defrag-Tools-37-JavaScript-Part-1?wt.mc_id=EntriesInArea)
10. [http://www.youtube.com/watch?v=tGr6xm9nUm4&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=tGr6xm9nUm4&feature=share&fb_ref=share)

11. <http://youtu.be/ypdnMpR5jZ8>

12. [http://www.youtube.com/watch?v=yRMsxyGsez4&list=PLfXiENmg6yyUpIVY9XV0kdbmBPx6PUm9\\_&feature=share&index=3&fb\\_ref=share](http://www.youtube.com/watch?v=yRMsxyGsez4&list=PLfXiENmg6yyUpIVY9XV0kdbmBPx6PUm9_&feature=share&index=3&fb_ref=share)

13. <http://goo.gl/QBWjvA>

14. [http://www.youtube.com/attribution\\_link?a=LtVczHAKqbM&u=%2Fwatch%3Fv%3DXRYN2xt11Ek%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=LtVczHAKqbM&u=%2Fwatch%3Fv%3DXRYN2xt11Ek%26feature%3Dshare&fb_ref=share)

15. [http://www.youtube.com/watch?v=DqMFX91ToLw&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=DqMFX91ToLw&feature=share&fb_ref=share)

16. [http://www.youtube.com/watch?v=cD9utLH3Q0k&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=cD9utLH3Q0k&feature=share&fb_ref=share)

17. <https://www.youtube.com/watch?v=DwYPG6vreJg>

18. [http://www.youtube.com/watch?v=PMfcsYzj-9M&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=PMfcsYzj-9M&feature=share&fb_ref=share)

19. [http://www.youtube.com/attribution\\_link?a=Cqtja3KchL0&u=%2Fwatch%3Fv%3D8aGhZQkoFbQ%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=Cqtja3KchL0&u=%2Fwatch%3Fv%3D8aGhZQkoFbQ%26feature%3Dshare&fb_ref=share)

20. [http://www.youtube.com/watch?v=zRZNb4GDOPI&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=zRZNb4GDOPI&feature=share&fb_ref=share)

21. [http://www.youtube.com/attribution\\_link?a=JeH0Vs6tcZw&u=%2Fwatch%3Fv%3D9IpeHgl\\_g90%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=JeH0Vs6tcZw&u=%2Fwatch%3Fv%3D9IpeHgl_g90%26feature%3Dshare&fb_ref=share)

22. <https://www.youtube.com/watch?v=LjIWRvOL7aM&list=PLrZrNeNx3kNHsaPfrpPo0A1W-MhJE6gOA&index=10>

23. [http://www.youtube.com/watch?v=WRADgQVSWG4&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=WRADgQVSWG4&feature=share&fb_ref=share)

24. [http://www.youtube.com/attribution\\_link?a=UiB8WtJBt5A&u=%2Fwatch%3Fv%3Dd\\_UuOVhuCF8%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=UiB8WtJBt5A&u=%2Fwatch%3Fv%3Dd_UuOVhuCF8%26feature%3Dshare&fb_ref=share)

25. [http://www.youtube.com/watch?v=3QuSKsr47f0&feature=share&list=PL00952AC35D0A4701&index=3&fb\\_ref=share](http://www.youtube.com/watch?v=3QuSKsr47f0&feature=share&list=PL00952AC35D0A4701&index=3&fb_ref=share)

26. <https://www.youtube.com/playlist?list=PL00952AC35D0A4701>

27. [http://www.youtube.com/watch?v=DwYPG6vreJg&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=DwYPG6vreJg&feature=share&fb_ref=share)

28. <http://www.amazon.ca/JavaScript-Patterns-Stoyan-Stefanov/dp/0596806752>

29. <https://www.youtube.com/watch?v=wtLJPvx7-ys>

30. [http://www.youtube.com/attribution\\_link?a=-U2QuDFQH48&u=%2Fwatch%3Fv%3Di9MHigUZKEM%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=-U2QuDFQH48&u=%2Fwatch%3Fv%3Di9MHigUZKEM%26feature%3Dshare&fb_ref=share)

31. [http://www.youtube.com/watch?v=DwYPG6vreJg&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=DwYPG6vreJg&feature=share&fb_ref=share)

32. [http://www.youtube.com/watch?v=\\_X86VyrEjc&sns=fb](http://www.youtube.com/watch?v=_X86VyrEjc&sns=fb)

33. <http://www.youtube.com/watch?v=Fekw8FwJcOk&sns=fb>

34. <http://www.amazon.com/The-Readable-Code-Theory-Practice/dp/0596802293>

35. [http://www.youtube.com/watch?v=YcylSiDoOio&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=YcylSiDoOio&feature=share&fb_ref=share)

36. [http://www.youtube.com/attribution\\_link?a=v2YJsPYGvZM&u=%2Fwatch%3Fv%3Dbo36MrBfTk4%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=v2YJsPYGvZM&u=%2Fwatch%3Fv%3Dbo36MrBfTk4%26feature%3Dshare&fb_ref=share)

37. [http://www.youtube.com/attribution\\_link?a=GCVjQsKVR5E&u=%2Fwatch%3Fv%3DhBVJbzAagfs%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=GCVjQsKVR5E&u=%2Fwatch%3Fv%3DhBVJbzAagfs%26feature%3Dshare&fb_ref=share)

38. [http://www.youtube.com/attribution\\_link?a=NHVbdKud8r0&u=%2Fwatch%3Fv%3Dq-718cnpI4k%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=NHVbdKud8r0&u=%2Fwatch%3Fv%3Dq-718cnpI4k%26feature%3Dshare&fb_ref=share)

39. [http://www.youtube.com/attribution\\_link?a=7r4t5Axk4pc&u=%2Fwatch%3Fv%3DskD1fjxSRog%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=7r4t5Axk4pc&u=%2Fwatch%3Fv%3DskD1fjxSRog%26feature%3Dshare&fb_ref=share)

40. [http://www.youtube.com/attribution\\_link?a=-05Nv51UgJU&u=%2Fwatch%3Fv%3DOTCuYzAw31Y%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=-05Nv51UgJU&u=%2Fwatch%3Fv%3DOTCuYzAw31Y%26feature%3Dshare&fb_ref=share)

41. [http://www.youtube.com/attribution\\_link?a=P6SF3zGOp4U&u=%2Fwatch%3Fv%3D78eexsReL9M%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=P6SF3zGOp4U&u=%2Fwatch%3Fv%3D78eexsReL9M%26feature%3Dshare&fb_ref=share)

42. [http://www.youtube.com/attribution\\_link?a=pxI27DzEb5I&u=%2Fwatch%3Fv%3DAkCpX2Pnj0w%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=pxI27DzEb5I&u=%2Fwatch%3Fv%3DAkCpX2Pnj0w%26feature%3Dshare&fb_ref=share)

43. [http://www.youtube.com/attribution\\_link?a=uklLhqKvXYA&u=%2Fwatch%3Fv%3D\\_\\_8eIX0QFXU%26feature%3Dshare%26list%3DPLndbWGuLoHeZVgHEBJGGtCqQaXpZJfv51%26index%3D4&fb\\_ref=share](http://www.youtube.com/attribution_link?a=uklLhqKvXYA&u=%2Fwatch%3Fv%3D__8eIX0QFXU%26feature%3Dshare%26list%3DPLndbWGuLoHeZVgHEBJGGtCqQaXpZJfv51%26index%3D4&fb_ref=share)

44. [http://www.youtube.com/attribution\\_link?a=387pTjKZkJg&u=%2Fwatch%3Fv%3Dsu-SdgebJCE%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=387pTjKZkJg&u=%2Fwatch%3Fv%3Dsu-SdgebJCE%26feature%3Dshare&fb_ref=share)

45. [http://www.youtube.com/attribution\\_link?a=9hzmX-Op57Q&u=%2Fwatch%3Fv%3D03EQu\\_2K2cs%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=9hzmX-Op57Q&u=%2Fwatch%3Fv%3D03EQu_2K2cs%26feature%3Dshare&fb_ref=share)

ef=share

46. [http://www.youtube.com/watch?v=K3q\\_y8H1ZTo&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=K3q_y8H1ZTo&feature=share&fb_ref=share)

47. [https://www.youtube.com/watch?v=5\\_-YukDEDBE&list=UUtXKDgv1AVoG88PL18nGXmw&index=427](https://www.youtube.com/watch?v=5_-YukDEDBE&list=UUtXKDgv1AVoG88PL18nGXmw&index=427)

48. [http://www.youtube.com/watch?v=YZCX-izCYMk&list=UUtXKDgv1AVoG88PL18nGXmw&feature=share&index=21&fb\\_ref=share](http://www.youtube.com/watch?v=YZCX-izCYMk&list=UUtXKDgv1AVoG88PL18nGXmw&feature=share&index=21&fb_ref=share)

49. <http://www.youtube.com/watch?v=draY63DasmQ>

50. [http://www.youtube.com/watch?v=acjvKJiOvXw&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=acjvKJiOvXw&feature=share&fb_ref=share)

51. [http://www.youtube.com/watch?v=HCR7i5F5L8c&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=HCR7i5F5L8c&feature=share&fb_ref=share)

52. [http://www.youtube.com/watch?v=0SARbwvhupQ&list=PLC0C39A4F2D9580F4&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=0SARbwvhupQ&list=PLC0C39A4F2D9580F4&feature=share&fb_ref=share)

53. [http://www.youtube.com/attribution\\_link?a=mI4cQMSJT4g&u=%2Fwatch%3Fv%3DseX7jYI96GE%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=mI4cQMSJT4g&u=%2Fwatch%3Fv%3DseX7jYI96GE%26feature%3Dshare&fb_ref=share)

54. [http://www.youtube.com/attribution\\_link?a=EqQZpKBee6c&u=%2Fwatch%3Fv%3DmHtdZgou0qU%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=EqQZpKBee6c&u=%2Fwatch%3Fv%3DmHtdZgou0qU%26feature%3Dshare&fb_ref=share)

55. <http://shop.oreilly.com/product/9781937785468.do>

56. [http://www.youtube.com/attribution\\_link?a=AmQCwXmb060&u=%2Fwatch%3Fv%3DR01Wnu-xKoY%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=AmQCwXmb060&u=%2Fwatch%3Fv%3DR01Wnu-xKoY%26feature%3Dshare&fb_ref=share)

57. <https://www.youtube.com/watch?v=4F72VULWFvc>

58. [http://www.youtube.com/attribution\\_link?a=m9mNGZ0AGLM&u=%2Fwatch%3Fv%3DDwYPG6vreJg%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=m9mNGZ0AGLM&u=%2Fwatch%3Fv%3DDwYPG6vreJg%26feature%3Dshare&fb_ref=share)

59. [http://www.youtube.com/watch?v=XcT4yYu\\_TTs&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=XcT4yYu_TTs&feature=share&fb_ref=share)

60. [http://www.youtube.com/watch?v=Y2Y0U-2qJMs&feature=share&fb\\_ref=share](http://www.youtube.com/watch?v=Y2Y0U-2qJMs&feature=share&fb_ref=share)

61. [http://www.youtube.com/attribution\\_link?a=6fbrVqm1kJ0&u=%2Fwatch%3Fv%3Dv2ifWcnQs6M%26feature%3Dshare&fb\\_ref=share](http://www.youtube.com/attribution_link?a=6fbrVqm1kJ0&u=%2Fwatch%3Fv%3Dv2ifWcnQs6M%26feature%3Dshare&fb_ref=share)

## Leetcode 109: convert sorted list to a binary search tree (2015-09-22 21:33)

Sept. 22, 2015

109 Convert sorted list to binary search tree (No. 109)

8/25/2015

Read the following blogs:

[1]<http://articles.leetcode.com/2010/11/convert-sorted-list-to-balanced-binary.html>

C #, bottom up, time  $O(n)$ , space  $O(\log n)$  solution - best solution:

[2][https://github.com/jianminchen/Leetcode\\_C-/blob/master/109ConvertSortedListToBinarySearchTreeB.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/109ConvertSortedListToBinarySearchTreeB.cs)

C #, top down, time  $O(n^2)$ , space  $O(\log n)$  solution - naive solution:

worked on code 2 times, first time, the calculation is kind of messy, then, worked on Leetcode question 108, get the idea to make it more simple; tips like  $\text{len}/2$  only shows once, afterwards, use  $m$  instead. Just need to improve coding, think to make it more abstract, simple.

[3][https://github.com/jianminchen/Leetcode\\_C-/blob/master/109ConvertSortedListToBinarySearchTreeC.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/109ConvertSortedListToBinarySearchTreeC.cs)

9/21/2015

Review the best solution, and then, totally forgot the bottom up solution idea. So, update the code with more comment.

Need to review more about bottom up/ top down solution in tree problems. Get more experience on bottom-up solution, read some articles about it.

9/22/2015

Go over one example to build some muscle memory about this bottom up,  $O(1)$  solution to find the root node in subtree function.

Sorted List:

1->2->3->4->5->6->7,

How to convert the above sorted list to a binary search tree?

Thought process:

1.

First, get length of the list, which is 7 in the above list;

2.

Secondly, define a recursive function called

constructBST(ref TreeNode head, int start, int end)

the above function definition, 3 arguments:

1.

First one is the reference of head node of sorted list,

2.

Start index of the list,

3. End index of the list

In the function, first define the base case:

head is null, or start < end, return null

start == end, return head

then, call the recursive function for left subtree:

head node is the same, start is the same, but end is len/2-1;

great tip comes in, the head node should move in the function, so that

root node can be accessed in the list using  $O(1)$ , instead of starting from very beginning.

One more statement:

head = head.next;

TreeNode root = head; // return this node as tree root node

Root.left = left subtree root node

Root.right = right subtree recursive function

constructBST(ref head.next, mid+1, end)

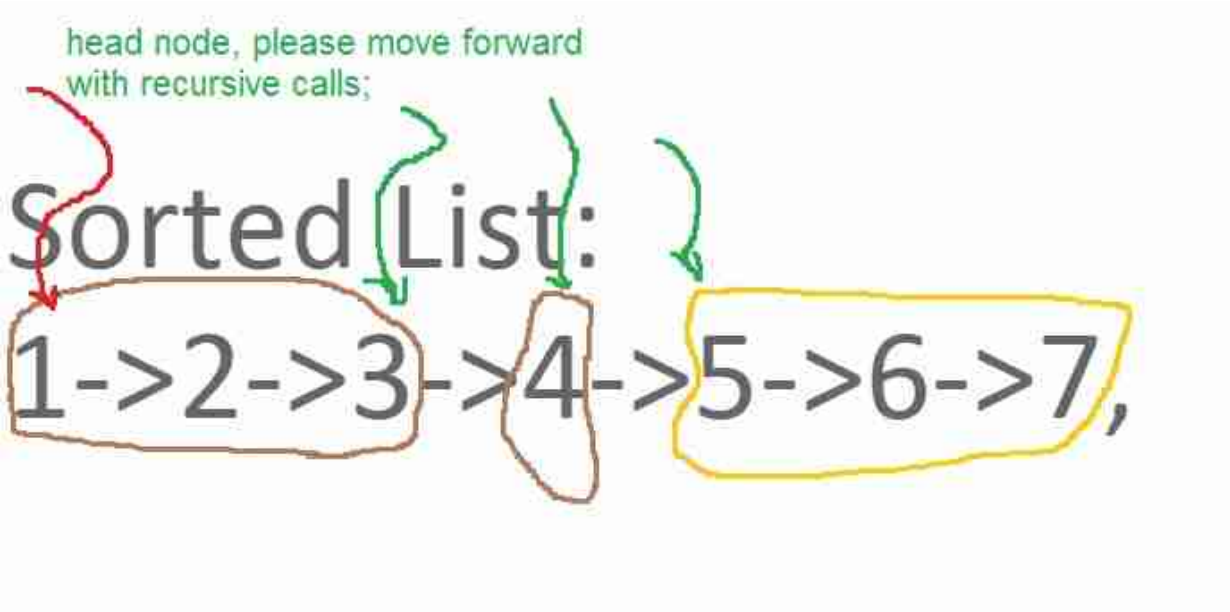
The tips to remember in the design, the recursive function should return the root node of the tree; secondly, input argument of linked list should use reference, and also head node moves in the recursive function, so it is  $O(1)$  to find the root node.

Just cannot believe that only call .next function once in the recursive function! How to argue that the move is only once? Therefore, the total calls of .next should be length of list. Total recursive function calls is n, length of list.

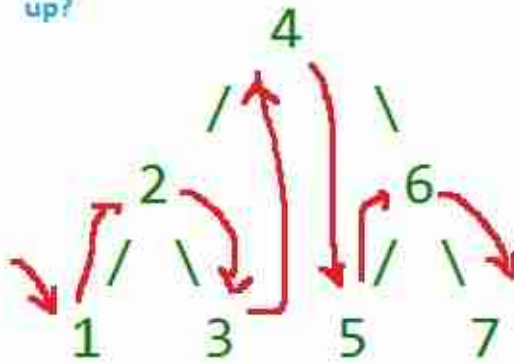
Debate why the recursive function has to return root node, and set up root node, connect its left/ right subtree root node.

Debate why the linked list head node is moving.

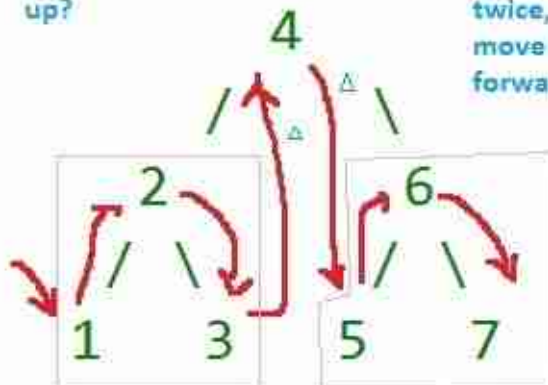
```
????????  
,  
????????????  
,  
????  
?  
?????????  
,  
?????????  
,  
?????  
,  
?????  
.  
????  
,  
?????????  
;  
?????  
,  
????  
.
```



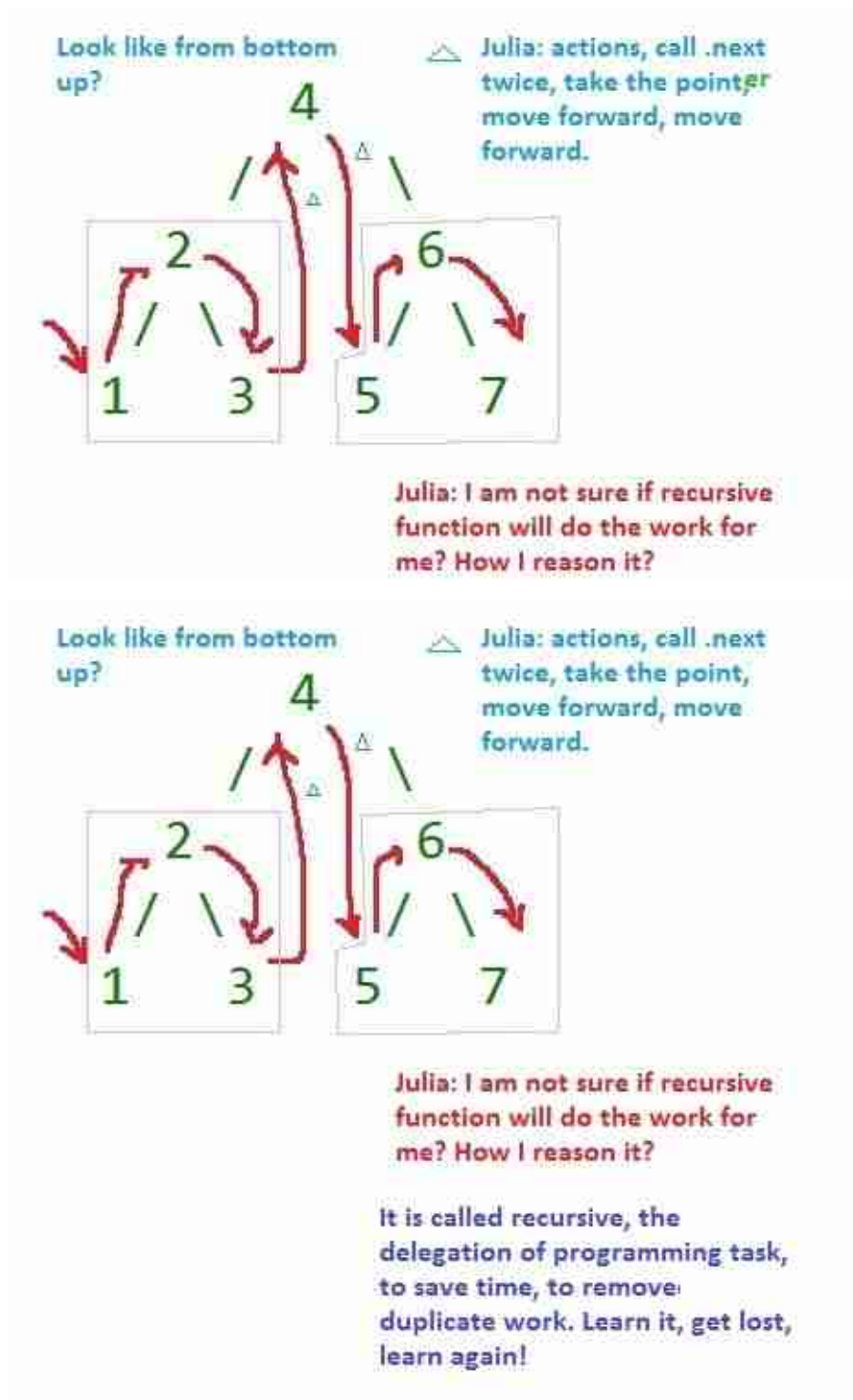
Look like from bottom  
up?



Look like from bottom  
up?



△ Julia: actions, call .next  
twice, take the pointer  
move forward, move  
forward.



The main point to understand the best solution using  $O(\ln N)$  space, the left subtree has to be built first, and then, head node can be retrieved as .next method call, root node can be set up, and then, left subtree can be built. So, left subtree, then right subtree, then the tree with root node. Bottom up.

Whereas sorted array to BST, the root node can be find right away, and then, tree can be set up top down. Julia is still confusing this top down / bottom up difference. :-) read more blogs.

Blogs:

1. use global variable to remember the head node of linked list, great idea:

[4]<http://www.jiuzhang.com/solutions/convert-sorted-list-to-binary-search-tree/>

2. another implementation using two pointers. Try it using C # later.

[5][https://siddontang.gitbooks.io/leetcode-solution/content/tree/convert \\_sorted \\_listarray \\_to \\_binary \\_search \\_tree.html](https://siddontang.gitbooks.io/leetcode-solution/content/tree/convert_sorted_listarray_to_binary_search_tree.html)

3. Java implementation, teach me how to use reference in Java or wrapper class. Good point!

[6]<http://rleetcode.blogspot.ca/2014/02/convert-sorted-list-to-binary-search.html>

4. Time and space complexity analysis - think about it - very clear analysis

[7][http://blog.unieagle.net/2012/12/14/leetcode %E9 %A2 %98 %E7 %9B %AE %EF %BC %9Aconvert-sorted-list-to-binary-search-tree/](http://blog.unieagle.net/2012/12/14/leetcode%E9%A2%98%E7%9B%AE%EF%BC%9Aconvert-sorted-list-to-binary-search-tree/)

5. Three implementation discussions

[8]<http://www.cnblogs.com/feiling/p/3267917.html>

6. Great explanation - bottom up / top down, and in order traversal/ post order traversal

[9]<https://leetcode-notes.wordpress.com/2013/11/23/convert-sorted-list-to-binary-search-tree/>

Implement the above 6 blogs using C #, and then, share the blog. After 1 month, check again and see if I can come out the bottom up solution in 5 minutes. If yes, stop; otherwise review again.

Dec. 24, 2015

Review the solution again.

How to recall the solution step by step?

1. Get list length <- travel the list once

2. Design the recursive function with a list, start and end two position; use the position of start/end to boundary case checking; also the function returns the root node of the tree.

3. Build left subtree

4. Find the middle of list as root <- how to get there?

Cannot traverse the list again to half length if you want optimized solution, since it is in recursive function,  $O(n \log n)$

Every recursive call the list pointer will traverse once

5. Connect root node to left child

6. Move list pointer to next one

7. Build right subtree

connect root node to right child as well.

Recap the importance of function design:

1. list with a pointer moving, starting from head

2. start, end position of linked list

3. in the function build a tree

4. return root node of the tree

1. <http://articles.leetcode.com/2010/11/convert-sorted-list-to-balanced-binary.html>

2. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/109ConvertSortedListToBinarySearchTreeB.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/109ConvertSortedListToBinarySearchTreeB.cs)

3. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/109ConvertSortedListToBinarySearchTreeC.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/109ConvertSortedListToBinarySearchTreeC.cs)



4. <http://www.jiuzhang.com/solutions/convert-sorted-list-to-binary-search-tree/>
5. [https://siddontang.gitbooks.io/leetcode-solution/content/tree/convert\\_sorted\\_listarray\\_to\\_binary\\_search\\_tree.html](https://siddontang.gitbooks.io/leetcode-solution/content/tree/convert_sorted_listarray_to_binary_search_tree.html)
6. <http://rleetcode.blogspot.ca/2014/02/convert-sorted-list-to-binary-search.html>
7. <http://blog.unieagle.net/2012/12/14/leetcode%E9%A2%98%E7%9B%AE%E7%BC%9Aconvert-sorted-list-to-binary-search-tree/>
8. <http://www.cnblogs.com/feiling/p/3267917.html>
9. <https://leetcode-notes.wordpress.com/2013/11/23/convert-sorted-list-to-binary-search-tree/>

## Leetcode: Convert sorted array to binary search tree (2015-09-25 20:54)

09/25/2015

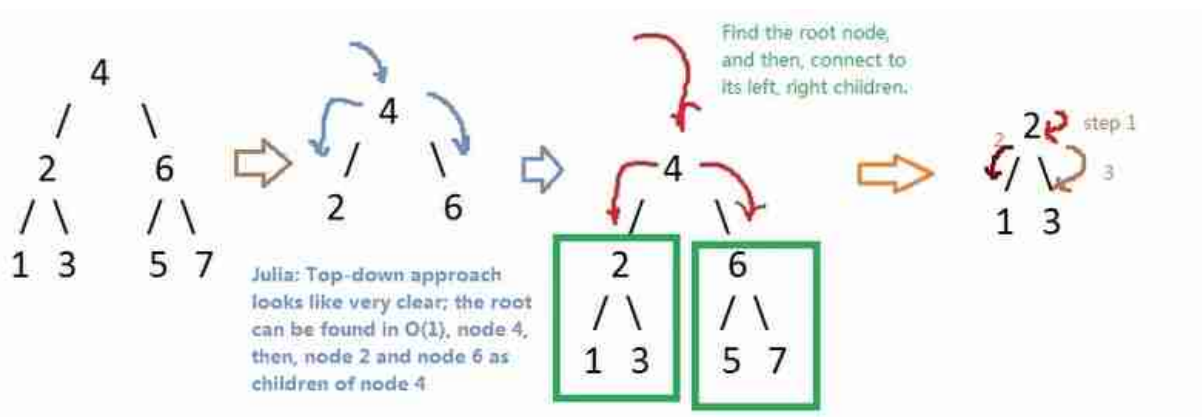
First practice on 08/25/2015

108 Convert sorted array to binary search tree (No. 108)

Julia's C # implementation: top-down approach

[1] [https://github.com/jianminchen/Leetcode\\_C-/blob/master/108ConvertSortedArrayToBinarySearchTree.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/108ConvertSortedArrayToBinarySearchTree.cs)

9/25/2015 Try to add some fun memory about top-down approach for binary search tree construction, then, present some diagrams.



my task is to make the coding practice more fun, and some funny part with naive drawing. Maybe, Julia could use the drawing to bring some comic style to serious coding challenges.

Julia's favorite blogs about this leetcode question:

1.

[2]<https://gist.github.com/Jeffwan/8982109>

Read this blog (1) and like the analysis, first time see the comment: "Conquer" above the return statement:

return node; Learn Divide and Conquer in this simple illustration.

2. [3]<http://blog.csdn.net/linhuanmars/article/details/23904883>

Because this blog's popularity, it is one of my favorites. One thing I like to discuss is about the base case, how this base case is making sense, and then minimum/ no redundancy. Let Julia argue about it.

Here is the code in the above blog:

```

1. private
   TreeNode helper(
       int
       [] num,
       int
       l,
       int
       r)
2. {
3. if
   (l>r)
4. return
   null
   ;
5. int
   m = (l+r)/
   2
   ;
6. TreeNode root = new
   TreeNode(num[m]);
7. root.left = helper(num,l,m- 1
   );
8. root.right = helper(num,m+ 1
   ,r);
9. return
   root;
10. }

```

Have some base case debates, and make this algorithm more entertainment:

why base case is:

if(l>r) return null,

if l==r, the above function will perform:

root = new TreeNode(num[l]), and also,

set up

root.left = RecursiveFunctionCall() -> go to base case, return null,

root.right is the same as root.left,

and then return root;

how come it is not:

```
if(l==r) return new TreeNode(num[l])
```

then, need to add checking before the helper call:

```
if(l<=m-1)
```

```
root.left = helper(num, l, m-1)
```

```
if(m+1<=r)
```

```
root.right = helper(num, m+1, r)
```

So, better to leave this if conditional checking to base case; at least there are 2 if checking, anyone of them implies  $l \leq r$ , so duplicate checking.

3.

[4]<http://www.acmerblog.com/leetcode-solution-convert-sorted-array-to-binary-search-tree-6247.html>

julia's comment:

the base case has redundant line:

Java code:

```
if(start == end) return
```

```
new
```

```
TreeNode(num[start]);
```

Here is the Java code:

```
public class Solution {
```

```
11 public static TreeNode sortedArrayToBST(int[] num) {
```

```
12 if(num == null || num.length == 0) return null;
```

```
13 return sortedArrayToBST(num, 0, num.length-1);
```

```
14 }
```

```
15
```

```
16 public static TreeNode sortedArrayToBST(int num[],int start, int end) {
```

```
17 if(start > end ) return null;
```

```
18 if(start == end) return new TreeNode(num[start]);
```

```
19 int mid = start + (end-start)/2;
```

```
20 TreeNode root = new TreeNode(num[mid]);
```

```
21 root.left = sortedArrayToBST(num, start, mid-1);
```

```
22 root.right = sortedArrayToBST(num, mid+1, end);
```

```
23 return root;
```

```
24 }
```

```
25 }
```

4. [5]<http://pisxw.com/algorithm/Convert-Sorted-Array-to-Binary-Search-Tree.html>

Remind me the BST, the left and right subtree difference is maximum 1.

5.

[6]<http://www.kunxi.org/notes/LeetCode/>

Look into the leetcode solution later. Should be a great one for me to catch up some programming tips.

1. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/108ConvertSortedArrayToBinarySearchTree.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/108ConvertSortedArrayToBinarySearchTree.cs)

2. <https://gist.github.com/Jeffwan/8982109>

3. <http://blog.csdn.net/linhuanmars/article/details/23904883>

4. <http://www.acmerblog.com/leetcode-solution-convert-sorted-array-to-binary-search-tree-6247.html>

5. <http://pisxw.com/algorithm/Convert-Sorted-Array-to-Binary-Search-Tree.html>

6. <http://www.kunxi.org/notes/LeetCode/>

---

## Study time - watch videos, and read articles (2015-09-25 23:04)

10/18/2015

REST+JSON API Design - Best Practices for Developers

[1]<https://www.youtube.com/watch?v=hdSrT4yjS1g>

How To Design A Good API and Why it Matters (Julia's rating: A+)

[2]<https://www.youtube.com/watch?v=aAb7hSCTvGw>

- more notes: How To Design A Good API and Why it Matters

Watched again on 10/21/2015, notes taken to share:

54:40/1:00:18

Use Appropriate Parameter and Return Types

- . Favor interface types over classes for input

- provides flexibility, performance

- . Use most specific possible input parameter type

- Moves error from runtime to compile time (Julia's comment: Great tip, reduce time on debugging, fix things in compiling time!)

- . Don't use string if a better type exists

- Strings are cumbersome, error-prone, and slow

- . Don't use floating point for monetary values

- Binary floating point causes inexact results!

- Use double (64 bits) rather than float (32bits)

- Precision loss is real, performance loss negligible

watched again on 10/21/2015, notes take to share:

49:19/1:00:18

Don't violate the Principle of Least Astonishment

Read more on this article:

[3]<http://programmers.stackexchange.com/questions/187457/what-is-the-principle-of-least-astonishment>

- end of notes for

Josh Bloch, Lord of the APIs - A Brief, Opinionated History of the API

[4]<https://www.youtube.com/watch?v=ege-kub1qtk>

[5]<https://www.youtube.com/watch?v=4YxnxmQS41s>

9/27/2015

SQL injection Myths & Fallacies: Best practices of defense

[6]<https://www.youtube.com/watch?v=o4dJ7hdA8fs>

Quickly go over the book "SQL antipatterns" in short future.

9/24/2015

Effective Java - Still Effective After All These Years

[7] <https://www.youtube.com/watch?v=V1vQf4qyMXg>

Julia's favorite time: first 15 minutes with two great coding examples; watch again later.

9/25/2015

Back to my life without writing code, enjoy the videos to catch up things about internet and technology, my favorite topic in the video: google CEO comment about competition, company vs. government, what the differences, How CEO thinks.

Eric Schmidt & Jared Cohen: The impact of Internet and Technology

[8]<https://www.youtube.com/watch?v=OwJTCHxjcjQ>

Get curious about .net framework - work on .NET framework and enjoy benefit last 5 years; should spend 10 hours to watch the .NET framework videos. Will spend more time when I take vacation from Oct. 2 - Oct. 16, 2015

[9]<https://www.youtube.com/watch?v=ywN0vTFJNcw> &list=PLnrS3jsiKkdKcChakQPJi1XdKyDt\_2jPa

9/28/2015

first time using search in github, and search Sudoku, and find some code to read:

[10][https://github.com/dartist/sudoku\\_solver/blob/master/benchmark/sudoku.cs](https://github.com/dartist/sudoku_solver/blob/master/benchmark/sudoku.cs)

[11]<http://norvig.com/sudoku.html>

<http://aspadvice.com/blogs/rbirkby/archive/2007/08/23/Silverlight-Sudoku-with-LINQ.aspx>

1. <https://www.youtube.com/watch?v=hdSrT4yjS1g>

2. <https://www.youtube.com/watch?v=aAb7hSCtvGw>

3. <http://programmers.stackexchange.com/questions/187457/what-is-the-principle-of-least-astonishment>

4. <https://www.youtube.com/watch?v=ege-kub1qtk>

5. <https://www.youtube.com/watch?v=4YxnxmQS41s>

6. <https://www.youtube.com/watch?v=o4dJ7hdA8fs>
  7. <https://www.youtube.com/user/UserGroupsatGoogle>
  8. <https://www.youtube.com/watch?v=0wJTCHxjcjQ>
  9. [https://www.youtube.com/watch?v=ywN0vTFJNcw&list=PLnrS3jsiKkdKcChakQPJi1XdKyDt\\_2jPa](https://www.youtube.com/watch?v=ywN0vTFJNcw&list=PLnrS3jsiKkdKcChakQPJi1XdKyDt_2jPa)
  10. [https://github.com/dartist/sudoku\\_solver/blob/master/benchmark/sudoku.cs](https://github.com/dartist/sudoku_solver/blob/master/benchmark/sudoku.cs)
  11. <http://norvig.com/sudoku.html>
- 

## Object oriented principles - S.O.L.I.D. (2015-09-26 00:10)

August 28, 2015

?????????????????Ž?????????????

,

????(??

,

???

!

??????-ĥ?

,

???????????

.

???

,

???

,

??????

,

?????

.

??????

,

??????????

?

??????w??

,

???????????????

.

SOLID principles:

- S – Single-responsibility principle
- O – Open-closed principle
- L – Liskov substitution principle
- I – Interface segregation principle
- D – Dependency Inversion Principle

??????????

:

[1]<https://scotch.io/bar-talk/s-o-l-i-d-the-first-five-principles-of-object-oriented-design>

Challenge myself, learn some OO design. Here are my focus from Jan. 2014 - Sept. 2015, Java Script -> Leetcode -> .NET framework, OO design. Julia learns to change her focus and become a good learning "wild animal" in IT industry.

Nov. 11, 2015

Need to work on the code seriously, start to apply those five principles: SOLID.

Here is the video studying on this remembrance day of Canada holiday,

Applying S.O.L.I.D. Principles in .NET/C #

[2]<https://www.youtube.com/watch?v=Whhi1C2PpaA>

Spent 60 minutes to watch this video:

Bob Martin SOLID Principles of Object Oriented and Agile Design

[3]<https://www.youtube.com/watch?v=TMuno5RZNeE>

video 1:12/1:23

Introduction to Dependency injection

[4][https://msdn.microsoft.com/library/hh323705\(v=vs.100\).aspx](https://msdn.microsoft.com/library/hh323705(v=vs.100).aspx)

Nov. 12, 2015

book chapter from uncle bob:

[5][http://objectmentor.com/resources/articles/Principles\\_and\\_Patterns.pdf](http://objectmentor.com/resources/articles/Principles_and_Patterns.pdf)

Lecture notes to read

[6]<http://www.it.uu.se/edu/course/homepage/ood/ht12/overview/principles/Lecture4.pdf>

Videos ( Julia's favorite)

[7][https://www.youtube.com/watch?v=5lqB0AwRMxg&index=13&list=PL8m4NUhTQU48oiGCSgCP1FiJEcg\\_xJzyQ](https://www.youtube.com/watch?v=5lqB0AwRMxg&index=13&list=PL8m4NUhTQU48oiGCSgCP1FiJEcg_xJzyQ)

[8]<http://code.tutsplus.com/series/the-solid-principles-cms-634>

[9]<http://butunclebob.com/ArticleS.UncleBob.PrinciplesOfOod>

1. <https://scotch.io/bar-talk/s-o-l-i-d-the-first-five-principles-of-object-oriented-design>

2. <https://www.youtube.com/watch?v=gwIS9cZlrhk>

3. <https://www.youtube.com/watch?v=TMuno5RZNeE>

4. [https://msdn.microsoft.com/library/hh323705\(v=vs.100\).aspx](https://msdn.microsoft.com/library/hh323705(v=vs.100).aspx)

5. [http://objectmentor.com/resources/articles/Principles\\_and\\_Patterns.pdf](http://objectmentor.com/resources/articles/Principles_and_Patterns.pdf)

6. <http://www.it.uu.se/edu/course/homepage/ood/ht12/overview/principles/Lecture4.pdf>
  7. [https://www.youtube.com/watch?v=5lqB0AwRMxg&index=13&list=PL8m4NUhTQU48oiGCSgCP1FiJEcg\\_xJzyQ](https://www.youtube.com/watch?v=5lqB0AwRMxg&index=13&list=PL8m4NUhTQU48oiGCSgCP1FiJEcg_xJzyQ)
  8. <http://code.tutsplus.com/series/the-solid-principles--cms-634>
  9. <http://butunclebob.com/ArticleS.UncleBob.PrinciplesOfOod>
- 

## ASP.NET framework (2015-09-27 22:03)

August 28, 2015

????

,

??????

ASP

?

ASP.NET

??

, .NET framework

????????

?

???-2K??

, ??????. ?

[1] <http://www.indiabix.com/technical/dotnet/asp-dot-net/>

Read the webpage: (Sept. 2, 2015)

[2] <https://www.microsoft.com/learning/en-ca/exam-70-486.aspx>

Watch MVC videos if I have a few of hours.

[3]<https://www.microsoftvirtualacademy.com/en-US/training-courses/developing-asp-net-mvc-4-web-applications-jump-start-8239>

1. <http://www.indiabix.com/technical/dotnet/asp-dot-net/>

2. <https://www.microsoft.com/learning/en-ca/exam-70-486.aspx>

3. <https://www.microsoftvirtualacademy.com/en-US/training-courses/developing-asp-net-mvc-4-web-applications-jump-start-8239>

---

## 1.2 October

### Good API video watching and notes taken (2015-10-22 00:04)

Oct. 22, 2015

48



Spent time to look into the note through the video, and then study more later:

How To Design A Good API and Why it Matters (Julia's rating: A+)

[1]<https://www.youtube.com/watch?v=aAb7hSctvGw>

#### 1. Watched again on 10/21/2015, notes taken to share:

54:40/1:00:18

Use Appropriate Parameter and Return Types

- . Favor interface types over classes for input
- provides flexibility, performance
- . Use most specific possible input parameter type
- Moves error from runtime to compile time (Julia's comment: Great tip, reduce time on debugging, fix things in compiling time!)
- . Don't use string if a better type exists
- Strings are cumbersome, error-prone, and slow
- . Don't use floating point for monetary values
- Binary floating point causes inexact results!
- Use double (64 bits) rather than float (32bits)
- Precision loss is real, performance loss negligible

#### 2. watched again on 10/21/2015, notes taken to share:

49:19/1:00:18

Don't violate the Principle of Least Astonishment

Read more on this article:

[2]<http://programmers.stackexchange.com/questions/187457/what-is-the-principle-of-least-astonishment>

3. watched again on 10/21/2015, notes taken to share:

42:31/1:00:18

Subclass only where it Makes Sense

. Subclassing implies substitutability (Liskov)

– Subclass only when is-a relationship exists

– Otherwise, use composition

. Public classes should not subclass other public classes for ease of implementation

Bad:

**Properties extends Hashtable**

**Stack extends Vector**

Good:

**Set extends Collection**

1. <https://www.youtube.com/watch?v=aAb7hSCtvGw>

2. <http://programmers.stackexchange.com/questions/187457/what-is-the-principle-of-least-astonishment>

---

## **The clean code talks - unit testing (2015-10-25 12:50)**

10/25/2015

Watch the google talk "The clean code talks - unit testing", write down notes and then learn better for next time.

video link:

[1]<https://www.youtube.com/watch?v=wEhu57pih5w>

**Video time:**

**02:02 / 32:07**

Hard to test code

Most People say:

Make things private

Using final keyword

Long methods

Real issues: (interview questions)

Mixing new with logic

Work in constructor

Global state

Singleton

Static Methods (procedure programming) (leaf method vs way-up method)

Deep inheritance (Divorce myself from inheritance at run time. )  
Too many conditionals ()

Procedure code - Seam - Polymorphism - Julia's comment: look into it, read more about it. 10/25/2015

**Video time:**

**14:35/32:07**

Progression of Testing:

1. Scenario / Large Tests

Test whole application by pretending to be a user  
slow / Flaky / Mostly happy paths

2. Functional / Medium Tests

External dependencies simulated  
Test class interaction

3. Unit / Small Tests

Focus on application logic  
Very Fast / No I/O / No Need for a debugger

4. Think of it more as a continuum than discrete

5. All tests levels important

Not all levels have the same probability of a bug  
Because smaller tests cover less need more of them

6. Testability more important as you get more focused

Video time:

17:05 /32:07

Different kinds of Tests

Execution time :

unit test -> Functional Tests -> Scenario

Test individual ->

Test collections of classes as subsystems ->

Test the whole system pretending to be a user

**Video time:**

**27:06/32:07**

How do you write hard to test code?

You mix object creation code with business logic

This will assure that a test case never can construct a graph of objects different from production. Hence nothing can be tested in isolation.

Unit test class -

Seam - friendly - try to understand the dependency injection

class depends on a lot of other classes, mocking, seam, learned through hacking,

class:  
object graph  
construction & lookup  
business logic

**27:36/32:07**

Take Away

Unit tests are a preferred way of testing

Unit tests require separation of instantiation of object graph from business logic under test

Questions:

1. Global variable - make test unpredictable, the order of test matters and so on and so forth.

Do not clean up after yourself, another test will fail;

Counter argument: framework than DI framework.

Monkey-patching - evil on the global state - dependency injection

C++ / Java / frameworks - dependency injection difference

And read the slideshow using the following link:

[2]<http://www.slideshare.net/avalanche123/clean-code-5609451>

[3]<http://misko.hevery.com/code-reviewers-guide/>

[4]<http://misko.hevery.com/presentations/>

[5]<http://misko.hevery.com/2008/10/21/dependency-injection-myth-reference-passing/>

1. <https://www.youtube.com/watch?v=wEhu57pih5w>
  2. <http://www.slideshare.net/avalanche123/clean-code-5609451>
  3. <http://misko.hevery.com/code-reviewers-guide/>
  4. <http://misko.hevery.com/presentations/>
  5. <http://misko.hevery.com/2008/10/21/dependency-injection-myth-reference-passing/>
- 

**Study notes - "The clean code talks - 'Global State and Singletons' " (2015-10-25 14:07)**

10/25/2015

Google talks:

The clean code talks - "Global State and Singletons"

[1]<https://www.youtube.com/watch?v=-FRm3VPhseI>

Global state:

Julia tries to understand global state in this talk:

Run same thing a multiple time, will get different test results;

5:40/54:08

A. Multiple Executions can product different results

1. Flakiness
2. Order of tests matters
3. Can not run test in parallel

B. Unbounded location of state

C. Hidden Global State in JVM

1. System.currentTimeMillis()
2. new Date()
3. Math.random()

D. Testing above code is hard

1. Therefore inject in doubles

8:49/54:08

Singleton: Good vs Bad

1. Application Global vs. JVM Global

2. Usually we have one application per JVM

A. Hence we incorrecly assume that:

Application Global state = JVM Global

3. Each test is a different configuration of a portion of our application

19:01/54:08

Deceptive API

1. API that lies about what it needs
2. Spooky action at a distance

Julia's comment, the talk is interesting and worth time to watch again later.

1. <https://www.youtube.com/watch?v=-FRm3VPhseI>

---

## reading time on Sunday - housing issue, health issue, problem solving (2015-10-25 15:51)

10/25/2015

Spent time to watch the video on this Sunday. Learn how billionaire educate their kids, what philosophy they hold, I do not have money but I definitely like the wisdom. Enjoy the perfect English and forget my concerns about money, Vancouver and all other things.

[1]<https://www.youtube.com/watch?v=aSL-ilskEFU>

Enjoy reading, and also find interesting things to read.

[2]<http://www.aol.com/article/2015/10/21/a-23-year-old-google-employee-lives-in-a-truck-in-the-companys/21251909/?ncid=txtlnkusaolp00001363>

[3]<http://frominsidethebox.com/>

10/28/2015

Melinda Gates - talk about puzzle, nonprofit, problem solving in general

Personal story to help parents to do rental business - so encouraging me to work hard as well - start from small business.

[4]<https://www.youtube.com/watch?v=i9Ifb3EGl9Q>

Melinda French Gates: What nonprofits can learn from Coca-Cola

[5]<https://www.youtube.com/watch?v=GIUS6KE67Vs>

10/31/2015

[6]<https://engineering.pinterest.com/blog/discover-pinterest-se-arch-and-discovery>

read linkedin profile about site reliability engineer work experience

[7][https://www.linkedin.com/profile/view?id=AAEAAAAxOf0BLtv8Mfz6NXAfym0v5\\_adk35-sCV4\\_&authType=name&authToken=EWTR&trk=prof-sb-browse\\_map-name](https://www.linkedin.com/profile/view?id=AAEAAAAxOf0BLtv8Mfz6NXAfym0v5_adk35-sCV4_&authType=name&authToken=EWTR&trk=prof-sb-browse_map-name)

Building Pinterest's Mobile Apps ()

[8]<http://www.infoq.com/presentations/pinterest-app>

[9]<https://engineering.pinterest.com/tech-talks>

How we've scaled dropbox

[10]<https://www.youtube.com/watch?v=PE4gwstWhmc>

Discover Pinterest: Mobile Engineering and Design

[11][#t=63m24s](https://www.youtube.com/watch?v=wzRyTmBxs7Y)

1:10/1:35

Nags - Nag placement, Home view placement, Pin Tutorial Card

Further reading Nov. 3, 2015

Responsive Web Design: Relying Too Much on Screen Size

[12]<http://www.lukew.com/ff/entry.asp?1816>

[13]<http://www.lukew.com/presos/>

Nov. 6, 2015

Discover Pinterest: Search and Discovery

[14][https://www.youtube.com/watch?v=f\\_JeAEBxpUs](https://www.youtube.com/watch?v=f_JeAEBxpUs)

Read the article

Venture Beat article.

[15]<http://venturebeat.com/2015/05/28/pinterest-improves-related-pins-with-deep-learning-plans-product-recommendations-using-object-recognition/>

Nov. 7, 2015

Google I/O 2012 - SQL vs NoSQL: Battle of the Backends

[16]<https://www.youtube.com/watch?v=rRoy6l4gKWU>

Julia spent time to learn NoSQL, and got some basic ideas using NoSQL.

12:46pm

Google I/O 2012 - Building Mobile App Engine Backends for Android, IOS and the Web

[17][https://www.youtube.com/watch?v=NU\\_wNR\\_UUn4](https://www.youtube.com/watch?v=NU_wNR_UUn4)

Google I/O 2011: HTML5 versus Android: Apps or Web for Mobile Development?

[18][https://www.youtube.com/watch?v=4f2Zky\\_YyyQ](https://www.youtube.com/watch?v=4f2Zky_YyyQ)

O'Reilly Webcast: Building Mobile HTML5 Apps in Hours, Not Days

[19]<https://www.youtube.com/watch?v=G0UC-C2Y7ME>

1. <https://www.youtube.com/watch?v=aSL-iIskEFU>
  2. <http://www.aol.com/article/2015/10/21/a-23-year-old-google-employee-lives-in-a-truck-in-the-companys/21251909/?ncid=txtlnkusaolp00001363>
  3. <http://frominsidethebox.com/>
  4. <https://www.youtube.com/watch?v=i9Ifb3EGl9Q>
  5. <https://www.youtube.com/watch?v=GLUS6KE67Vs>
  6. <https://engineering.pinterest.com/blog/discover-pinterest-search-and-discovery>
  7. [https://www.linkedin.com/profile/view?id=AAEAAAAxOf0BLtv8Mfz6NXAfym0v5adk35-sCV4&authType=name&authToken=EWTR&trk=prof-sb-browse\\_map-name](https://www.linkedin.com/profile/view?id=AAEAAAAxOf0BLtv8Mfz6NXAfym0v5adk35-sCV4&authType=name&authToken=EWTR&trk=prof-sb-browse_map-name)
  8. <http://www.infoq.com/presentations/pinterest-app>
  9. <https://engineering.pinterest.com/tech-talks>
  10. <https://www.youtube.com/watch?v=PE4gwstWhmc>
  11. <https://www.youtube.com/watch?v=wzRyTmBxs7Y#t=63m24s>
  12. <http://www.lukew.com/ff/entry.asp?1816>
  13. <http://www.lukew.com/presos/>
  14. [https://www.youtube.com/watch?v=f\\_JeAEBxpUs](https://www.youtube.com/watch?v=f_JeAEBxpUs)
  15. <http://venturebeat.com/2015/05/28/pinterest-improves-related-pins-with-deep-learning-plans-product-recommendations-using-object-recognition/>
  16. <https://www.youtube.com/watch?v=rRoy6I4gKWU>
  17. [https://www.youtube.com/watch?v=NU\\_wNR\\_UUn4](https://www.youtube.com/watch?v=NU_wNR_UUn4)
  18. [https://www.youtube.com/watch?v=4f2Zky\\_YyyQ](https://www.youtube.com/watch?v=4f2Zky_YyyQ)
  19. <https://www.youtube.com/watch?v=G0UC-C2Y7ME>
- 

**Back to leetcode code algorithms (2015-10-26 22:44)**

10/26/2015

Spent 2 weeks vacation in China from Oct. 2 - Oct. 16.

Enjoyed the tennis practice in the city of yichun, jiangxi province, and also the Rolex tennis Master, 2015 in Shanghai.





Start to read blogs about leetcode algorithms in the evening, back to training and practice. My favorite blogs to read tonight.

[1]<http://fisherlei.blogspot.ca/2015/10/leetcode-product-of-array-except-self.html>

[2]<http://fisherlei.blogspot.ca/2015/10/leetcode-different-ways-to-add.html>

[3]<http://fisherlei.blogspot.ca/2015/10/leetcode-valid-anagram-solution.html>

Read 6 more blogs from the above blog.

1. <http://fisherlei.blogspot.ca/2015/10/leetcode-product-of-array-except-self.html>
2. <http://fisherlei.blogspot.ca/2015/10/leetcode-different-ways-to-add.html>
3. <http://fisherlei.blogspot.ca/2015/10/leetcode-valid-anagram-solution.html>

---

### Leetcode question 241: Different ways to add parentheses (2015-10-28 21:40)

10/28/2015

Read the blog,

[1]<http://fisherlei.blogspot.ca/2015/10/leetcode-different-ways-to-add.html>

[2]<http://blog.csdn.net/guanzhongshan/article/details/48086695>

and write C # code, compile it, build it, pass online judge, check in Github, and then, write a new code using memorization.

1. First step, write down C # code, compile ok.

[3][https://github.com/jianminchen/Leetcode\\_C-/blob/master/241DifferentWaysToAddParentheses.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/241DifferentWaysToAddParentheses.cs)

2. Read the most popular blog about this leetcode question through Google,

[4]<http://blog.csdn.net/sbitswc/article/details/48546421>

and then, take Java code as a sample, write C # version of implementation.

[5][https://github.com/jianminchen/Leetcode\\_C-/blob/master/241DifferentWaysToAddParentheses\\_B.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/241DifferentWaysToAddParentheses_B.cs)

read the leetcode string algorithms blog (A plus):

[6]<http://blog.csdn.net/sbitswc/article/details/20429853>

The leetcode question 241 is just a medium question in difficulties.

Dec. 17, 2015

Review the solution, and then, need to write down the solution - a script to help solve the problem:

1.  $1+2*3$

-> read a substring from leftmost to convert it to number, for the example, read '1', integer 1, and then, visit '+' stop,  
-> now, let us use recursive function call to get the first part of substring before '+', and get second part of substring after '+', and then, two lists, each one of list1 will operate '+' with each one of list2, add into return list.

Not convincing, try again:

use binary tree to model this problem:

[7]<http://juliachencoding.blogspot.ca/2015/12/oo-principle-solid-open-close-principle.html>

So, get the root node - operator, and then, deal with left child, right child, and root node does the evaluate function.

1. <http://fisherlei.blogspot.ca/2015/10/leetcode-different-ways-to-add.html>

2. <http://blog.csdn.net/guanzhongshan/article/details/48086695>

3. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/241DifferentWaysToAddParentheses.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/241DifferentWaysToAddParentheses.cs)

4. <http://blog.csdn.net/sbitswc/article/details/48546421>

5. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/241DifferentWaysToAddParentheses\\_B.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/241DifferentWaysToAddParentheses_B.cs)

6. <http://blog.csdn.net/sbitswc/article/details/20429853>

7. <http://juliachencoding.blogspot.ca/2015/12/oo-principle-solid-open-close-principle.html>

---

## 1.3 November

### testing, automation and testing patterns (2015-11-08 12:51)

Nov. 8, 2015

Watch the video:

Automated Testing Patterns and Smells

[1]<https://www.youtube.com/watch?v=Pq6LHFM4JvE>

Julia is a developer in the city of Vancouver last 5 years, she enjoyed the understanding how to minimize time 70 % by following one simple principle: DRY - Do not repeat yourself.

For instance, to calculate something, the same code repeats twice; Just a simple example, Julia does not like to see the code repetition in the C # code, so she gives out a try to use base class, and then, merge to one function applying to the base class. But, all other repetitions are everywhere, so she cuts other places similar to the one, 2 to 1. So, do a math, the code needs change for cases 2 x 2 x 2, 3 place duplication, becoming 1 x 1 x 1; Done, reduce the 7/8 work to work with design, unit test, and bugs related to design. (similar ideas seen video, watched on Nov. 13, 2015 - [2][https://www.youtube.com/watch?v=bxQK7hcVyGs&list=PLD4GnSXHpKQ2\\_eKG5fKzszXs5hYcEsft7](https://www.youtube.com/watch?v=bxQK7hcVyGs&list=PLD4GnSXHpKQ2_eKG5fKzszXs5hYcEsft7))

Julia tries to contribute the ideas to show how important it is to following DRY principle, and then, reduce test cases, improve code quality, and then, shorten time to work on a project.

Back to the video, Julia likes to write down the notes from the video, she does not have time to read 600 pages book, but she knows the tip: get a good video, google talk, and then, invest 2-3 hours:

1. video time 16:07/59.33

What is a "Test smell"

1. duplicate code, hard coded values, etc.

A set pf symptoms of an underlying problem in test code. Like duplicate code, breaking DRY principle.

In the talk, using changing diaper as an example, how to know when to change diaper? When diaper stinks.

code smells - visible problems in test code

behavior smells - test behaving badly

project smells - testing-related problems visible to a project manager

2. video time 19:27 / 59:33

What's a "Test Pattern"?

A "test pattern" is a recurring solution to a test automation problem

- E.g. A "Mock Object" solves the problem of verifying the behavior of an object that should delegate behavior to other objects

Test Patterns occur at many levels:

- Test automation strategy pattern  
recorded test vs scripted test
- Test design Patterns  
implicit setup vs delegated setup
- Test coding patterns  
assertion method, creation method
- Language-specific test coding idioms  
expected exception test, constructor test

3. video time 20:28/59:33

Common code smells

- conditional test logic
- hard to test code
- obscure test
- test code duplication
- test logic in production

4. Example to explain, best part

video time 22:16/59:33

1. Better assertion

2. Hard-wired test data - what is the relationship between those data? What is math? Do we need math? 30, why it is not 1, 19.99, why it is not 2, 69.96, why it is not 3? 6 months later, what is business logic?

Hard-wired test data - will lead to fragile test

3. Introduce custom assert

Avoid any conditional test logic in the test method - make less test code.

Problem of conditional test logic is not sure what you are testing. <- Julia likes this argument  
use customer assert, make a compaction of the code.

video time 28:58 / 59:33

4. Automated fixture teardown

tear down logic is a smell, such as, to separate business logic with presence logic

explicitly create deleteAllTestObjects instead of those complicated logic checking - nested try methods.

5. video time 32:07 / 59:33

Obscure test - irrelevant information

create address

create .. vs createAnonymousCustomer vs createAnonymousInvoice

create customer createAnonymousInvoice

Hide detail in creation method - encapsulation, what is level of info - different levels info stays in one function  
- create more functions - each function much easy to be understandable, easy for unit test, variable scope less than a few lines

## 6. Test coverage, rapid test writing

Work outside in, create functions not existed yet, type test what you like to look

- Julia's comment:

*If the code she wrote, it seems that the task is not showed in Leetcode questions, that means the code is not abstracted, not using object-oriented programming, cannot be unit test easily, people cannot read it without too much memorization, one function does all kinds of tasks; results? just too much time wasted for development, maintenance, testing, and it is bad behavior, not a pro.*

## 7. video time

44:08/ 59:38

Reducing Erratic Tests - Shared Fixture

Build new Shared Fixture for each run

- Avoids Unrepeatable Tests
- When:
  - >> Lazy Setup
  - >> Setup Decorator
  - >> SuiteFixture Setup

Fragile Tests:

Causes:

Interface Sensitivity

- Every time you can change the SUT (S- , U- , T- tear down), tests won't compile or start failing
- You need to modify lots of tests to get things "Green" again
- Greatly increases the cost of maintaining the system

Behavior Sensitivity

- Behavior of the SUT changes but it should not affect test outcome
- Caused by being dependent on too much of the SUT's behavior.

Data sensitivity

Context sensitivity

- something outside the SUT changes  
e.g. system time/date, contents of another application

46:19/ 59:33

Hard to test code

. code can be hard to test for a number of reasons:

- too closely coupled to other software
- No interface provided to set state, observe state
- Only asynchronous interfaces provided

- . Root cause is lack of design for testability
- comes naturally with Test-Driven development
- Must be retrofitted to legacy (test-less) software

- . Temporary workaround is Test Hook
- Becomes test logic in production (code smell) not removed.

50:17/59:33

What Does it take to be successful?

Programming Experience

- + xUnit Experience
- + Testing Experience
- + Design for Testability
- Test smells
- + Test Automation Patterns
- + Fanatical Attention to Test Maintainability

read the website:

[3]<http://xunitpatterns.com/>

Julia's comment:

Source code should be only written for machine to understand - 20 %

Source code should be readable for herself after 6 months / years - 20 %

Source code should be minimal - DRY principle - All other principle for easy to understand, relax for herself - any time - 30 %

Source code should be fun to read, with documented unit test cases, log history etc. - ?

1. <https://www.youtube.com/watch?v=Pq6LHFM4JvE>

2. [https://www.youtube.com/watch?v=bxQK7hcVyGs&list=PLD4GnSXHpKQ2\\_eKG5fKzszXs5hYcEsft7](https://www.youtube.com/watch?v=bxQK7hcVyGs&list=PLD4GnSXHpKQ2_eKG5fKzszXs5hYcEsft7)

3. <http://xunitpatterns.com/>

## OO design principles - DRY - Don't repeat yourself principle (2015-11-14 15:34)

Nov. 14, 2015

It is so great to find something to work on this weekend of November. My favorite time to study the video:

The Don't repeat yourself principle, part 2

[1][https://www.youtube.com/watch?v=wweNdRuM64g&list=PLD4GnSXHpKQ2\\_eKG5fKzszXs5hYcEsft7&index=2](https://www.youtube.com/watch?v=wweNdRuM64g&list=PLD4GnSXHpKQ2_eKG5fKzszXs5hYcEsft7&index=2)

Video time: 30:15/31:07

Summary

- . Repetition breed errors and waste
- . Abstract repetitive logic in code
- . Related fundamentals:
- . template method pattern
- . command pattern
- . dependency inversion principle

Recommended reading:

- . The pragmatic programming: From Journeyman to Master
- . 97 Things Every Programmer Should Know.

So, Julia found one book to read this weekend, 97 Things Every Programmer Should Know.

Further reading: (Nov. 18, 2015)

Template method pattern:

[2][https://en.wikipedia.org/wiki/Template\\_method\\_pattern](https://en.wikipedia.org/wiki/Template_method_pattern)

1. [https://www.youtube.com/watch?v=wwNdRuM64g&list=PLD4GnSXHpKQ2\\_eKG5fKzszXs5hYcEsft7&index=2](https://www.youtube.com/watch?v=wwNdRuM64g&list=PLD4GnSXHpKQ2_eKG5fKzszXs5hYcEsft7&index=2)

2. [https://en.wikipedia.org/wiki/Template\\_method\\_pattern](https://en.wikipedia.org/wiki/Template_method_pattern)

---

## **book reading: 97 Things Every Programmer Should Know (I) (2015-11-15 13:05)**

Nov. 15, 2015.

I like to tell how I get here to read this book - 97 things every programmer should know. It is a long story, but it can be described as this flow:

Julia works on leetcode questions (January 2015) ->

Challenged on August, 2015 about SOLID OO principles (Only know / remember S, basic decoupling, work on interface)

-> procedural code's concern, try to follow DRY principles, Oct. 2015

-> Learn SOLID principles through videos in Nov., 2015:

-> Recommended reading: the book

Spent 3 hours to read the book on Sunday morning, take some notes about my favorite ideas:

1. Act with prudence - Seb Rose

"doing it right" vs "doing it quick"

Julia's comment: thinking about it later :-) Prudence means "be careful".

2. Apply Functional Programming Principles - Edward Garson

Julia's comment:

Look into the term: "high degree of referential transparency"

Questions from Julia, list a 3-5 principles of functional programming principles, most popular ones.

3. Ask, "What Would the User Do?" (You are not the user) - Giles Colborne

Julia's comment:

Look up the term: the false consensus bias - psychologists call it.

4. Automate your coding Standard - Filip van Laenen

Julia's comment: Look up the meaning of "antipatterns", things talked about are interesting: test coverage, coding standard. Read the article later.

5. Beauty is in simplicity - Jorn Olmheim

Julia's comment: Four things we strive for in our code:

Readability (1), maintainability (2), speed of development (3), the elusive quality of beauty (4)

Simple objects with a single responsibility, with simple, focused methods with descriptive names.

**Desirable goal is to have short methods of 5-10 lines of code.** (Julia likes this argument!)

6. Before you refactor - Rajith Attapattu

Julia's comment: great topic, and good to hear some advice.

7. Beware the Share - Udi Dahan

8. The Boy Scout Rule - Robert C. Martin (Uncle Bob)

Julia's favorite: Make it better: improve name of one variable, or split one function into two smaller functions. Or break a circular dependency, or add an interface to decouple policy from detail.

And more about team spirit, help one another and clean up after one another.

9. Check Your Code First before looking to blame others - Allan Kelly

Once you eliminate the impossible, whatever remains, no matter how improbable, must be the truth.

[1]<http://www.allankelly.net/>

10. Choose Your tools with Care - Giovanni Asproni

existing tools - components, libraries, and frameworks

[2]<http://www.atlantec.ie/giovanni-asproni-to-be-added/>

11. Code in the language of the Domain - Dan North

consulting service:

[3]<http://dannorth.net/about/>

12. Code is Design - Ryan Brush

[4]<http://conferences.oreilly.com/strata/stratany2014/public/schedule/speaker/138579>

13. Code layout matters - Steve Freeman

Julia is still working on the code layout; she thinks that the good layout helps, her favorite book: the art of readable code



## 15. Code Reviews - Mattias Karlsson

Code review - increase code quality and reduce defect rate.

used to be: Lead programmer does the review, architect reviews everything.

## 16 Coding with Reason - Yechiel Kimchi

[5][http://programmer.97things.oreilly.com/wiki/index.php/Coding\\_with\\_Reason](http://programmer.97things.oreilly.com/wiki/index.php/Coding_with_Reason)

Julia likes the idea -

1. Try to reason about software correctness by hand
2. Automated tools are preferable but not always possible
3. a middle path - reasoning semiformality about correctness

Julia likes this middle path, in detail,

To divide all the code under consideration into short sections - from a single line, such as a function call, to blocks of less than 10 lines - and argue about their correctness.

Julia likes to memorize those good advices, so just write it down word by word.

1. Avoid using goto statements, as they make remote sections highly interdependent.
2. Avoid using modifiable global variables, as they make all sections that use them dependent.
3. Each variable should have the smallest possible scope. For example, a local object can be declared right before its first usage.
4. Make objects immutable when relevant.
5. Make the code readable using spacing, both horizontal and vertical.
6. Make the code self-documenting by choosing descriptive name for objects, types, functions, etc.
7. If you need a nested section, make it a function. <- Julia's favorite: avoid nested section, add a function
8. Make your functions short and focus on a single task. <- **great advice, obey it.**

The old 24-line limit still applies. Although screen size and resolution have changed, nothing has changed in human cognition since 1960s. ( **24 lines, Julia can do it!** )

9. Function should have few parameters ( **four is a good upper bound**). This does not restrict the data communicated to functions: Grouping related parameters into a single object benefits from object invariants and saves reasoning, such as their coherence and consistency.

## 17. Convenience is Not an -ility - Gregor Hohpe

It is a great topic about API design. A good API follows a consistent level of abstraction, exhibits consistency and symmetry, and forms the vocabulary for an expressive language.

API design reading, found one through search:

[6]<http://lcsd05.cs.tamu.edu/slides/keynote.pdf>

## 18. Deliberate practice

[7]Leading Lean Software Development (Addison-Wesley Professional)

## 19. Domain-Specific Languages

[8]<https://about.me/mhunger>

## 20 Do not afraid to break things - Mike Lewis

## 21. Encapsulate Behavior, Not just State - Julia's favorite article

[9][http://www.researchgate.net/profile/Einar\\_Landre/publications](http://www.researchgate.net/profile/Einar_Landre/publications)

[10][http://programmer.97things.oreilly.com/wiki/index.php/Encapsulate\\_Behavior,\\_not\\_Just\\_State](http://programmer.97things.oreilly.com/wiki/index.php/Encapsulate_Behavior,_not_Just_State)

Examples:

It is easy to find a class with a single 3,000-line main method, or a class with only set and get method for its primitive attributes.

Analysis: do not understand the object-oriented thinking, do not take advantage of the power of objects as modeling constructs.

Given an example, 3 classes: Customer, Order, and Item.

A Customer object is the natural placeholder for the credit limit and credit validation rules.

An Order object knows about its associated Customer, and its addItem operation delegates the actual credit check by calling customer.ValidateCredit(Item.Price()).

Less experienced OO developer, design one objects: OrderManager or OrderService to wrap all business rules. In the design, Order, Customer, and Item are treated as little more than record types.

All logic is factored out of the classes and tied together in one large, procedural method with a lot of internal if-then-else constructs.

But, these methods are easily broken and almost are impossible to maintain. Because encapsulation is broken.

## 22. Hard work does not pay off - Olve Maudal

### 23. Know your limits

[11][http://www.boost.org/doc/libs/1\\_34\\_0/people/greg\\_colvin.htm](http://www.boost.org/doc/libs/1_34_0/people/greg_colvin.htm)

## 24. A message to the Future - Linda Rising

Her website:

[12]<http://www.lindarising.org/>

Julia really likes the short story written in the chapter. The problem solved is difficult, and the solution should be just as difficult for everyone (maybe even for themselves a few months after the code was written) to understand and maintain. But, the code should be easy to understand.

## 25. Only the code tells the truth - Peter Sommerlad

[13]<https://www.youtube.com/watch?v=XLsZkA77h8c>

Notes from video:

Less code = More Software

Let Julia read those sentences and laugh about mistakes she made as well:

1. Complexity is one of the biggest problems with software if not THE biggest.
2. It is much easier to create a complicated "solution" than to really solve a problem.
3. Much software complexity is accidental not inherent to the problem solved.
4. It starts in the small, one statement at a time.

5. Architects and developers need to value Simplicity!
  - . Good Abstractions are the key, as are
  - . Managing Dependencies (Avoid global variables)
6. Software needs to be simpler to solve more complex problems.
7. Simple software requires work and skill but pays off in the long run.

Famous Quotes by Sir C.A.R. (Tony) Hoare

1. Inside every large problem, there is a small program trying to get out.
2. There are two ways of constructing a software design:
  1. one way is to make it so simple that there are obviously no deficiencies, and
  2. the other way is to make it so complicated that there are no obvious deficiencies.The first method is far more difficult.

Good advice:

1. Strive for good names.
2. Structure your code with respect to cohesive functionality, which also naming.
3. Decouple your code to achieve orthogonality.
4. Write automated tests explaining the intended behavior and check the interfaces.
5. Refactor mercilessly when you learn how to code a simpler, better solution.
6. Make your code as simple as possible to read the understand.

- to be continued.

More reading about authors:

1. Enjoy reading the blog of one of authors:

To junior developer:

[14]<http://dearjunior.blogspot.ca/2014/11/make-implicit-concepts-explicit-in-code.html>

Note:

Eric Evans, the thought leader of Domain Driven Design.

[15]<http://dearjunior.blogspot.ca/2012/03/dry-and-false-negatives.html>

1. <http://www.allankelly.net/>
2. <http://www.atlantec.ie/giovanni-asproni-to-be-added/>
3. <http://dannorth.net/about/>
4. <http://conferences.oreilly.com/strata/stratany2014/public/schedule/speaker/138579>
5. [http://programmer.97things.oreilly.com/wiki/index.php/Coding\\_with\\_Reason](http://programmer.97things.oreilly.com/wiki/index.php/Coding_with_Reason)
6. <http://lcsd05.cs.tamu.edu/slides/keynote.pdf>
7. [https://en.wikipedia.org/wiki/Lean\\_software\\_development](https://en.wikipedia.org/wiki/Lean_software_development)
8. <https://about.me/mhunger>
9. [http://www.researchgate.net/profile/Einar\\_Landre/publications](http://www.researchgate.net/profile/Einar_Landre/publications)

10. [http://programmer.97things.oreilly.com/wiki/index.php/Encapsulate\\_Behavior,\\_not\\_Just\\_State](http://programmer.97things.oreilly.com/wiki/index.php/Encapsulate_Behavior,_not_Just_State)
  11. [http://www.boost.org/doc/libs/1\\_34\\_0/people/greg\\_colvin.htm](http://www.boost.org/doc/libs/1_34_0/people/greg_colvin.htm)
  12. <http://www.lindarising.org/>
  13. <https://www.youtube.com/watch?v=XLsZkA77h8c>
  14. <http://dearjunior.blogspot.ca/2014/11/make-implicit-concepts-explicit-in-code.html>
  15. <http://dearjunior.blogspot.ca/2012/03/dry-and-false-negatives.html>
- 

## The clean code talks (2015-11-18 10:05)

Nov. 18, 2015

Review the video about the testing, and try to get more from this lecture:

"The Clean Code Talks – Inheritance, Polymorphism, & Testing"

[1]<https://www.youtube.com/watch?v=4F72VULWFvc>

action items:

1. put sample code in C #, and then, check in git;
2. write down key words, and easy for review and follow as rules.

Julia's sample code in C #:

C # code using conditional implementation, one class Node

[2]<https://github.com/jianminchen/OODesignPractice/blob/master/ConditionalVersion.cs>

[3][https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OO\\_Design\\_CleanCodeTalk\\_ConditionalVersion.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OO_Design_CleanCodeTalk_ConditionalVersion.cs)

better solution: Node, OpNode, ValueNode:

[4][https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OO\\_Design\\_CleanCodeTalk\\_OpNode-ValueNode.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OO_Design_CleanCodeTalk_OpNode-ValueNode.cs)

optimal solution:

[5][https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OO\\_Design\\_CleanCodeTalk\\_OpNode-ValueNodeAndMore.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OO_Design_CleanCodeTalk_OpNode-ValueNodeAndMore.cs)

Perfect example to learn S.O.L.I.D. OO principles, open for extension, close for change. The above optimal solution does not have any if statement, and any new arithmetic operation just needs a new class, no need to touch existing code. Julia passes the learn Open/Close principle. Move on to next one, Liskov substitution principle!

Notes:

Premise - Most ifs can be replaced by polymorphism

Why?

Easy to read, test without ifs.

polymorphic systems are easier to maintain.

Julia: write down the sentence in the talk: supporting facts: one execution path, so easy to understand, test, and extend.

Use polymorphism

If an object should behave differently based on its state.

If you have to check the same conditions in multiple places.

- binding is not on compile time,

Use conditionals

Mainly to do comparisons of primitive objects: >, <, ==, !=

There other uses, but today we focus on avoiding if

Do not return null in the method

To be if free

Never return a null, instead return a Null object, e.g. an empty list

Don't return error codes, instead throw an Exception (Run Time please!)

Rampant (wild, unchecked) subclassing

Polymorphism uses subclassing

Be careful about runaway subclassing (another talk focuses on that)

avoid pitfall: inheritance hierarchy - too complex

State based behavior

Replace conditionals with polymorphism

You have a conditional that chooses different behavior depending on the type of an object.

Move each leg of the conditional to an overriding method in a subclass.

Make the original method abstract.

Example:

```
double getSpeed() {
    switch ( _type) {
    case EUROPEAN:
        return getBaseSpeed();
    case AFRICAN:
        return getBaseSpeed() - getLoadFactor() * _numberOfCoconuts;
    case NORWEGIAN_BLUE:
        return ( _isNailed)? 0 : getBaseSpeed( _voltage);
    }
    throw new RuntimeException ("Should be unreachable");
}
```

Suggestion to solve the problem: use subclasses

Isn't there already a type system?

An Exercise:

Model this!

1 + 2 \* 3

favorite interview question by the speaker:

Represent this as a tree

```
+
/\
1 *
/\
2 3
```

Node object to store the information

evaluate()

computes the result of an expression

Most of people come out solution like the following:

Using conditionals

```
class Node {
char operator;
double value;
Node left;
Node right;
double evaluate() {
switch(operator) {
case '#': return value;
case '+': return left.evaluate() + right.evaluate();
case '*' return left.evaluate() * right.evaluate();
case ... // edit this for each new operator
}
}
}
```

Big problem on this:

graphic representation,

Node

op:char

value: double

left: Node

right:Node

-----

evaluate():double

Julia could not figure out the analysis here <- first time to memorize this analysis, and also try to learn reasoning

Analyzing attributes

# + \*

function yes yes

value yes  
left yes yes  
right yes yes

Two different behaviors are fighting here, (see the above table),  
if you are the operation node, then you need your left and right child; whereas value node, you just need value.

Either you need value or you need left and right, but you never need both.  
One class have multiple tasks entangled, polymorphism is through if statement, not through polymorphism.

video time: 11:42/38:24

Let us break it up:

Node

-----

evaluate(): double

| |

ValueNode OpNode

value: double op: char

----- left: Node

evaluate: double right: Node

-----

evaluate(): double

As showing above, break Node into ValueNode and OpNode, so ValueNode does not have left and right child because of no meaning over there.

Operations and values

```
abstract class Node {  
    abstract double evaluate();  
}
```

```
class ValueNode extends Node {  
    double value;  
    double evaluate() {  
        return value;  
    }  
}
```

```
class OpNode extends Node {  
    char operator;  
    Node left;  
    Node right;  
    double evaluate() {  
        switch(operator) {  
            case '+': return left.evaluate() + right.evaluate();  
            case '-': return left.evaluate() - right.evaluate();  
            case ... // edit this for each new operator  
        }  
    }  
}
```

```
}  
}
```

How to extend this? Every time you add a new operator, need to hold on source code, and add code in switch statement, how to make it better?

Tree looks like:

```
OpNode  
+  
/\nValueNode OpNode  
1 *  
/\nValueNode ValueNode  
2 3
```

OpNode divides into AdditionNode and MultiplicationNode

```
OpNode  
-----  
left: Node  
right: Node  
-----  
evaluate(): double
```

```
AdditionNode MultiplicationNode  
-----  
evaluate(): double evaluate(): double
```

```
abstract class Node {  
    abstract double evaluate();  
}
```

```
class ValueNode extends Node {  
    double value;  
    double evaluate() {  
        return value;  
    }  
}
```

```
class abstract OpNode extends Node {  
    Node left;  
    Node right;  
    abstract evaluate();  
}
```

video time: 14:39/38:24



Operation classes

```
class AdditionNode extends OpNode {  
double evaluate() {  
return left.evaluate() + right.evaluate();  
}  
}
```

```
class MultiplicationNode extends OpNode {  
double evaluate() {  
return left.evaluate() * right.evaluate();  
}  
}
```

Now, the new tree diagram:

```
AdditionalNode  
+  
/\   
ValueNode MultiplicationNode  
1 *  
/\   
ValueNode ValueNode  
2 3
```

Julia's C # implementation:

[6][https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OODesign\\_CleanCodeTalk\\_OpNode-ValueNodeAndMore.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OODesign_CleanCodeTalk_OpNode-ValueNodeAndMore.cs)

Further exploration

Define toString() prints the infix expression placing parenthesis only when necessary.

Add new math operators: exponentiation, factorial, logarithm, trigonometry

Summary

A polymorphic solution is often better because:

1. new behavior can be added without having the original source code, and
2. each operation/concern is separated in a separate file which makes it easy to test/understand.

Prefer polymorphism over conditionals:

switch almost always means you should use polymorphism  
if is more subtle ... sometimes an if is just an if

Repeated Condition

22:36/38:24

## Two piles

piles of Objects pile of Construction

- . business logic . factories
- . the fun stuff . builders
- . Provider<T>
- . given the collaborators needed .created and provides collaborators (Dependency Injection)

## Construction

```
class Consumer {  
  Consumer(Update u) {... }  
}
```

```
class Factory {  
  Consumer build() {  
    Update u = FLAG_i18n_ENABLED? new I18NUpdate() : new NonI18NUpdate();  
  
    return new Consumer(u);  
  }  
}
```

## Benefits

Conditional is localized in one place

No more duplication

Separation of responsibilities, and global state

Common code is in one location

Testing independently easily, and in parallel

Looking at the subclasses makes it clear what the differences are

## When to use polymorphism

Behavior changes based on state

Parallel conditionals are in multiple places in code

Be pragmatic

You will still have some conditionals

Question and answers:

Argument: Easy to read switch statement vs a lot of subclasses

File over thousand line of length

Easy to create a new class

Single responsibility

Behavior is controlled by a lot of flags vs. a lot of classes collaboration

Julia's comment:

1. Julia watched the video over 3 times, she likes the teaching and sample code.
2. Julia likes to refactor the C # code, learn OO design. Strongly recommend this video to friends.
3. C # code using conditional implementation, one class Node

[7]<https://github.com/jianminchen/OODesignPractice/blob/master/ConditionalVersion.cs>

1. <https://www.youtube.com/watch?v=4F72VULWFvc>
2. <https://github.com/jianminchen/OODesignPractice/blob/master/ConditionalVersion.cs>
3. [https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/00Design\\_CleanCodeTalk\\_ConditionalVersion.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/00Design_CleanCodeTalk_ConditionalVersion.cs)
4. [https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/00Design\\_CleanCodeTalk\\_OpNodeValueNode.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/00Design_CleanCodeTalk_OpNodeValueNode.cs)
5. [https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/00Design\\_CleanCodeTalk\\_OpNodeValueNodeAndMore.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/00Design_CleanCodeTalk_OpNodeValueNodeAndMore.cs)
6. [https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/00Design\\_CleanCodeTalk\\_OpNodeValueNodeAndMore.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/00Design_CleanCodeTalk_OpNodeValueNodeAndMore.cs)
7. <https://github.com/jianminchen/OODesignPractice/blob/master/ConditionalVersion.cs>

---

## Study time - OO design, principles and testability (2015-11-21 12:24)

Nov. 21, 2015

Spent one hour to watch the video of topic: (Julia's rating A+) from 10:20am - 11:23am. Try to go outdoor to play tennis and get some workout in this freezing temperature. Come back later to get more notes written down, and help myself to continue to learn on this topic.

"Michael Feathers - the deep synergy between testability and good design"

[1]<https://www.youtube.com/watch?v=4cVZvoFGJTU>

some notes: (Julia's comment, I made all the mistakes in the talk, that is the reason I like it; start to learn OO design)

1. video time 11:43/50:49

Solving Design Problems

Solving Testing Problems

solving design problems means solving testing problems.

Testing pain

**State hidden in method**

You find yourself wishing you could access local variables in tests.

Your method is so long.

Analysis for reasons:

**Method are too long, SRP violations** <- Julia's comment: agree!

More than 20 lines <- not good

usually 5 - 10 lines

matching human cognitive ability - small thing, focus on one thing, little thing

video time 14:36/50:49

### **Difficult setup**

Instantiating a class involving instantiating 12 others

Need to factor the class into small pieces - you always can do that

Interface / Class / Object

**too much coupling** <- Julia's comment: made mistakes before, will be more careful!

concrete pain in the testing, but the problem is in the design.

video time: 17:25/ 50:49

### **Incomplete shutdown**

Piece of your code lose resources and you never realize it until you test.

In detail, for example, you got to the shop, after the work, does not clean up after yourself; in C++, resource leak. In test environment, class should take itself better.

**Classes Don't Take Care of themselves, Poor Encapsulation** <- Julia's comment: still remember in C++ destructor, need to free the memory.

video time: 19:46/ 50:49

### **State-Leaks Across Tests**

The state of one test affects another

For instance, share static mutable data through the tests. So, one test is depending on another test's actions.

**Singletons or other forms of Global mutable state** <- Julia: look into more later.

video time: 22:27/50:49

### **Framework frustration**

Testing in the presence of a framework is hard

**Insufficient domain separation** <- Julia's comment: watch again, example for domain separation.

video time: 24:39

### **Difficult mocking**

You find yourself writing mocks for objects returned by other objects

**Law of Demeter Violations** <- Julia's comment: get more familiar with the law.

For example, A ->B->C->D, a chain of dependencies to get something. Client code like A, B, C, D, those dependencies hurts you on compile time, also in conceptual, you have to know too many clients in order to do the piece of work. So, mocking is from fake A to fake B to fake C to fake D, real big pain.

video time: 26:58

### **Difficult Mocking - 2**

Hard to mock particular classes

Answer: **Introduce interface** <- Julia's comment: Good point!

video time: 28:07

### **Hidden effects:**

You cannot test a class because you have no access to the ultimate effect of its execution.

Reason: **Insufficient separation of concerns, encapsulation violation**

**Hidden inputs - same type of things comparing to Hidden effects.**

There is no way to instrument the setup conditions for a test through the API.

Reason: **Over-Encapsulation, insufficient separation of concerns.**

video time: 30:31

**Unwieldy parameter lists**

It is too much work to call a method or instantiate a class

Reason: **Too many responsibilities in a class or method**

video time: 32:07

**Insufficient Access**

You find yourself wishing you could test a private method.

Reason: **Too many responsibilities in a class**

**Test Thrash**

Many unit tests changes whenever you change your code

Symptoms: unit test always fails

reason: **Open / Close violations**

In detail, Close for modification / Open for extension

class, function, need to have a small, tight focus, (Julia agrees, matching cognitive principles, small tight focus!)

small piece, easy to reason

Why? **small function, small test.**

Good design follows cognitive principles. Small detail makes people easy to recognize.

**The Golden Hammer** <- another name for dependency injection.

**dependency injection**

**Groping Test Tools**

Best example Julia's favorite:

a team many years ago, every class member/method is public; what is problem?

Should be a lot of things encapsulated, and a small API to access it.

Make tests too brittle.

using medical condition to help understand the OO design,

people do not feel the pain - easy to break bones, and others, can not live long

So, every one of us hates pain, but pain is the learning experience.

video time: 45:24

**Testing isn't Hard. Testing is Easy in the Presence of Good Design.**

Saturday evening video:

Escaping the Technical Debt Cycle - Michael Feathers

[2]<https://www.youtube.com/watch?v=7hL6g1aTGvo>

Read the blog to understand Law of Demeter - "**Principle of least knowledge**"

[3]<http://javarevisited.blogspot.ca/2014/05/law-of-demeter-example-in-java.html>

[4]<http://javarevisited.blogspot.sg/2012/03/10-object-oriented-design-principles.html>

Fast App Dev using Dependency Injection, Code First EF, and SOLID Design - SVNUG Presentation 32 (Julia comment: surprisingly great! good presentation, code with demo. Learn a lot!)

[5]<https://www.youtube.com/watch?v=zY1kzXPD568>

1. <https://www.youtube.com/watch?v=4cVZvoFGJTU>

2. <https://www.youtube.com/watch?v=7hL6g1aTGvo>

3. <http://javarevisited.blogspot.ca/2014/05/law-of-demeter-example-in-java.html>

4. <http://javarevisited.blogspot.sg/2012/03/10-object-oriented-design-principles.html>

5. <https://www.youtube.com/watch?v=zY1kzXPD568>

---

## Study time - Learn dependency injection, and others (2015-11-22 12:49)

Nov.22, 2015

This Sunday morning, Julia spent time to work on learning dependency injection, and really had great time to get concrete ideas what to learn in this OO design principles, S.O.L.I.D. Here are the videos she watched:

1. Understanding Dependency Injection (DI) & IOC

[1]<https://www.youtube.com/watch?v=RVpADaFIIRw>

2. Dependency Injection using Microsoft Unity Application block ( DI IOC) - 30 minutes training (Julia's comment: so great the video, Julia follows every step and then understand whole idea using third party container to do dependency injection)

[2]<https://www.youtube.com/watch?v=FuAhnqSDe-o>

Julia tries to catch up a lot of things last 5 years, but OO principles, dependency injection is just a new thing for her. She likes the learning, and also enjoys the great teaching from those videos. Compared to Leetcode algorithms problem solving, this should be pass; in other words, not so difficult and intimidating. Most important, do something as well besides watching the video, maybe, short code to practice, sharing; a short note to help understand the topic in the video.

Julia likes to write code, and she believes that being a good software developer, also she has to develop the skills to write, enjoy writing the blogs :-), and also find different challenging tasks in the daily life.

More videos on Sunday afternoon from 4:00pm - 11:30pm, enjoyed Google employee presentation, and two Microsoft employees's.

1. Codemania 2014: Scott Hanselman - Angle Brackets, Curly Braces, JavaScript & Assembler

[3]<https://www.youtube.com/watch?v=k0e7R3fsdKM> (watch again later!)

2. Rob Ashton - "Javascript sucks and it doesn't matter"

[4][https://www.youtube.com/watch?v=PV\\_cFx29Xz0](https://www.youtube.com/watch?v=PV_cFx29Xz0)

3. Jon Skeet - "Back to basics: the mess we've made of our fundamental data types"

[5]<https://www.youtube.com/watch?v=l3nPJ-yK-LU>

4. Going Beyond Dependency Injection

[6]<https://www.youtube.com/watch?v=JfgP566BHW0>

5. 10 Rules of English Communication For developers

[7]<https://www.youtube.com/watch?v=7MvyvHRAUR0>

1. <https://www.youtube.com/watch?v=RVpADaFI1Rw>

2. <https://www.youtube.com/watch?v=FuAhnqSDe-o>

3. <https://www.youtube.com/watch?v=k0e7R3fsdKM>

4. [https://www.youtube.com/watch?v=PV\\_cFx29Xz0](https://www.youtube.com/watch?v=PV_cFx29Xz0)

5. <https://www.youtube.com/watch?v=l3nPJ-yK-LU>

6. <https://www.youtube.com/watch?v=JfgP566BHW0>

7. <https://www.youtube.com/watch?v=7MvyvHRAUR0>

---

## Testing and Refactoring Legacy Code (2015-11-25 22:55)

Nov. 25, 2015

Testing and refactoring Legacy Code ( Julia rating: very good! Definitely watch again, and practice demo code using C # language. )

[1][https://www.youtube.com/watch?v=\\_NnEIPO5BU0](https://www.youtube.com/watch?v=_NnEIPO5BU0)

Julia watched the video, now she knew better about dependency injection, design, and also unit test. The lecture is excellent. MockitoJUnitRunner/ Spring is used in the demo, TripDAO inject is demoed to add for better code.

The lecture in the video is to work on a legacy code in Java, and then, the code is analyzed, refactored, and test code is also added. Julia follows 100 % the thinking process, great time to learn. Will review the part to do refactoring in the video.

Notes taken down:

Craftsmen at work

- . Write readable and maintainable code
- Code must express business rules
- . Strive for simplicity
- . Know your tools well (i.e. frameworks, shortcuts)
- . Work in small and safe increments
- Commit often
- . Embrace changes, be brave
- . Boy scout rule / No broken windows

Book reading:

**software craftsmanship**

## Professionalism Pragmatism Pride

Sandro Mancuso

The verse I like it as well, "How it is done is as important as getting it done".

The story in early 90s to have his code reviewed by the manager in his 20s, really a good story.

200 lines of code, and the cases are the following:

1. allocate memory in one method and deallocate it in another method - risk of memory leak, temporal coupling
2. block of lines, reduce eight line to 2 by thinking harder
3. Try/ catch block too big
4. name of this variable / method, what do they mean?
5. Hard-coded bit - if we want to change where it points to, need to open it, change it, recompile it, and redeploy the entire application
6. Code duplication all over the place
7. A big method - how much we would need to keep in our heads if every single method were that big? What about making them smaller and naming them according to their behavior.
8. It is not respectful - a few lines of code, no one else could understand. No idea what the code does. some cryptic ode in there, trying to show how clever.

Julia had this kind of experience as well in 2014, but Julia now is a big fan of OO principles, SRP - single responsibility principle is her favorite one. And then, open/ close principle, how to estimate the probability of change and then create new class based on the odds, she just loves the idea.

Ref:

[2]<http://craftedsw.blogspot.ca/2012/12/screencast-testing-and-refactorin g.html>

1. [https://www.youtube.com/watch?v=\\_NnElP05BU0](https://www.youtube.com/watch?v=_NnElP05BU0)

2. <http://craftedsw.blogspot.ca/2012/12/screencast-testing-and-refactoring.html>

---

## Study time - watch CppCon videos (2015-11-29 15:46)

Nov. 29, 2015

Julia likes to get some idea how unit test can be done as a developer, and also in C++. She likes to adventure out and see if she can get ideas how to build up good habit to do unit test.

[1] <https://www.youtube.com/watch?v=XLsZkA77h8c>

Notes from video:

Less code = More Software

Let Julia read those sentences and laugh about mistakes she made as well:

1. Complexity is one of the biggest problems with software if not THE biggest.
2. It is much easier to create a complicated "solution" than to really solve a problem.
3. Much software complexity is accidental not inherent to the problem solved.
4. It starts in the small, one statement at a time.



5. Architects and developers need to value Simplicity!

- . Good Abstractions are the key, as are

- . Managing Dependencies (Avoid global variables)

6. Software needs to be simpler to solve more complex problems.

7. Simple software requires work and skill but pays off in the long run.

(4:06/5:49)

Quotes by Sir C.A.R. (Tony) Hoare (quick sort algorithm inventor in 1959/1960)

"Inside every large problem, there is a small problem trying to get out."

Bad design vs good way:

one way is to make it simple that there are no obviously no deficiencies, and the other way is

to make it so complicated that there are no obvious deficiencies.

The first methods is far more difficult

C++ test-driven development

[2] <https://www.youtube.com/watch?v=YrGSQXZmAXs>

[3] <https://github.com/PeterSommerlad>

Another video watched in Sunday evening:

CppCon 2014: Titus Winters "The Philosophy of Google's C++ code"

[4] <https://www.youtube.com/watch?v=NOCElcMcFik>

4k-ish C++ engineers in Google

CppCon 2015: Bjarne Stroustrup "Writing Good C++14"

[5] <https://www.youtube.com/watch?v=hEx5DNLWGgA>

CppCon 2015: Herb Sutter "Writing Good C++14... By Default" ( Julia rating: A+, Nov. 30, 10:00-11:11pm)

<https://www.youtube.com/watch?v=hEx5DNLWGgA>

CppCon 2015: Sean Parent "Better Code: Data Structures"

[6] <https://www.youtube.com/watch?v=sWgDk-o-6ZE>

[7] <https://github.com/sean-parent/sean-parent.github.io/wiki/Papers-and-Presentations>

[8] <https://channel9.msdn.com/Events/GoingNative/2013/Cpp-Seasoning>

[9] <http://www.codeproject.com/Articles/854127/Top-Beautiful-Cplusplus-std-Algorithms-Examples>

1. <https://www.youtube.com/watch?v=XLsZkA77h8c>

2. <https://www.youtube.com/watch?v=YrGSQXZmAXs>

3. <https://github.com/PeterSommerlad>

4. <https://www.youtube.com/watch?v=NOCElcMcFik>

5. <https://www.youtube.com/watch?v=hEx5DNLWGgA>

6. <https://www.youtube.com/watch?v=sWgDk-o-6ZE>

7. <https://github.com/sean-parent/sean-parent.github.io/wiki/Papers-and-Presentations>

8. <https://channel9.msdn.com/Events/GoingNative/2013/Cpp-Seasoning>

9. <http://www.codeproject.com/Articles/854127/Top-Beautiful-Cplusplus-std-Algorithms-Examples>

## 1.4 December

### C++ - write quick code in C++ (2015-12-03 00:22)

Dec. 2, 2015

Julia likes to watch more C++ videos before she plans to write more Leetcode question using C # programming language. She likes to learn C++ by going through CppCon videos. Amazed that it is so easy to find high quality talk through the conference.

Read the book in short future (180 pages):

A tour of C++

( Dec. 3, 2015, read 1 hour, review Union, Enumeration (2.4, 2.5) )

videos:

CppCon 2014: Herb Sutter "Back to the Basics! Essentials of Modern C++ Style"

[1]<https://www.youtube.com/watch?v=xnqTKD8uD64>

Writing Quick Code in C++, Quickly

[2]<https://www.youtube.com/watch?v=ea5DiCg8HOY>

CppCon 2015: Bjarne Stroustrup "Writing Good C++14"

[3]<https://www.youtube.com/watch?v=1OEu9C51K2A>

CppCon 2015: Gabriel Dos Reis "Contracts for Dependable C++"

[4]<https://www.youtube.com/watch?v=Hjz1eBx91g8> ( Julia's rating: A, easy to understand, and will follow good contracts - write precondition, postcondition, invariant)

video 34:12/55:48 (Dec.4, 2015)

Semantics

- Precondition [[expects: condition]]

1. Arguments are evaluated
2. Condition is evaluated
3. Out-of-contract counter-measure deployed if contract violated
4. First statement of the user-authored function body executed

- Postcondition: [[ensures: condition]]

1. User-authored function body executed (expected return)  
. Return expression, if any, evaluate
2. condition is evaluated
3. Out-of-contract counter-measure deployed if contract violated
4. Control transferred to caller.

CppCon 2015: Neil MacIntosh "Evolving array \_view and string \_view for safe C++ code"

[5]<https://www.youtube.com/watch?v=C4Z3c4Sv52U>

[6]<https://github.com/isocpp/CppCoreGuidelines/tree/master/talks>

1. <https://www.youtube.com/watch?v=xnqTKD8uD64>
  2. <https://www.youtube.com/watch?v=ea5DiCg8H0Y>
  3. <https://www.youtube.com/watch?v=10Eu9C51K2A>
  4. <https://www.youtube.com/watch?v=Hjz1eBx91g8>
  5. <https://www.youtube.com/watch?v=C4Z3c4Sv52U>
  6. <https://github.com/isocpp/CppCoreGuidelines/tree/master/talks>
- 

## Coding standards - quick review (2015-12-04 23:27)

Dec. 4 , 2015

[1][https://www.youtube.com/watch?v=zW-i9eVGU\\_k](https://www.youtube.com/watch?v=zW-i9eVGU_k)

CppCon 2015: Titus Winters " Lessons in Sustainability... "

video 38:00/1:09

Policies

You need ways to guide the codebase. What if everyone writes their own hash?

- . style guides. Strongly encourage consistency and safety
- . Code review, and take it seriously - encourage sane code
- . Best practices - lightly encouraged guidance
- . Readability - Require responsible supervision, mentorship
- . Churn policies - encourage responsible infrastructure change.

Read the book:

**C++ Coding Standards**

**101 Rules, Guidelines, and Best Practices**

written by Herb Sutter, Andrei Alexandrescu

Julia understood that being a smart software developer, better spend time to work on coding standard, design principles, rules first, and then, develop some quality product with confidence.

Find 10 rules - most favorite ones.

1. [https://www.youtube.com/watch?v=zW-i9eVGU\\_k](https://www.youtube.com/watch?v=zW-i9eVGU_k)
-

## C++, coding standards, and code refactoring (2015-12-05 11:54)

Dec. 5, 2015

Surprised that C++ is such a popular programming language, a few days study, Julia was so amazed and she likes to learn C++ again; a lot of coding standards, C++ guidelines also applies to the programming languages she uses, C #, OO programming.

CppCon 2014: James McNellis & Kate Gregory "Making C++ Code Beautiful"

[1] <https://www.youtube.com/watch?v=BiYliKliFvs>

CppCon 2014: James McNellis & Kate Gregory "Modernizing Legacy C++ Code"

[2] <https://github.com/CppCon/CppCon2014/tree/master/Presentations>

video 26:46/59:20

Keep Functions Linear

Functions that have mostly linear flow are...

... easier to understand

... easier to modify during maintenance and bug fix

Recommendations:

Eliminate complexity introduced by the preprocessor

Refactor functions to linearize and shorten them

Update any manual resource management to use RALL

Litter your code with the const qualifier

Convert C casts to C++ casts

Use algorithms instead of loops

[3] <https://github.com/isocpp/CppCoreGuidelines/blob/master/CppCoreGuidelines.md>

Surprised that C++ is such a popular programming language, a few days study, Julia was so amazed and she likes to learn C++ again; a lot of coding standards, C++ guidelines also applies to the programming she uses, C #, OO programming, and are great.

<https://github.com/isocpp/CppCoreGuidelines/blob/master/CppCoreGuidelines.md>

1. <https://www.youtube.com/watch?v=BiYliKliFvs>

2. <https://github.com/CppCon/CppCon2014/tree/master/Presentations>

3. <https://github.com/isocpp/CppCoreGuidelines/blob/master/CppCoreGuidelines.md>

---

## Algorithms, performance with data structures (2015-12-05 14:36)

Dec. 5, 2015

( 11:50am - )

CppCon 2014: Chandler Carruth "Efficiency with Algorithms, Performance with Data Structures"

[1]<https://www.youtube.com/watch?v=fHNmRkzxHWs>

code examples:

substring() -  $O(N^2)$  algorithm, to better algorithm – look for needle in hay algorithm – Next, Knuth-Morris-Pratt (a table to skip) – Finally, Boyer-Moore (use the end of the needle) (video time: 19:25/1:13:40)

Julia worked on this algorithm problem - actually it is one of leetcode questions on June 10, 2015:

[2][http://juliachencoding.blogspot.ca/search/label/Boyer-Moore %20algorithm](http://juliachencoding.blogspot.ca/search/label/Boyer-Moore%20algorithm) (question 3)

[3]<https://github.com/jianminchen/stringDemo/blob/master/Program.cs> (June 10, 2015, strstr function)

std::vector using vector::reserve call first

cache[key] – save a local variable to avoid duplicated calls 4 times

41:10/1:13:40

STD::LIST

doubly-linked list

Each node separately allocated

All traversal operations chase pointers to totally new memory

In most cases, every step is a cache miss

Only use this when you rarely traverse the list, but very frequently update the list

1. <https://www.youtube.com/watch?v=fHNmRkzxHws>

2. <http://juliachencoding.blogspot.ca/search/label/Boyer-Moore%20algorithm>

3. <https://github.com/jianminchen/stringDemo/blob/master/Program.cs>

---

## Cpp core guidelines (2015-12-07 22:57)

Dec. 7, 2015

[1]<https://github.com/isocpp/CppCoreGuidelines/blob/master/CppCoreGuidelines.md>

After 2-3 hours reading, Julia found out that she learns quickly through the reading comparing to Cpp conference videos. So, she plans to read the whole document - 452 pages, and then, memorize the guidelines; and then, she spends less time on videos.

Most of guidelines are well documented with readable examples. So, it is good to write down some study notes.

Her most favorite guideline on Dec. 7, 2015, is called to "**express the intent**", the first day reading.

P.3: Express intent

Here are things Julia likes, "**the index is exposed**", "**index outlives the scope of loop**", good warnings.

—

Reason

Unless the intent of some code is stated (e.g., in names or comments), it is impossible to tell whether the code does what it is supposed to do.

[2]

Example

```
int i = 0;
while (i < v.size()) {
// ... do something with v[i] ...
}
```

The intent of "just" looping over the elements of `v` is not expressed here. The implementation detail of an index is exposed (so that it might be misused), and `i` outlives the scope of the loop, which may or may not be intended. The reader cannot know from just this section of code.

Better:

```
for (const auto& x : v) { /* do something with x */ }
```

—

Second favorite tip:

F.2: A function should perform a single logical operation

Dec. 8, 2015

Favorite guidelines, review again.

P.4: Ideally, a program should be statically type safe

P.5: Prefer compile-time checking to run-time checking

I.4: Make interfaces precisely and strongly typed

ES.5: Keep scopes small

ES.20: Always initialize an object

ES.23: Prefer the

```
{ }
```

initializer syntax

ES.78: Always end a non-empty

case

with a

break

ES.41: If in doubt about operator precedence, parenthesize

ES.45: Avoid "magic constants"; use symbolic constants

ES.46: Avoid lossy (narrowing, truncating) arithmetic conversions

T.20: Avoid "concepts" without meaningful semantics

Dec. 22, 2015

**P.4: Ideally, the program should be statically type safe**

Problem areas: unions, casts, array decays, range errors, narrow conversions.

Alternatives:

unions - use invariant

casts - template can help  
array decay - use span  
range error - use span

type safe - a new keyword, and get some examples about the area.

further reading:

- [3]<http://stackoverflow.com/questions/208959/c-variant>
- [4][http://www.boost.org/doc/libs/1\\_36\\_0/doc/html/variant.html](http://www.boost.org/doc/libs/1_36_0/doc/html/variant.html)

Thinking in C++

- [5]<http://web.mit.edu/merolish/ticpp/TicV2.html>

C # type safe compile time

- [6]<http://stackoverflow.com/questions/6927642/is-there-a-name-for-this-pattern-c-compile-time-type-safety-with-params-arg>

- [7][https://en.wikipedia.org/wiki/Type\\_safety](https://en.wikipedia.org/wiki/Type_safety)
- [8][http://en.cppreference.com/w/cpp/language/dynamic\\_cast](http://en.cppreference.com/w/cpp/language/dynamic_cast)

## **P.6. What cannot be checked at compile time should be checkable at run time**

- [9]<http://okmij.org/ftp/Computation/Subtyping/Preventing-Trouble.html>
- [10]<http://okmij.org/ftp/Computation/Subtyping/Trouble.html> #Problem

To be continued.

1. <https://github.com/isocpp/CppCoreGuidelines/blob/master/CppCoreGuidelines.md>
  2. <https://github.com/isocpp/CppCoreGuidelines/blob/master/CppCoreGuidelines.md#example-3>
  3. <http://stackoverflow.com/questions/208959/c-variant>
  4. [http://www.boost.org/doc/libs/1\\_36\\_0/doc/html/variant.html](http://www.boost.org/doc/libs/1_36_0/doc/html/variant.html)
  5. <http://web.mit.edu/merolish/ticpp/TicV2.html>
  6. <http://stackoverflow.com/questions/6927642/is-there-a-name-for-this-pattern-c-compile-time-type-safety-with-params-arg>
  7. [https://en.wikipedia.org/wiki/Type\\_safety](https://en.wikipedia.org/wiki/Type_safety)
  8. [http://en.cppreference.com/w/cpp/language/dynamic\\_cast](http://en.cppreference.com/w/cpp/language/dynamic_cast)
  9. <http://okmij.org/ftp/Computation/Subtyping/Preventing-Trouble.html>
  10. <http://okmij.org/ftp/Computation/Subtyping/Trouble.html#Problem>
- 

## **reading book - Exceptional C++ (2015-12-08 22:12)**

Dec. 8, 2015

Start to read the book Exceptional C++ today, here is the link of the book:

- [1]<http://www.amazon.ca/Exceptional-Engineering-Programming-Problems-Solutions/dp/0201615622>

My best learning experience on first hour is this example:

#### Item 18. Code Complexity—Part 1

#### Item 19. Code Complexity—Part 2

The good exercise, count "Nonexceptional Code Paths", excellent metrics to calculate in the example. How many does Julia count? (keep it secret!)

```
String EvaluateSalaryAndReturnName( Employee e )
{
    if( e.Title() == "CEO" || e.Salary() > 100000 )
    {
        cout << e.First() << " " << e.Last() << " is overpaid" << endl;
    }
    return e.First() + " " + e.Last();
}
```

To be continued.

1. <http://www.amazon.ca/Exceptional-Engineering-Programming-Problems-Solutions/dp/0201615622>

---

### Coding principles, good/bad design (2015-12-10 21:10)

Dec. 10, 2015

Julia spent time to work on legacy code she wrote last few years, she chooses the **good** design this time. Share 3 things:

1.

One of favorite quotes by Sir C.A.R. (Tony) Hoare (quick sort algorithm inventor in 1959/1960)

"Inside every large problem, there is a small problem trying to get out."

2.

One of favorite talks:

Less code = More Software

Peter Somerland

[1] <https://www.youtube.com/watch?v=XLsZkA77h8c>

3.

And her favorite verse of design from Peter Somerland:

Bad design vs good way:

one way is to make it simple that there are obviously no deficiencies, and the other way is to make it so complicated that there are no obvious deficiencies.

The first methods is far more difficult.

1. <https://www.youtube.com/watch?v=XLsZkA77h8c>

---



## Learn C++ (2015-12-12 14:39)

Dec. 11, 2015

Spent two hours in the evening (7:00pm - 9:00pm) of Friday to watch C++ video. Great teaching, so Julia likes to follow up in the future to make the study a great one.

CppCon 2015: Herb Sutter "Writing Good C++14... By Default"

[1]<https://www.youtube.com/watch?v=hEx5DNLWGgA>

Action items:

1. Write down some notes
2. Read the book "tour of C++" first, and then, come back to watch the video again.

1. <https://www.youtube.com/watch?v=hEx5DNLWGgA>

---

## C++ - make simple tasks simple (2015-12-12 14:43)

Dec. 11, 2015

Spent two hours to watch the video of C++:

CppCon 2014: Bjarne Stroustrup "Make Simple Tasks Simple!"

[1]<https://www.youtube.com/watch?v=nesCaocNjtQ>

Action item:

1. Write down 1-2 examples with code in this blog, and then, understand better than watching.
2. Read the "tour of C++" book first.

1. <https://www.youtube.com/watch?v=nesCaocNjtQ>

---

## Book reading: a tour of C++ (2015-12-12 14:46)

Dec. 12, 2015

Enjoy 2 hours to read the book: tour of C++ on Saturday morning.

A Tour of C++

[1]<http://www.amazon.com/A-Tour-C-In-Depth/dp/0321958314>

Plan to spend 10 hours to read the book.

First 3 hours reading, here are favorite advice from the book:  
the chapter 9 - containers Page (104-105)

Advice:

1. Use vector as your default container
8. User `push_back()` or `resize()` on a container rather than `realloc` on an array
13. **To preserve polymorphic behavior of elements, store pointers** (Julia's rating: A)
14. Insertion operators, such as `insert()` and `push_back()` are often surprisingly efficient on a vector.
17. A map is usually implemented as a red-black tree.
18. An unordered `_map` is a hash table.
19. Pass a container by reference and return a container by value (Julia: look into this advice more)
20. For a container, use the `()`-initializer syntax for sizes and the `{ }`-initializer syntas for elements.
21. Prefer compact and contiguous data structure (Julia's rating: A)
22. A list is relatively expensive to traverse.
23. Use unordered containers if you need fast lookup for large amounts of data.
24. User ordered associative containers (e.g., map and set) if you need to iterate over their elements in order
25. Use unordered containers for element types with no natural order.
28. Know your standard-library containers and prefer them to hand-crafted data structures.

When Julia works on leetcode algorithms question early in 2015, she found out that most talent programmers tend to use C++ to write solution. Now, she is determined to finish this book reading: a tour of C++, she likes to master C++ some day.

Further reading on the advice 13 (Dec. 12, 2015):

[2]<http://stackoverflow.com/questions/141337/c-stl-should-i-store-entire-objects-or-pointers-to-objects>

[3]<http://stackoverflow.com/questions/22146094/why-should-i-use-a-pointer-rather-than-the-object-itself?rq=1>  
(Julia enjoys reading the blog, she spends over 30 minutes on this blog.)

Notes from the above stackoverflow blogs:

Dynamic allocation

You need the object to outlive the current scope

You need to allocate a lot of memory

Pointers

You need reference semantics

You need polymorphism

You want to represent that an object is optional

You want to decouple compilation units to improve compilation time

You need to interface with a C library

Polymorphic behavior

Reference semantics and avoiding copying

Resource acquisition  
More fine-grained life-time control

[4]<http://stackoverflow.com/questions/79923/what-and-where-are-the-stack-and-heap>

1. <http://www.amazon.com/A-Tour-C-In-Depth/dp/0321958314>
  2. <http://stackoverflow.com/questions/141337/c-stl-should-i-store-entire-objects-or-pointers-to-objects>
  3. <http://stackoverflow.com/questions/22146094/why-should-i-use-a-pointer-rather-than-the-object-itself?rq=1>
  4. <http://stackoverflow.com/questions/79923/what-and-where-are-the-stack-and-heap>
- 

### **Booking reading: Mobile First (2015-12-13 00:07)**

Dec. 12, 2015

Plan to spend at least 10 hours to read the book before 2016: Mobile First, Luke Wroblewski.

[1][http://www.amazon.ca/MOBILE-FIRST-GUIDE-STRAT%C3%89GIQUE-DESIGN/dp/2212134061/ref=sr\\_1\\_1?ie=UTF8&qid=1449990236&sr=8-1&keywords=mobile+first](http://www.amazon.ca/MOBILE-FIRST-GUIDE-STRAT%C3%89GIQUE-DESIGN/dp/2212134061/ref=sr_1_1?ie=UTF8&qid=1449990236&sr=8-1&keywords=mobile+first)

Videos provided by Luke Wroblewski:

[2][https://www.youtube.com/watch?v=Y-FMTPsgy\\_Y](https://www.youtube.com/watch?v=Y-FMTPsgy_Y)

Discover Pinterest: Mobile Engineering and Design

[3]<https://www.youtube.com/watch?v=wzRyTmBxs7Y>

Airbnb Design Talk with Luke Wroblewski

[4]<https://www.youtube.com/watch?v=iYsOKXvoiVM>

UX How-To with Luke Wroblewski

[5]<https://www.youtube.com/watch?v=xAKnPtbfnfY&list=PLg-UKERBljNy2Yem3RJkYL1V70dpzkysC>

1. [http://www.amazon.ca/MOBILE-FIRST-GUIDE-STRAT%C3%89GIQUE-DESIGN/dp/2212134061/ref=sr\\_1\\_1?ie=UTF8&qid=1449990236&sr=8-1&keywords=mobile+first](http://www.amazon.ca/MOBILE-FIRST-GUIDE-STRAT%C3%89GIQUE-DESIGN/dp/2212134061/ref=sr_1_1?ie=UTF8&qid=1449990236&sr=8-1&keywords=mobile+first)
2. [https://www.youtube.com/watch?v=Y-FMTPsgy\\_Y](https://www.youtube.com/watch?v=Y-FMTPsgy_Y)
3. <https://www.youtube.com/watch?v=wzRyTmBxs7Y>

4. <https://www.youtube.com/watch?v=iYsOKXvoiVM>
  5. <https://www.youtube.com/watch?v=xAKnPtbFNFY&list=PLg-UKERBljNy2Yem3RJkYL1V70dpzkysC>
- 

## **Book reading: 97 thing every programmer should know (II) (2015-12-13 23:50)**

Dec. 13, 2015

Continue to review the book "97 thing every programmer should know"

"Make Interfaces Easy to Use Correctly and Hard to Use Incorrectly" by Scott Meyers.

And then, plan to spend time to watch the videos:

Scott Meyers – The Most Important Design Guideline

[1]<https://www.youtube.com/watch?v=5tg1ONG18H8>

[2][https://www.youtube.com/watch?v=smqT9Io\\_bKo](https://www.youtube.com/watch?v=smqT9Io_bKo)

CppCon 2014: Scott Meyers "Type Deduction and Why You Care" (Julia: study this topic, and then, more foundation knowledge about OO languages. A++ )

[3]<https://www.youtube.com/watch?v=wQxj20X-tIU>

Scott Meyers - Effective Modern C++ part 1 (6 videos)

[4]<https://www.youtube.com/watch?v=fhM24zs1MFA&list=PLmxXlAVb5hkyq5njldMEPYdOqTAQPLChR>

Modern C++: What You Need to Know (Julia: watch this video first! Excellent talks with good data about data structures - List, map, set etc. )

[5]<https://www.youtube.com/watch?v=1oHEYk6xuvQ>

Swift vs Modern C++ (Julia: this is fun; learn a new language in one hour, swift, and get ideas how languages are different in the design)

[6]<https://www.youtube.com/watch?v=VxevGjZ8RuE>

1. <https://www.youtube.com/watch?v=5tg1ONG18H8>
  2. [https://www.youtube.com/watch?v=smqT9Io\\_bKo](https://www.youtube.com/watch?v=smqT9Io_bKo)
  3. <https://www.youtube.com/watch?v=wQxj20X-tIU>
  4. <https://www.youtube.com/watch?v=fhM24zs1MFA&list=PLmxXlAVb5hkyq5njldMEPYdOqTAQPLChR>
  5. <https://www.youtube.com/watch?v=1oHEYk6xuvQ>
  6. <https://www.youtube.com/watch?v=VxevGjZ8RuE>
- 

## **OO principle - S.O.L.I.D., Single Responsibility, Open/ close principle drill (2015-12-15 20:36)**

Dec. 15, 2015

Review the video about the testing, and try to get more from this lecture:

"The Clean Code Talks – Inheritance, Polymorphism, & Testing"

[1]<https://www.youtube.com/watch?v=4F72VULWFvc>

action items:

1. put sample code in C #, and then, check in git;
2. write down key words, and easy for review and follow as rules.

Julia's sample code in C #:

Problem statement:  $1 + 2 * 3$ , how to implement in OO design (using C # language)

Three solutions are provided, naive one, better one, optimal solution to apply S.O. principles.

1. Represent this as a tree

+

/ \

1 \*

/ \

2 3

Most of people come out solution like the following:

Using conditionals

```
class Node {
```

```
    char operator;
```

```
    double value;
```

```
    Node left;
```

```
    Node right;
```

```
    double evaluate() {
```

```
        switch(operator) {
```

```

case '#': return value;

case '+': return left.evaluate() + right.evaluate();

case '*' return left.evaluate() * right.evaluate();

case ... // edit this for each new operator

}

}

}

```

Big problem on this:

graphic representation,

Node

op:char

value: double

left: Node

right:Node

-----

evaluate():double

Julia could not figure out the analysis here <- first time to memorize this analysis, and also try to learn reasoning

Analyzing attributes

# + \*

function yes yes

value yes

left yes yes

right yes yes

Two different behaviors are fighting here, (see the above table), not matching Single Responsibility Principle.

if you are the operation node, then you need your left and right child; whereas value node, you just need value.

C # code using conditional implementation, one class Node

[2]<https://github.com/jianminchen/OODesignPractice/blob/master/ConditionalVersion.cs>

[3][https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OODesign\\_CleanCodeTalk\\_ConditionalVersion.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OODesign_CleanCodeTalk_ConditionalVersion.cs)

2.

Let us break it up:

Node

-----

evaluate(): double

| |

ValueNode OpNode

value: double op: char

----- left: Node

evaluate: double right: Node

-----

evaluate(): double

As showing above, break Node into ValueNode and OpNode, so ValueNode does not have left and right child because of no meaning over there.

Tree looks like:

OpNode

+

/\

ValueNode OpNode

1 \*

/\

ValueNode ValueNode

2 3

Operations and values

```
abstract class Node {
```

```
    abstract double evaluate();
```

```
}
```

```
class ValueNode extends Node {
```

```
    double value;
```

```
    double evaluate() {
```

```
        return value;
```

```
}
```



```
}
```

```
class OpNode extends Node {
```

```
char operator;
```

```
Node left;
```

```
Node right;
```

```
double evaluate() {
```

```
switch(operator) {
```

```
case '+': return left.evaluate() + right.evaluate();
```

```
case '-': return left.evaluate() - right.evaluate();
```

```
case ... // edit this for each new operator
```

```
}
```

```
}
```

```
}
```

better solution: Node, OpNode, ValueNode:

[4][https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OO\\_Design\\_CleanCodeTalk\\_OpNode-ValueNode.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OO_Design_CleanCodeTalk_OpNode-ValueNode.cs)

3. How to extend this? Every time you add a new operator, need to hold on source code, and add code in switch statement, how to make it better?

OpNode divides into AdditionNode and MultiplicationNode

OpNode

-----

left: Node

right: Node

-----

evaluate(): double

AdditionNode MultiplicationNode

-----

evaluate(): double evaluate(): double

```
abstract class Node {
```

```
    abstract double evaluate();
```

```
}
```

```
class ValueNode extends Node {
```

```
    double value;
```

```
    double evaluate() {
```

```
        return value;
```

```
    }
```

```
}
```

```
class abstract OpNode extends Node {
```

```
    Node left;
```

```
    Node right;
```

```
    abstract evaluate();
```

```
}
```

Operation classes

```
class AdditionNode extends OpNode {
```

```
double evaluate() {
```

```
return left.evaluate() + right.evaluate();
```

```
}
```

```
}
```

```
class MultiplicationNode extends OpNode {
```

```
double evaluate() {
```

```
return left.evaluate() * right.evaluate();
```

```
}
```

```
}
```

Now, the new tree diagram:

AdditionalNode

+

/\

ValueNode MultiplicationNode

1 \*

/\

ValueNode ValueNode

optimal solution: Node, OpNode, ValueNode, AdditionNode, MultiplicationNode

[5][https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OODesign\\_CleanCodeTalk\\_OpNode-ValueNodeAndMore.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OODesign_CleanCodeTalk_OpNode-ValueNodeAndMore.cs)

Perfect examples of  $1 + 2 * 3$ , 3 implementations, Julia learns O of S.O.L.I.D. OO principles, open for extension, close for change.

More detail, the above optimal solution does not have any if statement, and any new arithmetic operation just needs a new class, no need to touch existing code: Node, OpNode, AdditionNode, MultiplicationNode. For example, minus '-' operation, just add MinusNode. For easy to demo, all classes are in one .cs file, but each class should have it's own .cs file. :-)

Share the learning of Open/Close principle - great examples. Cannot wait to move on to next one, Liskov substitution principle!

Reference:

Blog:

[6]<http://juliachencoding.blogspot.ca/2015/11/the-clean-code-talks.html>

1. <https://www.youtube.com/watch?v=4F72VULWFvc>

2. <https://github.com/jianminchen/OODesignPractice/blob/master/ConditionalVersion.cs>

3. [https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OODesign\\_CleanCodeTalk\\_ConditionalVersion.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OODesign_CleanCodeTalk_ConditionalVersion.cs)

4. [https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OODesign\\_CleanCodeTalk\\_OpNodeValueNode.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OODesign_CleanCodeTalk_OpNodeValueNode.cs)

5. [https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OODesign\\_CleanCodeTalk\\_OpNodeValueNodeAndMore.cs](https://github.com/jianminchen/OODesignPractice-1-2-3-B/blob/master/OODesign_CleanCodeTalk_OpNodeValueNodeAndMore.cs)

6. <http://juliachencoding.blogspot.ca/2015/11/the-clean-code-talks.html>

## Book Reading: 97 Things Every Programmer Should Know (III) (2015-12-17 19:01)

Dec. 17, 2015

Today Julia reviewed the topic of "Continuous Learning", written by Clint Shank.

Here is the chapter web link:

[1][http://programmer.97things.oreilly.com/wiki/index.php/Clint\\_Shank](http://programmer.97things.oreilly.com/wiki/index.php/Clint_Shank)

[2][http://programmer.97things.oreilly.com/wiki/index.php/Continuous\\_Learning](http://programmer.97things.oreilly.com/wiki/index.php/Continuous_Learning)

Continuous learning is such a great topic, and Julia likes to write down some notes:

1. Write some code.
2. Get to know the frameworks and libraries you use. (Julia's comment: learn more about Asp.net framework)
3. A really good way to learn something is to teach or speak about it.
4. L

earn a new programming language every year.

Julia's ideas:

1. Continuous learning should also be creative. Should involve some code writing, algorithm problem solving, and good documentation - blogs, and good spirit - sharing through the blogs etc.
2. The learning should also be full of challenges. Go back to school is great, but learn to love what you do, learn by yourself, prepare your own curriculum - data structure, foundation core, design principles/coding standards, top conferences, outstanding lecturers top of world - interested area. Keep yourself motivated, determined. You can work on it over 16 hours a day if you love to be a programmer/research/IT worker. Cherish time and limited resource, and find top-quality material to work on.

For example, Julia found out that CppConf is a great conference with great talks. She just found out that in Nov. 2015, and started her own study on C++ again, and then, she was amazed to find out that those materials she works on are best of quality and surprisingly rewarding, and she also knew a lot of top engineers and top scholars through videos, and she learns great things about design etc.

Julia subscribed code school and pluralsight.com to study a lot of quality courses to catch up technologies starting Dec. 2015.

3. Learn to master programming languages, but also, go for beyond basics, such as, OO principles - S.O.L.I.D., C++ coding standards, when you apply those principles, code standards, save time to develop/ debug / maintain the software.

1. [http://programmer.97things.oreilly.com/wiki/index.php/Clint\\_Shank](http://programmer.97things.oreilly.com/wiki/index.php/Clint_Shank)

2. [http://programmer.97things.oreilly.com/wiki/index.php/Continuous\\_Learning](http://programmer.97things.oreilly.com/wiki/index.php/Continuous_Learning)

---

## **Book Reading: 97 Things Programmer Should Know (IV) (2015-12-17 21:58)**

Dec. 17, 2015

Review my favorite book chapter:

Wet Dilutes Performance Bottlenecks - Kirk Pepperdine

Julia's notes:

Don't Repeat Yourself - it codifies the idea that every piece of knowledge in a system should have a singular representation. The antithesis of DRY is WET (Write Every Time).

Favorite use case of violating DRY principle - use of collections

Action item:

1. Put similar code example in Github, and get the idea of DRY violation analysis

2. Read "Java Performance Tuning" Book - 4 hours in December, 2015

[1][https://github.com/jianminchen/DRY\\_Principle\\_Examples/blob/master/DRYPrinciple\\_Collections.cs](https://github.com/jianminchen/DRY_Principle_Examples/blob/master/DRYPrinciple_Collections.cs)

Julia likes to review the JVM - Java virtual machine, performance tuning again; the programmer should care about performance tuning.

Watch the video:

Kirk Pepperdine — The (not so) Dark Art of Performance Tuning

[2]<https://www.youtube.com/watch?v=5XWgjSHZlQw>

Building and Tuning High Performance Java Platforms (Julia's comment: Vmware, only watch first 40 minutes)

[3]<https://www.youtube.com/watch?v=IGS-rqSjmFk>

Writing Quick Code in C++, Quickly

[4]<https://www.youtube.com/watch?v=ea5DiCg8HOY&list=PLGvfHSgImk4ZbhoiE6OXQREiLrHV9FPJH>

1. [https://github.com/jianminchen/DRY\\_Principle\\_Examples/blob/master/DRYPrinciple\\_Collections.cs](https://github.com/jianminchen/DRY_Principle_Examples/blob/master/DRYPrinciple_Collections.cs)
  2. <https://www.youtube.com/watch?v=5XWgjSHZlQw>
  3. <https://www.youtube.com/watch?v=IGS-rqSjmFk>
  4. <https://www.youtube.com/watch?v=ea5DiCg8HOY&list=PLGvfHSgImk4ZbhoiE6OXQREiLrHV9FPJH>
- 

## Book Reading: Effective Java (2015-12-18 00:19)

Dec. 17, 2015

Effective Java is such a great book, I had wonderful time to read 1-2 hours while enjoying the travel on train back to my home town this October, 2015.

But, Julia had to motivate her more to complete reading the book - at least spend over 10 hours this holiday break to catch up some best ideas about programming.

Dec. 17, 2015 (20 minutes reading)

Item 52: Refer to objects by their interfaces (Julia's rating 1-10: 10)

Dec. 18, 2015 (2 hours reading 8:00pm-10:00pm)

Item 23: Don't use raw types in new code

Item 24: Eliminate unchecked warnings

Item 25: Prefer lists to arrays

Item 40: Design method signatures carefully

Item 45: Minimize the scope of local variables

Item 46: Prefer for-each loops to traditional for loops

Dec. 19, 2015 (2 hours reading 8:30am-10:30am)

Item 6: Eliminate obsolete object reference (read the item twice)

Actions: take sample code, and think about similar example, write a one in C #, check in github

Notes:

3 cases for memory leaks:

1. whenever a class manages its own memory,
2. caches - WeakHashMap
3. listeners and other callbacks

Careful code inspection or with the aid of a debugging tool known as a heap profiler

Action item 2: Try heap profiler in C #

Item 8: Obey the general contract when overriding equals (read again, 30 minutes)

Further reading:

WeakHashMap

[1]<http://www.ibm.com/developerworks/library/j-jtp11225/>

Read some code, and then, go back to the book.

[2]<http://www.codeproject.com/Articles/595160/Understand-Liskov-Substitution-Principle-LSP>

[3]<http://www.codeproject.com/Articles/648987/Violating-Liskov-Substitution-Principle-LSP>

[4]<http://www.codeproject.com/Articles/597870/Liskov-Substitution>

1. <http://www.ibm.com/developerworks/library/j-jtp11225/>

2. <http://www.codeproject.com/Articles/595160/Understand-Liskov-Substitution-Principle-LSP>

3. <http://www.codeproject.com/Articles/648987/Violating-Liskov-Substitution-Principle-LSP>

4. <http://www.codeproject.com/Articles/597870/Liskov-Substitution>

---

## **Book reading: Head first Design pattern (2015-12-20 21:45)**

Dec. 20, 2015

Start to read the book "Head first design pattern" - 600 pages book. Plan to spend 10 hours to read the book first.

Head first series books are my favorite ones. This time, I will document how I learn, what my favorite parts through the book. Encourage myself reading more of this book, I like to enforce myself to read first 200 pages first.

Dec. 20, 2 hours

Read page 1 - Page 38, reviewed strategy pattern, example:

client:

duck

Encapsulated fly behavior

FlyBehavior Interface -> 1. FlyWithWings class 2. FlyNoWay class

Encapsulated quack behavior

QuackBehavior -> 1. Quack class 2. Squeak class 3. MuteQuack class

Duck has-a feature, not is-a since HAS-A can be better than IS-A

Design Principle: Favor composition over inheritance

So, Duck client class is designed:

Duck

— — —

```
FlyBehavior flyBehavior
QuackBehavior quackBehavior
-
setFlyBehavior()
setQuackBehavior()
```

Dec. 27, 2015

Spent 2 hours to read the book, went through the 300 pages quickly.

**Motivation to read more pages of the book** (Dec. 28, 2015):

1. The book is well written, Julia, you will not forget the examples in each pattern.
2. The reading takes your some time, but later, when you develop the software, you will save the time.
3. Just relax, and have some reading, you will not get lost, cannot understand, or get bored easily, the learning is fun.
4. If you cannot understand the book, then, you will have a lot of trouble down the road. OO design is a must skill to have your career as a programmer.
5. Find some videos - courses in pluralsight, watch first, and then, get other people's help first; come back to read the book slowly.

[1][https://www.youtube.com/watch?v=acjvKJiOvXw&feature=share&fb\\_ref=share](https://www.youtube.com/watch?v=acjvKJiOvXw&feature=share&fb_ref=share)

1. [https://www.youtube.com/watch?v=acjvKJiOvXw&feature=share&fb\\_ref=share](https://www.youtube.com/watch?v=acjvKJiOvXw&feature=share&fb_ref=share)

---

## OO principles - The Open / Closed Principle (2015-12-22 22:58)

Dec. 22, 2015

Three Approaches to Achieve OCP

1. Parameters (Procedural Programming)
2. Inheritance / Template Method Pattern
3. Composition / Strategy Pattern

### 1. Parameters (Procedure Programming)

Allow client to control behavior specifics via a parameter  
Combined with delegates/lambad, can be very powerful approach

### 2. Inheritance / Template Method Pattern

Child types override behavior of a base class (or interface)

### 3. Composition / Strategy Pattern

Client code depends on abstraction  
Provides a "plug in" model



Implementations utilize inheritance; Client utilizes Composition

Read the blog:

[1]<http://code.tutsplus.com/tutorials/solid-part-2-the-open-closed-principle-net-36600>

[2]<http://www.objectmentor.com/resources/articles/ocp.pdf>

[3]<http://stackoverflow.com/questions/59016/the-open-closed-principle>

A friend told me that he subscribes

[4]<https://www.pluralsight.com/>

So, Julia subscribed code school first starting this Dec. 2015, and later pluralsight.com. Two schools are different.

Video watch:

[5][https://www.youtube.com/watch?v=MvyG\\_ODng3w](https://www.youtube.com/watch?v=MvyG_ODng3w)

Code School Live: JavaScript Best Practices Q &A

[6]<https://www.youtube.com/watch?v=k9xwJoprEMY>

Principles of Object Oriented Design (2 - The Open-Closed Principle) 28 minutes

[7][https://www.youtube.com/watch?v=qP6MroshYvM&index=8&list=PLD4GnSXHpKQ2\\_eKG5fKzsXs5hYcEsft7](https://www.youtube.com/watch?v=qP6MroshYvM&index=8&list=PLD4GnSXHpKQ2_eKG5fKzsXs5hYcEsft7)

I Learned HTML and CSS, Now What?

[8]<http://blog.codeschool.io/2014/09/30/learned-html-css-now/>

[9]<http://support.pluralsight.com/knowledgebase/articles/491274-what-s-offered-at-code-school-vs-pluralsight>

1. <http://code.tutsplus.com/tutorials/solid-part-2-the-open-closed-principle-net-36600>

2. <http://www.objectmentor.com/resources/articles/ocp.pdf>

3. <http://stackoverflow.com/questions/59016/the-open-closed-principle>

4. <https://www.pluralsight.com/>

5. [https://www.youtube.com/watch?v=MvyG\\_ODng3w](https://www.youtube.com/watch?v=MvyG_ODng3w)

6. <https://www.youtube.com/watch?v=k9xwJoprEMY>

7. [https://www.youtube.com/watch?v=qP6MroshYvM&index=8&list=PLD4GnSXHpKQ2\\_eKG5fKzsXs5hYcEsft7](https://www.youtube.com/watch?v=qP6MroshYvM&index=8&list=PLD4GnSXHpKQ2_eKG5fKzsXs5hYcEsft7)

8. <http://blog.codeschool.io/2014/09/30/learned-html-css-now/>

9. <http://support.pluralsight.com/knowledgebase/articles/491274-what-s-offered-at-code-school-vs-pluralsight>

---

**Code School: courses study (2015-12-27 15:10)**

Dec. 28, 2015

Spent hours (Dec. 23 - 3 hours, Dec. 27 - 3 hours) to watch courses on code school from Dec. 22, 2015 .  
Website: [1]<https://www.codeschool.com/>

## Path HTML/CSS

### Get Started With HTML and CSS

Course: Front-end Foundations

Course: Front-end Foundations

### Intermediate CSS

course: CSS Cross-Country

course: Journey into Mobile

course: Adventures in Web Animations

### CSS Preprocessors

course: Assembling Sass

course: Assembling Sass Part 2

### CSS Frameworks

course: Blasting Off with Bootstrap

### Design

course: Fundamentals of Design

## Paths: JavaScript

### Client-side Frameworks

course: Shaping up with Angular.js

course: Staying Sharp with Angular.js

course: Warming Up With Ember.js

More review:

## Google I/O 2013 - Design Decisions in AngularJS

[2][https://www.youtube.com/watch?v=HCR7i5F5L8c&feature=share&fb\\_ref=share](https://www.youtube.com/watch?v=HCR7i5F5L8c&feature=share&fb_ref=share)

1. <https://www.codeschool.com/>

2. [https://www.youtube.com/watch?v=HCR7i5F5L8c&feature=share&fb\\_ref=share](https://www.youtube.com/watch?v=HCR7i5F5L8c&feature=share&fb_ref=share)

## Think Fast, Talk Smart: Communication Techniques (2015-12-27 15:26)

Dec. 27, 2015

Take a break, and watch the video:

Think Fast, Talk Smart: Communication Techniques

[1]<https://www.youtube.com/watch?v=HANw168huqA>

Take some notes:

[2]<http://www.singjupost.com/think-fast-talk-smart-communication-techniques-by-matt-abrahams-full-transcript/>

Techniques:

1. Greet anxieties - it is normal, and then, take deep breathe, and think that it is a conversation, not performance. There is no a right way to do presentation.

2. First, start with a question.

Count how many 'f', and answer the questions.

3. Write down a series of questions to answer. Instead of a lot of bullets notes.

4. Use conversational language.

Bring ourselves into present moment, instead of future consequences

My favorite way to get present-oriented is to say tongue twisters.

Repeat after me. It's only three phrases. "I slit a sheet. A sheet I slit. On that sheet I sit. "Focus on saying right, in the presence moment.

"I slit a sheet. A sheet I slit. And on that slitted sheet I sit". Very good, no shifts. Excellent. Very good.

the first two steps of our process. First we get out of our own way, and we can reframe the situation as an opportunity.

The next phase is also hard, but very rewarding, and that is to slow down, and listen .

The first useful structure: problem-solution-benefit structure

Another structure: what? So what? Now what? Structure.

Julia's favorite notes:

"So what does this all mean? It means that we have, within our ability, the tools and the approaches, to help us in spontaneous speaking situations. The very first thing we have to do is manage our anxiety, because you can't be an

effective speaker if you don't have your anxiety under control. And we talked about how you can do that by greeting your anxiety, reframing as a conversation, and being in the present moment.

Once you do that, you need to practice a series of four steps, that will help you speak spontaneously. First you get out of your own way. I would love it if all of you, on your way from here to the football game, point at things and call them the wrong name. It'll be fun. If most of us do it, then it won't be weird. If only one and two of us do it, it'll be weird. Right.

Second. Give gifts. By that I mean see your interactions as ones of opportunity, not challenges.

Third, take the time to listen, listen. And then finally, use structures. And you have to practice these structures. I practice these structures on my kids. I have two kids. When they ask me questions, I usually answer them in what, so what, now what. "

Book: Speaking Up Without Freaking Out

More videos:

1.

[3]<https://www.youtube.com/watch?v=5naThX63pF0>

2.

Matt Abrahams Workshop Compelling and Confident Communication 1

[4]<https://www.youtube.com/watch?v=ENLZfjLoPEY>

Notes:

Knowledge

Domain and Audience knowledge means you...

Speak to benefits instead of features.

What do they know?

What do they expect?

What are their area of concern?

Relevant to audience - Reveal the relevance - 25 % of all the world money, into the bank, for example

Advance Analogies - Buy you a lot, link thing they already know

Invoke Imagination - Rember an event in the past, or in the future

Use "You" - use conversational language, engage them to what they know, using inclusive language

Kevin O'Leary Keynote at Notre Dame

[5]<https://www.youtube.com/watch?v=XBUQNYczj4A>

1. <https://www.youtube.com/watch?v=HAnw168huqA>
  2. <http://www.singjupost.com/think-fast-talk-smart-communication-techniques-by-matt-abrahams-full-transcript/>
  3. <https://www.youtube.com/watch?v=5naThX63pF0>
  4. <https://www.youtube.com/watch?v=ENLZfjLoPEY>
  5. <https://www.youtube.com/watch?v=XBUQNYczj4A>
- 

## **Vacation break in 2015 - from Dec. 28 to 31, 2015 (2015-12-29 23:20)**

Dec. 29, 2015

Try to have a good vacation. So, plan to learn skiing this winter, go out to do sports. Swimming, tennis, skiing, and all other activities.

Julia went out the Lonsdale Quay on Dec. 29 to catch a free shuttle to mountain Seymour, but she missed the shuttle. She had great time to enjoy beauty of city of Vancouver. She took the skytrain, seabus, and walked in the city.

Then, she decided to learn something about real estate, money, and Julia likes the teaching from Canadian business guru Kevin O'Leary.

Here are the list of videos watched:

[1]<http://www.chatelaine.com/living/budgeting/why-kevin-oleary-doesnt-plan-to-leave-any-money-to-his-kids/>

The Hour: Kevin O'Leary

[2][https://www.youtube.com/watch?v=OTZ\\_\\_3iuQU4](https://www.youtube.com/watch?v=OTZ__3iuQU4)

Kevin O'Leary on Occupy Wall Street

[3]<https://www.youtube.com/watch?v=JR6aCl7cZyg>

Kevin O'Leary on men, women, and money

[4]<https://www.youtube.com/watch?v=PkbWBDeLrJ4>

Kevin O'Leary's 'Cold, Hard, Truth' on Gold Investing

[5]<https://www.youtube.com/watch?v=aZOT6RwJYPc>

O'Leary: No cracks in Canada's housing, condos show insatiable demand

[6]<https://www.youtube.com/watch?v=9etKaSeBpxg>

Kevin O'Leary Real Estate Investment Advice Canadian TV

[7]<https://www.youtube.com/watch?v=35wZ-dda1a0>

Kevin O'Leary's Top 10 Rules For Success (Julia's favorite: No. 6, No. 8, No. 4)

[8]<https://www.youtube.com/watch?v=ItMr7vxvekU>

10 rules -

1. You have to sacrifice a lot
2. Trust your gut
3. Have diversification
4. Have a backup plan
5. Be a leader
6. Admit your weakness
7. Buy stocks with dividends
8. Differentiate between family, friends & money
9. Get outside of North America
10. Go with simple ideas

Kevin O'Leary on how to get ahead in the workplace

[9]<https://www.youtube.com/watch?v=bG6L5z7fMxE>

1. How to calculate the worth to the company?

Answer from Kevin:

Can you get the job done?

How fast can you get the job done?

How much ripple effect you create?

Can you work with others? Do you create hassle with other employees? These are matrix employer uses.

2. Can you brag your accomplishment to the superior?

Answer from Kevin:

1. What keep you employed is to keep your boss looks great. Pass your manager to brag yourself, it is like passing a car in free way.

2. Got to be a politician. When to take, when to give, when to say, when not to say.

Make your boss looks good. Help boss achieve his goal.

3. Go to work to make money, not make friendship

4. Code ethics - Dressing, people does not notice - people do not remember what you wear yesterday

KEVIN O'LEARY How To Nail a Job Interview ADVICE

[10]<https://www.youtube.com/watch?v=VvVUvWwUrzl>

1. Do not cut people off, let people finish talk; answer the question.

2. Do not be anxious. Be confident. You are in the interview, supposing that you will get the job. Look them in the eyes when you are asked a question.

3. What is weakness? Do not know what you need from me. I will work hard to figure out.

## Kevin O'Leary Keynote at Notre Dame

[11]<https://www.youtube.com/watch?v=XBUQNYczj4A>

What makes a great entrepreneur?

1. Prepared to make life/balance sacrifices.
2. Have a little knowledge about everything, but a lot about what they are selling.
3. Put shareholders first.
4. Love what they sell.
5. Use technology to make work more efficient.
6. Understand the business is a global competition.

1. <http://www.chatelaine.com/living/budgeting/why-kevin-oleary-doesnt-plan-to-leave-any-money-to-his-kids/>
  2. [https://www.youtube.com/watch?v=0TZ\\_\\_3iuQU4](https://www.youtube.com/watch?v=0TZ__3iuQU4)
  3. <https://www.youtube.com/watch?v=JR6aC17cZyg>
  4. <https://www.youtube.com/watch?v=PkbWBDeLrJ4>
  5. <https://www.youtube.com/watch?v=aZOT6RwJYPc>
  6. <https://www.youtube.com/watch?v=9etKaSeBpxg>
  7. <https://www.youtube.com/watch?v=35wZ-dda1a0>
  8. <https://www.youtube.com/watch?v=ItMr7vxvekU>
  9. <https://www.youtube.com/watch?v=bG6L5z7fMxE>
  10. <https://www.youtube.com/watch?v=VvVUvWwUrzI>
  11. <https://www.youtube.com/watch?v=XBUQNYczj4A>
-





## 2. 2016

### 2.1 January

#### Favorites of top 10 rules for success (2016-01-01 01:18)

Dec. 31, 2015

Stayed up to last minute of 2015, 12:00am, watching a few of videos about top 10 rules for success, and then, think about putting together a blog to share, for a successful year 2016.

Top 10 rules for success

1. Jessica Alba

2. Oprah Winfrey

**No. 5, Run the race as hard as you can**

Julia's favorite, do not look at others, things you cannot control; focus on yourself  
25 years talk show, hundreds of talk show failed.

9. Stay grounded

3. Bill Gates

**No. 7 Ask for advice**

Blind spot concern

**No. 8 Pick good people**

**No. 9 Don't procrastinate**

School, last 2 days study for exam

But in business world, it is not good.

**No. 10 Have a sense of humor**

Warren Buffett

No. 5 Have a margin of safety

No. 7 Schedule for your personality

Jack Ma

No 1. Get used to rejection

No. 9 Don't complain, look for opportunities

No. 2 Keep your dream alive

No. 3 Focus on culture

Not just money, to help others

No. 5 Get inspired

No. 7 Have a good name

No. 8 customers are #1

Share holders will leave quickly if there is bad thing happens

Jeff Beco

No. 1 Have no regrets

80 years old, think about things you try in young age. Internet, if I fail, no regret.

No.2 Follow your heart not your head

No. 3 Invest more in the product then marketing

No. 4 Pick a good name

No. 5 Stand for something

No. 6 Focus on the customer

No. 7 Focus on your passion

No. 8 Build a culture

No. 9 premium products at non-premium prices

Kindle, product - no profit, but user uses it, make money

No. 10 take a risk

Larry Page

No. 2 Don't be afraid of failure

work on 10 years

leadership training - fail fast

Take some risk, encourage to take more risk

No. 3 Stay organized

No. 4 **Concentrate on the long term**

tooth brush test ( twice a day)

gmail/ youtube

double the revenue on youtube, since a lot of people use it. But at the beginning, spend a lot

No. 5 Have a good idea

No. 6 Solve bigger problems

No. 7 Take on challenges

No. 8 **Don't settle**

other people do not tell

Be an expert in all areas

Search algorithm, spent 3-4 years on it, before Google has over 100 people work on

No. 9 Adapt to changes

No. 10 **Follow your dreams**

anxiety -> dream

about school, computer error about admission etc. /irrational worry, then anxiety, had a dream:

download entire web to a small computer lying around ->

after dream, stay up a few of hours, work on algorithms ->

set up a few weeks, thought at the beginning vs 1-2 years work ->

keep links -> not all web pages -> rank of pages -> google

Share the story of his personal dream -> developed a top of world company and technology - Google

Michale Jordan

No. 1 Keep working hard

No. 2 Ignite the fire

nothing can stop me

No. 3 Be different

No. 4 **Fail your way to success**

Fail over and over and over again.

No. 5 Have high expectations

No. 6 Be positive

bad thing/good thing mixes, no such thing called bad thing

No. 7 Be who you were born to be

Encourage to fail  
Take everything you are given,  
Not about shoes, it is about where you choose to go  
No. 8 Have a vision  
Mentally strong  
No. 9 Stop make excuses  
No. 10 **Practice** (no need to think when in the game, situation met before)  
At that moment, you do not have to think

Kobe Bryant  
No. 1 Make sacrifices  
No. 2 **One move at a time**  
How do I prepare? One piece a time, start from a small step  
No. 3 Trust your skills  
how to face criticism? Pressure?  
No. 4 Use your scars as a weapon  
open up to them  
No. 5, Focus on each day  
Do not think about the final result, focus on each day -> training, conditioning, ...  
No. 6 **Don't be afraid confrontation**  
Cannot please everyone  
Even hurt your feelings, tell something: small thing, like food in your teeth  
No. 7 Be competitive when it's hard  
No. 8 **Just keep going**  
career -> not chase championships  
good/bad moments, things will work out  
No. 9 **Thrive on being an outsider**  
No. 10 **Compete with yourself**  
a story about practice, 800 times, young but mature, weight training, conditioning

January 2, 2015

Vinod Khosla's Top 10 Rules For Success (22 minutes, co-founded Sun Microsystems from 1982 to 1984)

[1][https://www.youtube.com/watch?v=MsGgZp\\_PAzM](https://www.youtube.com/watch?v=MsGgZp_PAzM)

- No. 1. Be persistent
- No. 2. Keep innovating
- No. 3. Add value
- No. 4. Have the guts to follow your beliefs
- No. 5 Try and fail, but don't fail to try
- No. 6 Transcend what's traditional
- No. 7. Shake things up
- No. 8. Build a great team
- No. 9. Dare to be great
- No. 10. Be brutally honest

Facebook Documentary - Sheryl Sandberg's Top 10 Rules For Success

[2]<https://www.youtube.com/watch?v=qKRB8n6PULQ>

No. 1 Have impact

No. 2 Think big

No. 3 **Go for growth**

she joined google when there were only 200 people.

She worked for a person 23 years old.

Care less about level, but about growth.

No. 4 **Communicate authentically**

There is no truth. Your truth, my truth. It is objective.

Walk in room, say what you believe. Give people room to authenticate, give out their thoughts.

No. 5 **Hire big**

Hire for then, not just now. Then happens quickly.

**Hire overqualified, hire new graduate.**

No. 6 Don't just talk, really listen!

No. 7 Take responsibility

No one has complete control.

Learn to take responsibility. Late for work, it is not for traffic. But you do not consider the traffic issue.

No. 8 **Measure results, not face time**

Focus on result, not face time. But if you focus on face time, you reward face time.

People work hard, but may not focus on result.

No. 9 Find something you really believe in

No. 10 Careers are not ladders, but jungle gyms

Jeff Weiner's Top 10 Rules For Success

[3]<https://www.youtube.com/watch?v=do9jIFIFj08&list=PLiZj-lk9MmM0VWRGYCfuUCdyhKfU733WX&index=102>

Reference videos:

Michael Jordan's Top 10 Rules For Success

[4] <https://www.youtube.com/watch?v=NidqtkXq9Yg>

Kobe Bryant's Top 10 Rules For Success

[5][https://www.youtube.com/watch?v=5Nd\\_nUhUj2M](https://www.youtube.com/watch?v=5Nd_nUhUj2M)

More watching: ( January 2, 2015 )

Vinod Khosla: Failure does not matter. Success matters.

[6]<https://www.youtube.com/watch?v=HZcXup7p5-8>

Note to laugh: Strategy to get greencard: Convince them, otherwise, confuse them.

Notes: come out your priority, and then measure

in 1990s, 25 dinners with kids /month, try to achieve them; means, no sports out with friends in dinner time.

January 21, 2016 (20 minutes talk - after 10 rules, there are 8 minutes talk about her projects at Google, nice sharing)

Marissa Mayer's Top 10 Rules For Success

[7]<https://www.youtube.com/watch?v=02zYZ-JB2zQ>

[8]<http://www.businessinsider.com/google-cfo-ruth-porat-success-secrets-2015-12>

January 31, 2016

Tony Hsieh

[9][https://www.youtube.com/watch?v=49kJE4HJb\\_w](https://www.youtube.com/watch?v=49kJE4HJb_w)

1. Chase the vision, not the money (

things like to do, and does not make a dime in 10 years / and let the money follows;

do not chase paper, chase dream

Entrepreneurs:

What would you be passionate about doing for 10 years even if you never made a dime?

)

2. Hire the right people (

technical: meet the bar, but, hang out with? if the answer is no, then ...

work life balance/ so much time at work, better enjoy it

- good one/ medium one: strong culture - 10 core values - does not matter what value you are

- no judgement of values - but you have some values - figure out what values you are

)

3. Let your customers do marketing for you. (buy exposure /ads etc., make customer happy - word of mouth)

4. Think long term

5. Do thought experiments

6. Form close bond with your coworkers

7. Have self-confidence

8. Celebrate each person's individuality

9. Maximize customer experience (360 day return policy, contact information - opposite information, low technology, unsexy - phone calls)

10. Just be happy (lose everything, still be happy)

1. [https://www.youtube.com/watch?v=MsgGZp\\_PAzM](https://www.youtube.com/watch?v=MsgGZp_PAzM)

2. <https://www.youtube.com/watch?v=qKRB8n6PULQ>

3. <https://www.youtube.com/watch?v=do9jIFIFj08&list=PLiZj-Ik9MmMOVWRGYCfuUCdyhKfU733WX&index=102>

4. <https://www.youtube.com/watch?v=NidqtkXq9Yg>

5. [https://www.youtube.com/watch?v=5Nd\\_nUhUj2M](https://www.youtube.com/watch?v=5Nd_nUhUj2M)

6. <https://www.youtube.com/watch?v=HZcXup7p5-8>

7. <https://www.youtube.com/watch?v=02zYZ-JB2zQ>

8. <http://www.businessinsider.com/google-cfo-ruth-porat-success-secrets-2015-12>

9. [https://www.youtube.com/watch?v=49kJE4HJb\\_w](https://www.youtube.com/watch?v=49kJE4HJb_w)

---

## Divide and Conquer: Preorder traversal of ternary tree (2016-01-06 21:49)

January 6, 2016

??????

??????

,

????????

,

????

,

??

!

???

?, ???.

Be humble, learn from her mistakes in 2015. Julia is learning how to solve "divide and conquer" - write a perfect recursive function - without bug in base case - recursive calls.

Her lessons for recursive function design, divide and conquer solution in 2014, 2015:

1. **Do the work for root node, but do not do the work for its children** ; In other words, **a subproblem can be handled by a recursive call** .

2. **Do not repeat base case 'if' clause - a checking afterwards in the recursive function call** . Not necessary, except trying to use less stack.

Julia, recursive function design is like dealing with a family - single parent with children; show the work how to handle root node, that is almost done. Let recursive function to take care of children, do not do the work for children nodes. For example, binary tree, 2 children, call recursive function twice; Ternary tree, 3 children, and then, use 3 recursive calls.

Once again, "

**Do not do the work for children directly**

".

Action items:

1. Work on Leetcode questions again and start with simple questions.

Binary Tree Preorder traversal of ternary tree

[1] <https://github.com/jianminchen/TreeAlgorithms/blob/master/ternaryTreeTraversal.cs>

Binary Tree Inorder traversal:

[2]

<https://github.com/jianminchen/TreeAlgorithms/blob/master/TreeInorderTraversal.cs>

Binary Tree Post order traversal

[3]

<https://github.com/jianminchen/TreeAlgorithms/blob/master/TreePostOrderTraversal.cs>

Preorder traversal

[4]<https://github.com/jianminchen/TreeAlgorithms/blob/master/TreePreOrderTraversalB.cs>

More reading about ternary tree:

[5] [https://en.wikipedia.org/wiki/Ternary\\_search\\_tree](https://en.wikipedia.org/wiki/Ternary_search_tree)

[6] [https://en.wikipedia.org/wiki/Divide\\_and\\_conquer\\_algorithms](https://en.wikipedia.org/wiki/Divide_and_conquer_algorithms)

[7] <http://www.drdobbs.com/database/ternary-search-trees/184410528>

1. <https://github.com/jianminchen/TreeAlgorithms/blob/master/ternaryTreeTraversal.cs>

2. <https://github.com/jianminchen/TreeAlgorithms/blob/master/TreeInorderTraversal.cs>

3. <https://github.com/jianminchen/TreeAlgorithms/blob/master/TreePostOrderTraversal.cs>

4. <https://github.com/jianminchen/TreeAlgorithms/blob/master/TreePreOrderTraversalB.cs>

5. [https://en.wikipedia.org/wiki/Ternary\\_search\\_tree](https://en.wikipedia.org/wiki/Ternary_search_tree)

6. [https://en.wikipedia.org/wiki/Divide\\_and\\_conquer\\_algorithms](https://en.wikipedia.org/wiki/Divide_and_conquer_algorithms)

7. <http://www.drdobbs.com/database/ternary-search-trees/184410528>

---

## Bootstrap - CSS framework (2016-01-08 15:34)

January 8, 2016

Bootstrap - plan to spend 40 hours to read the material and some hands on experience.

[1]<http://getbootstrap.com/css/>

February 23, 2016

[2]<http://stackoverflow.com/questions/20007610/bootstrap-3-carousel-multiple-frames-at-once>

[3]<http://jsfiddle.net/paulalexandru/at606jpe/>

[4]<http://codepen.io/mephysto/pen/ZYVKRY>

Multiple frames in one slide, move multiple slides as well

[5][#">http://www.bootply.com/89193 #](http://www.bootply.com/89193)

bootstrap Slider / carousel with thumbnail navigation

[6]<http://www.bootply.com/XiWUwjbGtB>

1. <http://getbootstrap.com/css/>

2. <http://stackoverflow.com/questions/20007610/bootstrap-3-carousel-multiple-frames-at-once>

3. <http://jsfiddle.net/paulalexandru/at606jpe/>

4. <http://codepen.io/mephysto/pen/ZYVKRY>
  5. <http://www.bootply.com/89193>
  6. <http://www.bootply.com/XiWUwjBtB>
- 

## Leetcode algorithm 160-170 study (2016-01-10 12:29)

January 10, 2016

Write a lot of code, that is the only way to improve and become a better programmer. Julia likes to read blogs about Leetcode algorithm problems, and then, learn from others.

????, ??????????. ?????.

11:30am - 1:30am

Leetcode questions 161 - 170

August 22, 2015

### 161 One Edit distance

[1]<http://www.danielbit.com/blog/puzzle/leetcode/leetcode-one-edit-distance>

[2]<http://www.meetqun.com/thread-2854-1-1.html>

### 162 Find Peak Element

[3]<http://juliachencoding.blogspot.ca/2015/06/leetcode-question-find-peak-element.html>

need to write first C # implementation for this problem.

[4][https://siddontang.gitbooks.io/leetcode-solution/content/array/find\\_peak\\_element.html](https://siddontang.gitbooks.io/leetcode-solution/content/array/find_peak_element.html)

[5]<http://www.cnblogs.com/ganganloveu/p/4147655.html>

### 163 Missing Ranges

[6]<http://www.danielbit.com/blog/puzzle/leetcode/leetcode-missing-ranges>

### 164 Maximum Gap

Very clear explanation - an example to demo the algorithm (Julia: write a C # version)

[7]<http://easyleetcode.blogspot.ca/2015/01/leetcode-164-maximum-gap.html>

bucket sort

[8]<https://leetcode.com/discuss/18499/bucket-sort-java-solution-with-explanation-o-time-and-space>

radix sort

[9]<http://yucoding.blogspot.ca/2014/12/leetcode-question-maximum-gap.html>

[10]<http://bookshadow.com/weblog/2014/12/14/leetcode-maximum-gap/>

Julia, please practice this C++ code, and get some idea how advanced C++ works. (January 10, 2016)



[11]<http://ask.julyedu.com/question/213>

165 Compare Version Numbers

166 Fraction to Recurring Decimal

167 Two Sum II - Input array is sorted

168 Excel Sheet Column Title

169 Majority Element

170 Two Sum III - Data Structure Design

Read some blogs:

[12]<http://www.codeceo.com/article/24-soft-skills-for-programmer.html>

[13]<http://www.codeproject.com/Articles/1005428/Soft-Skills-are-so-important-to-Software-Engineers>

[14]<http://www.techweb.com.cn/network/system/2015-12-14/2240729.shtml>

Relax and read blogs on this website:

[15]<http://www.cnblogs.com/yuzhangcmu/p/4392084.html>

1. <http://www.danielbit.com/blog/puzzle/leetcode/leetcode-one-edit-distance>
2. <http://www.meetqun.com/thread-2854-1-1.html>
3. <http://juliachencoding.blogspot.ca/2015/06/leetcode-question-find-peak-element.html>
4. [https://siddontang.gitbooks.io/leetcode-solution/content/array/find\\_peak\\_element.html](https://siddontang.gitbooks.io/leetcode-solution/content/array/find_peak_element.html)
5. <http://www.cnblogs.com/ganganloveu/p/4147655.html>
6. <http://www.danielbit.com/blog/puzzle/leetcode/leetcode-missing-ranges>
7. <http://easyleetcode.blogspot.ca/2015/01/leetcode-164-maximum-gap.html>
8. <https://leetcode.com/discuss/18499/bucket-sort-java-solution-with-explanation-o-time-and-space>
9. <http://yucoding.blogspot.ca/2014/12/leetcode-question-maximum-gap.html>
10. <http://bookshadow.com/weblog/2014/12/14/leetcode-maximum-gap/>
11. <http://ask.julyedu.com/question/213>
12. <http://www.codeceo.com/article/24-soft-skills-for-programmer.html>
13. <http://www.codeproject.com/Articles/1005428/Soft-Skills-are-so-important-to-Software-Engineers>
14. <http://www.techweb.com.cn/network/system/2015-12-14/2240729.shtml>
15. <http://www.cnblogs.com/yuzhangcmu/p/4392084.html>

---

## Leetcode 51: N - Queen problems (2016-01-13 00:45)

January 12, 2016

Encourage myself to write blogs about leetcode questions. The following talk encourages me a lot.

[1] [https://www.youtube.com/watch?v=R22dJ7bn-pU &list=PLgYNPs-V9YFPqcnEvbly5hFE40BjxMbjw &index=2](https://www.youtube.com/watch?v=R22dJ7bn-pU&list=PLgYNPs-V9YFPqcnEvbly5hFE40BjxMbjw&index=2)  
Hiring Rockstars by Roger Philby

the person likes to write, and then the person must love to read, and then the person must be very curious and intellectual.

So, Julia likes to build up a good habit to write, and then, to read, and to be curious, and to be intellectual.

Julia is working on recursive function design, how to handle N-Queen problems. After 6 months, she totally forgets how to solve the problem; So, she continues to read more blogs about this algorithm.

Review the algorithm again. The following blog gives a good explanation, help me understand the algorithm better.

blog1 (Java programming language):

[2]<http://blog.csdn.net/linhuanmars/article/details/20667175>

Amazed that a person can work on algorithms problems and finish over 300 questions/year, people love algorithms and they must know the power of algorithms to save time and improve efficiency at work.

[3]<http://buttercola.blogspot.ca/search/label/Leetcode>

[4]<http://buttercola.blogspot.ca/2014/09/leetcode-n-queens.html>

Leetcode 314:

[5] [https://github.com/jianminchen/Leetcode\\_C/blob/master/BinaryTreeVerticalOrderTraversal\\_314.java](https://github.com/jianminchen/Leetcode_C/blob/master/BinaryTreeVerticalOrderTraversal_314.java)

Julia likes to work on the analysis of the algorithm, this pseudo code in the blog may help her quickly recover, from nervous to get familiar with the algorithm.

• -

[6]<http://yucoding.blogspot.ca/2013/01/leetcode-question-59-n-queens.html>

The following blog is well-written, and Julia should write a similar solution using C #, to express her appreciation of good variables names, good analysis. (Rating: 10 from 1-10)

[7] <https://polythinking.wordpress.com/2013/02/27/leetcoden-queens-i-and-ii/>

It is fun to read solutions in C++/Java.

Write another version of N-Queen problem in C # to entertain.

[8]

[https://github.com/jianminchen/Leetcode\\_C/blob/master/51NQueenProblem\\_B.cs](https://github.com/jianminchen/Leetcode_C/blob/master/51NQueenProblem_B.cs)

Good advice from the blog 1: [?????NP?????-s?????](#) [9]Sudoku Solver [?](#) [10]Combination Sum [?](#) [11]Combinations [?](#) [12]Permutations [?](#) [13]Word Break II [?](#) [14]Palindrome Partitioning [????????r????,???](#).

Feb. 2, 2016 It is also a DFS algorithm.

1. <https://www.youtube.com/watch?v=R22dJ7bn-pU&list=PLgYNPs-V9YFPqcnEvbIy5hFE40BjxMbjw&index=2>
  2. <http://blog.csdn.net/linhuanmars/article/details/20667175>
  3. <http://buttercola.blogspot.ca/search/label/Leetcode>
  4. <http://buttercola.blogspot.ca/2014/09/leetcode-n-queens.html>
  5. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/BinaryTreeVerticalOrderTraversal\\_314.java](https://github.com/jianminchen/Leetcode_C-/blob/master/BinaryTreeVerticalOrderTraversal_314.java)
  6. <http://yucoding.blogspot.ca/2013/01/leetcode-question-59-n-queens.html>
  7. <https://polythinking.wordpress.com/2013/02/27/leetcoden-queens-i-and-ii/>
  8. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/51NQueenProblem\\_B.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/51NQueenProblem_B.cs)
  9. <http://blog.csdn.net/linhuanmars/article/details/20748761>
  10. <http://blog.csdn.net/linhuanmars/article/details/20828631>
  11. <http://blog.csdn.net/linhuanmars/article/details/21260217>
  12. <http://blog.csdn.net/linhuanmars/article/details/21569031>
  13. <http://blog.csdn.net/linhuanmars/article/details/22452163>
  14. <http://blog.csdn.net/linhuanmars/article/details/22777711>
- 

## Google hiring Best Practices - How to interview others? (2016-01-16 14:00)

January 12, 2016

Google Hiring Best Practices by Allison Watson[1] <https://www.youtube.com/watch?v=U2plchEYZF4&list=PLgYNPs-V9YFPqcnEvbly5hFE40BjxMbjw&index=1>

Take some notes about the presentation:

### 4 key criteria:

#### 1. Intellectual curiosity, Problem solving:

very curious, understand problems; using hypothetical questions or behavior questions, really see how people break down the problem, can they see through, do they ask right types of questions

#### 2. Technical skills, abilities

#### 3. Leadership, not necessary people leadership, willing to take risks, having ownership

4. **People are nice, you want to work with them.** Do they want to engage with people? Work with part of team? Someone you can sit with 3 hours in the airport, wait for a flight and get along with them? Good sign that people can think good with the company.

5. Diversity - try to improve on this. Diverse workforce to reflect them.

A lot of consensus to make hiring decision

Questions:

#### 1. **How Google keep the culture up when company grows?** (Video time 7:00 - 9:00)

Always look for **people who are curious, engaged**, that is number one.

Make sure that we can offer them opportunities.

A lot of people move not for money, not because of money, move because of opportunities to grow, something interesting and challenging; get people always challenging themselves, improve themselves. Give them opportunities to grow, allow them take risks, fail, and then take more risks, and then you tend to find them always engaged.

In summary, what other values, fundamental values, stick to those values. Look for people embody the same thing.

## **2. How do you know the person is going to do well? (video time 9:00 - 11:32)**

**Explain the Jahn-Teller effect**, flip side of effect.

For example, some one comes in the room, smiling all the time, really nice; so you try to convince yourself why to hire this person. In the positive frame or mind.

However, do the opposite, people came from some university, look for reasons not to hire them. Be very conscious on that.

First thing as an interviewer is always thinking or rethinking, and checking, and asking several questions. Something seems off, keep asking, keep checking. Do not do it by yourself. Try to get at least 2 or 3 perspectives. And ideally, some one is not completely involved, have discussion afterwards.

Get 3 people in committee to discuss, objective,

what are the flags: be convinced.

Check that are you fair, not some one just likes you. Check that the person is good for company, good for the job.

Are there any place you can mitigate? Better to get right one. If all three of you are so so, do not do it. You are better to get convinced, and get a long time and get right person.

## **3. Do not do panic interview**

Can you tell last project successful? A lot of questions followed.

Hypothetical question: You can see the reasoning, how they will approach the work. But behavior question, a good story-teller will do better, (may not be a good fit for the job)

## **4. How to short list the applicants?**

The ways to shortlist, two things, you create job spec, seem to get a lot of unrelated resumes; Change the other job spec. Relevant job skills, is CV very jumpy? Make sure that this person can be committed, if the person moves a lot of jobs.

Try to see the achievements. Do a lot of phone screen, quick call; 5-10 minutes, you can quickly get what you want.

Huge emphasis on employee referrals. Recommendation and network.

## **5. How to make sure the managers do the good work on hiring?**

Train them; what criteria we are looking for.

Shadow interviews. Make sure that they are understanding process.

Ask to write feedback,

1. <https://www.youtube.com/watch?v=U2plchEYZF4&list=PLgYNPs-V9YFPqcnEvbIy5hFE40BjxMbjw&index=1>

---

## **Leetcode 314: Binary Tree Vertical Order Traversal (2016-01-16 14:04)**

**January 16, 2016**

First time after 9 years, I installed Eclipse, J2SE, and then, set up Java developer IDE Eclipse, ran the first Java program. Such a great ride to enjoy Java programming. Still remember that in 2006 - 2007, spent over 10 hours daily to read/ run/ maintain software using Java programming language, when I worked for a small startup called Siva Corp Inc. in the city of Delray Beach, Florida, USA.

Just bring all my good memory back about programming using Java.

[1]<http://buttercola.blogspot.ca/2014/09/leetcode-n-queens.html>

**Leetcode 314:**

[2][https://github.com/jianminchen/Leetcode\\_C-/blob/master/BinaryTreeVerticalOrderTraversal\\_314.java](https://github.com/jianminchen/Leetcode_C-/blob/master/BinaryTreeVerticalOrderTraversal_314.java)

Another blog using C++,

[3]<http://buttercola.blogspot.ca/2014/12/facebook-print-binary-tree-in-vertical.html>

1. <http://buttercola.blogspot.ca/2014/09/leetcode-n-queens.html>

2. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/BinaryTreeVerticalOrderTraversal\\_314.java](https://github.com/jianminchen/Leetcode_C-/blob/master/BinaryTreeVerticalOrderTraversal_314.java)

3. <http://buttercola.blogspot.ca/2014/12/facebook-print-binary-tree-in-vertical.html>

---

## **Cracking the Coding Interview (Part 1, 2 of 2) (2016-01-16 14:16)**

January 14, 2016

Take notes about the video

[1]<https://www.youtube.com/watch?v=LU75pxyUFj4>

### **Cracking the Coding Interview (Part 1 of 2)**

software programmer

Coding, problem solving, analytical skills and intelligent.

Look for your aptitude, but not just your knowledge, for those top companies.

Entrepreneur: start a marathon training group

The valuation of technical questions are relative to other candidates.

How You Are Judged

How did you do RELATIVE to other candidates on the SAME question?

More detail, coding questions, how fast you come out solution, how good your algorithm, how clean your code, how buggy your code.

## **PM Roles (program manager, project manager)**

- . Communication Skills
- . User-Focused Thinking
- . Passion for Technology
- . Analytical Skills
- . Technical Skills ( position dependent )

## **How We Review Resumes**

1. Pull resume out of giant stack
2. Spot-check: company names, positions, projects, schools.
3. Skim bullets to see if you've written real code

reject interview

4. Go to next resume & whine about how many more you have left.

"Glanced at," not read, 15-30 seconds

Behavior questions:

## **Communication Goals**

- . Answer the question.
- . Deliver a \*good\* answer. ()
- . Communicate well.

## **Preparing for Behavioral Qs**

- . Create Preparation Grid for Projects

OS Project Amazon Intern

Enjoyed

Hated

Most Challenging

Hardest Bug

Behavior Grid [for PM & less tech. roles]

**Two structures:**

1. First words is the answer.

2. Situation, Action, Result S. A. R.

Technical questions:

Coding algorithms - study very well

Out-of-schools

They are not testing your knowledge.

**Practice solving questions.**

- Don't memorize!

- See: CtCI & CareerCup.com

. Push yourself!

**Data structures**

How to implement

When to use (pros / cons)

Linked Lists Stacks Queues

Trees Tries Graphs

Vectors Heap Hashtables

## **Algorithms**

Implementation

Space vs. Time complexity

Quick Sort Merge Sort

Tree Insert / Find Binary Search

Breadth-First Search Depth-First Search

## **Concepts**

Not just a concept - know how to code!

Threading System Design & Scalability Memory Management

Recursion Probability + Combinatorics Bit Manipulation

Big O time

How to learn CS Fundamentals:

Necessary for "elite" tech companies

MIT Open Courseware

Freshman / sophomore level DS & Algo courses

Books

- CLRS (Algorithms)

**Part 2 of 2 - probably here?**

## **Types of "Serious" Questions**

1. Product Design Questions



## 2. Estimation Questions

## 3. Software Engineering Questions

- Coding & Algorithms
- Object Oriented Design
- Scalability
- Factual / Trivia / Language-Based

### **Problem Design Qs: Approach**

1. Ask questions to resolve ambiguity
2. Understand the user
3. Structure the problem
4. Solve piece by piece

### Estimation Questions:

It is not important to know the answers. About two things, problem solving and basic quantity skills.

How much Gmail business revenue annually?

The way to approach questions,

Ask questions, gmail or gmail.com

Revenue or net sales

### **Find structures:**

1. How many gmail account users?
2. Estimate how much dollar value per click
3. How many clicks

If I can guess 3 numbers we have, I can multiply them and get answer for you.

How do you figure out how many Gmail account users?

Break down, make some assumptions,

### **Estimation Qs: How to Approach**

1. Ask questions to resolve ambiguity

- Don't make assumptions (yet)

2. Outline / Structure Your Approach

3. Break down the components

- Assume numbers when necessary

- State assumptions explicitly

- Round numbers to make your math easier

How many population? 80 % uses gmail

Make reasonable guess

Take notes when you do it

Coding algorithms:

Ask questions - what the people are looking for

Talk aloud -

### **How to Solve Tough Problems (7:53 /22:56 - ? )**

1. Ask Questions!

- Questions are more ambiguous than they appear

2. Talk out loud

- Show us how you think

### 3. Think critically

- Does your algorithm really work? What's the space and time complexity?

### 4. Code slowly and methodically

- It's not a race.

Make too many mistakes, take too long, don't try to race it.

If your code is done, test your code; make sure that it really works

Find the bug, do not just fix the value; find what is wrong.

What does mean to write a good code?

### **What does a "good coder" do?**

- . Code in top-left of whiteboards

- . Pseudo code first (if necessary)

- . Be methodical. Don't try to rush.

- . Reasonably Bug Free

- Thorough testing (and careful fixing)

- Check for error conditions

- . Clean coding

- Use other functions

- Good use of data structure (define own if useful)

- Concise and readable

### **Algorithm Qs: Pattern Matching (13:10/22:56)**

**Q: Write code to reverse the order of words in a sentence.**

"dogs are cute"

"cute are dogs"

Similar to: reverse characters in a string.

"dogs are cute"

"etuc era sgod"

A: Reverse full string, then reverse each word.

Another approach: **Simplify and generalize**

**Q: Design algorithm to figure out if you can build a ransom note (array of strings) from a magazine ( array of strings).**

Simplify: what if we used characters instead of strings?

-> Build array of characters frequencies.

Generalize: how we can extend answer to words?

A: Build hashtable from word to frequency.

Next Approach: **Base case & Build**

**Q: Design algorithm to print subsets of set**

{a, b, c} -> { }, {a }, {b }, {c }, {a,b }, {a,c }, {b, c }, {a, b, c }

S( { } ) -> { }

S( {a } ) -> { }, {a }

S( {a,b } ) -> { }, {a }, {b }, {a,b }

S( {a,b,c } ) -> ?

A. Build S(n) by cloning S(n-1) and add n to the cloned sets.

## Final approach, data structure brain storm

Q: There are  $10^{10}$  possible phone #s. Explain how you could efficiently implement

assignSpecificNum(num) and assignAnyAvailableNum().

Array (sorted)? Too slow to remove num.

Linked list? Too slow to find specific num.

Hash table? Can't iterate through free nums.

Tree? Ah-ha! Binary search tree

Evaluation is relative.

Top 10 % - always get A, does not matter how hard the problem is.

. Write real code -

. Don't rush

1. <https://www.youtube.com/watch?v=LU75pxyUFj4>

---

## Leetcode 5: Longest substring palindrome (2016-01-16 14:20)

January 13, 2016

Read the blog:

[1]<http://blog.csdn.net/linhuanmars/article/details/20888595>

[2]<http://blog.csdn.net/linhuanmars/article/details/22777711>

**Please read 5-6 solution about this leetcode question, and then, collect all the wisdom. Try to have nice memory about the solution, some fun experience; therefore, it will be a quick and fun time to solve the similar problems in the future.**

Do not rush to finish more leetcode questions. Try to focus on simple problems.

There are 5 solutions discussed in the following blog, most important is to give overview of 5 analysis, what is brute force solution - time, space complexity.

One way to make review more fun is to read more than 10 - 20 solutions, and know all the ideas out there, and then, write some code; Reading is most important to help understand algorithms, through a simple problem, common interview question.

[3][http://articles.leetcode.com/2011/11/longest-palindromic-substring-par t-i.html](http://articles.leetcode.com/2011/11/longest-palindromic-substring-part-i.html)

#### 4 solutions in the above blog

one optimal solution - linear time solution in the following blog:

[4][http://articles.leetcode.com/2011/11/longest-palindromic-substring-par t-ii.html](http://articles.leetcode.com/2011/11/longest-palindromic-substring-part-ii.html)

This blog is in Chinese. Excellent! Try to summary a few tips to come out linear time optimal solution! Do something to get more involved.

[5]<http://www.felix021.com/blog/read.php?2040>

spent 10 minutes to read the article, enjoyed the reading; a good article with very well explained brute force solution, how good is to work on linear solution in time complexity.

[6][https://www.akalin.com/longest-palindrome-linear-tim\[7\]e](https://www.akalin.com/longest-palindrome-linear-time/)

Read the blog:

[8][http://www.acmerblog.com/leetcode-longest-palindromic-substring-5356.h tml](http://www.acmerblog.com/leetcode-longest-palindromic-substring-5356.html)

6th solution, a suffix tree solution:

[9]<http://www.allisons.org/ll/AlgDS/Tree/Suffix/>

First blog about this leetcode question:

[10][http://juliachencoding.blogspot.ca/2015/06/leetcode-longest-palndromi c-substring.html](http://juliachencoding.blogspot.ca/2015/06/leetcode-longest-palindromic-substring.html)

Another palindrome algorithm #9

[11][https://github.com/jianminchen/Leetcode \\_C-/blob/master/9palindromeNumber.cs](https://github.com/jianminchen/Leetcode_C/blob/master/9palindromeNumber.cs)

1. <http://blog.csdn.net/linhuanmars/article/details/20888595>

2. <http://blog.csdn.net/linhuanmars/article/details/22777711>

3. <http://articles.leetcode.com/2011/11/longest-palindromic-substring-part-i.html>
  4. <http://articles.leetcode.com/2011/11/longest-palindromic-substring-part-ii.html>
  5. <http://www.felix021.com/blog/read.php?2040>
  6. <https://www.akalin.com/longest-palindrome-linear-time>
  7. <https://www.blogger.com/null>
  8. <http://www.acmerblog.com/leetcode-longest-palindromic-substring-5356.html>
  9. <http://www.allisons.org/ll/AlgDS/Tree/Suffix/>
  10. <http://juliachencoding.blogspot.ca/2015/06/leetcode-longest-palindromic-substring.html>
  11. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/9palindromeNumber.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/9palindromeNumber.cs)
- 

## **pluralsight: online courses study (2016-01-16 14:24)**

### **January 16**

Just start to watch video courses on code school and pluralsight. And she likes to blog on this as well.

Julia did not have time / motivation /confidence to go over MVC in last 5 years work as a full time programmer, but in less than 3 hours, went over MVC course, and found out that technologies are so easy to understand, also affordable to get best education - courses ( \$40/month, compared to IT course \$500/each course, graduate research credit out-of-state \$800 - \$1000), and use it to apply at work.

The pluralsight.com service is such affordable, make me connect to best of talents in the world, catch up all the technologies we need, or just curious, and then, master one or two of them a time if need.

Life is such a journey, and then, you find out that you are isolated, work alone, tired, and suddenly, you are connected to others, get back with more confidence.

That is the journey Julia likes to start with pluralsight.com / codeschool.com.

### **Dec. 28, 2015**

[1]<http://juliachencoding.blogspot.ca/2015/12/code-school-courses-study.html>

started to watch code school videos, learned Angular.JS, Ember.JS, bootstrap CSS framework, reviewed CSS, Html etc. Great teaching from instructors, better than my experience - work full time 3 years + on CSS, html, Java Script, JQuery. Good to know how to teach, how to be teachable, and enjoy the lectures.

### **January 15, 2016**

Spent more than 3 hours to watch the videos on pluralsight.

### **"building applications with [2]ASP.NET MVC 4"**

### **January 28, 2016**

Building a Web App with ASP.NET 5, MVC 6, EF7 and Angular JS

[3]<https://app.pluralsight.com/library/courses/aspsdotnet-5-ef7-bootstrap-angular-web-app/table-of-contents>

### **[4] February 4, 2016**

Web API design

[5]<https://app.pluralsight.com/library/courses/web-api-design/table-of-contents>

Play-by-play learning AngularJS

[6]<https://app.pluralsight.com/library/courses/play-by-play-learning-angularjs-ken-cenerelli-john-papa/table-of-contents>

AngularJS: Get Started

AngularJS Fundamentals

[7]<https://app.pluralsight.com/library/courses/angularui-fundamentals/table-of-contents>

Feb. 10, 2016

AngularUI Fundamentals

3 hours video lectures

[8]<http://stevemichelotti.com/new-pluralsight-course-yeoman-fundamentals/>

reading:

[9]<https://github.com/angular-ui/ui-router/wiki>

February 14, 2016

### **Build your career with Michael Lopp**

And then, read the blog about Michael Lopp:

[10]<http://techcrunch.com/2015/08/01/a-conversation-with-michael-lopp-pinterests-head-of-engineering/>

[11]<http://randsinrepose.com/>

Good article to read:

[12]<http://recode.net/2014/06/10/pinterest-hires-long-time-apple-execs-to-lead-engineering-and-product-design/>

[13]<http://randsinrepose.com/archives/what-to-do-when-youre-screwed/>

[14]<http://randsinrepose.com/archives/the-culture-chart/>

[15]<http://randsinrepose.com/archives/>

February 16, 2016

So good to come cross this course, and then spend 3 hours to catch up responsive design in CSS. Save me a lot of time to catch up CSS.



Pluralsight:

Responsive In-Browser Web Page Design with HTML and CSS

[16] <https://app.pluralsight.com/library/courses/responsive-browser-web-page-design-html-css-2262/table-of-contents>

**March 1, 2016**

Another 3 hours on course:

Building a Web App with ASP.NET 5, MVC 6, EF7 and Angular JS

check other two courses:

1. JavaScript for C # developers
2. Front-end web development Quick Start

Plan to watch this video:

**<https://app.pluralsight.com/library/courses/csharp-best-practices-collection-generics/table-of-contents>**

1. <http://juliachencoding.blogspot.ca/2015/12/code-school-courses-study.html>
2. <http://asp.net/>
3. <https://app.pluralsight.com/library/courses/aspdotnet-5-ef7-bootstrap-angular-web-app/table-of-contents>
4. <https://app.pluralsight.com/library/courses/aspdotnet-5-ef7-bootstrap-angular-web-app/table-of-contents>
5. <https://app.pluralsight.com/library/courses/web-api-design/table-of-contents>
6. <https://app.pluralsight.com/library/courses/play-by-play-learning-angularjs-ken-cenerelli-john-papa/table-of-contents>
7. <https://app.pluralsight.com/library/courses/angularui-fundamentals/table-of-contents>
8. <http://stevemichelotti.com/new-pluralsight-course-yeoman-fundamentals/>
9. <https://github.com/angular-ui/ui-router/wiki>
10. <http://techcrunch.com/2015/08/01/a-conversation-with-michael-lopp-pinterests-head-of-engineering/>
11. <http://randsinrepose.com/>
12. <http://recode.net/2014/06/10/pinterest-hires-long-time-apple-execs-to-lead-engineering-and-product-design/>
13. <http://randsinrepose.com/archives/what-to-do-when-youre-screwed/>
14. <http://randsinrepose.com/archives/the-culture-chart/>
15. <http://randsinrepose.com/archives/>
16. <https://app.pluralsight.com/library/courses/responsive-browser-web-page-design-html-css-2262/table-of-contents>

---

## **Learn Google interview process - 25 things to learn (2016-01-17 01:12)**

**January 9, 2016**

Julia spent time again in 2016 to review the video. She likes to take some notes, and then, look into them. So, she remembers now that interview white board question solving should be fun, and should be creative, and should be easy to work with, since the interviewer likes to drop hints, and make you happy and see if you can solve the problem, or learn something from you as well.

Moishe Lettvin - What I Learned Doing 250 Interviews at Google

[1]<https://www.youtube.com/watch?v=r8RxkpUvxK0>

Moishe was in the hiring committee, and also taught the interview process.

Watch again in 2016, and this time Julia likes to take notes about 25 things about interview.

### 1. **It's Fun!**

Talk to new people, meeting new people is extremely fun, sharing expertise with people who have same knowledge - great fun.

Learn through interview, how to interact, no matter undergraduate dropout or PH.D.

2.

### **Except when it isn't**

3 hours for preparation, before and after; meet the strangers

Random things happens -

Candidate cries, cannot figure out the solution;

Some one just refused to write code; communication problems; People are jerks; people are refused to write code; for more senior position etc.

3.

### **Noisy, inaccurate, arbitrary**

(random, casual)

To write code on white board, it is difficult to do

some cascading problem

Noisy - interviewer may have a bad day, how to feel about interacting with a person

Interviewer may be in a bad day; you can get different results

If you are in border line, you are invited to come back in 1 year - google is just generous

5 people can say no if they interview any hired person

Google's response:

Set the bar very high - false positive, false negative - not hire a bad person

For example, one week, 8 of candidates, say no to all of them

### 4. **Data is a blessing and a curse**

no correlation between interviewee and ...

each interviewer give 1 - 4, 4 definitively hire, 2.0 - 3.0 range

### **You can't put a number on stuff that matters**

Written feedback - 3 or 4 pages narrative about the person

Same as IQ, it is not accurate

**You can't even ask about stuff that matters** - things to love work with, things about employee

How to test it in the interview?

Give them the question, see if she/he has passion, and their creative ability, their enthusiasm, maybe their sense of fun, pretty good habit to hack, come out the solution

Keypoint is like a criticism, not statistics.

#### 5. **It's a team effort, and isolation hurts**

One interviewer goes very deep in one area; two personalities clicks very well.

Microsoft - hiring manager is responsible for saying yes/ no - hiring bar goes down a little bit but Google, Hiring for Google, not for hiring for a team

Analysis of hiring of Microsoft:

not be on malicious, just need to hire some one; bar will keep going down; no check on it.

Analysis of hiring of Google:

very consistent.

#### 6. **Be prepared!**

When you have to give out interview, you should be prepared.

**Having a structure will help:**

Google has 4-5 hiring - interview process class - basic outline

10 minutes -

40 minutes - question

10 minutes - question to answer

More nervous than interviewee - first interview to give.

#### 7. **But, it's like jazz**

unexpected thing will happen, will go side ways;

Favorite interview questions:

Maybe they are heard; you can figure out quickly; improvise,

Maybe they do not like to write code

Maybe people have better ideas to write code

Maybe people can

Maybe be taught, interviewee may give out total out-of-surprise solution

#### 8. **Please make mistakes!**

It takes practice. calibrates, 3 people gives interviews together.

people did more than 25 interviews.

## 9. Every interview is a conversation (with goals)

It is not a test, or something like putting through and see if the interviewee can survive

Conversation has goals, which is ok; Do not just stress the interviewer out;

**Primary goal:**

**Happy candidate**

Make them happy

Do not get something from the candidate, but make them feel not good.

It does not take much to gain notoriety - people blog about the bad interview process

Interview is stressful

Survey: Do you have good experience of Google interview?

**Other goal: Answer "Would I love to work with this person?"**

Even the people is not in the same team, do you like to work with him; contribute the company, can he teach you, can they be taught by you, can he inspire you etc. ? Idea world, work with the people smart than me. You want to keep raising the bar.

Is this person having qualities, and then people love to work with? Can they contribute to the company

**Other goal: give answer to "Would you love to work at this company?"**

Talk about culture, how excited to work here.

Feel real Humanity sense during the interview while I am working here.

**Good interviewers are generous** - Ask good questions when interview goes on. Honest to candidate, work hard on candidate. Ask good questions in the interview goes on.

I learn something amazing, graph traversal problem for example, traverse the graph, and find all the words in them. Second part, implement a dictionary to look up those words quickly. Use tries for that.

That is what far the interviewer gets on that question. But he asked the candidate, he got through that approximately 8 minutes. What else you can do, he showed me that traverse the graph and trie in parallel, magic and beautiful thing.

cannot write down the detail.

Example:

1. Traverse graph, and build up a dictionary. A candidate spends 8 minutes to figure out. The candidate shows doing thing in parallel.

2. Random number, time stamp

superior smart, give me direction

3. Beauty math and engineering problem

Interviewer being prepared, knowledge

**10. Time is of the essence - counter point of generous of interviewer**

**Time management - easy to get away from you as an interviewer.**

Candidate may talk too much not important, not relate to the conversation the interviewer likes to have.

**A few thoughts on technical interviews:**

**11. Good question are like onions**

naive solution in the crust, you can peel back, deep, more interesting problems going on.

First technical interview question I asked, write a program to draw a circle -

Microsoft, 1994, super nervous, five minutes small talk

Write a program to draw a circle -  $\sin$ ,  $\cos$ ,  $2 * \pi$ ,

Interviewer says that it sucks, speed up

$\sqrt{\text{root}}$

All right, good job

- 15 minutes left, no square root - compute error - with right hint

compute error - high far away you are

congratulations: circle drawing algorithms by windows

questions goes to deeper and deeper, for the candidate, it is great. More iterate on something

Iterative on something

**12. Strive for higher bandwidth**

prefer for the white board to just talk to people, can get more data

First Microsoft interview, whole nervous whole day.

Just give me a notebook and problem, and let me work on the problems.

**13. Artifice is inevitable**

Being a candidate is not having a job.

Is this person enjoyable to be around? Is this person creative? Can this person learn?

**14. More signal; less noise**

Let candidate know what they are here for; let interviewee prepare

Your job as an interviewer is to help them show their best work. That is what you want to see. You are not just trying to make them fail.

**15. This might be the best we can do**

It probably isn't

Hack school is a good hiring tool - 2 months of coding, some one can finish or so on.

## 15. Have Fun

### Learn

Questions?

1. Machine learning helps the interviewer to decide ...  
Human input, the written feedback, hiring committee

2. Question: false positive

Worthy taking some risk on some one; Make a bad hire, cascading effects,

3. Microsoft hires for a team, Google hires for the company  
hiring committee -  
culture thing -

4. How to hire people with domain knowledge? 1 person with domain knowledge

5. Personality fit vs strong technique skills

Ability to lean and pick things up; demonstrate Java and Java Script knowledge, so he/she can pick up PHP.

People cannot get the expertise you need.

6. Don't hire assholes if he is perfect for the job.

During interview, he is a domain expert; Some one creates a toxic environment.

He is bad, drag people down;

7. Interviewers talk about, send an email for these emails.

Microsoft email chain - hiring or not hiring / quiet sure

Poised or blessed by your first interviewer - a senior people if he is.

Google does not allow to communicate, every one does not have context, what happened before. Without bias is better.

8. Group interview - 2 interviewer, one interviewee

2 shade interviews - 2 shade interviews, just observe, do not say a word.

9. If you have power to change, what to change?

Could you hang out? Could you talk? Can you communicate?

Do you have enough data for you to decide if you should hire?

10. calibrate? new bees? How much weight?

Good questions, exercise the candidate

here is the place the interviewer has to hint

Apply something

Write solid code

Talk through ...

11. Who is focusing on? before interview

After the interview, people talk about the impressions, sort of things.

2-3 pages - take up code sample - two pages typical for feedback

If every one goes to wrap up meeting,

12. Google does follow up, it takes months to make a decision; now it only takes weeks.

Stumble early on, give hints on - just more you ask question, the more you know how to give hints.

13. behavior questions:

agree general ideas how to do behavior questions.

Program interview exposed.

14. turnaround time - how quickly to get back to the candidate?

Microsoft - last interview - you have the hiring manager to interview you.

Google - April, interview, at the end of June.

6th edition of book:

[2][http://www.amazon.com/gp/product/0984782850/ref=olp\\_product\\_details?ie=UTF8 &me=](http://www.amazon.com/gp/product/0984782850/ref=olp_product_details?ie=UTF8 &me=)

1. <https://www.youtube.com/watch?v=r8RxkpUvxK0>

2. [http://www.amazon.com/gp/product/0984782850/ref=olp\\_product\\_details?ie=UTF8&me=](http://www.amazon.com/gp/product/0984782850/ref=olp_product_details?ie=UTF8&me=)

---

interview hr (2016-06-21 01:48:53)

This comment has been removed by a blog administrator.

interview hr (2016-08-18 19:04:01)

This comment has been removed by a blog administrator.

## **string function - strStr - BoyerMoore algorithm - needle in hay (2016-01-17 14:27)**

January 5, 2016

Read the Java code on the following website:

[1]<http://algs4.cs.princeton.edu/53substring/BoyerMoore.java.html>

Write a C # version, and check in github, and see if it will help to memorize the algorithm.

Read the webpage: (well written! now Julia knows two rules: bad character rule, the good suffix rule)

[2][https://en.wikipedia.org/wiki/Boyer%E2%80%93Moore\\_string\\_search\\_algorithm](https://en.wikipedia.org/wiki/Boyer%E2%80%93Moore_string_search_algorithm)

[3]<http://www.cs.tufts.edu/comp/150GEN/classpages/BoyerMoore.html>

previous study:

[4]<http://juliachencoding.blogspot.ca/2015/06/leetcode-strstr-function.html>

[5][http://juliachencoding.blogspot.ca/search/label/string %20functions %20review](http://juliachencoding.blogspot.ca/search/label/string%20functions%20review)

1. <http://algs4.cs.princeton.edu/53substring/BoyerMoore.java.html>
  2. [https://en.wikipedia.org/wiki/Boyer%E2%80%93Moore\\_string\\_search\\_algorithm](https://en.wikipedia.org/wiki/Boyer%E2%80%93Moore_string_search_algorithm)
  3. <http://www.cs.tufts.edu/comp/150GEN/classpages/BoyerMoore.html>
  4. <http://juliachencoding.blogspot.ca/2015/06/leetcode-strstr-function.html>
  5. <http://juliachencoding.blogspot.ca/search/label/string%20functions%20review>
- 

## Algorithms night (1) (2016-01-17 22:57)

January 17, 2016

Spent Sunday evening (8pm - 12pm) to work on algorithms questions. Just read the blogs, and get some ideas how to solve the problems.

### 1. String parenthesis to see if all of them are pair correctly

#### Leetcode 20: valid parentheses

[1]<http://buttercola.blogspot.ca/2014/07/leetcode-valid-parentheses.html>

[2]<http://www.shuatiblog.com/blog/2014/05/09/Valid-Parentheses/>

### 2. Count the number of palindromes in a string

(January 28, 2016)

Come back to write down ideas:

1. First of all, do not count duplicate.
2. Brute force solution:  
any substring of  $O(N^2)$  substrings to see if it is a palindrome;  
Add the substring of palindrome to a hashset if it is not in the hashset.  
And return the length of hashset
3. Use recursive solution - using subproblem to solve. Cannot filter out duplicate - not good
4. Better solution - use center point of string -  $2n + 1$ , and then, go over each one, add all palindromes substring.

Requirement: write a C # code in 10 minutes for the solution, using brute force one.

[3][https://github.com/jianminchen/AlgorithmsPractice/blob/master/NumberOf Distinct Palindromes.cs](https://github.com/jianminchen/AlgorithmsPractice/blob/master/NumberOf%20Distinct%20Palindromes.cs)

### 3. Write a function to detect if the string has the unique character.

Read the blog :

[4]<http://stackoverflow.com/questions/19484406/detecting-if-a-string-has-unique-characters-comparing-my-solution-to-cracking>

this blog provides good answers for the question of unique character:



Notice that this method doesn't allocate an array of booleans. Instead, it opts for a clever trick. Since there are only 26 different characters possible and there are 32 bits in an

int

, the solution creates an

int

variable where each bit of the variable corresponds to one of the characters in the string. Instead of reading and writing an array, the solution reads and writes the bits of the number.

Julia's comment: first blog about bit manipulation - surprising, after the reading, it is much easy to understand the code:

Two bit operation:

1<< val

|=

It is always helpful to understand the

public

static

boolean

isUniqueChars

(

String

str

)

{

if

```
(  
    str  
    .  
    length  
)
```

```
>
```

```
256
```

```
)
```

```
{
```

```
// NOTE: Are you sure this isn't 26?
```

```
return
```

```
false
```

```
;
```

```
}
```

```
int
```

```
    checker
```

```
146
```

```
=
```

```
0
```

```
;
```

```
for
```

```
(
```

```
int
```

```
  i
```

```
=
```

```
0
```

```
;
```

```
  i
```

```
<
```

```
  str
```

```
  .
```

```
length
```

```
  ());
```

```
  i
```

```
  ++)
```

```
{
```

```
int
    val
=
    str
.
charAt
(
    i
)
```

-

```
'a'
;
```

```
if
```

```
((
checker
&
```

```
(
1
```

{\textless}{\textless}

```
    val
  ))

>

0
)

return

false

;

checker

|=

(

1

{\textless}{\textless}

    val
  );

}

return
```

true

;

}

[5]<http://javahungry.blogspot.com/2014/11/string-has-all-unique-characters-java-example.html>

#### 4. A single linked list, delete a node but only know the node pointer which needs to be deleted. How to do it?

[6]<http://www.geeksforgeeks.org/in-a-linked-list-given-only-a-pointer-to-a-node-to-be-deleted-in-a-singly-linked-list-how-do-you-delete-it/>

#### 5. Leetcode/ Lintcode: binary tree maximum path sum

Leetcode 124: binary tree maximum path sum

[7]<http://fisherlei.blogspot.ca/2013/01/leetcode-binary-tree-maximum-path-sum.html>

Review the blog:

[8]<http://juliachencoding.blogspot.ca/2015/07/leetcode-binary-tree-maximum-path-sum.html>

#### 6. Minimum height of a binary search tree

[9]<http://www.programcreek.com/2013/02/leetcode-minimum-depth-of-binary-tree-java/>

1. <http://buttercola.blogspot.ca/2014/07/leetcode-valid-parentheses.html>

2. <http://www.shuatiblog.com/blog/2014/05/09/Valid-Parentheses/>

3. <https://github.com/jianminchen/AlgorithmsPractice/blob/master/NumberOfDistinctPalindromes.cs>

4. <http://stackoverflow.com/questions/19484406/detecting-if-a-string-has-unique-characters-comparing-my-solution-to-cracking>

5. <http://javahungry.blogspot.com/2014/11/string-has-all-unique-characters-java-example.html>

6. <http://www.geeksforgeeks.org/in-a-linked-list-given-only-a-pointer-to-a-node-to-be-deleted-in-a-singly-linked-list-how-do-you-delete-it/>

7. <http://fisherlei.blogspot.ca/2013/01/leetcode-binary-tree-maximum-path-sum.html>

8. <http://juliachencoding.blogspot.ca/2015/07/leetcode-binary-tree-maximum-path-sum.html>

9. <http://www.programcreek.com/2013/02/leetcode-minimum-depth-of-binary-tree-java/>

## Leetcode 317: Shortest distance from all buildings (2016-01-20 00:19)

January 19, 2016

Problem statement:

You want to build a house on an empty land which reaches all buildings in the shortest amount of distance. You can only move up, down, left and right. You are given a 2D grid of values 0, 1 or 2, where:

Each 0 marks an empty land which you can pass by freely.

Each 1 marks a building which you cannot pass through.

Each 2 marks an obstacle which you cannot pass through.

For example, given three buildings at (0,0), (0,4), (2,2), and an obstacle at (0,2):

```
1 - 0 - 2 - 0 - 1
| | | |
0 - 0 - 0 - 0 - 0
| | | |
0 - 0 - 1 - 0 - 0
```

The point (1,2) is an ideal empty land to build a house, as the total travel distance of 3+3+1=7 is minimal. So return 7.

code reference:

[1]<http://buttercola.blogspot.ca/2016/01/leetcode-shortest-distance-from-all.html>

Julia's practice using C # programming language

with **bugs** waiting to fix, good experience, spent more than 2 hours, from 9pm - 11pm to learn lessons.

[2][https://github.com/jianminchen/Leetcode\\_C-/blob/master/317ShortestDistanceFromAllBuildings\\_BugA.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/317ShortestDistanceFromAllBuildings_BugA.cs)

T

**ip 1: Julia likes the idea of " [3] express the intent**

**", so she added a new function called getNoOfBuildings().**

So, here is the version to **fix the bug** :

[4][https://github.com/jianminchen/Leetcode\\_C-/blob/master/317ShortestDistanceFromAllBuildings.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/317ShortestDistanceFromAllBuildings.cs)

Action item:

Rewrite the fill function, make the function do one task:

public

void

```

fill
(
int
origX,
int
origY,
int
x,
int
y,
int
curDistance,
int[][]
dist,
int[][]
reach,
int[][]
grid,
bool[][]
visited, Queue<Node> queue)

```

the above function is too complicated, Julia create a few bugs in the function.  
more than 8 arguments - too many!

#### **Here is the version after some work on function design**

- get rid of too many arguments
- function much more easy to read, understand
- less error-prone
- function to do one task only

[5][https://github.com/jianminchen/Leetcode\\_C-/blob/master/317ShortestDistanceFromAllBuildings\\_C.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/317ShortestDistanceFromAllBuildings_C.cs)

[6]<https://gist.github.com/jianminchen/e8d660831f7b04adfb84>

#### **How to make BFS algorithm design experience fun, memorable ?**

- Have more discussion on the blog. Welcome comments.

#### **Lessons Julia learned through debugging:**



1. using ref or not in C # language - does not matter
2. function design - simple, is important to review, avoid bugs
3. Queue is used for BFS, and when the node is added to queue, please also log the distance, and use it for next round. - Main bug - calculation. Since each node is in different distance to the building node.

More about BFS:

#### 1. BFS - how to think about the algorithm ?

Tips: Try to start from simple things, like count how many building, empty lands, obstacles; then, figure out which one is meaningful to work with first.

3 choices - buildings, empty land, obstacle. Choose one and see if you can make progress

buildings - very good. Do some simple task, put all buildings in a List data structure, easy to iterate.

empty land - kind of hard to do BFS - the question is to ask a node to all building distance sum - not sum - but the minimum of sum.

So, start from each building to do BFS

#### 2. BFS - how to design a BFS function?

So many bugs in the first 2-3 hours coding, it takes 30 minutes to understand other people's idea and code.

Go through bugs one by one later. (To be continued)

#### Encourage herself to work on BFS algorithms in the future:

Some facts:

1. Julia could not figure out how to handle this BFS problem on January 19, 2016, she has to read other people's solution first, and also analysis, and then, start to work on coding.

2. BFS - Ask questions:

start from where - how many of them - what order to do BFS? order matter or not?

3. Julia could not finish the BFS algorithm writing in less than 30 minutes. A lot of bugs, confusion with BFS function design, and edge case handling.

#### Past experience on BFS:

Julia is still working on BFS algorithms, and like to get more experience if possible. Here is another BFS practice she did in June 2015.

[7]<http://juliachencoding.blogspot.ca/2015/06/breadth-first-search-algorithm.html>

More reading about Leetcode 317:

[8]<https://asanchina.wordpress.com/2016/01/03/317-shortest-distance-from-all-buildings-2/>

BFS algorithm review:

[9]<https://asanchina.wordpress.com/category/bfs-%E5%B9%BF%E5%BA%A6%E4%BC%98%E5%85%88%E6%90%9C%E7%B4%A2/>

1. <http://buttercola.blogspot.ca/2016/01/leetcode-shortest-distance-from-all.html>
  2. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/317ShortestDistanceFromAllBuildings\\_BugA.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/317ShortestDistanceFromAllBuildings_BugA.cs)
  3. <http://juliachencoding.blogspot.ca/search/label/code%20standard>
  4. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/317ShortestDistanceFromAllBuildings.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/317ShortestDistanceFromAllBuildings.cs)
  5. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/317ShortestDistanceFromAllBuildings\\_C.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/317ShortestDistanceFromAllBuildings_C.cs)
  6. <https://gist.github.com/jianminchen/e8d660831f7b04adfb84>
  7. <http://juliachencoding.blogspot.ca/2015/06/breadth-first-search-algorithm.html>
  8. <https://asanchina.wordpress.com/2016/01/03/317-shortest-distance-from-all-buildings-2/>
  9. <https://asanchina.wordpress.com/category/bfs-%E5%B9%BF%E5%BA%A6%E4%BC%98%E5%85%88%E6%90%9C%E7%B4%A2/>
- 

## Book Reading: C# book Effective C# (2016-01-20 21:38)

January 20, 2016

Julia came cross this C # book:

Effective C # (Covers C # 4.0): 50 Specific Ways to Improve Your C #

[1] <http://www.amazon.com/Effective-Covers-4-0-Specific-Development/dp/0321658701>

Read item 2:

Item 2: Prefer readonly to const

Need to spend time to read very well-written articles about C #, and enjoy learning.

Try to find more books about C # to improve algorithm writing - code, design and more.

1. <http://www.amazon.com/Effective-Covers-4-0-Specific-Development/dp/0321658701>
- 

## Leetcode questions 20: Valid Parentheses (2016-01-20 21:44)

January 20, 2016

Leetcode 20: Valid parentheses

[1] <http://blog.csdn.net/fightforyourdream/article/details/13011825>

Julia"- C # code:

[2] [https://github.com/jianminchen/Leetcode\\_C-/blob/master/20ValidParentheses.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/20ValidParentheses.cs)

January 20, 2016

[3] [https://github.com/jianminchen/Leetcode\\_C/blob/master/20ValidParentheses\\_B.cs](https://github.com/jianminchen/Leetcode_C/blob/master/20ValidParentheses_B.cs)

**favorite blogs to read: January 20, 2016**

**Great time to go over different ideas to implement the algorithm, using Map in C++, Hashmap, and see so many talents through those code -**

[4] <http://bangbingsyb.blogspot.ca/2014/11/leetcode-valid-parentheses.html>

[5] <http://www.acmerblog.com/leetcode-solution-valid-parentheses-6316.html>

[6] [http://blog.csdn.net/foreverling/article/details/49685177?hmsr=toutiao.io &utm\\_medium=toutiao.io &utm\\_source=toutiao.io](http://blog.csdn.net/foreverling/article/details/49685177?hmsr=toutiao.io&utm_medium=toutiao.io&utm_source=toutiao.io)

[7] <http://segmentfault.com/a/1190000003481208>

[8] <http://harriefeng.github.io/algo/leetcode/valid-parentheses.html>

<http://www.jiuzhang.com/solutions/valid-parentheses/>

So,

ready to review more about parentheses:

Leetcode 32: longest valid parentheses -

[9] <http://codeganker.blogspot.ca/2014/03/longest-valid-parentheses-leetcode.html>

Julia's blog on this question:

Leetcode question 22: Generate Parentheses

[10] [https://github.com/jianminchen/Leetcode\\_C/blob/master/GenerateParentheses\\_No22.cs](https://github.com/jianminchen/Leetcode_C/blob/master/GenerateParentheses_No22.cs)

Leetcode question 32:

32 Longest Valid Parentheses

[11] <http://codeganker.blogspot.ca/2014/03/longest-valid-parentheses-leetcode.html>

[12] <http://bangbingsyb.blogspot.ca/2014/11/leetcode-longest-valid-parentheses.html>

[13] <http://blog.csdn.net/worldwindjp/article/details/39460161>

[14] <http://shanjiixin.blogspot.ca/2014/04/longest-valid-parentheses-leetcode.html>

C # code:

[15] [https://github.com/jianminchen/Leetcode\\_C/blob/master/32LongestValidParentheses.cs](https://github.com/jianminchen/Leetcode_C/blob/master/32LongestValidParentheses.cs)

1. <http://blog.csdn.net/fightforyourdream/article/details/13011825>

2. [https://github.com/jianminchen/Leetcode\\_C/blob/master/20ValidParentheses.cs](https://github.com/jianminchen/Leetcode_C/blob/master/20ValidParentheses.cs)

3. [https://github.com/jianminchen/Leetcode\\_C/blob/master/20ValidParentheses\\_B.cs](https://github.com/jianminchen/Leetcode_C/blob/master/20ValidParentheses_B.cs)

4. <http://bangbingsyb.blogspot.ca/2014/11/leetcode-valid-parentheses.html>

5. <http://www.acmerblog.com/leetcode-solution-valid-parentheses-6316.html>

6. [http://blog.csdn.net/foreverling/article/details/49685177?hmsr=toutiao.io&utm\\_medium=toutiao.io&utm\\_source=](http://blog.csdn.net/foreverling/article/details/49685177?hmsr=toutiao.io&utm_medium=toutiao.io&utm_source=)

toutiao.io

7. <http://segmentfault.com/a/1190000003481208>
  8. <http://harrifeng.github.io/algo/leetcode/valid-parentheses.html>
  9. <http://codeganker.blogspot.ca/2014/03/longest-valid-parentheses-leetcode.html>
  10. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/GenerateParentheses\\_No22.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/GenerateParentheses_No22.cs)
  11. <http://codeganker.blogspot.ca/2014/03/longest-valid-parentheses-leetcode.html>
  12. <http://bangbingsyb.blogspot.ca/2014/11/leetcode-longest-valid-parentheses.html>
  13. <http://blog.csdn.net/worldwindjp/article/details/39460161>
  14. <http://shanjiaxin.blogspot.ca/2014/04/longest-valid-parentheses-leetcode.html>
  15. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/32LongestValidParentheses.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/32LongestValidParentheses.cs)
- 

## Algorithm night (2): 10 questions to read (2016-01-20 22:08)

January 20, 2016

Plan to spend 2 hours (9pm - 12pm) to read 10 blogs about algorithms, each blog 10 minutes.

1.

### Leetcode: Coin Change

[1]<http://www.geeksforgeeks.org/dynamic-programming-set-7-coin-change/>

Think about **recursive function** solution first, and then DP solution to remove redundant calculation

[2]<http://blog.csdn.net/kenden23/article/details/17636599>

[3]<http://buttercola.blogspot.ca/2016/01/leetcode-coin-change.html>

2.

### Leetcode: Maximum product of word lengths

[4]<https://www.hrwhisper.me/leetcode-maximum-product-of-word-lengths/>

[5]<http://buttercola.blogspot.ca/2016/01/leetcode-maximum-product-of-word-lengths.html>

Brute force analysis, **bit manipulation** - encode string using integer, and then, optimize the algorithm - instead of two strings' comparison, using two integer bit manipulation - And operator - clever solution.

3.

### Leetcode questions: Power of 2 / Power of 3

[6]<http://www.cnblogs.com/grandyang/p/4623394.html>

Power of 3

[7]<http://my.oschina.net/Tsybius2014/blog/601048>

[8]<http://buttercola.blogspot.ca/2016/01/leetcode-power-of-three.html>

4.

Leetcode 325: Maximum size subarray sum to K (**Level: easy**)

[9][https://asanchina.wordpress.com/2016/01/05/325-maximum-size-sub array-sum-equals-k/](https://asanchina.wordpress.com/2016/01/05/325-maximum-size-sub-array-sum-equals-k/)

[10][http://buttercola.blogspot.ca/2016/01/leetcode-maximum-size-subarray- sum.html](http://buttercola.blogspot.ca/2016/01/leetcode-maximum-size-subarray-sum.html)

January 21, 2016

5.

### **BFS - algorithm**

[11]<http://www.geeksforgeeks.org/find-number-of-islands/>

Read the blog, understand the algorithm. Rate: 10

<http://www.jiuzhang.com/solutions/find-the-connected-component-in-the-undirected-graph/>

<http://algorithms.wtf/number-of-connected-components-in-an-undirected-graph/>

[12][http://buttercola.blogspot.ca/2016/01/leetcode-number-of-connected- components.html](http://buttercola.blogspot.ca/2016/01/leetcode-number-of-connected-components.html)

6.

### **Sparse Matrix Multiplication**

[13][http://www.cs.cmu.edu/ scandal/cacm/node9.html](http://www.cs.cmu.edu/scandal/cacm/node9.html)

[14]<https://leetcode.com/discuss/71912/easiest-java-solution>

[15]<http://buttercola.blogspot.ca/2016/01/leetcode-sparse-matrix-multiplication.html>

7.

### **Leetcode 310: Minimum Height Trees**

[16][https://leetcode.com/problems/minimum-height-trees/ #](https://leetcode.com/problems/minimum-height-trees/#)

read the blog:

[17][http://bookshadow.com/weblog/2015/11/26/leetcode-minimum-height-trees /](http://bookshadow.com/weblog/2015/11/26/leetcode-minimum-height-trees/)

[18]<https://leetcode.com/discuss/71656/c-solution-o-n-time-o-n-space>

[19][http://likemyblogger.blogspot.ca/2015/11/leetcode-310-minimum-hei ght-trees.html](http://likemyblogger.blogspot.ca/2015/11/leetcode-310-minimum-height-trees.html)

[20][http://buttercola.blogspot.ca/2016/01/leetcode-minimum-height-trees. html](http://buttercola.blogspot.ca/2016/01/leetcode-minimum-height-trees.html)

8.

### **Leetcode: Remove duplicate letters**

[21]<http://bookshadow.com/weblog/2015/12/09/leetcode-remove-duplicate-letters/>  
3 solutions are discussed in the following blog:

[22]<https://www.hrwhisper.me/leetcode-remove-duplicate-letters/>

[23]<https://leetcode.com/discuss/73777/easy-to-understand-iterative-java-solution>

[24]<http://buttercola.blogspot.ca/2016/01/leetcode-remove-duplicate-letters.html>

9.

### **Leetcode: Longest increasing subsequence**

[25]<http://blog.csdn.net/kenden23/article/details/17632821>

spend 10 minutes to read the blog:

[26]<http://www.kancloud.cn/kancloud/data-structure-and-algorithm-notes/73075>

[27]<http://www.geeksforgeeks.org/dynamic-programming-set-3-longest-increasing-subsequence/>  
Previous blog:

[28]<http://juliachencoding.blogspot.ca/2015/06/dynamic-programming-longest-increasing.html>

Write some C # code for this algorithm later.

[29]<http://buttercola.blogspot.ca/2015/12/leetcode-longest-increasing-subsequence.html>

NLogn solution, better than DP solution, Recursive solution

[30]<http://www.geeksforgeeks.org/longest-monotonically-increasing-subsequence-size-n-log-n/>

Set Goals:

1. Know how to solve it using recursive solution first;
2. And then, understand DP solution, how to build the formula, avoid duplication;
3. And then, know where to find optimal solution here,  $O(n \log n)$ ,

10.

### **Leetcode: Binary Tree Longest Consecutive Sequence**

Excellent chance to practice recursive function

[31]<https://segmentfault.com/a/1190000003957798>

[32]<http://buttercola.blogspot.ca/2015/12/blog-post.html>

1. <http://www.geeksforgeeks.org/dynamic-programming-set-7-coin-change/>

2. <http://blog.csdn.net/kenden23/article/details/17636599>

3. <http://buttercola.blogspot.ca/2016/01/leetcode-coin-change.html>

4. <https://www.hrwhisper.me/leetcode-maximum-product-of-word-lengths/>

5. <http://buttercola.blogspot.ca/2016/01/leetcode-maximum-product-of-word-lengths.htm>

6. <http://www.cnblogs.com/grandyang/p/4623394.html>
7. <http://my.oschina.net/Tsybius2014/blog/601048>
8. <http://buttercola.blogspot.ca/2016/01/leetcode-power-of-three.html>
9. <https://asanchina.wordpress.com/2016/01/05/325-maximum-size-subarray-sum-equals-k/>
10. <http://buttercola.blogspot.ca/2016/01/leetcode-maximum-size-subarray-sum.html>
11. <http://www.geeksforgeeks.org/find-number-of-islands/>
12. <http://buttercola.blogspot.ca/2016/01/leetcode-number-of-connected-components.html>
13. <http://www.cs.cmu.edu/~scandal/cacm/node9.html>
14. <https://leetcode.com/discuss/71912/easiest-java-solution>
15. <http://buttercola.blogspot.ca/2016/01/leetcode-sparse-matrix-multiplication.html>
16. <https://leetcode.com/problems/minimum-height-trees/>
17. <http://bookshadow.com/weblog/2015/11/26/leetcode-minimum-height-trees/>
18. <https://leetcode.com/discuss/71656/c-solution-o-n-time-o-n-space>
19. <http://likemyblogger.blogspot.ca/2015/11/leetcode-310-minimum-height-trees.html>
20. <http://buttercola.blogspot.ca/2016/01/leetcode-minimum-height-trees.html>
21. <http://bookshadow.com/weblog/2015/12/09/leetcode-remove-duplicate-letters/>
22. <https://www.hrwhisper.me/leetcode-remove-duplicate-letters/>
23. <https://leetcode.com/discuss/73777/easy-to-understand-iterative-java-solution>
24. <http://buttercola.blogspot.ca/2016/01/leetcode-remove-duplicate-letters.html>
25. <http://blog.csdn.net/kenden23/article/details/17632821>
26. <http://www.kancloud.cn/kancloud/data-structure-and-algorithm-notes/73075>
27. <http://www.geeksforgeeks.org/dynamic-programming-set-3-longest-increasing-subsequence/>
28. <http://juliachencoding.blogspot.ca/2015/06/dynamic-programming-longest-increasing.htm>
29. <http://buttercola.blogspot.ca/2015/12/leetcode-longest-increasing-subsequence.html>
30. <http://www.geeksforgeeks.org/longest-monotonically-increasing-subsequence-size-n-log-n/>
31. <https://segmentfault.com/a/1190000003957798>
32. <http://buttercola.blogspot.ca/2015/12/blog-post.html>

---

## Algorithm night (3) (2016-01-21 22:54)

January 21, 2016

Find 10 challenging algorithms questions, and then go over them one by one. It is fun, and also good learning experience.

She wishes that she could have more than 100 algorithm night, reviewed 1000 algorithms already, and each time, just so relax to go over different ideas how to solve them, by reading, and by good hints and teaching through blogs. At that time, she will be good thinker about algorithms. Wishful thinking, get back to reality (5 algorithms, 2 hours, a lot of headache, and could not figure out what is brute force solution.), work on 3rd algorithm night.

Be humble, be stupid, and learn from mistakes.

### 1. Leetcode 295: Find medium from data stream

[1]<https://segmentfault.com/a/1190000003709954>

[2]<https://gist.github.com/jianminchen/31572a64b2af48e3ccce>

[3]<http://buttercola.blogspot.ca/2015/12/leetcode-find-median-from-data-stream.html>

Julia, read Java priorityQueue class and get some ideas about the class design:

[4]<http://www.programcreek.com/2009/02/using-the-priorityqueue-class-example/>

[5]<http://stackoverflow.com/questions/683041/java-how-do-i-use-a-priority-queue>

understand priority queue first, read the lecture notes:

[6]<http://www.eecs.wsu.edu/~ananth/CptS223/Lectures/heaps.pdf>

## 2. min stack

[7]<http://buttercola.blogspot.ca/2015/11/zenefits-mini-stack.html>

3. `float sumPossibility(int dice, int target)` `sumPossibility(dice, target)` brute force `dfs` `O(6^dice)` - `dp` - memorized search

[8]<https://gist.github.com/diegozeng/6c964f623bbeaa716526>

[9]<http://www.labnol.org/internet/github-gist-tutorial/28499/>

## 4. Possible triangle

[10]<http://www.geeksforgeeks.org/find-number-of-triangles-possible/>

[11]<http://stackoverflow.com/questions/8110538/total-number-of-possible-triangles-from-n-numbers>

Argue about how to reduce time complexity from  $O(N^3)$  to  $O(N^2)$ , how exactly  $i, j, k$  three variable,  $k$  is only from 0 to  $n-1$  once for each  $i$ , nothing to do with  $j$ . So, it is time for  $i$  - from 1 to  $N$ , and time for  $j$  from 1 to  $N$  two loops, and then for  $i - N$ , and  $k$  from 1 to  $N$  ( **skip j - big point! Cannot figure out easily.** )

5.

**Top K int in a large stream** (This can be done in  $O(n)$ )

[12]<http://stackoverflow.com/questions/185697/the-most-efficient-way-to-find-top-k-frequent-words-in-a-big-word-sequence>

## 6. Leetcode solutions - blog reading

Just read as many solution and see if there are something new to write down here:

[13]<http://www.cnblogs.com/grandyang/p/4606334.html> ( No. 1 - No. 280)

Review Leetcode questions 1 - 20 (January 24, 2016)

So, here are **some notes about DFS - backtracking problem solving:**

1. DFS vs BFS, DFS usually involves backtracking as well. So, remember backtracking.
2. How to design the DFS function?



A. return result - one of arguments:

`lList<string> res`

is a good idea, also, it can be member variable of class Solution.

B. Convert a char to integer '1' -> 1

quickest way is `c - '0'`

C. Backtracking is the key to ensure the bug free

D. Java, C # using `StringBuilder` to construct string, `append`

Review solutions:

[14]<http://codesniper.blogspot.ca/2015/03/sticky-post-all-of-my-published.html>

Java solution: Leetcode 1 - 120

1. <https://segmentfault.com/a/1190000003709954>

2. <https://gist.github.com/jianminchen/31572a64b2af48e3ccce>

3. <http://buttercola.blogspot.ca/2015/12/leetcode-find-median-from-data-stream.html>

4. <http://www.programcreek.com/2009/02/using-the-priorityqueue-class-example/>

5. <http://stackoverflow.com/questions/683041/java-how-do-i-use-a-priorityqueue>

6. <http://www.eecs.wsu.edu/~ananth/CptS223/Lectures/heaps.pdf>

7. <http://buttercola.blogspot.ca/2015/11/zenefits-mini-stack.html>

8. <https://gist.github.com/diegozeng/6c964f623bbeaa716526>

9. <http://www.labnol.org/internet/github-gist-tutorial/28499/>

10. <http://www.geeksforgeeks.org/find-number-of-triangles-possible/>

11. <http://stackoverflow.com/questions/8110538/total-number-of-possible-triangles-from-n-numbers>

12. <http://stackoverflow.com/questions/185697/the-most-efficient-way-to-find-top-k-frequent-words-in-a-big-word-sequence>

13. <http://www.cnblogs.com/grandyang/p/4606334.html>

14. <http://codesniper.blogspot.ca/2015/03/sticky-post-all-of-my-published.html>

---

## Leetcode 17: Letter Combinations of a phone number (DFS) (2016-01-24 15:02)

January 24, 2016

### 17 Letter Combinations of a phone number (DFS)

Julia likes to focus on basic things about algorithms, she tries to focus on recursive function design, BFS, DFS, backtracking. Here is her favorite DFS algorithm, she tries to build more fun memory about DFS algorithm, every time she works on DFS algorithm, she is so happy and eager to share her 2 cents, learned from Leetcode blogs - all her favorite blogs.

[1]<http://www.cnblogs.com/grandyang/p/4452220.html>

Analysis from the above blog:

????????????-h????????2?9????-X???????????????????????????????????? [2] Path Sum II ?????????? ?  
[3]Subsets II ?????? ? [4]Permutations ??? ? [5]Permutations II ?????? ? [6]Combinations ??? ? [7] Com-

combination Sum [8] Combination Sum II Recursion  
level

Her practice is too weak in 2015.

[9][https://github.com/jianminchen/Leetcode\\_C/blob/master/LetterCombinationOfAPhoneNumber.cs](https://github.com/jianminchen/Leetcode_C/blob/master/LetterCombinationOfAPhoneNumber.cs)

#### Missing in the last practice :

1. Use iterative solution, but lack of analysis - (comment area, hard to review)
2. Should focus on DFS solution, basic things.

#### New practice in January 2016:

1. using recursive to solve the problem, it is fast and quick way to solve it in 10 minutes.
2. Only need to write a few lines of code.
3. Need to design recursive function.
4. Need to do backtracking,
5. Need to do char, string, int a few types, and also do type conversion.

Julia's practice, it took her 27 minutes to write down, compile, and have some comment written.

[10][https://github.com/jianminchen/Leetcode\\_C/blob/master/17LetterCombinationOfAPhoneNUmber\\_DFS.cs](https://github.com/jianminchen/Leetcode_C/blob/master/17LetterCombinationOfAPhoneNUmber_DFS.cs)

January 24, Read more blogs on this question:

subset, combination backtracking  
digits[i]

Iterative solution:

[11]<http://bangbingsyb.blogspot.ca/2014/11/leetcode-letter-combinations-of-phone.html>

Java, using hashmap - Good idea

[12]<http://blog.welkinlan.com/2015/10/25/letter-combinations-of-a-phone-number-leetcode-java/>

DFS

[13][http://simpleandstupid.com/2014/10/16/letter-combinations-of-a-phone-number-leetcode- %E8 %A7 %A3 %E9 %A2 %98 %E7 %AC %94 %E8 %AE %B0/](http://simpleandstupid.com/2014/10/16/letter-combinations-of-a-phone-number-leetcode-%E8%A7%A3%E9%A2%98%E7%AC%94%E8%AE%B0/)

So, Julia is not good at recursive function design as well. So, she likes to search a blog and find the most important tip she can grasp in next 20 minutes.

Array initialization can be in one line instead of more than 8 lines

[14]<http://yucoding.blogspot.ca/2013/01/leetcode-question-42-letter.html>

DFS, backtracking is clear in the code.

[15]<http://leetcode.blogspot.ca/2014/02/letter-combinations-of-phone-number-java.html>

Java code, member of class - hashmap - static - first time read static - declare variables sharing static in state-

ment. (missing backtracking? or does not matter)

[16]<https://github.com/rffffff007/leetcode/blob/master/Letter%20Combinations%20of%20a%20Phone%20Number.java>

very well written,

[17]<http://codesniper.blogspot.ca/2015/01/17-letter-combinations-of-phone-number.html>

using vector instead of hashmap

[18]<http://yumei165.blogspot.ca/2013/04/letter-combinations-of-phone-number-c.html>

Good analysis - read the analysis in the following blog -

- [http://www.jiuzhang.com/solutions/letter-combinations-of-a-phone-number/](#), [http://www.jiuzhang.com/solutions/letter-combinations-of-a-phone-number/](#)
- [http://www.jiuzhang.com/solutions/letter-combinations-of-a-phone-number/](#) (Julia's comment -> backtracking)
- [http://www.jiuzhang.com/solutions/letter-combinations-of-a-phone-number/](#)-Ö[http://www.jiuzhang.com/solutions/letter-combinations-of-a-phone-number/](#), [http://www.jiuzhang.com/solutions/letter-combinations-of-a-phone-number/](#)void. [http://www.jiuzhang.com/solutions/letter-combinations-of-a-phone-number/](#).
- [http://www.jiuzhang.com/solutions/letter-combinations-of-a-phone-number/](#)-ÿc++[http://www.jiuzhang.com/solutions/letter-combinations-of-a-phone-number/](#), [http://www.jiuzhang.com/solutions/letter-combinations-of-a-phone-number/](#)const String

[19]<http://harifeng.github.io/leetcode/letter-combinations-of-a-phone-number.html>

[http://www.jiuzhang.com/solutions/letter-combinations-of-a-phone-number/](#)

[20]<http://www.jiuzhang.com/solutions/letter-combinations-of-a-phone-number/>

Analysis from the following blog:

### Understand the problem:

The problem gives a digit string, return all possible letter combination that the number could represent. Note the relative order of the number should be reflected in the corresponding strings. For instance, "23", 2 is ahead of 3, so abc should be in front of def as well. For a brute force solution, we can iterate all possible combinations, with the time complexity of  $O(n^m)$ , where  $n$  is the number of characters for each digit,  $m$  is the length of the digit string.

### Recursive Solution:

It is a typical recursive problem, you can mimic the solution of Subset.

[21]<http://buttercola.blogspot.ca/2014/09/leetcode-letter-combinations-of-a-phone-number.html>

use this one as my example to write a good solution. Good style!

[22]<https://segmentfault.com/a/1190000003766442>

Excellent code style.

[23]<http://shanjiaxin.blogspot.ca/2014/02/letter-combinations-of-a-phone-number.html>

So, **put together something for DFS /backtracking algorithm - favorite tips :**

1. DFS, do not forget backtracking, using Chinese words, " 回溯 " .

2. Recursive function design:

Return result C #: IList<string>,

put it as input argument or member variable of class Solution

3. Remember a tip to convert a char to int, very basic convert using this:

Char c (no one can remember the ascii value of '0', but use type conversion c-'0', char to int; in other words, compare to relative char, get the value by type conversion automatically).

**c - '0'**

4. dictionary declaration, common and easy way is to declare one dimension array:

```
String[] table = { " ", " ", "abc", "def", "ghi", "jkl", "mno", "pqrs", "tuv", "wxyz" };
```

5. using C # or Java,

string

class is not good, using StringBuilder class instead.

6. Remember one or two DFS algorithms, use them to relax and help to figure out design issue, difficulty level of DFS problem solving - honestly, it is easy and quick solution, less time-consuming compared to an iterative solution, more time-consuming DP with memorization solution.

1. <http://www.cnblogs.com/grandyang/p/4452220.html>

2. <http://www.cnblogs.com/grandyang/p/4042156.html>

3. <http://www.cnblogs.com/grandyang/p/4310964.html>

4. <http://www.cnblogs.com/grandyang/p/4358848.html>

5. <http://www.cnblogs.com/grandyang/p/4359825.html>

6. <http://www.cnblogs.com/grandyang/p/4332522.html>

7. <http://www.cnblogs.com/grandyang/p/4419259.html>

8. <http://www.cnblogs.com/grandyang/p/4419386.html>

9. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/LetterCombinationOfAPhoneNumber.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/LetterCombinationOfAPhoneNumber.cs)

10. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/17LetterCombinationOfAPhoneNUmber\\_DFS.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/17LetterCombinationOfAPhoneNUmber_DFS.cs)

11. <http://bangbingsyb.blogspot.ca/2014/11/leetcode-letter-combinations-of-phone.html>

12. <http://blog.welkinlan.com/2015/10/25/letter-combinations-of-a-phone-number-leetcode-java/>

13. <http://simpleandstupid.com/2014/10/16/letter-combinations-of-a-phone-number-leetcode-%E8%A7%A3%E9%A2%98%E7%AC%94%E8%AE%B0/>

14. <http://yucoding.blogspot.ca/2013/01/leetcode-question-42-letter.html>

15. <http://rleetcode.blogspot.ca/2014/02/letter-combinations-of-phone-number-java.html>

16. <https://github.com/rfffffffff007/leetcode/blob/master/Letter%20Combinations%20of%20a%20Phone%20Number.java>
  17. <http://codesniper.blogspot.ca/2015/01/17-letter-combinations-of-phone-number.html>
  18. <http://yumei165.blogspot.ca/2013/04/letter-combinations-of-phone-number-c.html>
  19. <http://harriheng.github.io/algo/leetcode/letter-combinations-of-a-phone-number.html>
  20. <http://www.jiuzhang.com/solutions/letter-combinations-of-a-phone-number/>
  21. <http://buttercola.blogspot.ca/2014/09/leetcode-letter-combinations-of-phone.html>
  22. <https://segmentfault.com/a/1190000003766442>
  23. <http://shanjiaxin.blogspot.ca/2014/02/letter-combinations-of-phone-number.html>
- 

## Leetcode 1, 15, 16: Two sum, 3 sum, 3 sum closest (2016-01-24 20:44)

January 24, 2016

Review Leetcode solution, Two sum, 3 sum, 3 sum closest.

[1][http://codesniper.blogspot.ca/2014/12/two-sum-leetcode\\_31.html](http://codesniper.blogspot.ca/2014/12/two-sum-leetcode_31.html)

[2]<http://codesniper.blogspot.ca/2015/01/3sum-leetcode.html>

[3]<http://codesniper.blogspot.ca/2015/01/3sum-closest-leetcode.html>

May 17, 2016

Julia likes to write some code after she reads the blog:

[4]<http://www.cnblogs.com/TenosDolt/p/3649607.html>

Write C # practice on Leetcode 16 - 3 sum closest solution:

[5]<https://gist.github.com/jianminchen/d23598296e7ba665b6d45ae4acfeb600>

First practice on Leetcode 16 - 3 sum closest solution on

[6][https://github.com/jianminchen/Leetcode\\_C-/blob/master/3sumCloset.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/3sumCloset.cs)

Question and Answer:

1. What do you learn through the practice this time?

Julia learns the importance to do static analysis on the code. Do not rush, make sure every variable/ name is making sense, every line of code is best she can present, every executable path should be examined. Talk about the work as well.

1. [http://codesniper.blogspot.ca/2014/12/two-sum-leetcode\\_31.html](http://codesniper.blogspot.ca/2014/12/two-sum-leetcode_31.html)
2. <http://codesniper.blogspot.ca/2015/01/3sum-leetcode.html>
3. <http://codesniper.blogspot.ca/2015/01/3sum-closest-leetcode.html>
4. <http://www.cnblogs.com/TenosDoIt/p/3649607.html>
5. <https://gist.github.com/jianminchen/d23598296e7ba665b6d45ae4acfeb600>
6. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/3sumCloset.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/3sumCloset.cs)

---

## Sunday - Algorithm day (4) - 25 algorithms study (2016-01-31 12:07)

Feb. 13, 2016

Algorithm Sunday, work on it, not just Google. Think by yourself first, work hard on it first.

### 1. Maximum number of overlapping intervals

For example – { (0,2), (3, 7), (4,6), (7,8), (1,5) }. The maximum number of intervals overlapped is 3 during (4,5).

Recall Leetcode merge interval

Good solutions maybe: Using two arrays, one is to sort interval by start, another is by end.

### 2. Merge two sorted linked lists

**Recursive solution:**

[1]<http://stackoverflow.com/questions/10707352/interview-merging-two-sorted-singly-linked-list>

Iterative solution:

[2]<http://stackoverflow.com/questions/10707352/interview-merging-two-sorted-singly-linked-list>

[3]<http://www.geeksforgeeks.org/merge-two-sorted-linked-lists/>

Julia's practice:

Recursive solution:

[4]<https://github.com/jianminchen/AlgorithmsPractice/blob/master/MergeTwoSortedSinglyLinkedList.cs>

### 3. LRU Cache Miss

Read blogs about LRU first:

[5]<http://www.programcreek.com/2013/03/leetcode-lru-cache-java/>

[6]<https://leetcode.com/discuss/20139/java-hashtable-double-linked-list-with-touch-of-pseudo-nodes>

This blog is excellent. Julia will write the implementation in C #.

[7]<https://gist.github.com/Jeffwan/9926551>

[8]<http://www.acmerblog.com/leetcode-lru-cache-lru-5745.html>

[9]<http://www.jiuzhang.com/solutions/lru-cache/>

### 4. Leetcode 64: Minimum Path Sum

[10]<http://codingmelon.com/2015/12/12/minimal-path-sum-from-root-to-leaf/>

### 5. Disk scheduling algorithm: Round Robin

### 6. K closest point

[11]<https://shepherdyan.wordpress.com/category/algorithms-and-data-structures/page/4/>

[12]<https://shepherdyan.wordpress.com/2014/10/16/%E8%AE%B0%E5%9C%A82014/>

## 7. Rearrange array - [ ] [ ] [ ] [ ] [ ]

**count inversions :**

[15]

idea:

## 10. Iterative binary tree traversal

[18]<https://shepherdyan.wordpress.com/2014/08/09/google-bst-insertr-search-delete/>

[19]<https://shepherdyan.wordpress.com/2014/08/08/trie-java-implementation/>

[20]<https://shepherdyan.wordpress.com/2014/08/07/moores-voting-algorithm/>

## [21]

[22]

## Decimal representation

## 15. Implement list, stack and map

**16.**

[26]<https://shepherdyan.wordpress.com/2014/08/03/heap-sort/>

[27]<http://pages.cs.wisc.edu/~vernon/cs367/notes/11.PRIORITY-Q.html>

[28]<https://sheperdyuan.wordpress.com/2014/08/03/sorting-algorithms/>

Julia's blog and her quick sorting partition explanation with a test case:

## quick sort algorithm practice:

## 18. Merge Intervals

[31]<https://leetcode.com/problems/merge-intervals/>

## 168



BFS - graph traversal

????? ????api????node ??????million to billion ??????response?????-z\?

## 20. validBST

????dup ?? dup return false ????? failed ??corner case

**21. Given a tree, find the smallest subtree that contains all of the tree's deepest nodes.**

a

/ | \

b c d

/ \ |

e f g

// \

h i j

depth of tree: 4

deepest nodes:[h,i,j]

least common ancestor of [h,i,j]: b

?????e???.???????f??.

return: b

leetcode????? lowest common ancestor of binary tree

**22. insert node in circular list ,**

**23. LRU miss count**

?? maxSize? input int array

????miss cout

example? size = 4? input array ?1?2?3?4?5?4?1?

1 miss

2 miss

3 miss

4 miss

?????e???.??????.???

5 miss ?? 1

4 hit ?4???????

1 miss ?? 2

???????????

February 2, 2016

Read blogs:

[32]<http://blog.csdn.net/lycorislqy/article/details/49218977>

[33]<http://yikun.github.io/2015/04/03/%E5%A6%82%E4%BD%95%E8%AE%BE%E8%AE%A1%E5%AE%9E%E7%8E%B0%E4%B8%80%E4%B8%AALRU-Cache%E7%BC%9F/>

February 2, 2016

[34]<http://blog.csdn.net/rollycoris/article/category/5884445>

[35]<http://blog.csdn.net/rollycoris/article/category/5884447>

Julia's practice:

[36]<https://github.com/jianminchen/AlgorithmsPractice/blob/master/LRUCacheMissCount.cs>

**24.**

8 cell

example

ps

example

day0: (0)0,1,1,1(0)

day1: (0)1,1,0,1(0)

day2: (0)1,1,0,0(0)

**25. Longest path from root to leaf in binary tree**

Diameter

[37]<http://www.geeksforgeeks.org/diameter-of-a-binary-tree/>

Feb. 17, 2016

Check out topcoder training about testing cases, maybe, use it for a while for practice.

[38]<https://www.topcoder.com/blog/how-quora-founder-and-ex-facebook-cto-a-dam-dangelo-became-a-topcoder/>

[39]<https://www.zhihu.com/question/40463511>

1. <http://stackoverflow.com/questions/10707352/interview-merging-two-sorted-singly-linked-list>

2. <http://stackoverflow.com/questions/10707352/interview-merging-two-sorted-singly-linked-list>

3. <http://www.geeksforgeeks.org/merge-two-sorted-linked-lists/>

4. <https://github.com/jianminchen/AlgorithmsPractice/blob/master/MergeTwoSortedSinglyLinkedList.cs>
5. <http://www.programcreek.com/2013/03/leetcode-lru-cache-java/>
6. <https://leetcode.com/discuss/20139/java-hashtable-double-linked-list-with-touch-of-pseudo-nodes>
7. <https://gist.github.com/Jeffwan/9926551>
8. <http://www.acmerblog.com/leetcode-lru-cache-lru-5745.html>
9. <http://www.jiuzhang.com/solutions/lru-cache/>
10. <http://codingmelon.com/2015/12/12/minimal-path-sum-from-root-to-leaves/>
11. <https://shepherdyan.wordpress.com/category/algorithms-and-data-structures/page/4/>
12. <https://shepherdyan.wordpress.com/2014/10/16/%E8%AE%B0%E5%9C%A82014/>
13. <https://shepherdyan.wordpress.com/2014/07/23/linkedin-k-closest-points/>
14. <https://shepherdyan.wordpress.com/2014/08/22/count-inversions/>
15. <https://shepherdyan.wordpress.com/2014/08/22/count-inversions/>
16. <https://shepherdyan.wordpress.com/2014/09/06/rearrange-array-in-alternating-positive-negative-items-with-o1-extra-space/>
17. <https://shepherdyan.wordpress.com/2014/08/09/iterative-binary-tree-traversal/>
18. <https://shepherdyan.wordpress.com/2014/08/09/google-bst-insert-search-delete/>
19. <https://shepherdyan.wordpress.com/2014/08/08/trie-java-implementation/>
20. <https://shepherdyan.wordpress.com/2014/08/07/moores-voting-algorithm/>
21. <https://shepherdyan.wordpress.com/2014/08/07/first-missing-positive/#respond>
22. <https://shepherdyan.wordpress.com/2014/08/22/count-inversions/>
23. <https://shepherdyan.wordpress.com/2014/08/07/first-missing-positive/>
24. <https://shepherdyan.wordpress.com/2014/09/06/g%E5%AE%B6%E9%9D%A2%E7%BB%8F%E9%A2%98-decimal-representation/>
25. <https://shepherdyan.wordpress.com/2014/07/24/linkedin-%E9%9D%A2%E7%BB%8F%E9%A2%98-implement-liststackmap/>
26. <https://shepherdyan.wordpress.com/2014/08/03/heap-sort/>
27. <http://pages.cs.wisc.edu/~vernon/cs367/notes/11.PRIORITY-Q.html>
28. <https://shepherdyan.wordpress.com/2014/08/03/sorting-algorithms/>
29. <http://juliachencoding.blogspot.ca/2015/06/3-way-partition-problem-dutch-national.html>
30. <https://github.com/jianminchen/AlgorithmsPractice/blob/master/quickSortAlgorithm.cs>
31. <https://leetcode.com/problems/merge-intervals/>
32. <http://blog.csdn.net/lycorislqy/article/details/49218977>
33. <http://yikun.github.io/2015/04/03/%E5%A6%82%E4%BD%95%E8%AE%BE%E8%AE%A1%E5%AE%9E%E7%8E%B0%E4%B8%80%E4%B8%AALRU-Cache%E7%BC%9F/>
34. <http://blog.csdn.net/rollycoris/article/category/5884445>
35. <http://blog.csdn.net/rollycoris/article/category/5884447>
36. <https://github.com/jianminchen/AlgorithmsPractice/blob/master/LRUCacheMissCount.cs>
37. <http://www.geeksforgeeks.org/diameter-of-a-binary-tree/>
38. <https://www.topcoder.com/blog/how-quora-founder-and-ex-facebook-cto-adam-dangelo-became-a-topcoder/>
39. <https://www.zhihu.com/question/40463511>

---

## 2.2 February

**quicksort: a practice makes difference (2016-02-02 21:53)**

February 2, 2016

Julia likes to work on basic things on algorithm problem solving, like brute force solution, recursive function design, divide and conquer solution, partitioning; So, she can write code everyday on algorithm.

Review the quicksort algorithm, and then, practice using C # programming language. So many times Julia went through the review of quick sort algorithm, but to write it in less than 10 minutes, without a bug, without stress, takes more dedicated effort - maybe write a blog to share will help.

First review the blog - the quick algorithm written in Java programming language.

[1]<https://shepherdyuan.wordpress.com/2014/08/03/sorting-algorithms/>

**Let us describe how quick sort works: - explain it using 5 minutes - 10 minutes to write the code**

1. First, using partition, and then, divide and conquer, use recursive function calls to solve the problem.

Most important technique, is to design a partition - 2-way partition, how to design the partition, choose pivot point, and how to do partition step by step.

Let us work on a simple example and have discussion, show the procedure of partition.

Let us say that array 1, 2, 3, 4, 5, 6, sorted

and then, we like to derive the above sorted array using this one, 1, 3, 2, 5, 4, 6

The idea of pivot position, is to choose **last number** in the array, for example, choose last one in the array, 6,

the position of pivot point, left side  $\leq 6$ , right side  $> 6$

So, do you see the problem, 6 is not ideal case. No swap, second partition with 0 values; in other words, all values go to first partition.

Let us work on another order: 1, 6, 2, 5, 4, 3,

**Overview of test case design** 1, 6, 2, 5, 4, 3 :

Tips:

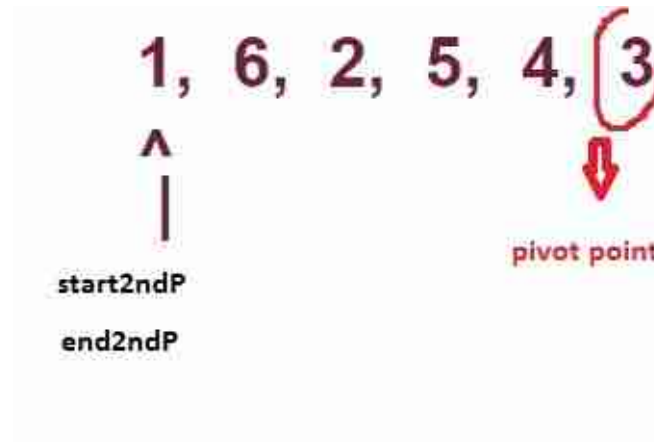
left partition: 2 values, right partition 3 values, partition only needs two swaps, first swap for left partition, second one is for pivot point positioning. <- remember my choice, design test case. Julia tried to make her test case easy to follow.

So, this way, the test case is set up to demo the algorithm clearly and efficient enough.

work on last one in the array, 3, using it as pivot: 1, 2 left side,

6, 5, 4 right side; and then, put the pivot point in-between, leave as is, left/ right side goes to subproblem, use recursive call.

Use a diagram to show progress:



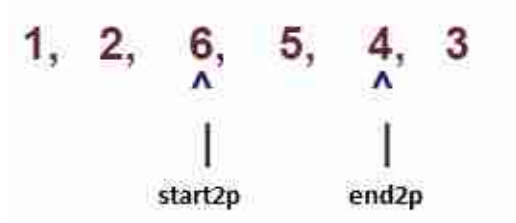
current <- iterate from start - 0 to last one - right -1, leave A[right] for pivot point

another pointer -> mark first node > pivot point value 3, s

So, two pointers, one is moving to iterate whole array except last one - pivot point.

Second one is to mark the position of first node > pivot value, let us call it **start2ndP** .

After review, two pointers will mark the right partition, so change the variable name using start2ndp, end2ndp.



right side start and end position.

6 5 4

start2ndP - 2

end2ndP - 4 // not 5, start from 0

left partition:

left -> start2ndP -1

Does this tip help to program? Relax a little bit.

Julia's practice:

Quick sort algorithm writing in C # - less than 20 minutes (18 minutes to write the code with comments)

[2]<https://github.com/jianminchen/AlgorithmsPractice/blob/master/quickSortAlgorithm.cs>

[3][https://github.com/jianminchen/AlgorithmsPractice/blob/master/quickSortAlgorithm\\_B.cs](https://github.com/jianminchen/AlgorithmsPractice/blob/master/quickSortAlgorithm_B.cs)

Julia, spend 1 hours to read the webpage, and enjoy the great lecture. Reading is much important than coding. And then, work on some questions in the lecture.

[4]<http://algs4.cs.princeton.edu/23quicksort/>

February 3, 2016

After reading her favourite lecture notes ([5]<http://algs4.cs.princeton.edu/23quicksort/>), Julia likes to respond something to entertain quicksort algorithm practice, put something together with her own thinking based on discussion from reading material:

Fact 1:

**Q1. Quick sort algorithm will work even in worst case, no dead loop. Why?**

Arguments:

1. Each recursive function, at least one value is resolved, no more work for the value. That is **pivot point**. So, at most, each recursive call solves one value, n recursive call will solve all n values in the array. Is it fun to design the algorithm? Yes.

Q2.

**Quicksort function design - only solve one value to position correctly, in the whole array.** This is not efficient?

Arguments:

1. This is the fact. And it works fine. And if you remember that, you can write a quick sort algorithm in less than 5 minutes.

2. So, position one value, partition array into two sections.

**Q3. Partition can use different strategies, only important and should work on more is to find one, work out as fast as you can. And make it easy to remember. What is your advice?**

Advice:

Choose last one in the array as a pivot point, and then, find the partition - swap and maintain a partition (each value in the partition > pivot value) just on right side of array. Use two pointers to find the partition two values - start, end position.

**Q4. Is this algorithm using in-place without extra space ?**

Answer:

Yes, the answer is staying in the original array, no extra space ( $O(1)$ , not  $O(n)$ ). So, array is used to store result, only swap two nodes' value if need.

Facts: at most how many swap? it depends.  $O(n^2)$

How many recursive calls? n

Because it is using in-place, the quicksort function design takes 3 input arguments, original array, start, end.

If you cannot come out best design using in-place, you may need extra time.

Feb. 4, 2016

## Q5. Work on Leetcode - partition list

86. Partition list

[6]<http://www.cnblogs.com/springfor/p/3862392.html>

328. Odd Even Linked List

1. <https://shepherdyan.wordpress.com/2014/08/03/sorting-algorithms/>
  2. <https://github.com/jianminchen/AlgorithmsPractice/blob/master/quickSortAlgorithm.cs>
  3. [https://github.com/jianminchen/AlgorithmsPractice/blob/master/quickSortAlgorithm\\_B.cs](https://github.com/jianminchen/AlgorithmsPractice/blob/master/quickSortAlgorithm_B.cs)
  4. <http://algs4.cs.princeton.edu/23quicksort/>
  5. <http://algs4.cs.princeton.edu/23quicksort/>
  6. <http://www.cnblogs.com/springfor/p/3862392.html>
- 

## Algorithm: Merge two sorted singly linked list (2016-02-03 00:43)

February 2, 2016

**Always work on simple problem. Enjoy the practice. Recursive function design is most important one.**

### recursive solution:

[1]<http://stackoverflow.com/questions/10707352/interview-merging-two-sorted-singly-linked-list>

### Iterative solution:

[2]<http://stackoverflow.com/questions/10707352/interview-merging-two-sorted-singly-linked-list>

[3]<http://www.geeksforgeeks.org/merge-two-sorted-linked-lists/>

Julia's practice:

### Recursive solution:

[4]<https://github.com/jianminchen/AlgorithmsPractice/blob/master/MergeTwoSortedSinglyLinkedList.cs>

1. <http://stackoverflow.com/questions/10707352/interview-merging-two-sorted-singly-linked-list>
  2. <http://stackoverflow.com/questions/10707352/interview-merging-two-sorted-singly-linked-list>
  3. <http://www.geeksforgeeks.org/merge-two-sorted-linked-lists/>
  4. <https://github.com/jianminchen/AlgorithmsPractice/blob/master/MergeTwoSortedSinglyLinkedList.cs>
-

## Algorithm: Count the number of palindromes in a string (2016-02-03 21:39)

### Count the number of palindromes in a string

January 28, 2016

Write down ideas:

1. First of all, do not count duplicate.
2. Brute force solution:  
any substring of  $O(N^2)$  substrings to see if it is a palindrome;  
Add the substring of palindrome to a hashset if it is not in the hashset.  
And return the length of hashset
3. Use recursive solution - using subproblem to solve. Cannot filter out duplicate - not good
4. Better solution - use center point of string -  $2n + 1$ , and then, go over each one, add all palindromes substring.

Requirement: write a C # code in 10 minutes for the solution, using brute force one.

[1]<https://github.com/jianminchen/AlgorithmsPractice/blob/master/NumberOfDistinctPalindromes.cs>

1. <https://github.com/jianminchen/AlgorithmsPractice/blob/master/NumberOfDistinctPalindromes.cs>

---

## Algorithm: reading blog (2016-02-03 22:25)

February 2, 2016

Think about learning Python today, since Julia likes to read Python code and get some great ideas in the code. She starts to read the blogs from Leetcode question 331 to 1 in descending order.

As a programmer, she spent over 6 months to learn Javascript in 2014, and then she loves to read Javascript code now. So, spend 10 - 20 minutes a day to learn Python, one day she will love to read Python code.

Here is the blog she starts to read.

[1]<http://bookshadow.com/leetcode/>

[2]<https://www.hrwhisper.me/leetcode-algorithm-solution/>

[3]<http://www.cnblogs.com/grandyang/p/4606334.html>

[4]<http://www.cnblogs.com/EdwardLiu/tag/Leetcode/>

[5]<http://www.jiuzhang.com/problem/>

Here is the log of her reading:

**Feb. 3, 2016**

Leetcode: 331, 330, 329, 328, 327, 326

**331. Verify Preorder serialization of a Binary Tree**

idea: Use stack



### 330. Patching Array

Read the blog, understand the idea:

[6]<https://leetcode.com/discuss/82822/solution-explanation>

### 329. Longest Increasing Path in a matrix

idea: go through each node in the matrix, BFS search, and get minimum one;

### 328. Odd Even Linked List (Easy)

Julia's comment: think about  $O(N)$  space solution first (using array, and then, easy to go through), and then, work on  $O(1)$  space solution, the following blog helps:

[7]<http://www.cnblogs.com/EdwardLiu/p/5138199.html>

Feb. 4, 2016

### 327 Count of Range Sum

Still confused about the question, the example is also hard to understand.

### 324. Wiggle Sort II

Julia figured out the easy way to do it,  $O(n)$  time,  $O(1)$  space. So motivated. Next time, think about algorithm by yourself, start from brute force, and then, work towards requirement - efficient solution.

[8]<https://segmentfault.com/a/1190000003783283>

```
public
```

```
class
```

```
Solution
```

```
{
```

```
public
```

```
void
```

```
wiggleSort
```

```
( int [] nums)
```

```
{
```

```
for
```

```
(
```

```
int
```

```
i =
```

```
1
```

```
; i < nums.length; i++) {
```

```
// nums[i] < nums[i - 1] || nums[i] > nums[i - 1]
```

```
if
```

```
((i %
```

```
2
```

```

==
1
& & nums[i] < nums[i-
1
]) || (i %
2
==
0
& & nums[i] > nums[i-
1
])) {
int
tmp = nums[i-
1
]; nums[i-
1
] = nums[i]; nums[i] = tmp; } } }

```

322 **Coin change**  
 [9]<http://www.cnblogs.com/grandyang/p/5138186.html>

Dynamic Programming 'dp[i] = min(dp[i], dp[i - coins[j]] + 1);

```

dp[i] = min(dp[i], dp[i - coins[j]] + 1);
);

```

coins[j] - coins[j] i dp r 1 dp Φp

321. **Find Maximum number**  
 [10]<https://www.hrwhisper.me/leetcode-create-maximum-number/>  
 319

**Bulb Switch**  
 [11]<http://www.cnblogs.com/grandyang/p/5100098.html>

Analysis from the above blog:

5X' - √

???? XXXXX

??? √√√√

??? √X√X√

??? √XXXX√

??? √XX√√

??? √XX√X

????????????1?4????????????????-gJ????????????????n????? Δ????n????????-  
????????????n????????????n?36????????(1,36), (2,18), (3,12), (4,9), (6,6), ??????????????Ü????????????  
????????????-Xψ????????????????(6,6)????6???????? o?????????????Q????????????-  
????????????-??????????????????????????????

????????1?n????????????-4??force brute????1????????n???

February 7, 2-16

Please understand the question, if the problem is too complex, then the understanding may not be not correct.

### Leetcode 318: Maximum Product of Word Length

Given a string array words , find the maximum value of length(word[i]) \* length(word[j]) where the two words do not share common letters. You may assume that each word will contain only lower case letters. If no such two words exist, return 0.

[12]<https://www.hrwhisper.me/leetcode-maximum-product-of-word-lengths/>

Solution 1:

????????-pÖ????????????

- ???????0
- ??-i???? words[i].length() \* words[j].length()

Solution 2:

????????-?int ?????????6????

- elements[i] |= 1 << (words[i][j] - 'a'); //?words[i][j] ?26????????1
- elements[i] & elements[j] // ?????????? ? ? ?AND????

read the blog to understand bit operation, take some time to refresh the memory:

[13]<http://www.cnblogs.com/onlyac/p/5155881.html>

????????words????max {Length(words[i]) \* Length(words[j]) },?words[i]?words[j]????-?????a-  
z????

????????words[i]????a-z????words[i]?words[j]???max {Length(words[i]) \* Length(words[j]) }?

??a-z26?-f-bool flg[26]??int??int?32? ?flg? ? &?1?-  
<<??“abcd” ?int? 0000 0000 0000 0000 0000 0000 1111?“wxyz” ?int? 1111 0000 0000 0000 0000  
0000 0000 0000?? &??0? ????????

1. <http://bookshadow.com/leetcode/>
2. <https://www.hrwhisper.me/leetcode-algorithm-solution/>
3. <http://www.cnblogs.com/grandyang/p/4606334.html>
4. <http://www.cnblogs.com/EdwardLiu/tag/Leetcode/>
5. <http://www.jiuzhang.com/problem/>
6. <https://leetcode.com/discuss/82822/solution-explanation>
7. <http://www.cnblogs.com/EdwardLiu/p/5138199.html>
8. <https://segmentfault.com/a/1190000003783283>
9. <http://www.cnblogs.com/grandyang/p/5138186.html>
10. <https://www.hrwhisper.me/leetcode-create-maximum-number/>
11. <http://www.cnblogs.com/grandyang/p/5100098.html>
12. <https://www.hrwhisper.me/leetcode-maximum-product-of-word-lengths/>
13. <http://www.cnblogs.com/onlyac/p/5155881.html>

---

## Netflix company culture - take some notes (2016-02-05 21:56)

February 5, 2016

Netflix culture - Julia likes to write down and then think about them.

[1][#1](http://www.fastcompany.com/3056187/the-future-of-work/the-woman-who-created-netflixs-enviable-company-culture?partner=cnnmoney&linkId=21014664)

### Netflix culture: Freedom & Responsibility

Enron, whose leader went to jail, and which went bankrupt from fraud, had these values displayed in their lobby:

Integrity  
Communication  
Respect  
Excellence

The actual company values,  
as opposed to the nice-sounding values, are shown by **who gets rewarded, promoted, or let go**

Actual company values are **the behavior and skills that are valued in fellow employees**.  
At Netflix, we particularly value the following nine behaviors and skills in our colleagues...  
... meaning we hire and promote people who demonstrate these nine

### Judgement

You make wise decisions (people, technical, business, and creative) despite ambiguity

You identify root cause, and get beyond treating symptoms

You think strategically, and can articulate what you are, and are not, trying to do

You smartly separate what must be done well now, and what can be improved later

### **Communication**

You listen well, instead of reacting fast, so you can better understand

You are concise and articulate in speech and writing

You treat people with respect independent of their status or disagreement with you

You maintain calm poise in stressful situations

### **Impact**

You accomplish amazing amounts of important work

You demonstrate consistently strong performance so colleagues can rely upon you

You focus on great values rather than on process

You exhibit bias-to-action, and avoid analysis-paralysis

### **Curiosity**

You learn rapidly and eagerly

You seek to understand our strategy, market, customers, and suppliers

You are broadly knowledgeable about business, technology and entertainment

You contribute effectively outside of your specialty

### **Innovation**

You re-conceptualize issues to discover practical solutions to hard problems

You challenge prevailing assumptions when warranted, and suggest better approaches

You create new ideas that prove useful

You keep us nimble by minimizing complexity and finding time to simplify

### **Courage**

You say what you think even if it is controversial

You make tough decisions without agonizing

You take smart risks

You question actions inconsistent with our values

**Passion**

You inspire others with thirst for excellence

You care intensely about Netflix's success

You celebrate wins

You are tenacious

**Honesty**

You are known for candor and directness

You are non-political when you disagree with others

You only say things about fellow employees you will say to their face

You are quick to admit mistakes

**Selflessness**

You seek what is best for Netflix, rather than best for yourself or your group

You are ego-less when searching for the best ideas

You make time to help colleagues

You share information openly and proactively

**Loyalty is Good**

Loyalty is good as a stabilizer

People who have been stars for us, and hit a bad patch, get a near term pass because we think they are likely to become stars for us again

**Hard Work - Not Relevant**

We don't measure people by how many hours they work or how much they are in the office

We do care about accomplishing great work

Sustained B-level performance, despite "A for effort", generates a generous severance package, with respect

Sustained A-level performance, despite minimal effort, is rewarded with more responsibility and great pay

**Brilliant Jerks**

Some companies tolerate them

For us, cost to effective teamwork is too high

Diverse styles are fine - as long as person embodies the 9 values.

**The Rare Responsible Person**

Self motivating  
Self aware  
Self disciplined  
Self improving  
Acts like a leader  
Doesn't wait to be told what to do  
Picks up the trash lying on the floor

Responsible People  
Thrive on Freedom, and are worthy of Freedom

[2]<http://www.ted.com/watch/ted-institute/ted-bcg/patty-mccord-lessons-from-a-silicon-valley-maverick>

1. <http://www.fastcompany.com/3056187/the-future-of-work/the-woman-who-created-netflixs-enviable-company-culture?partner=cnnmoney&linkId=21014664#1>
  2. <http://www.ted.com/watch/ted-institute/ted-bcg/patty-mccord-lessons-from-a-silicon-valley-maverick>
- 

## Do not complain - a tip worthy to share (2016-02-06 00:28)

February 5, 2016

Chinese new year is coming in 2 days. And then, spent a few hours to watch videos about Canadian immigrant story.

"Do not complain", the video about Robert Herjavec, a Canadian story, an immigrant story.

- [1]<https://www.youtube.com/watch?v=u5uD62Plakg>  
[2]<https://www.youtube.com/watch?v=-s9qJ7ATP7w>  
[3]<https://www.youtube.com/watch?v=sYprh2qkeDY>

1. Be great on one thing
2. **Never complain** (Never complain, no one cares.  
short story: Appreciate the opportunity. The father was laid off from sweeping the floor job, did not take unemployment pay when Robert tried to fill the form for his dad, his father did not want anything from the Canada which gives him opportunity. 20 dollars his dad came to Canada with wife and son, taking a boat with a suitcase)
3. Just keep going
4. Create value for your customers
5. Become the person others want to know
6. Listen to yourself
7. Leave your emotions out of it
8. Be able to adapt
9. Find what makes you tick
10. You are in control of everything

### Robert Herjavec: The will to win

[4]<https://www.youtube.com/watch?v=7GxQ9KoaUJ0>

Do not classify people loser. People give up, being a loser. Being a winner is easy, being a loser, keep going. Everything can go, go wrong for me.

**Mental strength** . Train for marathon for 20 days. The way you gain confidence, you just do it.

Just do it. You gain confidence by your experience.

**If you are successful, money will follow** .

Be efficient. Adaptability. If you are dropped in jungle, you will survive.

**With kindness - kills the competition**

[5]<https://www.youtube.com/watch?v=M9Mf2s4OJSU>

Love tech industry - every 3 years it reinvents itself.

**What is the purpose of business** ? Create customer, create value for them. ( Julia's still thinking about it)

**Leave emotion out of it. Be angry, make bad decisions** .

Listening - what people try to say to you.

**Let go** - success is not good at everything. Find thing you are good at. Let go others. The world will reward you with very narrow knowledge.

**Be world class on one thing** . Do not worry about your weakness.

The minute you find that you are successful, it is the end of it. Keep going. Be master everything you choose, react with.

Conclusion:

**Do not complain how time-consuming to improve skills by working on Leetcode algorithms.**

Try to stay calm when performing algorithms. Do not give in nervousness.

Be mental toughness on problem solving. Always focus on current problem. Not worry about next one, or other thing. Try to do small, simple, stupid things first, see if it leads an idea or solution, and then, try to improve, and optimize it if need.

1. <https://www.youtube.com/watch?v=u5uD62Plakg>

2. <https://www.youtube.com/watch?v=-s9qJ7ATP7w>

3. <https://www.youtube.com/watch?v=sYprh2qkeDY>

4. <https://www.youtube.com/watch?v=7GxQ9KoaUJ0>

5. <https://www.youtube.com/watch?v=M9Mf2s4OJSU>

---

**Sunday - Reading time - Career Advice (2016-02-07 11:15)**

February 7, 2016

Working on Leetcode algorithm, Julia took so many breaks when she worked on 5 Leetcode questions on Feb. 6 evening, she found out that she needs to build up mental toughness on the problem solving - using algorithm/ data structure. She wonders why she needs to read something not meaningful while solving those problems, why the algorithm cannot be nature in her life.



So, she turns to advice from her favorite politician, Mitt Romney's top 10 rules for success, a Harvard graduate's advice this Sunday.

[1][https://en.wikipedia.org/wiki/Mitt\\_Romney](https://en.wikipedia.org/wiki/Mitt_Romney)

**1. Have clear objectives -**

Write down clear objectives.

What is my objective to work on Leetcode algorithms?

1. **Constant Reinvent herself** - as a software programmer.
- 2.
- 3.

**2. Stop thinking, start doing**

**3. Failures are inevitable**

**4. Have a life coach**

**5. Do what you enjoy**

More detail: English major -> business school -> Law degree (Harvard law school)

**Do what you enjoy -> not lead more money**

**6. Launch out into the deep**

Master teaches Peter how to fish (Luke 5:4 "Put out into deep water, and let down the nets for a catch." )

Metaphor for the life, do not live in shallow, live in deep. Educate you, service others

**7. Keep your life in perspective**

Perspective is good friend. Find ways in perspective.

Do not study on Sunday. Personal time.

Do not work at home. Devote time to the family. Really focus on the importance things of life.

**8. Devote time for family**

**9. Do your present job well**

Secret to the advancement.

**10. Your choices shape the lives of other people**

And then, spent half hour to read the article. Great time to read. Julia found out that so many interesting things to read in the article, she started to find the joy of life - reading, expand the knowledge, and know the world: education, father and son, and career advice from the father, and many more.

[2][https://en.wikipedia.org/wiki/Mitt\\_Romney](https://en.wikipedia.org/wiki/Mitt_Romney)

1. [https://en.wikipedia.org/wiki/Mitt\\_Romney](https://en.wikipedia.org/wiki/Mitt_Romney)

2. [https://en.wikipedia.org/wiki/Mitt\\_Romney](https://en.wikipedia.org/wiki/Mitt_Romney)

---

## Sunday - Reading time (2) (2016-02-07 12:11)

February 7, 2016

Favorite time to read, learn and understand business world. Have some coaches. Today, Julia writes down another 10 rules to success, Marcus Lemonis.

Marcus Lemonis

[1]<https://www.youtube.com/watch?v=TyOL1rkjl5w>

### 1. **Make business plan**

get a piece of paper, write down the idea, and get feedback:

Here is my product and service.

Here is how I attack the market.

Here is how I beat the competition.

Here is how I learn from my competitors.

Here is I take from my customers.

Here is the people I need to surround myself with.

Answer those questions.

### 2. **Have enough working capital**

### 3. **Create a to do list for tomorrow**

Good habit to follow -

Write down a list to complete at night, and complete it before noon. Afternoon, maybe, a wild wild west.

What you have to do, house keeping.

### 4. **Take care of your employees**

Take care of your employees - they will put customers first.

### 5. **Have the knowledge**

Follow the passion. Work for other for a while. Better have a partner.

### 6. **Constantly reinvent yourself**

Bold step, no matter how old you are. As an entrepreneur.

End up like Sears, not existing any more. Not evolving, will die.

7.

### **Always stay one step ahead**

### 8. **Stick to the grind**

Business owner, team member.

### 9. **Push through the fear of failure**

Fear of failure - cannot ignore them, admit my vulnerability when he turns old around 40s years old.

### 10. **Stand out**

1. <https://www.youtube.com/watch?v=Ty0L1rkjl5w>

---

## Leetcode 318: Maximum Product of Word Length (2016-02-07 14:46)

February 7, 2016

There are a several of stages to go through on Leetcode 318 problem solving today.

10 minutes to read question and think about solution, confused about requirement(????????) ->

20 minutes to read blogs to understand solutions ->

10 minutes to know the detail to implement (1) ->

20 minutes to implement without bugs (2) ->

10 minutes to implement with more clear code with bugs (3) ->

10 minutes debug to read code again, debugging to pinpoint bug ->

60 minutes to work on a better idea - optimal idea (5) ->

work on exceeded time

issues

(20 minutes, instead of 60+ minutes) (6)

Action items:

1. Always work on coding, using Visual Studio. Try to improve coding.

### Leetcode 318: Maximum Product of Word Length

Given a string array words , find the maximum value of  $\text{length}(\text{word}[i]) * \text{length}(\text{word}[j])$  where the two words do not share common letters. You may assume that each word will contain only lower case letters. If no such two words exist, return 0.

[1]<https://www.hrwhisper.me/leetcode-maximum-product-of-word-lengths/>

#### Solution 1:

????????????????-pö????????????????

- `0`
- `words[i].length() * words[j].length()`

Julia's practice:

[2][https://github.com/jianminchen/Leetcode\\_C/blob/master/318MaximumProductOfWordLength.cs](https://github.com/jianminchen/Leetcode_C/blob/master/318MaximumProductOfWordLength.cs)

### Solution 2:

```
int maxProductOfWordLengths(char** words, int wordsSize) {
    int max = 0;
    for (int i = 0; i < wordsSize; i++) {
        for (int j = i + 1; j < wordsSize; j++) {
            if (words[i][0] < words[j][0]) {
                max = max > words[i].length() * words[j].length() ? max : words[i].length() * words[j].length();
            }
        }
    }
    return max;
}
```

- `elements[i] |= 1 << (words[i][j] - 'a');` // `words[i][j]` 26个字母
- `elements[i] & elements[j]` // 两个单词的公共字母

read the blog to understand bit operation, take some time to refresh the memory:

[3]<http://www.cnblogs.com/onlyac/p/5155881.html>

```
int maxProductOfWordLengths(char** words, int wordsSize) {
    int max = 0;
    for (int i = 0; i < wordsSize; i++) {
        for (int j = i + 1; j < wordsSize; j++) {
            if (words[i][0] < words[j][0]) {
                max = max > words[i].length() * words[j].length() ? max : words[i].length() * words[j].length();
            }
        }
    }
    return max;
}
```

```
int maxProductOfWordLengths(char** words, int wordsSize) {
    int max = 0;
    for (int i = 0; i < wordsSize; i++) {
        for (int j = i + 1; j < wordsSize; j++) {
            if (words[i][0] < words[j][0]) {
                max = max > words[i].length() * words[j].length() ? max : words[i].length() * words[j].length();
            }
        }
    }
    return max;
}
```

```
int maxProductOfWordLengths(char** words, int wordsSize) {
    int max = 0;
    for (int i = 0; i < wordsSize; i++) {
        for (int j = i + 1; j < wordsSize; j++) {
            if (words[i][0] < words[j][0]) {
                max = max > words[i].length() * words[j].length() ? max : words[i].length() * words[j].length();
            }
        }
    }
    return max;
}
```

Here is julia's practice:

[4][https://github.com/jianminchen/Leetcode\\_C/blob/master/318MaximumProductOfWordLength\\_B.cs](https://github.com/jianminchen/Leetcode_C/blob/master/318MaximumProductOfWordLength_B.cs)

Read the webpage about precedence and order of evaluation:

[5]<https://msdn.microsoft.com/en-us/library/2bxt6kc4.aspx>

### Solution 3:

Just for fun, write another version of bit manipulation implementation, but Julia created 3 bugs in the row before she writes a correct version.

Instead of using two words do not contain same char,

**if ((s1 & s2) == 0) return true;**

she tried to write a version to check any char from a to z, one by one check version:

[6][https://github.com/jianminchen/Leetcode\\_C/blob/master/318MaximumProductOfWordLength\\_C.cs](https://github.com/jianminchen/Leetcode_C/blob/master/318MaximumProductOfWordLength_C.cs)

Design takes practice. She thought about and then wrote, and then failed 3 times!

1. <https://www.hrwhisper.me/leetcode-maximum-product-of-word-lengths/>
2. [https://github.com/jianminchen/Leetcode\\_C/blob/master/318MaximumProductOfWordLength.cs](https://github.com/jianminchen/Leetcode_C/blob/master/318MaximumProductOfWordLength.cs)
3. <http://www.cnblogs.com/onlyac/p/5155881.html>
4. [https://github.com/jianminchen/Leetcode\\_C/blob/master/318MaximumProductOfWordLength\\_B.cs](https://github.com/jianminchen/Leetcode_C/blob/master/318MaximumProductOfWordLength_B.cs)
5. <https://msdn.microsoft.com/en-us/library/2bxt6kc4.aspx>
6. [https://github.com/jianminchen/Leetcode\\_C/blob/master/318MaximumProductOfWordLength\\_C.cs](https://github.com/jianminchen/Leetcode_C/blob/master/318MaximumProductOfWordLength_C.cs)

## Leetcode 322: Coin Change (2016-02-08 01:10)

February 8, 2016

Julia likes to build a good fun memory about dynamic programming design, coding experience. Let this one - coin change build up good memory about Dynamic Programming.

### 322 Coin change

[1]<http://www.cnblogs.com/grandyang/p/5138186.html>

Dynamic Programming 'dp[i] = min(dp[i], dp[i - coins[j]] + 1);

```
dp[i] = min(dp[i], dp[i - coins[j]] + 1);
```

coins[j] - coins[j] i - coins[j] dp[i] r 1 dp Φp

To be continued.

1. <http://www.cnblogs.com/grandyang/p/5138186.html>

## Leetcode 319: Bulb Switch (2016-02-08 01:14)

February 8, 2016

319

### Bulb Switch

[1]<http://www.cnblogs.com/grandyang/p/5100098.html>

Analysis from the above blog:

5 'X' - '√'

X X X X X

√ √ √ √

√ X √ X √

√ X X X √

√ X X √ √

√ X X √ X

????????????????1??4????????????????????-gJ????????????????????n?????X????n????????-  
????????????n????????????n?36????????(1,36), (2,18), (3,12), (4,9), (6,6), ??????????????Ü????????????  
????????????-Xψ????????????????????(6,6)????6????????? o?????????????Q????????????-  
????????????????-??????????????????????????????????  
????????????1?n????????????????-4??force brute????1????????n????

To be continued.

1. <http://www.cnblogs.com/grandyang/p/5100098.html>

**Leetcode : Wiggle Sort (2016-02-08 01:20)**

February 8, 2016

**Wiggle Sort**

Given an unsorted array nums, reorder it in-place such that nums[0] <= nums[1] >= nums[2] <= nums[3]....

Julia figured out the easy way to do it, O(n) time, O(1) space. So motivated. Next time, think about algorithm by yourself, start from brute force, and then, work towards requirement - efficient solution.

[1]<https://segmentfault.com/a/1190000003783283>

```
public
class
Solution
{
    public
    void
    wiggleSort
    ( int [] nums)
    {
        for
        (
        int
        i =
        1
        ; i < nums.length; i++) {
            // ?????????nums[i] < nums[i - 1]????nums[i] > nums[i - 1]
```

```

((i %
2
==
1
& & nums[i] < nums[i-
1
]) || (i %
2
==
0
& & nums[i] > nums[i-
1
])) {
int
tmp = nums[i-
1
]; nums[i-
1
] = nums[i]; nums[i] = tmp; } } }

```

To be continued.

1. <https://segmentfault.com/a/1190000003783283>

---

Unknown (2016-03-20 02:56:30)

I'm not sure this is a correct answer. Try input with {4,3,2,2,1,0}. your code gives {3, 4, 2, 2, 0, 1}. Can you please clarify?

Jianmin Chen (2016-03-21 12:36:33)

This comment has been removed by the author.

Jianmin Chen (2016-03-22 12:37:50)

This comment has been removed by the author.

Jianmin Chen (2016-03-22 21:33:02)

Thanks for asking. I noticed that the algorithm is for Wiggle Sort Algorithm, not Leetcode 324 Wiggle Sort II.

## Leetcode 328: Odd Even Linked List (Easy) (2016-02-08 01:22)

February 8, 2016

### 328. Odd Even Linked List (Easy)

Julia's comment: think about  $O(N)$  space solution first (using array, and then, easy to go through), and then, work on  $O(1)$  space solution, the following blog helps:

[1]<http://www.cnblogs.com/EdwardLiu/p/5138199.html>

To be continued.

1. <http://www.cnblogs.com/EdwardLiu/p/5138199.html>

---

### Leetcode 329: Longest increasing path in matrix (2016-02-08 01:27)

February 8, 2016

problem statement:

Given an integer matrix, find the length of the longest increasing path. From each cell, you can either move to four directions: left, right, up or down. You may NOT move diagonally or move outside of the boundary (i.e. wrap-around is not allowed).

#### 329. Longest Increasing Path in a matrix

##### Example 1:

```
nums = [
  [
    9
  ,9,4],
  [
    6
  ,6,8],
  [
    2
  ,
  1
  ,1]
]
```

Return 4

The longest increasing path is [1, 2, 6, 9].

##### Example 2:

```
nums = [
  [
    3
  ,
  192
```



```

4
,
5
],
[3,2,
6
],
[2,2,1]
]

```

Return 4

The longest increasing path is [3, 4, 5, 6]. Moving diagonally is not allowed.

idea: go through each node in the matrix, BFS search, and get minimum one;

read blog:

[1]<http://bookshadow.com/weblog/2016/01/20/leetcode-longest-increasing-path-matrix/>

June 20, 2016

1st practice using C #:

[2]<https://gist.github.com/jianminchen/b984ccba8dabbb0bb78160c6e5c6e8b4>

blog:

[3]<http://juliachencoding.blogspot.ca/2016/06/leetcode-329-longest-increasing-path-in.html>

1. <http://bookshadow.com/weblog/2016/01/20/leetcode-longest-increasing-path-matrix/>

2. <https://gist.github.com/jianminchen/b984ccba8dabbb0bb78160c6e5c6e8b4>

3. <http://juliachencoding.blogspot.ca/2016/06/leetcode-329-longest-increasing-path-in.html>

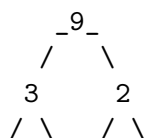
## Leetcode 331: Verify Preorder serialization of a Binary Tree (2016-02-08 01:39)

Feb. 8, 2016

Serialization of a binary tree, great algorithm to review.

### 331. Verify Preorder serialization of a Binary Tree

One way to serialize a binary tree is to use pre-order traversal. When we encounter a non-null node, we record the node's value. If it is a null node, we record using a sentinel value such as #.



```

  4   1   #   6
 / \ / \ / \
# # # #   # #

```

For example, the above binary tree can be serialized to the string "9,3,4,#,#,1,#,#,2,#,6,#,#", where # represents a null node.

idea: Use stack

Here is the blog she starts to read.

[1]<http://bookshadow.com/leetcode/>

[2]<https://www.hrwhisper.me/leetcode-algorithm-solution/>

[3]<http://www.cnblogs.com/grandyang/p/4606334.html>

[4]<http://www.cnblogs.com/EdwardLiu/tag/Leetcode/>

To be continued.

1. <http://bookshadow.com/leetcode/>
2. <https://www.hrwhisper.me/leetcode-algorithm-solution/>
3. <http://www.cnblogs.com/grandyang/p/4606334.html>
4. <http://www.cnblogs.com/EdwardLiu/tag/Leetcode/>

## Leetcode 295: Find median from data stream (2016-02-09 00:22)

February 8, 2016

It is always very important to write some code and then get the experience to master a new algorithm.

### Leetcode 295: Find median from data stream

[1] <https://segmentfault.com/a/1190000003709954>

[2] <https://gist.github.com/jianminchen/31572a64b2af48e3ccce>

[3] <http://buttercola.blogspot.ca/2015/12/leetcode-find-median-from-data-stream.html>

Julia, read Java PriorityQueue class and get some ideas about the class design:

[4] <http://www.programcreek.com/2009/02/using-the-priorityqueue-class-example/>

[5] <http://stackoverflow.com/questions/683041/java-how-do-i-use-a-priorityqueue>

understand priority queue first, read the lecture notes:

[6] <http://www.eecs.wsu.edu/~ananth/CptS223/Lectures/heaps.pdf>

To be continued.

1. <https://segmentfault.com/a/1190000003709954>
  2. <https://gist.github.com/jianminchen/31572a64b2af48e3ccce>
  3. <http://buttercola.blogspot.ca/2015/12/leetcode-find-median-from-data-stream.html>
  4. <http://www.programcreek.com/2009/02/using-the-priorityqueue-class-example/>
  5. <http://stackoverflow.com/questions/683041/java-how-do-i-use-a-priorityqueue>
  6. <http://www.eecs.wsu.edu/~ananth/CptS223/Lectures/heaps.pdf>
- 

## Algorithm: Write a function to detect if the string has the unique character (2016-02-09 00:25)

February 8, 2016

So excited to have chance to write code for a new algorithm.

Write a function to detect if the string has the unique character.

Read the blog :

[1]<http://stackoverflow.com/questions/19484406/detecting-if-a-string-has-unique-characters-comparing-my-solution-to-cracking>

this blog provides good answers for the question of unique character:

Notice that this method doesn't allocate an array of booleans. Instead, it opts for a clever trick. Since there are only 26 different characters possible and there are 32 bits in an

int

, the solution creates an

int

variable where each bit of the variable corresponds to one of the characters in the string. Instead of reading and writing an array, the solution reads and writes the bits of the number.

Julia's comment: first blog about bit manipulation - surprising, after the reading, it is much easy to understand the code:

Two bit operation:

$1 \ll \text{val}$

$| =$

It is always helpful to write a few of small function:

```
int toInt ( c har c
)
{
```

```
    return c-'a';
```

```
}
```

```
int OneShiftLeftNbits (int val)
```

```
{
```

```
    return 1 << val;
```

```
}
```

```
int checkNthBit(int val)
```

```
{
```

```
    // ...
```

```
}
```

```
public
```

```
static
```

```
boolean
```

```
    isUniqueChars
```

```
(
```

```
String
```

```
    str
```

```
)
```

```
196
```

```
{

if

(
str
.
length
()

>

256

)

{

// NOTE: Are you sure this isn't 26?

return

false

;
```

```
}
```

```
int
```

```
    checker
```

```
=
```

```
0
```

```
;
```

```
for
```

```
(
```

```
int
```

```
    i
```

```
=
```

```
0
```

```
;
```

```
    i
```

```
<
```

```
    str
```

```
.
```

```
length
```

```
());
```

```
    i
```

```
198
```

```
++)
```

```
{
```

```
int
```

```
val
```

```
=
```

```
str
```

```
.
```

```
charAt
```

```
(
```

```
i
```

```
)
```

```
-
```

```
'a'
```

```
;
```

```
if
```

```
((
```

```
checker
```

```
&
```

(

1

{\textless}{\textless}

val

))

>

0

)

return

false

;

checker

|=

(

1

{\textless}{\textless}

val

);

200



```
}
```

```
return
```

```
true
```

```
;
```

```
}
```

[2][http://javahungry.blogspot.com/2014/11/string-has-all-unique-character s-java-example.html](http://javahungry.blogspot.com/2014/11/string-has-all-unique-character-s-java-example.html)

To be continued.

1. <http://stackoverflow.com/questions/19484406/detecting-if-a-string-has-unique-characters-comparing-my-solution-to-cracking>
  2. <http://javahungry.blogspot.com/2014/11/string-has-all-unique-characters-java-example.html>
- 

## Algorithm: Possible Triangle (2016-02-09 00:29)

February 8, 2016

### Possible triangle

[1]<http://www.geeksforgeeks.org/find-number-of-triangles-possible/>

[2]<http://stackoverflow.com/questions/8110538/total-number-of-possible-triangles-from-n-numbers>

Argue about how to reduce time complexity from  $O(N^3)$  to  $O(N^2)$ , how exactly  $i, j, k$  three variable,  $k$  is only from 0 to  $n-1$  once for each  $i$ , nothing to do with  $j$ . So, it is time for  $i$  - from 1 to  $N$ , and time for  $j$  from 1 to  $N$  two loops, and then for  $i - N$ , and  $k$  from 1 to  $N$  ( **skip j - big point! Cannot figure out easily.** )

To be continued.

1. <http://www.geeksforgeeks.org/find-number-of-triangles-possible/>
  2. <http://stackoverflow.com/questions/8110538/total-number-of-possible-triangles-from-n-numbers>
-

## Leetcode 310: Minimum Height Trees (2016-02-09 00:41)

### Leetcode 310: Minimum Height Trees

[1] <https://leetcode.com/problems/minimum-height-trees/> #

read the blog:

[2] <http://bookshadow.com/weblog/2015/11/26/leetcode-minimum-height-trees/>

[3] <https://leetcode.com/discuss/71656/c-solution-o-n-time-o-n-space>

[4] <http://likemyblogger.blogspot.ca/2015/11/leetcode-310-minimum-height-trees.html>

[5] <http://buttercola.blogspot.ca/2016/01/leetcode-minimum-height-trees.html>

To be continued.

1. <https://leetcode.com/problems/minimum-height-trees/>

2. <http://bookshadow.com/weblog/2015/11/26/leetcode-minimum-height-trees/>

3. <https://leetcode.com/discuss/71656/c-solution-o-n-time-o-n-space>

4. <http://likemyblogger.blogspot.ca/2015/11/leetcode-310-minimum-height-trees.html>

5. <http://buttercola.blogspot.ca/2016/01/leetcode-minimum-height-trees.html>

---

## Leetcode 316: Remove duplicate letters (2016-02-09 00:46)

### Leetcode: Remove duplicate letters

Given a string which contains only lowercase letters, remove duplicate letters so that every letter appear once and only once. You must make sure your result is the smallest in lexicographical order among all possible results.

Example:

Given "bcabc"

Return "abc"

Given "cbacdcbc"

Return "acdb"

Blogs to read:

[1] <http://bookshadow.com/weblog/2015/12/09/leetcode-remove-duplicate-letters/>

3 solutions are discussed in the following blog:

[2] <https://www.hrwhisper.me/leetcode-remove-duplicate-letters/>

[3] <https://leetcode.com/discuss/73777/easy-to-understand-iterative-java-solution>

[4] <http://buttercola.blogspot.ca/2016/01/leetcode-remove-duplicate-letters.html>

To be continued.

1. <http://bookshadow.com/weblog/2015/12/09/leetcode-remove-duplicate-letters/>

2. <https://www.hrwhisper.me/leetcode-remove-duplicate-letters/>

3. <https://leetcode.com/discuss/73777/easy-to-understand-iterative-java-solution>
  4. <http://buttercola.blogspot.ca/2016/01/leetcode-remove-duplicate-letters.html>
- 

## Leetcode: Longest increasing subsequence (2016-02-09 00:48)

February 8, 2016

### Leetcode: Longest increasing subsequence

[1] <http://blog.csdn.net/kenden23/article/details/17632821>

spend 10 minutes to read the blog:

[2] <http://www.kancloud.cn/kancloud/data-structure-and-algorithm-notes/73075>

[3] <http://www.geeksforgeeks.org/dynamic-programming-set-3-longest-increasing-s subsequence/>

Previous blog:

[4] <http://juliachencoding.blogspot.ca/2015/06/dynamic-programming-longest-increasing.html>

I

Write some C # code for this algorithm later.

[5] <http://buttercola.blogspot.ca/2015/12/leetcode-longest-increasing-subsequence.html>

NLogn solution, better than DP solution, Recursive solution

[6] <http://www.geeksforgeeks.org/longest-monotonically-increasing-subsequence -size-n-log-n/>

Set Goals:

1. Know how to solve it using recursive solution first;
2. And then, understand DP solution, how to build the formula, avoid duplication;
3. And then, know where to find optimal solution here,  $O(n \log n)$ ,

Julia, write some code - That is most important!

To be continued.

February 11, 2016

**More important, work on brute force solution first, and then, come out ideas to improve. Coding is next stage.**

Julia likes to work on the algorithm by herself, not depending on her experience/ memory of solved problems. So, she tried in 20 minutes to come out the idea:

**Write down her analysis steps roughly:**

1. Think how to use brute force solution first,

For array 0-n, brute force solution, how many choice for length 1 subsequence - choose 1 from n,  
length i subsequence - choose i from n, and then, in total, it is  $2^n$  calculation

2. next, she thought about and had difficulty to write down and developed the idea, in 10 minutes, she decided to work on an example to help, easily she did figure out one algorithm besides brute force:

Given an array, size of 7, `int[] A = {`

1, 8, 3, 2, 5, 4, 7 }

for subarray of size 6, 1, 3, 5 - longest increasing subsequence, maximum 3,

so longest one for size of 7 is 1, 3, 5, 7

But, if the array is changed to: 1, 8, 3, 2, 8, 4, 7, and then, maximum subsequence of subarray (size of 6) has two choices:

**1, 3, 8** <- because  $7 < 8$ , no increment

but, **1, 2, 4** <- because  $7 > 4$ , increment 1, so longest subsequence is 4, so need to maintain all the sequences with length 3

And then, try to record the number of each element in the array - maximum value ends to i

1 8 3 2 8 4 7

1 2 2 2 **3 3** 4 <- so, when you work on 7, you can check value ends on 3 first, to see if it can make 4, otherwise, go for 3, decrease one by one. extra space is  $O(N)$

one more example:

2 8 3 1 8 4 7

1 2 2 **1**

**3 3** 4 <- so, when you work on 7, you can only check value ends on 3, extra space is  $O(N)$

**Julia, that is the way you should work on the algorithm. There are thousands of problems, you cannot spend time on each one of them, you have to learn how to solve a new problem by yourself. And later, write down your improvement on the original thoughts.**

**Think about space to store the value - array, extra map for key - value list. To simplify, just go through the array from 0 to i-1, and then, compare  $A[i]$  to  $A[j]$  (j from 0 to i-1), if it is bigger, then compute maximum value.  $O(N^2)$  time complexity.**

**February 12, 2016**

**Work on the DP solution formula - how to calculate the value based on previous one**

**Formula of DP:**

**And then, one more advance to optimal solution (read the blog and get the idea:**

**[7]**

**[8] <http://www.geeksforgeeks.org/longest-monotonically-increasing-subsequence-size-n-log-n/>**

**):**

**For each sequence ends at same distance, for example, 2, you only need to keep the minimum one for future to compare,**

**Here is the detail:**

2 8 3 1 8 4 7

1 2 2 **1**

**3 3** 4

204

Length =1

2

1

Length = 2

2 8

2 3

what if only keep 2 3,

Length

= 3

2 3 8

2 3 4

so, for every length, keep the minimum value sequence, save space, reduce time complexity.

**How much advance in time complexity for this work?**

**Julia, you are very close to the optimal solution !**

1. <http://blog.csdn.net/kenden23/article/details/17632821>
2. <http://www.kancloud.cn/kancloud/data-structure-and-algorithm-notes/73075>
3. <http://www.geeksforgeeks.org/dynamic-programming-set-3-longest-increasing-subsequence/>
4. <http://juliachencoding.blogspot.ca/2015/06/dynamic-programming-longest-increasing.htm>
5. <http://buttercola.blogspot.ca/2015/12/leetcode-longest-increasing-subsequence.html>
6. <http://www.geeksforgeeks.org/longest-monotonically-increasing-subsequence-size-n-log-n/>
7. <http://www.geeksforgeeks.org/longest-monotonically-increasing-subsequence-size-n-log-n/>
8. <https://www.blogger.com/null>

---

**Video watching: The Future of Analytics (2016-02-10 22:05)**

Feb. 10, 2016

Spent one hour to browse through the linkedin profiles, and then, she came cross the profile of Joseph Sirosh - Corporate Vice President, Data Group at Microsoft ([1]<https://www.linkedin.com/in/joseph-sirosh-39803b1>), and learned a few things through the search. Watch this presentation:

**The Future of Analytics**

Speaker: Joseph Sirosh

[2]<https://channel9.msdn.com/events/Cortana-Analytics-Suite/CA-Suite-Workshop-10-11SEP15/The-Future-of-Analytics>

Julia likes the presentation, the metaphor used in the talk: about 30 years ago, tailor to customize the clothes for each one; now, mass manufacturer for all body shapes. So, data analytics now is like 30 year clothing industry, but in the future, it is not a big deal.

[3]<http://www.pcworld.com/article/3006609/why-microsofts-data-chief-thinks-current-machine-learning-tools-are-like-tailored-shirts.html>

And she likes the music in last 2 minutes in the above presentation. Remind her the favorite song:

Genie Bouchard 2014 Montage

[4]<https://www.youtube.com/watch?v=JZJ8K5857dk>

1. <https://www.linkedin.com/in/joseph-sirosh-39803b1>

2. <https://channel9.msdn.com/events/Cortana-Analytics-Suite/CA-Suite-Workshop-10-11SEP15/The-Future-of-Analytics>

3. <http://www.pcworld.com/article/3006609/why-microsofts-data-chief-thinks-current-machine-learning-tools-are-like-tailored-shirts.html>

4. <https://www.youtube.com/watch?v=JZJ8K5857dk>

---

## Sunday: Reading time (3) (2016-02-13 10:58)

February 13, 2016

Being a software programmer, Julia likes to improve her English along the way. She started to bring writing into her daily life, write a small topic making sense everyday. She enjoyed reading the blogs today:

[1]<http://randsinrepose.com/archives/weblog-writing/>

[2]<http://www.amazon.com/gp/product/0596155409?ie=UTF8&tag=beigee-20&linkCode=as2&camp=1789&creative=9325&creativeASIN=0596155409>

[3]<http://randsinrepose.com/archives/what-to-do-when-youre-screwed/>

Here is the video she watched:

Robert Johnson's Top Rules For Success (African billionaire - own TV network)

[4][https://www.youtube.com/watch?v=HaUe-YGAK\\_E](https://www.youtube.com/watch?v=HaUe-YGAK_E)

1. Build relationships

**People like to do business they like**

2. Get the capital you need

3. Keep Revenues up, costs down

4. **Make Friends Before You Need Them**

5. Stop Consuming, Start saving

Debt: you consume more than you are worth.

Postpone unnecessary, impulsive spending

6. Stand for something

7. Get to scale

8. **#Believe in yourself**

People follow leader, people invest on leaders

Sense of confidence, sense of leadership

9. **Make hard choices**

**People cannot afford the bill in restaurant, keep ordering, or until he can run away avoiding the bill.**

10. Partner with suppliers

Trade equity with services.

Mark Cuban

: Only Morons Start a Business on a Loan

[5]<https://www.youtube.com/watch?v=KYneLGRTgy8>

Mark Cuban's Advice to High Schoolers and College Grads

[6]<https://www.youtube.com/watch?v=UgdNTzul27k>

For high school graduate, do not pay overpriced education to get into debts over 40,000.

For graduate, you may not get best job. You paid money to learn. Now it is your chance to get paid to learn.

You do not need perfect job, you can make some money, also you can learn, add to your profile of knowledge and add to your foundation. As much as we want, know exactly, what mission of our life is, chosen profession going to be in 20 years, **You are get paid to learn**, and learn more about yourself, and business, and see what it takes.

Mark Cuban: The best advice I never got

[7]<https://www.youtube.com/watch?v=XuCCBiYLoDw>

Mark Cuban's 12 Rules for Startups

[8]<https://www.youtube.com/watch?v=camXWnD4QcI>

1. Don't start a company unless it's an obsession or something you love

2. If you have an exit strategy, it's not an obsession.

3. Hire people who you think will love working there.

4. Sales will cure all.

Know how your company will make money and how you will actually make sales.

5. Know your core competencies and focus on being great at them. Pay up for people in your core competencies. Get the best.

6. **An espresso machine? Are you kidding me? Coffee is for closers. There are 24 hours in a day, and if people like their jobs, they'll find ways to use as much of it as possible to do their jobs.**

7. **No private offices**

**Open office spaces keep everyone in tune with what's going on and keep the energy up.**

8. **About technology**

9. **Keep the organization**

If you have managers reporting to managers in a startup, you will fail.

10. Never buy swag.

A sure sign of failure for a startup is when someone sends out logo-embroidered polo shirts.

11. Never hire a PR firm

Whenever you see any information related to your field, get the email of the person publishing it and send them a message introducing yourself and the company. They'll welcome hearing from the founder instead of some PR flack.

12. Make the job fun for employees.

Keep a pulse on the stress levels and accomplishments of your people and reward them.

The 7 Sleep habits of Successful Entrepreneurs

Sleep affects moods, increases risk of psychiatric disorders and depression, cardiovascular disease and lowers immune system health.

1. Avoid alcohol before bedtime

2. Turn off electronics before bedtime.

3. Write your worries away.

4. Create the perfect ambience.

5. Exercise (release serotonin, dopamine to brain)

6. Avoid sugar before bedtime, but select protein and fat

7. Wake up to the light.

Larry Page Q &A Zeitgeist Americas 2012

(38 minutes)

[9]<https://www.youtube.com/watch?v=4Mzlp6mlaC4> &feature=youtu.be

Google map, no-one driving car, Android device, tradition education, youtube, etc.

Mark Cuban at USC | Full Interview | 2015

[10]<https://www.youtube.com/watch?v=fs9Gr8Gj6Cg>

Give out advice: Remember 3 things, key to success:

1. Find something you love to do, good at it.

2. Put yourself in other people's shoes. Think about for other people.

3. **The way to succeed is to remove the stress around you.** Those people start to work with you, and then you just do the work.

1. <http://randsinrepose.com/archives/weblog-writing/>

2. <http://www.amazon.com/gp/product/0596155409?ie=UTF8&tag=beigee-20&linkCode=as2&camp=1789&creative=9325&creativeASIN=0596155409>

3. <http://randsinrepose.com/archives/what-to-do-when-youre-screwed/>

4. [https://www.youtube.com/watch?v=HaUe-YGAK\\_E](https://www.youtube.com/watch?v=HaUe-YGAK_E)

5. <https://www.youtube.com/watch?v=KYneLGRTgy8>

6. <https://www.youtube.com/watch?v=UgdNTzul27k>



7. <https://www.youtube.com/watch?v=XuCCBiYLoDw>
  8. <https://www.youtube.com/watch?v=camXWnD4QcI>
  9. <https://www.youtube.com/watch?v=4Mzlp6mIaC4&feature=youtu.be>
  10. <https://www.youtube.com/watch?v=fs9Gr8Gj6Cg>
- 

## Pluralsight: Responsive In-Browser Web Page Design with HTML and CSS (2016-02-16 19:39)

February 16, 2016

So good to come cross this course, and then spend 3 hours to catch up responsive design in CSS. Save me a lot of time to catch up CSS/ responsive design.

Pluralsight:

Responsive In-Browser Web Page Design with HTML and CSS

[1]<https://app.pluralsight.com/library/courses/responsive-browser-web-page-design-html-css-2262/table-of-contents>

A lot of thing to read:

[2]<https://css-tricks.com/scale-svg/> (Feb. 21, 2016)

1. <https://app.pluralsight.com/library/courses/responsive-browser-web-page-design-html-css-2262/table-of-contents>
  2. <https://css-tricks.com/scale-svg/>
- 

## Leetcode 312: Burst Balloons (2016-02-17 23:27)

February 17, 2016

Spent 10 minutes to work on an example,  $int[] A = \{1,2,3,4,5\}$ , and see how to work out maximum coins?

Need to figure out Dynamic Programming formula, how to get the analysis done in a reasonable way? The problem is rated as medium, hard. So, take time to figure out. Since most of questions are easy to medium, let Julia put some creative, passion and other important things into the analysis, get more confident on hard questions.

Feb. 18, 2016 continue to work on the analysis

One phrase Julia likes is " **Things have to do is called stress, but thing love to do is called passion** ". Make this analysis is Julia's first passion practice - how to approach the problem?

Try to work on brute force solution first.

work on example,  $\text{int[] } A = \{1, 2, 3, 4, 5\}$

First one to burst, 5 choices: any one of them, so

case 1: burst 1, left:  $\{2, 3, 4, 5\}$

2: burst 2, left:  $\{1, 3, 4, 5\}$

3: burst 3, left:  $\{1, 2, 4, 5\}$

4: burst 4, left:  $\{1, 2, 3, 5\}$

5: burst 5, left:  $\{1, 2, 3, 4\}$

And choose max value from the above 5 case. Each case is a subproblem, find max value from array size of 4. <- not exactly <- ? **Julia, miss something important here**

Use start, end index of array, and max value is denoted as  $P(\text{start}, \text{end})$

so,  $P(0, 4)$  can be divided into 5 cases:

case 1: burst 1, left:  $P(1, 4)$ , how to calculate the value,  $A[0]$ ,  $A[1]$ ,  $P(1, 4)$ , value =  $A[0] * A[1] + P(1, 4)$

case 2: burst 2, left:  $P(0, 0)$ ,  $P(2, 4)$ , value =  $P(0, 0) + A[0] * A[1] * A[2] + P(2, 4)$

...

case 5: burst 5, left:  $P(0, 3)$ , value =  $P(0, 3) + A[3] * A[4]$ ,

So, the max value is  $P(0, 4)$ , the value is max value of subproblems. Formula:

$\max \{P(0, k-1) + \text{product}(k) + P(k+1, n)\}$ ,  $k = 0, 1, \dots, n$

Now, the dynamic programming formula is constructed.

So, write the design of DP - memorization,  $P(\text{start}, \text{end})$  - declare two dimension array  $\text{vector}[n][n]$

Feb. 19, 2016

**Still need to work on implementation**, go for DP - how to build loops?

Go for recursive solution -

Go for Divide and Conquer solution -

Here is the blog she starts to read.

[1]<http://bookshadow.com/leetcode/>

[2]<https://www.hrwhisper.me/leetcode-algorithm-solution/>

[3]<http://www.cnblogs.com/grandyang/p/4606334.html>

[4]<http://www.cnblogs.com/EdwardLiu/tag/Leetcode/>

[5]<http://www.jiuzhang.com/problem/>

Similar problem like Floyd shortest distance.

1. <http://bookshadow.com/leetcode/>
  2. <https://www.hrwhisper.me/leetcode-algorithm-solution/>
  3. <http://www.cnblogs.com/grandyang/p/4606334.html>
  4. <http://www.cnblogs.com/EdwardLiu/tag/Leetcode/>
  5. <http://www.jiuzhang.com/problem/>
- 

## **New School - HackerRank - algorithm: Beautiful Pairs (2016-02-21 13:09)**

February 21, 2016

HackerRank is Julia's new school - her new lover. But, problem statement is too long to read, and the test cases are not very clear.

Julia starts a new journey with HackerRank, she likes the algorithm problem solving.

She spent 2 hours on Sunday morning to work on an easy problem in a " **101 Hack Feb 2016** ".

Here is the question:

### **Beautiful pairs**

#### **Problem Statement:**

[1]<https://github.com/jianminchen/HackRank/blob/master/beautifulPairs/beautifulPairs.jpg>

Here is her answer with a bug - failed 2 of 6 test cases:

[2]<https://github.com/jianminchen/HackRank/blob/master/beautifulPairs/beautifulPairs.cs>

#### **Performance review:**

40 minutes - calm down, read problem statement - **she has to understand the problem first** .

Failed 2 cases - that is the value of HackerRank - good practice.

#### **Lesson learned:**

20 minutes coding,

20 minutes bug fix: Array size: 1000->1001 to remove run time error,

the score: from 3.8 to 15.80.

Wrong answer for 2 test cases.

Here is the perfect version - fixed the bug.

[3][https://github.com/jianminchen/HackRank/blob/master/beautifulPairs/beautifulPairs\\_BugFix.cs](https://github.com/jianminchen/HackRank/blob/master/beautifulPairs/beautifulPairs_BugFix.cs)

### Conclusion:

Each school is different, HackerRank is cool! Julia, just be humble. Make mistakes, always work on easy question, work on the first one in next 5-10 practice. One question a time.

Julia, you have to go through hackerRank contests, go through training - article detailed on this:

[4]<https://www.facebook.com/notes/10151298476823920>

- You'll learn how to critically analyze your work. Because you don't get credit for solving a problem until the code you write can generate the correct results for a large input set (and you don't know what that input set looks like), you're forced to think about things such as time complexity, memory usage, and nasty corner cases. Importantly, most of this work happens outside the context of a debugger. Debuggers are invaluable tools for figuring out why a given piece of code is buggy, but it's better if you can write bug-free code in the first place. In an interview situation, candidates who can't statically analyze their code generally have trouble showing their solution is correct (or figuring out why it's not).

[5]<https://medium.com/@dpup/whiteboarding-4df873dbba2e#.ps92c99lb>

[6]<https://code.google.com/codejam/contests.html>

[7]<https://code.google.com/codejam/contest/6224486/dashboard#s=a&a=0>

Be a better programmer to grow in your current job -

[8]<http://blog.hackerrank.com/3-ways-crush-technical-interview/>

[9]<http://dandreamsofcoding.com/2014/03/18/dissecting-an-interview-question/>

[10]<http://dandreamsofcoding.com/2015/01/09/dissecting-an-interview-question-math-is-hard/>

[11]<http://dandreamsofcoding.com/2014/08/01/dissecting-an-interview-question-reconstructing-a-tree/>

1. <https://github.com/jianminchen/HackRank/blob/master/beautifulPairs/beautifulPairs.jpg>

2. <https://github.com/jianminchen/HackRank/blob/master/beautifulPairs/beautifulPairs.cs>

3. [https://github.com/jianminchen/HackRank/blob/master/beautifulPairs/beautifulPairs\\_BugFix.cs](https://github.com/jianminchen/HackRank/blob/master/beautifulPairs/beautifulPairs_BugFix.cs)

4. <https://www.facebook.com/notes/10151298476823920>

5. <https://medium.com/@dpup/whiteboarding-4df873dbba2e#.ps92c99lb>

6. <https://code.google.com/codejam/contests.html>

7. <https://code.google.com/codejam/contest/6224486/dashboard#s=a&a=0>

8. <http://blog.hackerrank.com/3-ways-crush-technical-interview/>

9. <http://dandreamsofcoding.com/2014/03/18/dissecting-an-interview-question/>

10. <http://dandreamsofcoding.com/2015/01/09/dissecting-an-interview-question-math-is-hard/>

11. <http://dandreamsofcoding.com/2014/08/01/dissecting-an-interview-question-reconstructing-a-tree/>

---

## HackerRank - New School - Algorithm: Fix the cycles (2016-02-21 21:37)

Feb. 21, 2016

Another 2 hours on hackerRank:

### **Fix the cycles :**

Problem Statement:

[1]<https://github.com/jianminchen/HackRank/blob/master/fixTheCycles/fix-the-cycles-English.jpg>

Solution:

There are 4 cycles which share same edge: D->A, so if set D->A big enough then any cycle can be positive.

4 cycles:

A B C D A

A C D A

A B D A

A C D A

Simple solution.

30 minutes to read - **take too long!**

10 minutes to write code,

[2]<https://github.com/jianminchen/HackRank/blob/master/fixTheCycles/FixCycles.cs>

### **Short term target of training on hackerRank :**

Easy question,

reading time: 10 minutes,

code time: 10 minutes.

Know how to fix bugs, make more points

1. <https://github.com/jianminchen/HackRank/blob/master/fixTheCycles/fix-the-cycles-English.jpg>

2. <https://github.com/jianminchen/HackRank/blob/master/fixTheCycles/FixCycles.cs>

---

## **HackerRank - New School - Nice code sprint on algorithm challenges (2016-02-21 21:39)**

Feb. 21, 2016

Sign up for another test:

### **Juniper coding challenge**

[1]<https://www.hackerrank.com/juniper-codesprint>

Work on 4 questions:

[2]<https://www.hackerrank.com/contests/juniper-codesprint/challenges/leonardo-and-substring>

The blog documented her experience:

[3]<http://juliachencoding.blogspot.ca/2016/02/code-challenge-count-of-substring.html>

1. <https://www.hackerrank.com/juniper-codesprint>
  2. <https://www.hackerrank.com/contests/juniper-codesprint/challenges/leonardo-and-substring>
  3. <http://juliachencoding.blogspot.ca/2016/02/code-challenge-count-of-substring.html>
- 

## HackerRank - New School - JavaScript (2016-02-22 00:12)

February 21, 2016

Just sign up JavaScript week 2. It is a new year - 2016, make JavaScript programming language my favorite language. Spend time to play with code challenge on JavaScript.

Read more about JavaScript and get prepared.

To entertain - learning more about JavaScript:

[1]<http://rauschma.2ality.com/publications.html>

Speaking JavaScript: (read online)

[2]<http://speakingjs.com/es5/index.html>

1. <http://rauschma.2ality.com/publications.html>
  2. <http://speakingjs.com/es5/index.html>
- 

## Mock interview experience (I) (2016-02-24 00:06)

February 23, 2016

Julia never had chance to give other people code interview before, and then, she will have one. Excited! Finally, she got her first experience.

Algorithm for the interview being an interviewer:

### ***Find The Duplicates***

Given two arrays of US social security numbers: Arr1 and Arr2 of lengths n and m respectively, how can you most efficiently compute an array of all persons included on both arrays?

Solve and analyze the complexity for 2 cases:

1.  $m \approx n$  - lengths are approximately the same
2.  $m \gg n$  - one is much longer than the other

Julia worked on the question by herself first, around 20 minutes (no coding) before she reads :

1. Brute force solution, time complexity:  $O(nm)$ , go through two loops, in other words, go through Arr1, and for each element in Arr1, check if it is in Arr2.

2. Can we do better than  $O(nm)$ ? Of course.

case 1:  $m$

$\approx$

$n$

- lengths are approximately the same

sort Arr1, it takes time  $O(n \log n)$ ;

and then, for each node in Arr2, find it is duplicated one in Arr1. Use binary search, each search takes  $O(\log n)$ ,  $m$  elements, so time complexity is  $O(m \log n)$

So, time complexity in total:  $O(n \log n) + O(m \log n)$

$\approx O(n \log n)$ , which is better than brute force one:  $O(nm)$  time complexity

<- Julia, you missed another important step: Ask if the arrays are sorted or not?

<- Julia, if two arrays are sorted, what is the optimal time complexity?

Linear time <-  $O(n+m)$  <- you missed the opportunity to ask yourself the question.

<- Julia, algorithm time complexity - How to say that? Level 1 (sorted array time complexity), level 2, asking level 2 in detail (assuming that two arrays are sorted, now what? linear vs  $n \log m$  vs  $n^2$ ).

---

## Algorithm contests? Competitive programming? (2016-02-24 23:32)

Feb. 24, 2016

Algorithm contests are great tools to use. It takes more critical thinking, and more attention to detail in order to gain points.

Julia did two easy questions on two hackerRank contests last weekend, she could not perform on easy question in 20 minutes each. She actually spent over 60 minutes, hard to concentrate, felt stressed. So, she did some research and look into articles talking about programming contests and how she can improve in short time:

3 popular contests sites:

1. **HackerRank**
2. **TopCoder**
3. **CodeJam**

Her favorite article detailed on this -

**Preparing for a technical interview with programming contests**

:

[1] <https://www.facebook.com/notes/10151298476823920>

- You'll learn how to **critically analyze** your work. Because you don't get credit for solving a problem until the code you write can generate the correct results for a large input set (and you don't know what that input set looks like), **you're forced to think about things** such as time complexity, memory usage, and nasty corner cases. Importantly, most of this work happens outside the context of a debugger. Debuggers are invaluable tools for figuring out why a given piece of code is buggy, but it's better if you can write bug-free code in the first place. In an interview situation, **candidates who can't statically analyze their code** generally have trouble showing their solution is correct (or figuring out why it's not).

[2]<https://medium.com/@dpup/whiteboarding-4df873dbba2e#.ps92c99lb>

[3]<https://code.google.com/codejam/contests.html>

[4]<https://code.google.com/codejam/contest/6224486/dashboard#s=a&a=0>

[5]<http://blog.hackerrank.com/3-ways-crush-technical-interview/>

[6]<https://www.quora.com/How-has-competitive-programming-helped-you-get-a-job>

[7]<http://www.redgreencode.com/12-reasons-to-study-competitive-programming/>

[8]<https://www.quora.com/What-is-the-best-strategy-to-improve-my-skills-in-competitive-programming-in-2-3-months>

[9]<https://www.quora.com/How-did-Anudeep-Nekkanti-become-so-good-at-competitive-programming>

Things to do in next month:

**Work on contests every week, 2 - 4 hours. Try to aim for 20 hours experience first.**

1. <https://www.facebook.com/notes/10151298476823920>

2. <https://medium.com/@dpup/whiteboarding-4df873dbba2e#.ps92c99lb>

3. <https://code.google.com/codejam/contests.html>

4. <https://code.google.com/codejam/contest/6224486/dashboard#s=a&a=0>

5. <http://blog.hackerrank.com/3-ways-crush-technical-interview/>

6. <https://www.quora.com/How-has-competitive-programming-helped-you-get-a-job>

7. <http://www.redgreencode.com/12-reasons-to-study-competitive-programming/>

8. <https://www.quora.com/What-is-the-best-strategy-to-improve-my-skills-in-competitive-programming-in-2-3-months>

9. <https://www.quora.com/How-did-Anudeep-Nekkanti-become-so-good-at-competitive-programming>

---

## React and Flux for Angular Developers (2016-02-26 00:01)

February 25, 2016

Spent 1 hour to watch video on pluralsight.com.

"React and Flux for Angular Developers" by

Read something related to catch up ideas about coding using React.js.



## HackerEarth: first algorithm practice - Milly Chocolate (2016-02-27 01:26)

Feb. 26, 2016

Spent more than 1 hour to work on an algorithm problem on HackerEarth.

Problem statement:

[1]<https://www.hackerearth.com/druva-sdet-hiring-challenge/algorithm/milly-and-chocolates-4/>

Milly loves to eat chocolates. She buys only those food items which contain some amount or percentage of chocolate in it. She has purchased  $N$  such food items and now she is planning to make a new food item by her own. She will take equal proportions of all of these  $N$  food items and mix them. Now she is confused about the percentage of chocolate that this new food item will have. Since she is busy in eating the chocolates so you have to help her in this task.

Input

First line of the input will contain  $T$  (no. of test cases). Every test case will contain two lines. First line will contain  $N$  (no. of food items) and the second line will contain  $N$  space separated  $P_i$  values denoting the percentage of chocolate in  $i^{\text{th}}$  food item.

Output

For every test case, print the percentage of chocolate that will be present in the new food item.

Note : Your answer should be exactly up to 8 decimal places which means that if your answer is 2.357 then you have to print 2.35700000 or if your answer is 2.66666666 .... then you have to print 2.66666667

Constraints

$1 \leq T \leq 5$

$1 \leq N \leq 5 \cdot 10^5$

$0 \leq P_i \leq 100$

SAMPLE INPUT

[2] [3]

1

3

80 30 90

SAMPLE OUTPUT

[4] [5]

66.66666667

Time Limit: 1 sec(s) for each input file.

Memory Limit: 256 MB

Source Limit: 1024 KB

Marking Scheme: Marks are awarded if any testcase passes.

Allowed Languages: C, CPP, CLOJURE, CSHARP, GO, HASKELL, JAVA, JAVASCRIPT, JAVASCRIPT\_NODE, LISP, OBJECTIVEC, PASCAL, PERL, PHP, PYTHON, RUBY, R, RUST, SCALA

Julia's practice:

**1 second for each file, but, the time is over 1 second .** Need to work on speed! (Julia likes the challenge! )

[6]<https://github.com/jianminchen/hackerEarth/blob/master/MillyandChocolates/MillyAndChocolate.cs>

Julia read editorial of algorithm, and know the better solution:

[7]<https://www.hackerearth.com/problem/algorithm/milly-and-chocolates-4/editorial/>

( will try her own version as well later. Definitely, the code will take less time to write! Good learning tool - Thanks, hackerEarth !)

So, to fix TLE error ( more than 1 second for each file), Julia wrote a bug free version:

[8][https://github.com/jianminchen/hackerEarth/blob/master/MillyandChocolates/MillyAndChocolate\\_BugFix.cs](https://github.com/jianminchen/hackerEarth/blob/master/MillyandChocolates/MillyAndChocolate_BugFix.cs)

Read this web page:

[9]<https://www.hackerearth.com/@pranjuldb/activity/hackerearth/>

Julia's hackerEarth profile:

[10]<https://www.hackerearth.com/@jianminchen.fl>

1. <https://www.hackerearth.com/druva-sdet-hiring-challenge/algorithm/milly-and-chocolates-4/>
2. <https://he-s3.s3.amazonaws.com/media/hackathon/druva-sdet-hiring-challenge/problems/6ac399f4-b-sample-input-6ac395b.txt?Signature=Xstt6sAIHsJYXZ0cb%2FX4GyjswEY%3D&Expires=1456565396&AWSAccessKeyId=AKIAJLE6MUHDYS3HN6YQ>
3. <https://www.blogger.com/null>
4. <https://he-s3.s3.amazonaws.com/media/hackathon/druva-sdet-hiring-challenge/problems/6acb8ee8-b-sample-output-6acb8ab.txt?Signature=7YuX5qEh0tbxCAm4QNC4fbpN03o%3D&Expires=1456565396&AWSAccessKeyId=AKIAJLE6MUHDYS3HN6YQ>
5. <https://www.blogger.com/null>
6. <https://github.com/jianminchen/hackerEarth/blob/master/MillyandChocolates/MillyAndChocolate.cs>
7. <https://www.hackerearth.com/problem/algorithm/milly-and-chocolates-4/editorial/>
8. [https://github.com/jianminchen/hackerEarth/blob/master/MillyandChocolates/MillyAndChocolate\\_BugFix.cs](https://github.com/jianminchen/hackerEarth/blob/master/MillyandChocolates/MillyAndChocolate_BugFix.cs)
9. <https://www.hackerearth.com/@pranjuldb/activity/hackerearth/>
10. <https://www.hackerearth.com/@jianminchen.fl>

---

## HackerRank: Pangram (2016-02-28 01:49)

Feb. 28, 2016

Work on the pangram problem on hackerRank. Julia spent 20 minutes to work on the solution.

Problem statement: (Easy question)

[1]<https://www.hackerrank.com/challenges/pangrams>

Also, read the topic:

<https://www.hackerrank.com/challenges/pangrams/topics>

Here is Julia's practice.

[2]<https://github.com/jianminchen/HackRank/blob/master/pangram/Pangram.cs>

1. <https://www.hackerrank.com/challenges/pangrams>

2. <https://github.com/jianminchen/HackRank/blob/master/pangram/Pangram.cs>

---

## code challenge: count of substring (2016-02-28 18:44)

February 28, 2016

Problem statement:

[1]<https://www.hackerrank.com/contests/juniper-codesprint/challenges/leonardo-and-substring>

Problems solved in the progression of coding:

1. **Runtime error** - exceed time limit

naive solution - compare each substring if it contains 00 or 11

2. **Console.ReadLine** only reads up to 256 chars, the input is up to 100000 chars.

3. **Recursive calls - stack overflow** - string length is up to 100000

4. **Using iterative solution** to replace recursive solution

First, wrote a solution in 20 minutes, but **Time exceeding limit - TLE error**.

Solution 1:

[2][https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo %20and %20the %20Substring/CountSubstringNotIn00or11\\_TLE.cs](https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo%20and%20the%20Substring/CountSubstringNotIn00or11_TLE.cs)

Solution 2:

[3][https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo %20and %20the %20Substring/ConstructSubstgringNotIn00or 11\\_upTo256.cs](https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo%20and%20the%20Substring/ConstructSubstringNotIn00or11_upTo256.cs)

So, write second version using recursive to avoid redundant calculation: **stack overflow problem**

Solution 3:

[4][https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo %20and %20the %20Substring/ConstructSubstringNotIn00or11\\_stackOverFlow.cs](https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo%20and%20the%20Substring/ConstructSubstringNotIn00or11_stackOverflow.cs)

Then, wrote third version with iterative solution:

Solution 4: ( **HackerRank embedded C # executable - wrong answer, but Visual express is ok! Cannot figure**

out! )

[5][https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo%20and%20the%20Substring/ConstructSubstringNotIN00or11\\_B.cs](https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo%20and%20the%20Substring/ConstructSubstringNotIN00or11_B.cs)

Spent more than 4 hours on this easy question. Totally invest 3 hours nonstop on Sunday afternoon on this problem solving.

What we say to encourage this behavior - **have guts to fail** . This is just the practice.

1. <https://www.hackerrank.com/contests/juniper-codesprint/challenges/leonardo-and-substring>
  2. [https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo%20and%20the%20Substring/CountSubstringNotIN00or11\\_TLE.cs](https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo%20and%20the%20Substring/CountSubstringNotIN00or11_TLE.cs)
  3. [https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo%20and%20the%20Substring/ConstructSubstgringNotIN00or11\\_upTo256.cs](https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo%20and%20the%20Substring/ConstructSubstgringNotIN00or11_upTo256.cs)
  4. [https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo%20and%20the%20Substring/ConstructSubstringNotIN00or11\\_stackOverflow.cs](https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo%20and%20the%20Substring/ConstructSubstringNotIN00or11_stackOverflow.cs)
  5. [https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo%20and%20the%20Substring/ConstructSubstringNotIN00or11\\_B.cs](https://github.com/jianminchen/HackRank/blob/master/JuniperCodeChallengeFeb2016/Leonardo%20and%20the%20Substring/ConstructSubstringNotIN00or11_B.cs)
- 

## Blogs Reading: Algorithm problems (2016-02-28 23:42)

Feb. 28, 2016

Julia likes to use interview questions to broaden knowledge, make herself a good thinker.

- [1]<http://dandreamsofcoding.com/2014/03/18/dissecting-an-interview-question/>
- [2]<http://dandreamsofcoding.com/2015/01/09/dissecting-an-interview-question-math-is-hard/>
- [3]<http://dandreamsofcoding.com/2014/08/01/dissecting-an-interview-question-reconstructing-a-tree/>

*Action items :*

Julia, you should write down some notes, and put your 2 cents in. Entertain the ideas from the author.

**Randomly selected question: (spend 30 minutes to think, Feb. 26, 2016)**

[4]<http://www.spoj.com/problems/BFBASE/>

February 26, 2016

[5]<https://www.hackerearth.com/druva-sdet-hiring-challenge/problems/>

[6]<https://www.hackerearth.com/algorithms-qualifiers-round-1/problems/>

1. <http://dandreamsofcoding.com/2014/03/18/dissecting-an-interview-question/>
2. <http://dandreamsofcoding.com/2015/01/09/dissecting-an-interview-question-math-is-hard/>

3. <http://dandreamsofcoding.com/2014/08/01/dissecting-an-interview-question-reconstructing-a-tree/>
  4. <http://www.spoj.com/problems/BFBASE/>
  5. <https://www.hackerearth.com/druva-sdet-hiring-challenge/problems/>
  6. <https://www.hackerearth.com/algorithms-qualifiers-round-1/problems/>
- 

## Pluralsight: AngularUI Fundamentals (2016-02-29 18:46)

Feb. 10, 2016

### AngularUI Fundamentals

3 hours video lectures

[1]<http://stevemichelotti.com/new-pluralsight-course-yeoman-fundamentals/>

reading:

[2]<https://github.com/angular-ui/ui-router/wiki>

1. <http://stevemichelotti.com/new-pluralsight-course-yeoman-fundamentals/>
  2. <https://github.com/angular-ui/ui-router/wiki>
- 

## Mental toughness training - small talk how to perform algorithm as well (2016-02-29 18:50)

Feb. 29, 2016

Julia went through mental toughness training in her tennis practice by herself last 5 years. She has played over hundreds of single/ double tennis matches ever since. Every time, she failed over and over again on mental toughness checking - her mind just does its own trip - being distracted, thinks about the win/ loss, things happened at work/ church/ friends/ family, she reminded herself, recovered and focused on the current, back to sports, current game, current point - it is tough, that is the reason called mental toughness training. **Do not even think about this game, this set, or last point, only this current moment .**

In more technical term, mental toughness is - Only think about what she can control, stand on the ground, current moment, thing is happening right now.

Good thing is that she develops the great attitude to live at the moment, and kick some depression ass/ negative thoughts' ass, and then be a super happy and smart person again.

How about algorithm problem solving skills? coding skills? One of her problems is to deal with algorithms problem solving in one hour. How does she stay in the current moment, spend one hour?

She still feels frustrated after a week, but she takes time to think about it.

Here are her action items:

1. Try to solve easy problem always.
2. Have some training - fast coding, on hackerRank.

---

## 2.3 March

### Angular - put a web app together in short future (2016-03-01 23:26)

March 1, 2016

So busy to study courses on pluralsight.com and code school, try to build a web app in short future. Here is the short video to get motivated to use Angular.

[1]<https://channel9.msdn.com/Events/Visual-Studio/Connect-event-2015/051>

[2]<https://channel9.msdn.com/Events/ASPNET-Events/ASPNET-Fall-Sessions/Brad-Green-from-Google?ocid=player>

1. <https://channel9.msdn.com/Events/Visual-Studio/Connect-event-2015/051>

2. <https://channel9.msdn.com/Events/ASPNET-Events/ASPNET-Fall-Sessions/Brad-Green-from-Google?ocid=player>

---

### Mock interview experience (2016-03-02 00:32)

March 1, 2016

Julia likes to document her first mock interview using C # on March 28, 2016. She got this question to be interviewed: **BST Successor Search** .

Given a node

n

in a binary search tree, explain and code the most efficient way to find the successor of

n

.

Analyze the run time complexity of your solution.

How did Julia work out the problem with the help from the interviewer?

Julia was nervous, so she did not think too much. Ask to discuss the successor. What is the successor?

Then, she was told that a tree

4

3 5

4's successor is 5, and 3's successor is 4.

Julia should work on some test cases, and make complete understanding of algorithm first.

But, she did not.

She was told to write some code about it. Julia then asked to start from root node, and then, was told that Node class:

```
Class Node {
```

```
Node left;
```

```
Node right;
```

```
Node parent;
```

```
int value;
```

```
}
```

So, the function has to take argument node n - any node in the tree, instead of root node.

code with bugs - work on simple BST:

4

3 5

4's successor is 5, and 3's successor is 4.

[1][https://github.com/jianminchen/AlgorithmsPractice/blob/master/BSTSuccessor\\_Step1.cs](https://github.com/jianminchen/AlgorithmsPractice/blob/master/BSTSuccessor_Step1.cs)

And then, here are the conversations:

Interviewer: There will be more than 1 cases failing;

show a tree

4

2

3

3's successor is 4, not null. 2 is left child of 4, and then 3 is right child of 2. Go up to parents.

Julia: So, if we work on the following tree, will all bugs be fixed?

4

2 6

1 3 5 7

Interviewer: 3's successor is 4, and 4's successor is 5, not 6.

Julia: Ok, let me add two more functions to help.

For node 3, the successor of 3 is 4. // to fix bug, add function

**checkRelationship**

while retrieving grand parents.

For node 4, the successor of 4 is 6. // to fix bug, add function called

**getLeftMostNode**

()

[2][https://github.com/jianminchen/AlgorithmsPractice/blob/master/BST\\_Successor\\_Step2.cs](https://github.com/jianminchen/AlgorithmsPractice/blob/master/BST_Successor_Step2.cs)

The interviewer asked Julia cleaned up the code, removed unnecessary variables, put return statement in while loop.

At the end, the interviewer told that Julia was the best one; he interviewed 7 other people, maybe using the same question. Julia was so excited, motivated after hearing the feedback. Julia, she did write very clear, readable code, with logic perfect through mock interview.

And then, Julia was asked the run time analysis. The balanced search tree, the run time is  $O(\log N)$ , but if it is not balanced, can be up to  $N$ .

But, lessons learned:

1. Always work on test cases, discuss algorithm, make sure both agrees, then, start to write code;
2. Work on simple test case, and then, extend the algorithm to fit the complicate case.
3. Start from simple case, write code to make it work on simple case.

And then, discuss bugs, and fix the bugs with more functions.

It took 28 minutes to finish most of coding.

4. Julia was too nervous, she could not think about successor clearly at the beginning. So, she asked to have some discussion about successor. Interviewer showed her one simple example.

Surprisingly, write in two steps, make the coding so easy.

1. [https://github.com/jianminchen/AlgorithmsPractice/blob/master/BSTSuccessor\\_Step1.cs](https://github.com/jianminchen/AlgorithmsPractice/blob/master/BSTSuccessor_Step1.cs)

2. [https://github.com/jianminchen/AlgorithmsPractice/blob/master/BST\\_Successor\\_Step2.cs](https://github.com/jianminchen/AlgorithmsPractice/blob/master/BST_Successor_Step2.cs)

---

**HackerRank: HourRank 6 (2016-03-03 00:36)**

March 2, 2016



Julia spent more than 1 hour to work on first question on HackerRank: HourRank 6, Lisa's workbook.

It's a one-hour rated contest with 3-4 algorithmic challenges. Are you fast and accurate enough to win it?

So, by any means, Julia finished the first algorithm, and passed all test cases. 20 minutes to read the question, and then, 20 minutes to write, and over 20 minutes to debug, fix bugs to pass basic test case.

[1]<https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-workbook>

Here is the code she wrote:

[2][https://github.com/jianminchen/HackRank/blob/master/Lisa's %20workbook/LisaWorkbook.cs](https://github.com/jianminchen/HackRank/blob/master/Lisa's%20workbook/LisaWorkbook.cs)

So, she read other people's code. Best performers in top 20 use less than 5 minutes.

Julia read a few of great code to use loops. So, she now has more ideas to implement the algorithm.

Her favorite ones:

No. 1 ( #6):

[3]<https://gist.github.com/jianminchen/6e4265a1d73b41f544a7>

No. 2 ( #12):

[4]<https://gist.github.com/jianminchen/7cc380fed68fbc57c5>

No. 3 **Julia's favorite solution** -

[5]<https://gist.github.com/jianminchen/599248d93d71a0155a0b>

Action item:

1. Work on basics - for loops; Julia, learn from others - basic C programming, it is quick and fast.

2. C++, less time to write compared to C #.

3. A lot of room to improve,

60 minutes -> 20 minutes, <- **reading time from 20 -> 5 minutes** .

20 minutes to 10 minutes, <- **clean code, no bug** .

20 minutes to 5 minutes.

4. It is like the sports. Julia, you have to practice.

Some workout:

No. 1 ( #6):

[6]<https://gist.github.com/jianminchen/6e4265a1d73b41f544a7>

Add some comments to the solution:

```
int
main () {page = 1 ; scanf (
"
%d %d
"
, &n, &k); for (i= 1 ; i<=n ; i++) { // Go through each chapter scanf (
"
%d
"
, &x); for (j= 1 ; j<=x ; j++) { // go through each problem if (page == j) ans++; // check if it is special if (j % k == 0 )
page++; // check if problem starts on a new page } if (x % k != 0 ) page++; // edge case: do not forget. start a new
page for next chapter //printf(" %d\n",page); } printf (
"
%d
\n
"
,ans); return
0 ; }
No. 2 ( #12)
```

[7]<https://gist.github.com/jianminchen/7cc380fed68fcbcd57c5>

```
int
main () {
cin >> n >> k; for (i = 0 ; i < n; i++) cin >> a[i];ll ans = 0 ; for (i = 0 ; i < n; i++) // go through each chapter { for (j = 1 ; j <=
a[i]; j+=k) // go through each problem in the chapter, increment by k { // in other words, go through first problem
in each page m++; if (m >= j & & m <= min (j+k- 1 ,a[i])) // Problem number range includes page no m ans++; } }cout
<< ans << endl; return
0 ; }No. 3
[8]https://gist.github.com/jianminchen/599248d93d71a0155a0b
```

using
namespace
226

```

std
;
typedef
long
long ll; typedef
long
double ld; ll n,m,k,q,d;ll T;std::vector<ll> A; int
main () { ll c1,c2,c3,c4,c5,c6; ll x,y,z; ll a,b,c,e; ll s; std::cin >> n >> k; ll pn = 1 ; ll ans = 0 ; for (c1= 0 ;c1<n;c1++) { // go through each chapter std::cin >> a; ll t = 0 ; while (t < a) { // go through each problem in the chapter if (pn > t & & pn <= min (t+k,a)) // page number is in problem index range {ans++; } pn++; t += k; // increment problem index k once } std::cout << ans <<
"
\n
"
; return
0 ; }

```

1. <https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-workbook>
2. <https://github.com/jianminchen/HackRank/blob/master/Lisa's%20workbook/LisaWorkBook.cs>
3. <https://gist.github.com/jianminchen/6e4265a1d73b41f544a7>
4. <https://gist.github.com/jianminchen/7cc380fed68fbc57c5>
5. <https://gist.github.com/jianminchen/599248d93d71a0155a0b>
6. <https://gist.github.com/jianminchen/6e4265a1d73b41f544a7>
7. <https://gist.github.com/jianminchen/7cc380fed68fbc57c5>
8. <https://gist.github.com/jianminchen/599248d93d71a0155a0b>

---

## HackerRank: Bear and Steady Gene (I) (2016-03-04 01:05)

March 4, 2016

Problem statement:

[1]<https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>

Editorial:

Let's first think when Limak can choose some particular interval (substring). We should care about the remaining letters, both in the prefix and the suffix. If there are more than

n

/

4

remaining letters of one type then we can't get a steady string. Otherwise, we know exactly how many letters of each type are missing and we can fill the removed interval with these exact letters. So, the interval can be chosen only if the remaining part doesn't contain more than

$n$

/

4

letters of some type.

For each possible starting index of the interval let's find the nearest (leftmost) possible ending index. You can create four segment trees, one for each letter. Then, for fixed interval you can check in

$O$

(

1

)

whether there are at most

$n$

/

4

remaining letters of each type. So, for each starting index you can binary search the earliest good ending index. Segment trees and binary search give us

$O$

(

$n$

$\log$

2

$n$

)

. Let's make it faster.

First thing is to get rid of segment trees. It's enough to store prefix (and maybe suffix) sum for each letter, so you don't need segment trees anymore. It's ok to get AC but you can change one more thing. As we move with the starting index to the right, the ending index also moves to the right (or it doesn't change). So, we can use the two pointers technique to get

$O$

(

$n$

)

solution. You can check codes below for details.

### Julia's practice :

[2]<https://gist.github.com/jianminchen/80723bae951328a690bb>

score 15 out of 50, there are 2 run time error, failed a few of test cases. The implementation is no better than brute force solution.

### C # code implementation to study:

[3]<https://gist.github.com/jianminchen/153eab0defae014842e8>

Readable code, with some analysis.

[4]<https://gist.github.com/jianminchen/fae9142eff9a6f9643fc>

### Java code implementation to study:

[5]<https://gist.github.com/jianminchen/d52ced38de0bffa2d9e8>

### C++ code to study:

[6]<https://gist.github.com/jianminchen/d7370b7e73013ee708be>

write this one as well <- **Great code, very readable!**

[7]<https://gist.github.com/jianminchen/c6b51207f9cc9b083573>

[8]<https://gist.github.com/jianminchen/80638c0db328d0098f3b>

Binary search algorithm: (Java)

[9]<https://gist.github.com/jianminchen/d01faa03ca9b06696db3>

### Comment:

**The brute force solution will not pass the test cases.**

It takes a lot of time to read 108 people's code - all scores 50 / 50. It is fun to read all 108 solution scoring 50/50.

Julia, take time to play with this algorithm.

1. <https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>
2. <https://gist.github.com/jianminchen/80723bae951328a690bb>
3. <https://gist.github.com/jianminchen/153eab0defae014842e8>
4. <https://gist.github.com/jianminchen/fae9142eff9a6f9643fc>
5. <https://gist.github.com/jianminchen/d52ced38de0bffa2d9e8>
6. <https://gist.github.com/jianminchen/d7370b7e73013ee708be>
7. <https://gist.github.com/jianminchen/c6b51207f9cc9b083573>
8. <https://gist.github.com/jianminchen/80638c0db328d0098f3b>
9. <https://gist.github.com/jianminchen/d01faa03ca9b06696db3>

## HackerRank: Bear Steady Gene (II) (2016-03-05 17:22)

March 4, 2016

Problem statement:

[1]<https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>

A gene is represented as a string of length

$n$

(where

$n$

is divisible by

4

), composed of the letters

A

,

C

,

T

, and

G

. It is considered to be steady if each of the four letters occurs exactly

$n/4$

times. For example,

GACT

and

AAGTCCT

are both steady genes.

**Julia's 1st practice :**

[2]<https://gist.github.com/jianminchen/80723bae951328a690bb>

score 15 out of 50, there are 2 run time error, failed a few of test cases.

The algorithm ends up in time complexity  $O(n^2)$ , close to brute force solution -  $O(n^2)$

**C # code** implementation to study:

Readable code, with some analysis.

[3]<https://gist.github.com/jianminchen/fae9142eff9a6f9643fc>

Work on two pointers, sliding window, so time complexity is  $O(2n) = O(n)$ .

Let us go over two pointers algorithm here using example:

GAAATAAA,

A - count of A, denoted as  $cA = 6$ ,  $n/4 = 2$ , so we have to change 4 of A to other thing.

A -  $6 - 2 = 4$ , 4 of change

C -  $0 - 2 = -2$

T -  $1 - 2 = -1$

G -  $1 - 2 = -1$

So, at least minimum is 4, but a substring containing 4 of A, shortest one is AAAA. But "AAAA" is not a substring of "GAAATAAA"

G A A A T A A A

0

start

end

Let us find the substring starting from 0, but will include all 4 of A, and then, rest of string will not include any char of "ACGT" more than  $n/4$ .

GAAATA, string length is 6.

start - 0, index of 0

end - A, index of 5

continue to move start to next one, 1, then, G is moved out from substring, adjust count of chars.

AAATA can be the substring, so the length is  $\min(6, 5) = 5$ . Do not need to move end pointer.

next step, start = 2, missing one of A, and then, end has to move to next one until the string fits into requirement.

Every thing should be covered in 20 minutes, problem reading, the design of algorithm, the coding.

So, **Julia had second practice**, and the code scores 50 out of 50 this time.

[4]<https://gist.github.com/jianminchen/61dfe437f82edb9793fc>

### Conclusion :

After coding, Julia understands the algorithm much better.

1. count of array for substring,  $cB=[4, -2, -1, -1]$ , so in other words, substring has to contain at least 4 'A', C, G, T does not matter.

So, using CB array, sliding windows of substring GAAATA, each time, end index is moving forward, tracking the

substring's char by taking off from CB; in other words, GAAATA, CB will be [0, -2, -2, -2].

Now, it is easy to understand that GAAATA will be added to the solution set, the length is 6. line 51, if

```
(cBAAllLessThan1(cB))
```

<-add substring to solution set, compare length<- easy to understand now.

2. the code in [5]<https://gist.github.com/jianminchen/fae9142eff9a6f9643fc> has discussion:

```
switch (line[ 0 ]) { case
```

```
' A '
```

```
: A-; break ; case
```

```
' C '
```

```
: C-; break ; case
```

```
' G '
```

```
: G-; break ; case
```

```
' T '
```

```
: T-; break ; }
```

Julia removed those detail discussion, but her code takes more time. Because she created a bug in her writing. Debugging takes time.

3. Julia's practice in 2015, sliding windows, two pointers:

[6][http://juliachencoding.blogspot.ca/search/label/slide %20window](http://juliachencoding.blogspot.ca/search/label/slide%20window)

April 27, 2016

change C # program to make variables more meaningful, matching the design:

code -> GENES <- global string array

function name matches design of two pointers moving.

[7]<https://gist.github.com/jianminchen/b8263048c297473319c23836e9468c14>

searchStrArray -> searchStrNumber, more meaningful.

[8]<https://gist.github.com/jianminchen/b23c4f606a101b9aeec71eff3268db32>

1. <https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>

2. <https://gist.github.com/jianminchen/80723bae951328a690bb>

3. <https://gist.github.com/jianminchen/fae9142eff9a6f9643fc>

4. <https://gist.github.com/jianminchen/61dfe437f82edb9793fc>

5. <https://gist.github.com/jianminchen/fae9142eff9a6f9643fc>

6. <http://juliachencoding.blogspot.ca/search/label/slide%20window>

7. <https://gist.github.com/jianminchen/b8263048c297473319c23836e9468c14>

8. <https://gist.github.com/jianminchen/b23c4f606a101b9aeec71eff3268db32>



## HackerRank: Bear and Steady Gene algorithm (III) (2016-03-05 22:28)

March 5, 2016

Problem statement:

[1]<https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>

A gene is represented as a string of length

$n$

(where

$n$

is divisible by

4

), composed of the letters

A

,

C

,

T

, and

G

. It is considered to be steady if each of the four letters occurs exactly

$n/4$

times. For example,

GACT

and

AAGTGCCT

are both steady genes.

**Julia's 1st practice :**

[2]<https://gist.github.com/jianminchen/80723bae951328a690bb>

score 15 out of 50, there are 2 run time error, failed a few of test cases.

The algorithm ends up in time complexity  $O(n^2)$ , close to brute force solution -  $O(n^2)$

**C # code** implementation to study:

[3]<https://gist.github.com/jianminchen/6c0ee8af6e88964410ab>

Work on two pointers, sliding window, so time complexity is  $O(2n) = O(n)$ .

Let us go over two pointers algorithm here using example:

GAAATAAA,

gene string "ACTG"

var a = [3, 0, 0, 0, 2, 0, 0, 0]

So, Target[] = [4, 0, 0, 0] - 4 of A is need, but C, T, G do not matter

If all element in Target == 0, then return 0

Left, right two pointer (l, r), starting from 0

declare a sum array

Start a loop, loop on r

Add a[r]'s char into Sum array,

r++, r moves to next one

inside the first loop, another loop for r, continue to move forward

if sums not reaches target. - 4 of them  $\geq$  target

**Julia's comment:** For the test case, GAAATAAA, find substring: GAAATA

inside the first loop, another loop for left pointer - l

condition:  $l < r$

if sums array is bigger than target array, then remove left pointer char count,

then, move left pointer to next one

then, the substring is the reaching the target,

**Julia's comment: GAAATA -> AAATA**

and compare to existing smallest length

AAATA is the smallest one so far.

**Conclusion :**

This algorithm is much easy to follow than the one in

[4]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-bear-steady-gene-ii.html>

**Reference:**

**C # code** implementation to study:

[5]<https://gist.github.com/jianminchen/6c0ee8af6e88964410ab>

1. <https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>
  2. <https://gist.github.com/jianminchen/80723bae951328a690bb>
  3. <https://gist.github.com/jianminchen/6c0ee8af6e88964410ab>
  4. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-bear-steady-gene-ii.html>
  5. <https://gist.github.com/jianminchen/6c0ee8af6e88964410ab>
- 

**HackerRank: Bear and Steady Gene algorithm (IV) (2016-03-05 23:22)**

March 5, 2016

Problem statement:

[1]<https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>

A gene is represented as a string of length

n

(where

n

is divisible by

4

), composed of the letters

A

,

C

,

T

, and

G

. It is considered to be *steady* if each of the four letters occurs exactly

n

4

times. For example,

GACT

and

AAGTGCCT

are both steady genes.

Julia did 3 practice code using C #, but she likes one of solutions using Java, and then, she spent 5 minutes to convert it to C #.

study code:

**Java code** implementation to study:

[2]<https://gist.github.com/jianminchen/d52ced38de0bffa2d9e8>

Julia's fourth practice:

use two pointers, sliding window, linear time solution

[3]<https://gist.github.com/jianminchen/0892107fc0f6b6bec7a0>

Julia's comment:

Julia learned a few things through the code:

1. Declare cnt array using size of 256, ascii code is 256. So, ACGT should be ok.
2. Declare a string "ACGT", and then the code uses 'A', 'C', 'G', 'T' only once.
3. Only two loops, first loop, variable l - left pointer from 0 to n-1.  
second while loop, move r pointer if cnt array is bigger than mx - minimum

while

(cnt[

' A '

236

```

] > mx || cnt[
' C '
] > mx || cnt[
' T '
] > mx || cnt[
' G '
] > mx) {

```

How to paraphrase this conditional checking?

Let us go through the test case "GAAATAAA",

```
cnt['A'] = 6, cnt['C'] = 0, cnt['G'] = 1, cnt['T'] = 1
```

```
mx = 2,
```

starting from G, and then, while loop <- get in,

```
G-> GA->...->GAAATA, cnt['A'] = 2, cnt['G'] = 0,
```

now, it is time to exit while loop. <- making sense. Because cnt array contains the count

for rest of substring "GAAATA", requirement is "none of 'ACGT' is more than 2".

April 14, 2016

I put the for loop into a standalone function, and then, add some comment for the function, explain the design. Basically, it is using sliding window.

Here is the link of code, which passes HackerRank as well.

[4]<https://gist.github.com/jianminchen/9b02beab326b2bfcd4b524f219d2946f>

**statistics:**

This post is one of most viewed so far, over 130 views, from March 5 to April 13, 2016.

1. <https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>
2. <https://gist.github.com/jianminchen/d52ced38de0bffa2d9e8>
3. <https://gist.github.com/jianminchen/0892107fc0f6b6bec7a0>
4. <https://gist.github.com/jianminchen/9b02beab326b2bfcd4b524f219d2946f>

Sidious (2016-04-13 19:43:02)

This comment has been removed by the author.

Sidious (2016-04-13 19:43:44)

Hi Julia. I'm having a hard time understanding the logic behind your code. Could you please what exactly happens inside the for loop? What each variable and array signifies? How the logic works exactly? That'd be pretty helpful. Thanks!

Jianmin Chen (2016-04-13 22:34:07)

Sidious, thank you for the comment. I put the for loop into a standalone function, and then, add some comment for the function, explain the design. Basically, it is using sliding window.

Here is the link of code, which passes HackerRank as well.

<https://gist.github.com/jianminchen/9b02beab326b2bfcd4b524f219d2946f>

## HackerRank: Bear And Steady Gene - binary search algorithm (V) (2016-03-06 00:45)

March 5, 2016

Problem statement:

[1] <https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>

A gene is represented as a string of length

$n$

(where

$n$

is divisible by

4

), composed of the letters

A

,

C

,

T

, and

G

. It is considered to be steady if each of the four letters occurs exactly

$n$

4

times. For example,

238

GACT

and

AAGTGCCT

are both steady genes.

Study code using binary search:

[2]<https://gist.github.com/jianminchen/d01faa03ca9b06696db3>

[3]<https://gist.github.com/jianminchen/395eb9e76fe19cc9338f>

Comment:

Binary search is better than linear search using two pointers.

Brute force  $O(n^2)$  -> linear search  $O(n)$  -> Binary search  $O(\log n)$

Let us walk through this binary search algorithm using test case GAAATAAA.

int low = 0;

int high = n; // n = 8

int need = 8/4 = 2

int[] cnt = {6, 0, 1, 1} // "ACGT"

mid = 4,

so, copy cnt to tmp[],

and then, first half, take it away, see left half meet the standard:

tmp[i] <= 2, i = 0, ..., 3

because first half contains 3 'A', and then, tmp[0] = 3,

deal with second half, go through a loop:

get confused, let us debug through the code.

**debug through the code, still confuse!**

**Java code:**

[4]<https://gist.github.com/jianminchen/d01faa03ca9b06696db3>

**C # code**

[5]<https://gist.github.com/jianminchen/76dffc51880a80279f25>

**April 28, 2016**

Come back to the problem on binary search solution, figure out the design:

First, go through two test cases: "GTTCCAAA" and "GAAATTCC"

1. "GTTCCAAA",

Binary search is doing this way:

Biggest value of search string length is 8.

First, divide range from 0 to 8 into half, 4

Find first string with length 4,

one is "GTTC", one is "CAAA",

and then, remove count of first half, rest string (two parts, separated) "CAAA", since A is repeated 3 times, not in the range;

instead of stopping, wrongly conclude that 4 is not high value, go through all the possible substring with length 4, by sliding window of search string: "GTTC" forward from left to right, but keep the same window size  
"GTTC" -> "TTCC"->"TCCA"->stop here, since "TCCA" is removed from counting of GENES="ACGT", fit into the requirement.

Next, low=0, high=4, mid = 2,

Because both test cases with one 'A' to replace, Julia figured out through the debugging:

The slide window of fixed length technique,

How to slide?

Which direction to slide?

Kind of clever in design.

**Time complexity analysis** : length search using binary search,  $n$  - length of string,  $O(\log n)$  times; each search for length  $m$ , go over each string once, since using calculated counting array, only do add one/ remove one at a time, one char only does the work once. So, it is  $O(n)$  on this.

**Total time complexity** :  $O(n \log n)$

Conclusion: this binary search is not better than linear search is previous blog (IV).

1. <https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>
  2. <https://gist.github.com/jianminchen/d01faa03ca9b06696db3>
  3. <https://gist.github.com/jianminchen/395eb9e76fe19cc9338f>
  4. <https://gist.github.com/jianminchen/d01faa03ca9b06696db3>
  5. <https://gist.github.com/jianminchen/76dffc51880a80279f25>
- 

**Sunday watching: Jamie Dimon to HBS MBA Class of 2009 (2016-03-06 17:02)**

March 6, 2016

Spend time to relax this Sunday, and enjoy the video from Harvard business school:

[1]<https://www.youtube.com/watch?v=9T9Kp4NE5l4>

Take some notes, and enjoy the teaching.

1. <https://www.youtube.com/watch?v=9T9Kp4NE5l4>
-



## HackerRank: Bear And Steady Gene - Binary Search (II) (2016-03-07 00:09)

March 6, 2016

Problem statement:

[1] <https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>

A gene is represented as a string of length

$n$

(where

$n$

is divisible by

4

), composed of the letters

A

,

C

,

T

, and

G

. It is considered to be steady if each of the four letters occurs exactly

$n$

4

times. For example,

GACT

and

AAGTGCCT

are both steady genes.

Study code using binary search:

cannot figure out the design - confused!

[2] <https://gist.github.com/jianminchen/d01faa03ca9b06696db3>

**This one is easy to follow**

[3] <https://gist.github.com/jianminchen/395eb9e76fe19cc9338f>

Julia's C # practice code:

[4]<https://gist.github.com/jianminchen/af1b3c064c523444463f>

Algorithm talk:

Work on test case GAAATAAA, let me explain the idea using binary search.

```
A1 = Math.Max(0, A1 - n / 4); // test case: GAAATAAA, A1 = 4
```

```
C1 = Math.Max(0, C1 - n / 4); // C1 = 0
```

```
G1 = Math.Max(0, G1 - n / 4); // G1 = 0
```

```
T1 = Math.Max(0, T1 - n / 4); // T1 = 0
```

```
if (A1 == 0 && C1 == 0 && G1 == 0 && T1 == 0)
```

```
{
```

```
    return 0; //
```

```
}
```

```
int ans = n; // default value is maximum value - string length
```

```
for (int i = 0; i < n; i++) // go through a loop, start from 0 to n-1.
```

```
{
```

```
    if (A[n] - A[i] < A1 || C[n] - C[i] < C1 || G[n] - G[i] < G1 || T[n] - T[i] < T1) // substring position from i to n , check the count of A, C, G, T
```

```
        break;
```

```
    int l = i + 1, r = n; // left, right two pointer
```

```
    while (l < r)
```

```
    {
```

```
        int mid = (l + r - 1) / 2;
```

```
        if (A[mid] - A[i] < A1 || C[mid] - C[i] < C1 || G[mid] - G[i] < G1 || T[mid] - T[i] < T1) // not enough
```

```
            l = mid + 1; // set left pointer to mid + 1
```

```
        else
```

```
            r = mid;
```

```
    }
```

```
    ans = Math.Min(ans, l - i); // substring is from i to l ,
```

```
}
```

```
return ans;
```

In other words, GAAATAAA,

i = 0, l = 1, r = 8,

find ans = Math.Min(ans, l-i) <- **ans = 6, l = 6**

i = 1,

find ans = Math.Min(ans, l-i) <- **ans = 5, l = 6, i=1**

i = 2,

find ans = Math.Min(ans, l-i) <- **ans = 5, l = 7, i=2**

```
i=3;  
find ans = Math.Min(ans, l-i) <- ans = 5, l = 8, i=3
```

```
i=4  
break the for loop  
if (A[n] - A[i] < A1 || C[n] - C[i] < C1 || G[n] - G[i] < G1 || T[n] - T[i] < T1)  
  
break;
```

since  $A[n] - A[i] = 3 < A1 = 4$ , TAAA, we need the substring at least 4 of A

**comment:**

This algorithm uses extra space for ACGT count for each i from 0 to n-1, total space is less than 1MB.

```
int [] A = new  
int [ 500500 ]; int  
[] C =  
new  
int  
[  
500500  
];  
int  
[] G =  
new  
int  
[  
500500  
];  
int  
[] T =  
new  
int  
[  
500500  
];
```

4 bytes for int, then  $500K \times 4 \times 4 = 8M$  bytes

1. <https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>
  2. <https://gist.github.com/jianminchen/d01faa03ca9b06696db3>
  3. <https://gist.github.com/jianminchen/395eb9e76fe19cc9338f>
  4. <https://gist.github.com/jianminchen/af1b3c064c523444463f>
- 

## **Pluralsight: JavaScript for C# developer (2016-03-08 00:41)**

March 7, 2016

Two hours on **JavaScript for**

**C # developer**

**on pluralsight (9:30pm - 11:30pm)** . Review JavaScript quickly.

Got 3 of 8 questions correct. <- That is the excellent teaching tool - Learning Check

### **1. How can you specify functions overload in JavaScript ?**

You cannot <- julia chose the one -

But, the correct answer is: You would interrogate the arguments object to determine what parameters were passed in.

Actually, on March 7, Julia wrote a function with overloading function. English, still, is julia's second language.

### **2. How does JavaScript support property setters and getters for objects ?**

Object.defineProperty

### **3. What is the JavaScript alternative to using C # interfaces ?**

Duck Typing

4. Is casting required in JavaScript?

5. What defines a scope in JavaScript?

6. How is the C #'s var keyword different from the JavaScript var keyword.

7. Does JavaScript support classes?

8. Is C #'s for each equivalent to JavaScript's for...in syntax?

Plant to watch another JavaScript course on pluralsight: **structuring JavaScript**.

---

## **Pluralsight: Structuring JavaScript (2016-03-08 20:47)**

March 8, 2016

2 hours on course: Structuring JavaScript on pluralsight, from 8:00pm - 10:00pm.

Julia is a big fan of pluralsight online courses. She spent time to read JavaScript books in 2014 over 6 months; while this time the teaching is more effective with excellent teaching.

She still works on how to take courses on pluralsight. She likes to take notes, and tracks her progression.

Take learning check later.

[1]<https://app.pluralsight.com/score/learning-check/structuring-javascript/next>

[2]<http://weblogs.asp.net/dwahlin/techniques-strategies-and-patterns-for-structuring-javascript-code>

[3]<http://weblogs.asp.net/dwahlin/techniques-strategies-and-patterns-for-structuring-javascript-code-the-prototype-pattern>

[4]<http://weblogs.asp.net/dwahlin/techniques-strategies-and-patterns-for-structuring-javascript-code-revealing-module-pattern>

5 out 10 incorrect questions:

Which of the following code samples demonstrate **creating closure** :

Answer:

```
function closureFunc()
{
  var d = new Date();
  return function() {
    myDiv.data = d.getDay();
  }
}
```

Related clip: Closure Demo 2

JavaScript's " **this** " keyword always represents the current object instance being used.

Answer:

False

Related Clip: Using the Revealing Prototype Pattern with 'this'

Which pattern is **the best one** to use with JavaScript code?

Answer:

There isn't one "best solution"

What features does the **Revealing Module Pattern** offer?

Answer: All of above

Define public and private members

Can include a self-calling function

Encapsulates code

What is a key difference between the **Prototype** and **Revealing Prototype patterns** ?

Answer:

Revealing Prototype allows public and private members to be defined.

Notes to review:

**The Prototype Pattern**

Pros:

Leverage JavaScript's built-in features  
"Modularize" code into re-useable objects  
Variables/ functions taken out of global namespace  
Functions loaded into memory once  
Possible to "override" functions through prototyping

Cons:

"this" can be tricky  
Constructor separate from prototype definition

1. <https://app.pluralsight.com/score/learning-check/structuring-javascript/next>
  2. <http://weblogs.asp.net/dwahlin/techniques-strategies-and-patterns-for-structuring-javascript-code>
  3. <http://weblogs.asp.net/dwahlin/techniques-strategies-and-patterns-for-structuring-javascript-code-the-prototype-pattern>
  4. <http://weblogs.asp.net/dwahlin/techniques-strategies-and-patterns-for-structuring-javascript-code-revealing-module-pattern>
- 

## CodeSchool: AngularJS Tutorial (2016-03-09 23:24)

March 9, 2016

Spent best time in the evening from 9:00pm - 11:00pm to watch the code school video, AngularJS Tutorial tonight.

Need to find some material to study for AngularJS, and then start to memorize some content, such as AngularJS cheat sheet. Spend 20 minutes a day to memorize the cheatsheet.

[1]<http://www.cheatography.com/proloser/cheat-sheets/angularjs/>

[2]<http://www.opitz-consulting.com/fileadmin/redaktion/veroeffentlichungen/pdf/CheatSheet-angularJS.pdf>

81 contributor - great resource for new learner like Julia.

[3]<https://github.com/jmcunningham/AngularJS-Learning/blob/master/README.md>

[4]<http://weblogs.asp.net/dwahlin/learning-angularjs-by-example-the-customer-manager-application>

March 10, 2016

12 minutes of video: AngularJS Tutorial

[5][https://www.youtube.com/watch?v=WuiHuZq\\_cg4](https://www.youtube.com/watch?v=WuiHuZq_cg4)

Plan to watch the video on March 10 evening time:

[6]AngularJS Fundamentals (Pluralsight) - 6 hours 15 minutes total (paid)

[7]<http://app.pluralsight.com/author/deborah-kurata>

Read Angular short book:

[8]<http://www.angularjsbook.com/angular-basics/chapters/introduction/>

March 22, 2016

[9]<http://weblogs.asp.net/dwahlin/video-tutorial-angularjs-fundamentals-in-60-ish-minutes>

1. <http://www.cheatography.com/proloser/cheat-sheets/angularjs/>
  2. <http://www.opitz-consulting.com/fileadmin/redaktion/veroeffentlichungen/pdf/CheatSheet-angularJS.pdf>
  3. <https://github.com/jmcunningham/AngularJS-Learning/blob/master/README.md>
  4. <http://weblogs.asp.net/dwahlin/learning-angularjs-by-example-the-customer-manager-application>
  5. [https://www.youtube.com/watch?v=WuiHuZq\\_cg4](https://www.youtube.com/watch?v=WuiHuZq_cg4)
  6. <http://app.pluralsight.com/courses/angularjs-fundamentals>
  7. <http://app.pluralsight.com/author/deborah-kurata>
  8. <http://www.angularjsbook.com/angular-basics/chapters/introduction/>
  9. <http://weblogs.asp.net/dwahlin/video-tutorial-angularjs-fundamentals-in-60-ish-minutes>
- 

## **Pluralsight: Angular Fundamentals (2016-03-11 00:40)**

March 10, 2016

Work on "Angular Fundamentals" 6 hours course. Julia, be patient, you will have great time to play with AngularJS, and then write a great web app in 2016.

Hours spent on AngularJS: 3 hours.

Take some notes, and then, follow up with some reading, play with some code using AngularJS as well.

---

## **small talk: what you can control, 3 variables from Michael Bloomberg (2016-03-11 21:20)**

March 11, 2016

Julia likes the advice from Michael Bloomberg. You cannot control how lucky you are, you may not control what you will do in next 10 years.

3 things you can control, influence, and success in your life, variables:

1. **how hard you work**,
2. **how honest you are**
3. **how well you deal with others**.

Cannot control how lucky you are. The harder you work, the lucky you get.

Michael Bloomberg's Top 10 Rules For Success

[1]<https://www.youtube.com/watch?v=gNnE-fQrqqA>

1. Don't jump too quickly

2. Share the credit

Really get ahead, to share the credit.

3. Delegate

4. Work hard

5. Question everything

6. Focus on people

7. Be honest with yourself

8. Be prepared

9. Don't look over your shoulder

10. Collaborate

1. <https://www.youtube.com/watch?v=gNnE-fQrqqA>

---

## **HackerRank: String algorithm - funny string (2016-03-12 19:03)**

March 12, 2016

Julia likes to work on easy questions on HackerRank - string. So, she can develop more confident to write code.

Funny string:

[1]<https://gist.github.com/jianminchen/e577e9180305f0343fd5>



1. <https://gist.github.com/jianminchen/e577e9180305f0343fd5>

---

## **HackerRank - String algorithm - Alternating Characters (2016-03-12 19:32)**

March 12, 2015

Alternating characters, Julia's practice:

[1]<https://gist.github.com/jianminchen/e4863f946534124c7229>

1. <https://gist.github.com/jianminchen/e4863f946534124c7229>

---

## **HackerRank: String algorithm - Game Throne. (2016-03-12 21:52)**

March 12, 2016

Great workout! Julia made a few mistakes, and then, fixed the issues:

[1]<https://gist.github.com/jianminchen/e7038fa7e59c96378675>

Julia, your code is not short enough. Study other people's submission:

1. using Dictionary
2. declare array size of 26, not 256
3. other things - check string input etc.
4. using string manipulation, from an empty string, remove char if finding it, otherwise, append the char.

C #:

1. [2]<https://gist.github.com/jianminchen/6bf133d5c42e18260410>
2. [3]<https://gist.github.com/jianminchen/bd3834da7b28ae619ee2>
3. [4]<https://gist.github.com/jianminchen/16e9837b7b6b1e1a721c>

Read a few of solutions using Java, C++. Great experience, relax and have some fun.

1. <https://gist.github.com/jianminchen/e7038fa7e59c96378675>
  2. <https://gist.github.com/jianminchen/6bf133d5c42e18260410>
  3. <https://gist.github.com/jianminchen/bd3834da7b28ae619ee2>
  4. <https://gist.github.com/jianminchen/16e9837b7b6b1e1a721c>
-

## HackerRank - strings - GemStones (2016-03-12 23:26)

March 12, 2016

GemStones:

[1]<https://gist.github.com/jianminchen/aa85318f91fbcdc8f74d>

Julia, you do not need to use jagged array to store all the string, you can use one array, and then, keep adding to the array the count.

So good to learn from other submission - Julia likes HackerRank, quickly get updated with more ideas/ implementation / more informed.

### 1. Improvement 1 :

Here is other person's implementation - **less space, beat your solution!**

[2]<https://gist.github.com/jianminchen/cb0886705d99423a321f>

So, Julia, write another one - short one -

```
not: int[][] countA = new int[len][];
```

```
int[] sumA = new int[26];
```

space improvement, code is much short.

Julia wrote second implementation using one dimension int[26] instead of int[len][]

[3]<https://gist.github.com/jianminchen/10ece1c63d85e6ae3e12>

### 2. Improvement 2 :

Here is one of solution using

#### bit manipulation

- Great idea - try to use it as well!

Java

[4]<https://gist.github.com/jianminchen/c3d56eedcb794b0fa496>

Julia uses C # to implement the bit manipulation:

[5]<https://gist.github.com/jianminchen/aba5bad049353738a520>

one comment on line 39:

```
int
```

```
x = -
```

```
1
```

```
;
```

```
250
```

// julia, debug the code, and learn the idea to use bit manipulation

why  $x = -1$ ?

recall -1 is FFFF in bit expression; since  $1+1 = 0$ , so

FFFF

1

———

0000

### 3 solutions - space complexity using jagged array to one dimension array to one integer.

1. <https://gist.github.com/jianminchen/aa85318f91fbc8c8f74d>

2. <https://gist.github.com/jianminchen/cb0886705d99423a321f>

3. <https://gist.github.com/jianminchen/10ece1c63d85e6ae3e12>

4. <https://gist.github.com/jianminchen/c3d56eedcb794b0fa496>

5. <https://gist.github.com/jianminchen/aba5bad049353738a520>

---

### Sunday study time: Good advice how to have good work ethnics (2016-03-13 11:51)

March 13, 2016

Know one billionaire story, how he gives out advice how to do thing better. Self-made billionaire. Amazed that he likes to ride bicycle 60 miles a day and then keep fit.

Alan Michael Sugar Top 10 Rules For Success

[1][https://www.youtube.com/watch?v=J7ZXs5Lrs\\_0](https://www.youtube.com/watch?v=J7ZXs5Lrs_0)

1. Your destiny is in your hands

When he was young, his family lived in a council flat.

2. Recognize failure

After leaving school at 16, he worked as a statistician at the Ministry of Education.

3. Seize opportunities

He started selling electrical goods out of a van which he had bought with his savings of £50.

4. Have a good work ethic

5. Stick to what you know

6. Appreciate your team

7. **Have discipline**

8. **Track your progress**

9. Analyze your marketplace

He is a fan of and the former owner of Tottenham Hotspur.

10. Let your product do the talking

Billionaire Lord Sugar's Rich lifestyle and story (1 hour video)

[2]<https://www.youtube.com/watch?v=PuuAkoxlWmA>

Weight loss advice - that is much more important than how to be great at work.

[3]<http://www.express.co.uk/life-style/health/276673/Lord-Sugar-s-magic-formula-for-losing-weight>

**If you enjoy what you do, don't be afraid of expressing your enthusiasm. Enjoyment is infectious.** - Alan Sugar

1. [https://www.youtube.com/watch?v=J7ZXs5Lrs\\_0](https://www.youtube.com/watch?v=J7ZXs5Lrs_0)

2. <https://www.youtube.com/watch?v=PuuAkoxlWmA>

3. <http://www.express.co.uk/life-style/health/276673/Lord-Sugar-s-magic-formula-for-losing-weight>

---

**HackerRank: string algorithm - Make it anagram (2016-03-13 14:41)**

March 13, 2016

Make it anagram

252

Julia's practice:  
C # language

[1]<https://gist.github.com/jianminchen/050b64f483c9a251d472>

Read other's submission:

1. C #, using Dictionary, HashSet - Speically helpful, when you are not sure what Latin chars are. You just declare in general Dictionary, HashSet.

[2]<https://gist.github.com/jianminchen/5c6f6a70a1be4f71a98a>

2. C #, using List class, remove method:

[3]<https://gist.github.com/jianminchen/a2258616e596c0328660>

3. C #, kind of functional programming, new style to Julia - like JavaScript programming style

[4]<https://gist.github.com/jianminchen/2909916d9435f69c036b>

4. using one array, first string each char - add to the array, second string each char - take away from the array, sum of abs value of each item.

[5]<https://gist.github.com/jianminchen/fbf8e9049d3a1539ee87>

5. Using IList interface, distinct method, and functional programming - Lambda Expression <- Julia, you should try this code by yourself. It should be quickly picked up.

[6]<https://gist.github.com/jianminchen/fc8b4309efe00d67a640>

read this webpage to refresh Lambda expression:

[7]<https://msdn.microsoft.com/en-CA/library/bb397687.aspx>

6. Sort two strings first, and then, using two pointers - sliding forward

[8]<https://gist.github.com/jianminchen/41e6cbd30228e9ba7204>

7. using string operation, remove a char from a string - for any char in both two strings

[9]<https://gist.github.com/jianminchen/0138e2425592638dcf7a>

1. <https://gist.github.com/jianminchen/050b64f483c9a251d472>
  2. <https://gist.github.com/jianminchen/5c6f6a70a1be4f71a98a>
  3. <https://gist.github.com/jianminchen/a2258616e596c0328660>
  4. <https://gist.github.com/jianminchen/2909916d9435f69c036b>
  5. <https://gist.github.com/jianminchen/fbf8e9049d3a1539ee87>
  6. <https://gist.github.com/jianminchen/fc8b4309efe00d67a640>
  7. <https://msdn.microsoft.com/en-CA/library/bb397687.aspx>
  8. <https://gist.github.com/jianminchen/41e6cbd30228e9ba7204>
  9. <https://gist.github.com/jianminchen/0138e2425592638dcf7a>
- 

## HackerRank: string algorithm - Anagram (2016-03-13 22:41)

March 13, 2016

Anagram

Julia's C # implementation:

[1]<https://gist.github.com/jianminchen/f3c48ed9f3b16e8c5928>

Julia made a few tries before she noticed that she needs to figure out the formula:

```
for
(
int
i = 0; i < SIZE; i++)
{
if
(sumA[i] > 0)
// add count of chars in array sumA but not in sumB
// axxbbbxx,
// axxb -> bbxx, change a to b, that is it!
// axxb a 1, b 1, x 2
// bbxx, a 0, b 2, x 2
// formula ->
count += (sumA[i] >= sumB[i]) ? (sumA[i] - sumB[i]) : 0;
}
```

Another approach is to add all the differences, and then, divided by 2

C # submission code to study:

1. String class contains, Split functions etc.

Split function - return array length to get the count of any char in the string.

[2]<https://gist.github.com/jianminchen/adbfc2d809fb5b2bac78>

2. add all the difference between two strings, and then divide it by 2

[3]<https://gist.github.com/jianminchen/7bbe86bbb83787d6b98b>

3. Using Dictionary, KeyValuePair class

[4]<https://gist.github.com/jianminchen/78346475b6a7ce5d1681>

4. Read more Lambda expression code in C #

[5]<https://gist.github.com/jianminchen/9d121bd95266db41dfa8>

5. using StringBuilder, C # code

[6]<https://gist.github.com/jianminchen/d972656068fa8088ae70>

6. use string.Remove function

[7]<https://gist.github.com/jianminchen/65687cefd2b107ec5e23>

Java Code:

1. [8]<https://gist.github.com/jianminchen/794ffee7726df6062a1f>

2. Maybe not smart idea, but it works - declare a string

`String alph = "abcdefghijklmnopqrstuvwxyz";`

[9]<https://gist.github.com/jianminchen/0cfa60bac880f2bba10f>

Julia likes to read code, any language in submission. She could

not stop

reading, she has read more than 50 solutions, totally opened to so many creative ideas.

1. <https://gist.github.com/jianminchen/f3c48ed9f3b16e8c5928>

2. <https://gist.github.com/jianminchen/adbfc2d809fb5b2bac78>

3. <https://gist.github.com/jianminchen/7bbe86bbb83787d6b98b>

4. <https://gist.github.com/jianminchen/78346475b6a7ce5d1681>

5. <https://gist.github.com/jianminchen/9d121bd95266db41dfa8>

6. <https://gist.github.com/jianminchen/d972656068fa8088ae70>

7. <https://gist.github.com/jianminchen/65687cefd2b107ec5e23>

8. <https://gist.github.com/jianminchen/794ffee7726df6062a1f>

9. <https://gist.github.com/jianminchen/0cfa60bac880f2bba10f>

## Mock interview (4th practice): Matrix Spiral Print (2016-03-16 22:49)

March 16, 2016

Problem statement:

Given a 2D array (matrix) named

M

, print all items of

M

in a spiral order, clockwise.

For example:

```
M = 1 2 3 4 5  
    6 7 8 9 10  
    11 12 13 14 15  
    16 17 18 19 20
```

The clockwise spiral print is: 1 2 3 4 5 10 15 20 19 18 17 16 11 6 7 8 9 14 13 12

Julia worked on solution using brute force solution.

[1]<https://gist.github.com/jianminchen/aa7a35df305b05f5d90a>

Evaluation from the mock interviewer:

Problem Solving: answering correctly, without much help or hints



Okay

Got a brute force solution

- 

Coding: bug-less, clean, readable, reusable and maintainable code



So and So

256



Completed the code with several bugs

- 

Communication: clarity of your answers and line of reasoning



Doubted

I mostly didn't understand my peer

- 

Working Together: peer's motivation to be your colleague



Maybe

If I have to

- 

Creativity: original or innovative thinking

- Not Applicable -

- 

Things you did well:

Asked clarifying questions to understand the problem Broke down the problem to solve

- 

Things you should work on:

explain the solution before actually start coding

***So, Julia worked on more to come out better idea, to avoid bugs, make coding interesting.***

*Come out better idea after the mock interview.*

Here is new solution:

Actually, there is only one variable, which is  $i$ , from 0 to  $(N+1)/2$ ,  $N$  is how many columns in the matrix

So, the circle is a rectangle with points

$N$  - how many columns

$M$  - how many rows

Four corner: LT, LR, BR, BL

LT coordinates:  $(i, i)$

LR coordinates:  $(i, N-1-i)$

BR coordinates:  $(M-1-i, N-1-i)$

BL coordinates:  $(M-1-i, i)$

To test the correctness, just use  $i = 0$ ;

And then, you only need to design a function to iterate through

private static void leftToRight(Coordinate[] A)

TopToDown, RightToLeft, and DownToUp

$i$

0 1 2

----->

1

2 3 4 5

6 7 8 9 10

11 12 13 14 15

16 17 18 19 20

The above case, LT = (0,0), LR = (0, 4), BR = (3, 4), BL = (3, 0)

Julia, design the algorithm using one variable, and then simplify the algorithm, reduce the time to write and avoid bugs.

[2]<https://gist.github.com/jianminchen/7d775438e2d0d316f77d>

Actually, one more condition:

LT, BR, two pointers, make sure that  $M-1-i \geq i$ ,  $N-1-i \geq i$ ; in other words, left top pointer is above the bottom right pointer.

therefore, it should be  $i \leq \text{Math.Min}((M-1)/2, (N-1)/2)$

Weakness:

1. Jagged array initialization - take more than 5 minutes, look up internet;

2. Design has issue - only variable  $i$ , from 0 to  $(N+1)/2$ , <- original thought

should be:

0 to  $\text{Math.Min}((M-1)/2, (N-1)/2)$

Debug the test case, and then, find the bug; it takes extra 10 minutes, run through several test cases, and then another 10 minutes.

3. Good design - extra checking - save time to debug, fix the bug. Always think about more checking.

Ref: 2015 June - Julia's practice

[3]<http://juliachencoding.blogspot.ca/2015/06/leetcode-spiral-array-print-out.html>

1. <https://gist.github.com/jianminchen/aa7a35df305b05f5d90a>

2. <https://gist.github.com/jianminchen/7d775438e2d0d316f77d>

3. <http://juliachencoding.blogspot.ca/2015/06/leetcode-spiral-array-printout.html>

## Mock interview (practice III) - Award budget cut (2016-03-17 00:49)

March 17, 2016

Julia likes to figure out best ways to help herself grow as a software programmer. She likes to find peers to work together, solve the algorithm problems. She never had chance to interview others as a software programmer before 2016, she started to practice now.

So far, she did practice 4 times.

Lessons learned:

Learned to calm down, and set a small goal for each interview:

1. First 5-10 minutes, work on a test case, come out the solution to solve the test case; <- **People are complaining about Julia about this .**
2. Communicate the ideas using the test case, make sure that both are clear how to solve the problem;
3. Ask permission to write code
4. Code for test case, and then, extend the solution, fix the bug etc.

Mock Interview 3 - Award budget cut

The awards committee had planned to give  $n$  research grants this year, out of a its total yearly budget.

However, the budget was reduced to  $b$  dollars. The committee members has decided to affect the minimal number of highest grants, by applying a maximum cap  $c$  on all grants: every grant that was planned to be higher than  $c$  will now be  $c$  dollars.

**Help the committee to choose the right value of  $c$  that would make the total sum of grants equal to the new budget.**

Given an array of grants  $g$  and a new budget  $b$ , explain and code an efficient method to find the cap  $c$ .

Analyze the time and space complexity of your solution.

Julia got feedback:

YOU AS AN INTERVIEWEE

•

Problem Solving: answering correctly, without much help or hints



Strong

Got it with only few hints

•

Coding: bug-less, clean, readable, reusable and maintainable code



Ninja

Flawless code: readable, reusable & maintainable

•

Communication: clarity of your answers and line of reasoning



Okay

I was able to understand some parts

•

Working Together: peer's motivation to be your colleague



Yes

It will be nice

•

Creativity: original or innovative thinking



Picasso!

Re-invented the solution

•

Things you did well:

- You started out with examples and walked through them to find a solution to the problem - You considered the base cases (`arr == null`, `arr.Length <= 0`, `sum < b...`), this is very good practice - You came up with the sorting idea by yourself
- You found the final complexity of the algorithm - You nailed the linear search

•

Things you should work on:

- You calculated the sum at each call of the function, making the algorithm  $O(n^2)$  - You got a little bit mixed up with the indices, but overall you did it correctly - You need to work on communication, and mainly english as I had trouble understanding you at times (and you had trouble understanding me too); I assume you'd do much better with a native speaker though, it was a bit harder for me to understand your accent because english is not my main language

---

**Hacker Rank: Two Strings - thinking in C# 15+ ways (2016-03-18 21:41)**

March 18, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/two-strings>

Julia likes to try a new way to train herself to expand C #/ C++ / Java / JavaScript languages, by reading the solutions, followed up with some study about small topics, make learning more fun.

Coding - write code, read code, memorize some classical code.

Julia's C # implementation:

1. [2]<https://gist.github.com/jianminchen/f73e37b764147cd19c7b>

Julia reads 10 **C # solution** , C++, Java, JavaScript solutions, she likes to have this extensive reading on software code, and then, build up her daily routine as a software programmer. Coding is most important skills,

C # - Read first, then write

C++ - Read more C++ code, prepare to write any time

Java - Read more

JavaScript - Julia tries to read more JavaScript code, focus on reading for next 2 months

She also likes the following C # solution by other submissions:

[3]<https://www.hackerrank.com/challenges/two-strings/leaderboard/filter/?language=csharp>

Before you read the solution, can you think about using C # HashSet, Dictionary, String.Contains, HashSet.Overlap method, string.IndexOf, Hashtable, string.Intersect etc. solve the problem?

**HashSet,  
Dictionary,  
String.Contains,  
HashSet.Overlap method,  
string.IndexOf,  
Hashtable,  
string.Intersect  
Using two pointer to solve the problem**

**2. use hashset - more efficient**

[4]<https://gist.github.com/jianminchen/acedbb7cb86cf1c00131>

read hashset constructor: <- **Excellent, Julia is learning to write new C # code**

[5][https://msdn.microsoft.com/en-us/library/bb301504\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/bb301504(v=vs.110).aspx)

ToList

[6][https://msdn.microsoft.com/en-us/library/bb342261\(v=vs.100\).aspx](https://msdn.microsoft.com/en-us/library/bb342261(v=vs.100).aspx)

[7][https://msdn.microsoft.com/library/bb534972\(v=vs.100\).aspx](https://msdn.microsoft.com/library/bb534972(v=vs.100).aspx)

**3. using string.intersect method**

[8]<https://gist.github.com/jianminchen/cece566bd69963533e80>

**4. using C #, var, foreach, Any method - interesting to read**

[9]<https://gist.github.com/jianminchen/50fc6b5a13b7d62dfa1d>

## 5. **IEnumerable - interesting to read the code**

[10]<https://gist.github.com/jianminchen/cdba6a97769db30c2e76>

Read IEnumerable: C++ analog - duck typing

[11]<http://stackoverflow.com/questions/8764643/is-there-a-standard-c-equivalent-of-ienumerable-in-c>

IEnumerable - Java analog

[12]<http://stackoverflow.com/questions/362367/java-arrays-generics-java-equivalent-to-c-sharp-ienumerable>

## 6. **Two hashsets**

[13]<https://gist.github.com/jianminchen/aa87f82f0247c4b59ab6>

## 7. **HashSet - overlaps method** - code is readable

[14]<https://gist.github.com/jianminchen/b28fab138ad778bc4326>

read HashSet overlap api - learn some design:

[15][https://msdn.microsoft.com/en-us/library/bb355623\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/bb355623(v=vs.110).aspx)

## 8. **Array, Array.BinarySearch method** - it is not time efficient solution - but code is reusable.

[16]<https://gist.github.com/jianminchen/775cf5e16d065e66feae>

## 9. **Declare new struct data, use byte type** , and code is beautiful, succinct.

[17]<https://gist.github.com/jianminchen/9d80b3995adf0b01f9d7>

## 10. **use two pointers, move forward if need separately** - can be reused.

[18]<https://gist.github.com/jianminchen/7de04dcfb8740ee19412>

## 11. Two things Julia likes:

'z'-'a' in array declaration,

second one is to use one array `int[26]`, not 2; second one is char array

[19]<https://gist.github.com/jianminchen/de154d443a1434094260>

bool

[] founded =

new

bool

[

'z'

-

'a'

262

```
+  
1  
];
```

char

```
[ ] secondWord = Console.ReadLine().ToCharArray();
```

## 12. Use Hashtable

[20]<https://gist.github.com/jianminchen/213335b4dec3988facfe>

## 13. Use string.Contains() method

[21]<https://gist.github.com/jianminchen/6388585c990a42ea76ae>

## 14. Use Dictionary class

[22]<https://gist.github.com/jianminchen/95b701e6d2146da13c1d>

Dictionary<int, char>

Dictionary<

int

,

char

> dex =

new

Dictionary<

int

,

char

>());

Dictionary method ContainsValue()

## 15. string.ToCharArray(), Distinct() of Char Array, ToList(), ToArray(), string.IndexOf methods

[23]<https://gist.github.com/jianminchen/7562c5dd310508d48f58>

## 16. use string.Contains , two dimension array

[24]<https://gist.github.com/jianminchen/365eb2b69e93573c0b5d>

## 17. use Dictionary<char, bool>

[25]<https://gist.github.com/jianminchen/a91504a793f795886e23>

18. use **StringBuilder** to concatenate output

[26]<https://gist.github.com/jianminchen/1fcc46803321f007eb1b>

Read the discussion **StringBuilder vs string to concatenation**

[27]<http://stackoverflow.com/questions/1612797/string-concatenation-vs-string-builder-performance>

[28]<https://support.microsoft.com/en-us/kb/306822>

More reading: 1

19.

use " " to concatenate two input strings and then .Split them

[29]<https://gist.github.com/jianminchen/591738de12e21e591634>

More Julia likes:

IEnumerable -

[30][https://www.hackerrank.com/rest/contests/master/challenges/two-strings/hackers/cdkmoose/download\\_solution](https://www.hackerrank.com/rest/contests/master/challenges/two-strings/hackers/cdkmoose/download_solution)

Statistics:

More than 3 hours to work on this study, thinking in C # -

Advice for C # programmers:

Julia enjoyed the study. Very focus, try to understand the solution, how other people think in C #. It is the big world, millions programmers in the world today, you have to go out to reach them, some wisdom out there, teach you to learn C # programming language so many ways. Just dig into the code and then find out.

No one person can put together so many ideas for a simple problem.

Julia is proud of herself, come to some simple idea to train herself in algorithm thinking. Low cost, and easy to access, and good quality code.

More reading:

18. use **StringBuilder** to concatenate output

Here are good ideas for Julia to be a better C # developer as the anonymous code writer in 18:

(From webpage: [31]<https://support.microsoft.com/en-us/kb/306822>)



1. the benefits of using the StringBuilder class over traditional concatenation techniques
2. C/C++ strcat() - to allocate a large character array as a buffer and copy string data into the buffer.
3. In .NET framework, a string is immutable; it cannot be modified in place. <- **immutable explanation!**
4. **The C # + concatenation operator builds a new string and causes reduced performance when it concatenates large amounts of text.**
5. **.NET framework, a StringBuilder class is optimized for string concatenation.**  
same as using a character array in C/C++, as well as automatically growing the buffer size (if needed) and tracking the length for you.
6. **Reuse existing StringBuilder class rather than reallocate each time you need one. This limits the growth of the heap and reduces garbage collection. StringBuilder makes more efficient use of the heap than using the + operator.**

1. <https://www.hackerrank.com/challenges/two-strings/submissions/code/18506948>
2. <https://gist.github.com/jianminchen/f73e37b764147cd19c7b>
3. <https://www.hackerrank.com/challenges/two-strings/leaderboard/filter/language=csharp>
4. <https://gist.github.com/jianminchen/acedbb7cb86cf1c00131>
5. [https://msdn.microsoft.com/en-us/library/bb301504\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/bb301504(v=vs.110).aspx)
6. [https://msdn.microsoft.com/en-us/library/bb342261\(v=vs.100\).aspx](https://msdn.microsoft.com/en-us/library/bb342261(v=vs.100).aspx)
7. [https://msdn.microsoft.com/library/bb534972\(v=vs.100\).aspx](https://msdn.microsoft.com/library/bb534972(v=vs.100).aspx)
8. <https://gist.github.com/jianminchen/cece566bd69963533e80>
9. <https://gist.github.com/jianminchen/50fc6b5a13b7d62dfa1d>
10. <https://gist.github.com/jianminchen/cdba6a97769db30c2e76>
11. <http://stackoverflow.com/questions/8764643/is-there-a-standard-c-equivalent-of-ienumerable-in-c>
12. <http://stackoverflow.com/questions/362367/java-arrays-generics-java-equivalent-to-c-sharp-ienumerable>
13. <https://gist.github.com/jianminchen/aa87f82f0247c4b59ab6>
14. <https://gist.github.com/jianminchen/b28fab138ad778bc4326>
15. [https://msdn.microsoft.com/en-us/library/bb355623\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/bb355623(v=vs.110).aspx)
16. <https://gist.github.com/jianminchen/775cf5e16d065e66feae>
17. <https://gist.github.com/jianminchen/9d80b3995adf0b01f9d7>
18. <https://gist.github.com/jianminchen/7de04dcfb8740ee19412>
19. <https://gist.github.com/jianminchen/de154d443a1434094260>
20. <https://gist.github.com/jianminchen/213335b4dec3988facfe>
21. <https://gist.github.com/jianminchen/6388585c990a42ea76ae>
22. <https://gist.github.com/jianminchen/95b701e6d2146da13c1d>
23. <https://gist.github.com/jianminchen/7562c5dd310508d48f58>
24. <https://gist.github.com/jianminchen/365eb2b69e93573c0b5d>
25. <https://gist.github.com/jianminchen/a91504a793f795886e23>
26. <https://gist.github.com/jianminchen/1fcc46803321f007eb1b>
27. <http://stackoverflow.com/questions/1612797/string-concatenation-vs-string-builder-performance>
28. <https://support.microsoft.com/en-us/kb/306822>
29. <https://gist.github.com/jianminchen/591738de12e21e591634>
30. [https://www.hackerrank.com/rest/contests/master/challenges/two-strings/hackers/cdkmoose/download\\_solution](https://www.hackerrank.com/rest/contests/master/challenges/two-strings/hackers/cdkmoose/download_solution)
31. <https://support.microsoft.com/en-us/kb/306822>

## HackerRank: Two string - thinking in JavaScript over 10 ways (2016-03-19 22:22)

March 19, 2016

A new drill for JavaScript practice - **Two string on HackerRank** , read JavaScript submissions.

First step is to read code, talk about it.

Problem Statement:

[1]<https://www.hackerrank.com/challenges/two-strings/submissions/code/18506948>

JavaScript solutions she chooses to read:

[2]<https://www.hackerrank.com/challenges/two-strings/leaderboard/filter/language=javascript>

**Solution 1:** [3]<https://gist.github.com/jianminchen/12a0caba6512b206e276>

Review Array.shift function:

[4][http://www.w3schools.com/jsref/tryit.asp?filename=tryjsref\\_shift](http://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_shift)

JavaScript Array.splice method:

[5][http://www.w3schools.com/jsref/jsref\\_splice.asp](http://www.w3schools.com/jsref/jsref_splice.asp)

[6][http://www.w3schools.com/jsref/jsref\\_indexof.asp](http://www.w3schools.com/jsref/jsref_indexof.asp)

process.stdin - try to figure out what it is

[7]<http://stackoverflow.com/questions/26460324/how-to-work-with-process-s-stdin-on>

[8]<http://stackoverflow.com/questions/9231847/node-js-how-to-detect-an-empty-stdin-stream>

**Solution 2:**

[9]<https://gist.github.com/jianminchen/660a28565839246c2c83>

**Solution 3:**

[10]<https://gist.github.com/jianminchen/25edcfe155c8859a922c>

(continued on March 21, 2016)

**Solution 4:**

[11]<https://gist.github.com/jianminchen/c896d4a598b92d127726>

**Solution 5:**

**Structured code, using prototype, simulated class function, new/ this etc., two sliding pointers, make me laugh**

[12]<https://gist.github.com/jianminchen/8f1d0a7d1765ba5cdc83>

read one more article to entertain the idea of using new/ this in JavaScript:

[13]<http://stackoverflow.com/questions/5224295/javascript-the-good-parts-how-to-not-use-new-at-all>

**Solution 6:**

**JavaScript, things I like: using `j== first.length` to determine YES/ NO, the big problem of coding - repeat same code in if/ else code.**

[14]<https://gist.github.com/jianminchen/8899cb7587b0389e6ff6>

line 6 - line 26 - Julia likes to swap two strings to ensure `str1.length < str2.length`

**Solution 7:**

swap two string if need.

[15]<https://gist.github.com/jianminchen/10f5cfa5c1ab22070922>

**Solution 8:**

**write a function to add unique chars into the array** - Julia likes the function.

[16]<https://gist.github.com/jianminchen/fe302cf65ac762e7f3eb>

object literal { } - act like a hashmap

search function time complexity  $O(N^2)$  - not efficient

A discussion on object literal -

[17]<http://stackoverflow.com/questions/17486854/how-to-create-a-method-in-object-literal-notation>

**Solution 9:**

**use jagged array - good practice**

[18]<https://gist.github.com/jianminchen/8dca6d1fd8fe96b7323c>

**Solution 10:**

**using object literal - creating a hashtable, and then, first string goes into as Hashtable - add it if not in; second string, add one on existing one.**

[19]<https://gist.github.com/jianminchen/ee9fbf9612c8ada8ccae>

**code can be refactored, early return line 23, not wait until on line 30 .**

**Solution 11:**

`Array.prototype.ForEach`

[20]<https://gist.github.com/jianminchen/69119913341c9bea8b0c>

[21]<https://gist.github.com/jianminchen/bfe92a2556aab268d250>

read this blog to understand the function:

[22][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Array/forEach](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/forEach)

[23][https://msdn.microsoft.com/en-us/library/ff679980\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/ff679980(v=vs.94).aspx)

still concern about value, index, array arguments in ForEach function.

spent 10 minutes to go over the topic:

[24]<http://neversaw.us/2011/01/16/not-your-fathers-javascript/>

Good article, stopped here: (continue later)

[25]<http://neversaw.us/2011/01/16/not-your-fathers-javascript/#binding-is-great>

Array iteration:

[26][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Array#Iteration\\_methods](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array#Iteration_methods) (20 minutes reading)

Too much detail, just 5 minutes reading:

[27]<http://andrewdupont.net/2006/05/18/javascript-associative-arrays-considered-harmful/>

[28]<http://www.less-broken.com/blog/2010/12/lightweight-javascript-dictionaries.html>

### **Solution 12:**

**using string to contain 26 characters, then look up each one in two string, use Array.indexOf method**

[29]<https://gist.github.com/jianminchen/e9667ffa4ff65a6c49b8>

### **Solution 13:**

use RegExp expression

[30]<https://gist.github.com/jianminchen/7bdc395289554ca1c779>

### **Solution 14:**

use Binary search

[31]<https://gist.github.com/jianminchen/b4e6a00926b5db8138ea>

**Statistics :** Hour spent: 8 hours

April 4, 2016

Julia needs to warmup JavaScript every week, she likes the drill; After 8 hours practice using the drill, she noticed that the code she wrote before was so bad, immediately, she spent 2 days to rewrite some of them, in middle of March.

1. <https://www.hackerrank.com/challenges/two-strings/submissions/code/18506948>
2. <https://www.hackerrank.com/challenges/two-strings/leaderboard/filter/language=javascript>
3. <https://gist.github.com/jianminchen/12a0caba6512b206e276>
4. [http://www.w3schools.com/jsref/tryit.asp?filename=tryjsref\\_shift](http://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_shift)
5. [http://www.w3schools.com/jsref/jsref\\_splice.asp](http://www.w3schools.com/jsref/jsref_splice.asp)
6. [http://www.w3schools.com/jsref/jsref\\_indexof.asp](http://www.w3schools.com/jsref/jsref_indexof.asp)
7. <http://stackoverflow.com/questions/26460324/how-to-work-with-process-stdin-on>
8. <http://stackoverflow.com/questions/9231847/node-js-how-to-detect-an-empty-stdin-stream>
9. <https://gist.github.com/jianminchen/660a28565839246c2c83>
10. <https://gist.github.com/jianminchen/25edcfe155c8859a922c>
11. <https://gist.github.com/jianminchen/c896d4a598b92d127726>
12. <https://gist.github.com/jianminchen/8f1d0a7d1765ba5cdc83>
13. <http://stackoverflow.com/questions/5224295/javascript-the-good-parts-how-to-not-use-new-at-all>
14. <https://gist.github.com/jianminchen/8899cb7587b0389e6ff6>
15. <https://gist.github.com/jianminchen/10f5cfa5c1ab22070922>
16. <https://gist.github.com/jianminchen/fe302cf65ac762e7f3eb>
17. <http://stackoverflow.com/questions/17486854/how-to-create-a-method-in-object-literal-notation>

18. <https://gist.github.com/jianminchen/8dca6d1fd8fe96b7323c>
  19. <https://gist.github.com/jianminchen/ee9fbf9612c8ada8ccae>
  20. <https://gist.github.com/jianminchen/69119913341c9bea8b0c>
  21. <https://gist.github.com/jianminchen/bfe92a2556aab268d250>
  22. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Array/forEach](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/forEach)
  23. [https://msdn.microsoft.com/en-us/library/ff679980\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/ff679980(v=vs.94).aspx)
  24. <http://neversaw.us/2011/01/16/not-your-fathers-javascript/>
  25. <http://neversaw.us/2011/01/16/not-your-fathers-javascript/#binding-is-great>
  26. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Array#Iteration\\_methods](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array#Iteration_methods)
  27. <http://andrewdupont.net/2006/05/18/javascript-associative-arrays-considered-harmful/>
  28. <http://www.less-broken.com/blog/2010/12/lightweight-javascript-dictionaries.html>
  29. <https://gist.github.com/jianminchen/e9667ffa4ff65a6c49b8>
  30. <https://gist.github.com/jianminchen/7bdc395289554ca1c779>
  31. <https://gist.github.com/jianminchen/b4e6a00926b5db8138ea>
- 

## **BootStrap CSS framework: Building Responsive UI with Bootstrap 6-hours - Full Sample code (2016-03-20 11:19)**

March 20, 2016

Plan to spend 6 hours to watch this course: (Spent time from 10:00am - 4:00pm)

[1][https://mva.microsoft.com/en-us/training-courses/building-responsive-ui-with-bootstrap-8378?l=BDfMAHIz\\_3004984382](https://mva.microsoft.com/en-us/training-courses/building-responsive-ui-with-bootstrap-8378?l=BDfMAHIz_3004984382)

One hour on March 20, 2016

Go over [2][bootswatch.com](http://bootswatch.com)

[3]<https://mobirise.com/bootstrap-carousel/>

[4]<http://getbootstrap.com/javascript/#carousel>

Julia likes to find the bootstrap carousel multiple frames at once -

[5]<http://stackoverflow.com/questions/20007610/bootstrap-3-carousel-multiple-frames-at-once>

[6]<http://www.bootply.com/92514>

1. [https://mva.microsoft.com/en-us/training-courses/building-responsive-ui-with-bootstrap-8378?l=BDfMAHIz\\_3004984382](https://mva.microsoft.com/en-us/training-courses/building-responsive-ui-with-bootstrap-8378?l=BDfMAHIz_3004984382)
2. <http://bootswatch.com/>
3. <https://mobirise.com/bootstrap-carousel/>

4. <http://getbootstrap.com/javascript/#carousel>
  5. <http://stackoverflow.com/questions/20007610/bootstrap-3-carousel-multiple-frames-at-once>
  6. <http://www.bootply.com/92514>
- 

## Reading time: blogs (2016-03-20 22:53)

March 20, 2016

Blogs to read in next week:

The 80/20 rule basically states that it will take you 20 % of a project's time to achieve 80 % of the desired effect, and then the remaining 80 % of time to just get the last 20 % right.

[1]<https://medium.com/@maebert/9-things-i-learned-as-a-software-engineer-c2c9f76c9266#.xhxvsi00>

How to work smart in the office, share space, avoid interruption etc.?

[2]<https://www.facebook.com/notes/will-hughes/how-to-level-up-as-a-developer/10153879894028632>

War for tech talent - unbelievable high salary, benefits.

[3]<http://www.andiamogo.com/war-for-tech-talent>

[4]<http://blog.alinelerner.com/>

[5]<http://www.interviewing.io/>

[6]<http://www.gainlo.co/#!/>

[7]<http://blog.gainlo.co/>

1. <https://medium.com/@maebert/9-things-i-learned-as-a-software-engineer-c2c9f76c9266#.xhxvsi00>

2. <https://www.facebook.com/notes/will-hughes/how-to-level-up-as-a-developer/10153879894028632>

3. <http://www.andiamogo.com/war-for-tech-talent>

4. <http://blog.alinelerner.com/>

5. <http://www.interviewing.io/>

6. <http://www.gainlo.co/#!/>

7. <http://blog.gainlo.co/>

---

## Leetcode 238? Product of Array except itself (2016-03-21 22:50)

March 21, 2016

Worked on the algorithm LeetCode 238:

Product of an array

[1][http://juliachencoding.blogspot.ca/2015/07/two-algorithms-questions.ht ml](http://juliachencoding.blogspot.ca/2015/07/two-algorithms-questions.html)

[2]<http://fisherlei.blogspot.ca/2015/10/leetcode-product-of-array-excep t-self.html>

Julia likes to share some tips about writing the code, assuming that you know the optimal solution is using dynamic programming (DP), and extra space is  $O(N)$ .

For example,

Array: [1, 2, 3, 4], denoted as `arr[]`.

for each  $i$ ,  $i = 0$  to 3,  
product of  $i$ , denoted as  $P[i]$ ,  
 $\text{leftP}[i] = \text{arr}[0] * \text{arr}[1] * \dots * \text{arr}[i-1] = \text{leftP}[i-1] * \text{arr}[i-1]$ .  
 $\text{rightP}[i] = \dots$   
 $P[i] = \text{leftP}[i] * \text{rightP}[i]$

write C # code:

```
1 public static int[] getProduct(int[] arr) // not A, match description: arr <- complaint 1, style
2 {
3 if(arr== null || arr.Length==0) return null;
4
5 int n = arr.Length;
6 int[] res = new int[n];
7
8 int tmp =1;
9 for(int i=0; i< n; i++)
10 {
11 if( i ==0) // <- complaint 2:
12 res[0] = 1; // <- complaint 2: not necessary, remove if clause
13
14 }
```

So, write again starting from line 9. Remember the analysis, only thing you have to do, one multiplication, one extra variable to store previous result - tmp.

Code should match the analysis <- **Julia learned the lesson.**

```
8 int tmp = 1;
9 for(int i=0; i< n; i++)
10 {
11 res[i] = tmp; // it works when i = 0;
12 tmp *= arr[i];
13 }
```

that is the code for `leftP[i]`.

Next practice:

[3]<https://www.hackerrank.com/challenges/computing-the-correlation>

[4]<https://www.hackerrank.com/challenges/unbounded-knapsack>

1. <http://juliachencoding.blogspot.ca/2015/07/two-algorithms-questions.html>
  2. <http://fisherlei.blogspot.ca/2015/10/leetcode-product-of-array-except-self.html>
  3. <https://www.hackerrank.com/challenges/computing-the-correlation>
  4. <https://www.hackerrank.com/challenges/unbounded-knapsack>
- 

## **Pluralsight: Angular Fundamentals (2016-03-22 21:29)**

March 22, 2016

Spent hours to watch course "Angular Fundamentals". 6 hours video. Be patient, and take some notes.

[1]<https://app.pluralsight.com/library/courses/angularjs-fundamentals/table-of-contents>

Introduction to Angular - 25m 20s

Angular Controller & Markup - 83 minutes

Creating and Using Angular Services - 83 minutes

Angular Routing - 51 minutes - March 22

Creating Custom Angular Directives - 76 minutes

Test Angular - 89 minutes

1. <https://app.pluralsight.com/library/courses/angularjs-fundamentals/table-of-contents>
- 

## **AngularJs fundamentals (2016-03-22 21:40)**

March 22, 2016

Angular one hour video



[1]<http://weblogs.asp.net/dwahlin/video-tutorial-angularjs-fundamentals-in-60-ish-minutes>

Spent hours to read code examples.

1. <http://weblogs.asp.net/dwahlin/video-tutorial-angularjs-fundamentals-in-60-ish-minutes>

---

### **Pluralsight: Play by Play: Modernizing C++ Code with Kate Gregory (2016-03-23 20:37)**

March 23 ,2016

Another 2 hours to work on the course:

Play by Play: Modernizing C++ Code with Kate Gregory

Julia learned a few things.

1. For loop - code review and concern, comparison to foreach range and for
  2. While loop
  3. Using the nullptr keyword
  - 4.
  - 5.
  - 6.
  - 7.
  - 8.
  - 9.
  - 10.
- 

### **Learning AngularJS - developer org document (2016-03-23 20:38)**

March 23, 2016

Spend more than 20 hours already to learn AngularJS. Julia started to read the developer document, and play with some code.

Spent time to read the document first:

[1] <https://docs.angularjs.org/guide/introduction>

[2]<https://docs.angularjs.org/guide/directive>

plan to watch AngularJS video from Microsoft:

[3][https://mva.microsoft.com/en-US/training-courses/introduction-to-angularjs-8682?l=qJ8KNLH1\\_1804984382](https://mva.microsoft.com/en-US/training-courses/introduction-to-angularjs-8682?l=qJ8KNLH1_1804984382)

1. <https://docs.angularjs.org/guide/introduction>

2. <https://docs.angularjs.org/guide/directive>

3. [https://mva.microsoft.com/en-US/training-courses/introduction-to-angularjs-8682?l=qJ8KNLH1\\_1804984382](https://mva.microsoft.com/en-US/training-courses/introduction-to-angularjs-8682?l=qJ8KNLH1_1804984382)

---

## Learning JavaScript (2016-03-23 20:39)

March 23, 2016

Favorite book in 2015: JavaScript and some PHP.

Book:

JavaScript and PHP

2/10/2015 - 3/27/2015

[1] [2] <https://github.com/jianminchen/JavaScriptAndPHP>

Need to go back to review the practice again, review what Julia learned in 2015.

1. <https://github.com/jianminchen/JavaScriptAndPHP>

2. <https://github.com/jianminchen/JavaScriptAndPHP>

---

## Pluralsight: C++ Core Guidelines and the Guideline Support Library (2016-03-23 20:40)

March 23, 2016 - 2 hours course

First Look: C++ Core Guidelines and the Guideline Support Library

Two hours lecture:

Great talk with code example: "express the intent", "const", "null"

Tips Julia likes:

Install Nuget:

C++ Core Checker

[1] <https://www.nuget.org/packages/Microsoft.CppCoreCheck/>

code analysis: <- check warnings

Extra Clang Tools 3.8 documentation

[2] <http://clang.llvm.org/extra/clang-tidy/checks/list.html>

clang-Tidy Checks

Most guidelines can't be checked by a tool

Those that can are gathered into profiles

At least two tools exist today to check your code against profiles:

**CppCoreCheck for Visual Studio**

**Clang-tidy**

Keep learning is much fun to use [www.pluralsight.com](http://www.pluralsight.com). - 2016 Julia Chen

[3] <http://www.gregcons.com/kateblog/>

Plan to watch courses provided by Kate Gregory - Julia's favorite C++ teacher:

C++ Advanced Topics

C++ Fundamentals and Part 2

Using StackOverflow and Other StackExchange Sites

Play by Play: Modernizing C++ Code with Kate Gregory

Reference:

[4] <http://juliachencoding.blogspot.ca/2015/12/cpp-core-guidelines.html>

1. <https://www.nuget.org/packages/Microsoft.CppCoreCheck/>

2. <http://clang.llvm.org/extra/clang-tidy/checks/list.html>

3. <http://www.gregcons.com/kateblog/>
  4. <http://juliachencoding.blogspot.ca/2015/12/cpp-core-guidelines.html>
- 

## **Pluralsight: C++ Advanced Topics (2016-03-23 22:14)**

March 23, 2016

Pluralsight:

C++ Advanced Topics

[1]<https://www.pluralsight.com/courses/adv-cpp>

Avoid Manual Memory Management

Use Lambdas

Use Standard Containers

Use Standard Algorithms

Embrace Move Semantics

Follow Style Rules

Consider the PImpl Idiom

Stop Writing C with Classes

Julia is still trying to figure out ways to know best teachers in C++/ C#. Where to find them? Julia, stay focus, find several articles from stackOverflow about C++, related to Lambdas, Move semantics, have some serious reading, then, come back to the video.

Watch some videos:

[2][https://www.youtube.com/playlist?list=PL\\_AKIMJc4roXG7rOmqsB\\_wDG1btCzhS8F](https://www.youtube.com/playlist?list=PL_AKIMJc4roXG7rOmqsB_wDG1btCzhS8F)

1. <https://www.pluralsight.com/courses/adv-cpp>

2. [https://www.youtube.com/playlist?list=PL\\_AKIMJc4roXG7rOmqsB\\_wDG1btCzhS8F](https://www.youtube.com/playlist?list=PL_AKIMJc4roXG7rOmqsB_wDG1btCzhS8F)

---

## **HackerRank: Two string - thinking in Cplusplus over 15 ways (2016-03-24 19:33)**

March 24, 2016

Julia likes to get idea how to write modern C++, she started to read at least 50 C++ solutions a week.

Always look for brilliant ideas to solve a simple problem. Take time to read other people's code, and then, find really excellent ideas.

**Algorithm problem solving -> More ideas -> Good thinker -> Confidence.**

Julia, think in C++ - solve problems using the following:

1. bit operation

2. array int[26]
3. class set
4. class unordered\_set, functions: reserve, insert, count
5. class bitset, size\_t
5. cin vs gets, pointer - some calculation

Problem statement:

[1]<https://www.hackerrank.com/challenges/two-strings>

#### **Solution 1:**

Here it is, namespace std, >>, string class, set<char>, auto, &x, for(auto &x: A), modern C++ style

[2]<https://gist.github.com/jianminchen/484bd7705844da26ecd0>

#### **Solution 2: using bit operation - beautiful code, genius!**

source code reference:

[3]<https://problemsolvingnotes.wordpress.com/page/2/>

[4]<https://www.hackerrank.com/MarioYC>

ACM ICPC World Finalist 2012, 2013

count - reference:

[5]<http://www.cplusplus.com/reference/algorithm/count/>

sync with stdio

[6][http://www.cplusplus.com/reference/ios/ios\\_base/sync\\_with\\_stdio/](http://www.cplusplus.com/reference/ios/ios_base/sync_with_stdio/)

using | inclusive OR

[7]<http://www.cplusplus.com/doc/tutorial/operators/>

solution in C++:

[8]<https://gist.github.com/jianminchen/a0724f1fbbe5c621cd63>

using namespace std;

```
int main() {
    ios::sync_with_stdio(0);
```

```
    string A,B;
```

```
    int T;
```

```
    cin >> T;
```

```
    int cont[26];
```

```
    while(T--) {
        cin >> A >> B;
```

```
        memset(cont,0,sizeof cont);
```

```

for(int i = 0; i < A.size(); ++i)
    cont[ A[i] - 'a' ] |= 1; // Julia's comment: set first bit as 1, using | - inclusive OR

for(int i = 0; i < B.size(); ++i)
    cont[ B[i] - 'a' ] |= 2; // Julia's comment: set second bit as 1, using | - inclusive OR

if(count(cont, cont + 26, 3) > 0) cout << "YES\n";

// check array cont - if there is any number is 3, if found, return true
else cout << "NO\n";
}

return 0;
}

```

### Solution 3:

[9]<https://gist.github.com/jianminchen/6b443714e61f1c8c1382>

### Solution 4:

[10]<https://gist.github.com/jianminchen/dcf885d390939a08ae2>

### Solution 5:

[11]<https://gist.github.com/jianminchen/13c2d3832f1cbd53433c>

### Solution 6: set class, insert, find, end functions

read set class end function:

[12]<http://www.cplusplus.com/reference/set/set/find/>  
 set class insert function:  
 [13]<http://www.cplusplus.com/reference/set/set/insert/>

[14]<https://gist.github.com/jianminchen/1fd8c8e9acdc72f18d10>

### Solution 7:

[15]<https://gist.github.com/jianminchen/f4b6dcea9936347192e9>

### Solution 8: unordered\_set class, reserved function, insert, count function

unordered\_set class reserve function: [16][http://www.cplusplus.com/reference/unordered\\_set/unordered\\_set/reserve/](http://www.cplusplus.com/reference/unordered_set/unordered_set/reserve/)

[17]<https://gist.github.com/jianminchen/a3865ec039c092476492>

### Solution 9: cin vs gets, scanf, pointer - calculation

warm up with some concepts:

C++ gets:

[18]<http://www.cplusplus.com/reference/cstdio/gets/>

compare two solutions - difference

[19]<https://gist.github.com/jianminchen/f35171b0c2ddcd403c2b>

code is better! - [20]<https://www.hackerrank.com/avolchek> 24 Gold

[21]<https://gist.github.com/jianminchen/d9c0d7a25ca2e24ca17b>

#### **Solution 11:**

##### **warm up with std::fill function template:**

[22]<http://www.cplusplus.com/reference/algorithm/fill/>

very strong C++ coder:

[23]<https://www.hackerrank.com/Informatimukas>

[24]<https://gist.github.com/jianminchen/ff3fd27b8c1245322adf>

#### **Solution 12: bitset, size\_t**

[25]<https://gist.github.com/jianminchen/765ba74888057b887cac>

#### **Solution 13: vector class template**

[26]<https://gist.github.com/jianminchen/c0e3a44edc31b3804331>

#### **Solution 14:**

##### **very readable - two functions**

[27]<https://gist.github.com/jianminchen/43f23e82ee04076c7496>

1. <https://www.hackerrank.com/challenges/two-strings>
2. <https://gist.github.com/jianminchen/484bd7705844da26ecd0>
3. <https://problemsolvingnotes.wordpress.com/page/2/>
4. <https://www.hackerrank.com/MarioYC>
5. <http://www.cplusplus.com/reference/algorithm/count/>
6. [http://www.cplusplus.com/reference/ios/ios\\_base/sync\\_with\\_stdio/](http://www.cplusplus.com/reference/ios/ios_base/sync_with_stdio/)
7. <http://www.cplusplus.com/doc/tutorial/operators/>
8. <https://gist.github.com/jianminchen/a0724f1fbbe5c621cd63>
9. <https://gist.github.com/jianminchen/6b443714e61f1c8c1382>
10. <https://gist.github.com/jianminchen/dcf885d390939a08ae2>
11. <https://gist.github.com/jianminchen/13c2d3832f1cbd53433c>
12. <http://www.cplusplus.com/reference/set/set/find/>
13. <http://www.cplusplus.com/reference/set/set/insert/>
14. <https://gist.github.com/jianminchen/1fd8c8e9acdc72f18d10>
15. <https://gist.github.com/jianminchen/f4b6dcea9936347192e9>
16. [http://www.cplusplus.com/reference/unordered\\_set/unordered\\_set/reserve/](http://www.cplusplus.com/reference/unordered_set/unordered_set/reserve/)
17. <https://gist.github.com/jianminchen/a3865ec039c092476492>
18. <http://www.cplusplus.com/reference/cstdio/gets/>
19. <https://gist.github.com/jianminchen/f35171b0c2ddcd403c2b>
20. <https://www.hackerrank.com/avolchek>
21. <https://gist.github.com/jianminchen/d9c0d7a25ca2e24ca17b>
22. <http://www.cplusplus.com/reference/algorithm/fill/>
23. <https://www.hackerrank.com/Informatimukas>
24. <https://gist.github.com/jianminchen/ff3fd27b8c1245322adf>

25. <https://gist.github.com/jianminchen/765ba74888057b887cac>  
26. <https://gist.github.com/jianminchen/c0e3a44edc31b3804331>  
27. <https://gist.github.com/jianminchen/43f23e82ee04076c7496>

---

## Leetcode 33: Search in sorted rotated array (2016-03-25 11:32)

March 23, 2016

understand the algorithm - good analysis graph in the following blog:

[1]<http://fisherlei.blogspot.ca/2013/01/leetcode-search-in-rotated-sorted-array.html>

using this analysis in the blog:

First, Julia, you have to find out which half is sorted; only one of them;

Second, use sorted half to determine if the search is in the sorted half or not;

Then, you go to sorted half/ unsorted half.

Code can be optimized, if it is in sorted half, then, go to normal binary search

otherwise, go to unsorted - modified binary search - use recursive to write a solution first.

[2]<http://bangbingsyb.blogspot.ca/2014/11/leetcode-search-in-rotated-sorted-array.html>

Julia, in your practice, you discuss that the middle point is a peak element/ valley element or not; and then, both halves are sorted, which is the special case.

Lesson learned:

1. In your analysis, you have to make the test case as simple as possible, as you see in the above blog:  
use 0 -7,

0 1 2 4 5 6 7

6 7 0 1 2 4 5 0 0 0 0 0 0 0 0

0 1 2 3 4 5 6 7

2

4 5 6 7 0

1

0 0 0 0 0 0 0 0 0 0 0 0 0 0

2. And then, you have to be careful, how many cases are then discussed. The above, two cases, using 0 as search element



3.

`sorted[mid] > target != A[mid]`

Actually, you should add one more restriction, search half is sorted in ascending order.

because the half of 6 7 0 1,  $6 > 1$ , in the descending half, it must include going up and then going down. but,

in the ascending half, 1 2 4 5, just pure ascending. <- you can argue that, it is a fact!

1. <http://fisherlei.blogspot.ca/2013/01/leetcode-search-in-rotated-sorted-array.html>

2. <http://bangbingsyb.blogspot.ca/2014/11/leetcode-search-in-rotated-sorted-array.html>

---

### HackerRank: Two string - thinking in Java (2016-03-25 20:16)

March 25, 2016

Read other people's ideas. Understand other people by reading their code. Julia likes to read some Java programming language code for 1-2 hours, she came cross people's code, amazed by ideas from people working in Facebook, Amazon, and amazed that people have GOLD prize on HackerRank.

Julia likes to be able to write very readable, most understandable code with shortest time as possible.

problem statement:

[1]<https://www.hackerrank.com/challenges/two-strings>

Here are a few Java solution she likes:

#### Solution 1:

[2]<https://gist.github.com/jianminchen/6f029bb909ed7d85c012>

#### Solution 2:

source code reference:

[3]<https://www.hackerrank.com/winger>

People like to use bit operation, most likely they are expert and then have some Gold in HackerRank.

bit operation

[4]<https://gist.github.com/jianminchen/3d2acf64725cd73c79db>

use one integer to store 26 characters, 1 bit one char from a to z.

```
private static int f(String a) {
    int r = 0;
    for (char c : a.toCharArray()) {
        r |= 1 <{\textless}{\textless} (c - 'a');    //
    }
    return r;
}
```

Julia's comment: 1 left shift (c-'a') times

Solution 3:

[5]<https://gist.github.com/jianminchen/ff846e884ef9e9e2856a>

Solution 4:

It is interesting to read InputStream, need to warm up on Java class

[6]<https://gist.github.com/jianminchen/21ffc64093fd5952647c>

Solution 5:

use HashSet, Set, Character, String classes

[7]<https://gist.github.com/jianminchen/1eb02a940f7fa4b6a3f7>

1. <https://www.hackerrank.com/challenges/two-strings>
2. <https://gist.github.com/jianminchen/6f029bb909ed7d85c012>
3. <https://www.hackerrank.com/winger>
4. <https://gist.github.com/jianminchen/3d2acf64725cd73c79db>
5. <https://gist.github.com/jianminchen/ff846e884ef9e9e2856a>
6. <https://gist.github.com/jianminchen/21ffc64093fd5952647c>
7. <https://gist.github.com/jianminchen/1eb02a940f7fa4b6a3f7>

---

## HackerRank: String - Sherlock and anagrams (I) (2016-03-27 09:02)

March 27, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/sherlock-and-anagrams>

Difficulty: Moderate

This problem solving gets hot. Julia found something she struggled a lot. When Julia spent more than 2 hours on a problem in the Saturday evening, she knew that she is in trouble. She needs to be trained, and she needs a mentor.

Time Spent: March 26, 2016 Saturday evening 9:30 - 11:30

Sunday morning 9:00 - 12:00

Several mistakes to fix:

1. Julia, improve your analysis on test cases from HackerRank
2. Julia, understand Anagram requirement.
3. loop index issues

Julia's practice:

[2]<https://gist.github.com/jianminchen/dd2290f3a7b67e5168c9>

Go over test case again:

1. abba,

Let's say  $S[i, j]$  denotes the substring( $i, j-i+1$ )

$S[0,1] = "ab"$ ,

$S[0,2] = "abb"$ ,

$S[1,1] = "b"$

$S[4,4] = "b"$

$S[1,2] = "ab"$

$S[3,4] = "ba"$

For  $S = abba$ , anagrammatic pairs are:

$\{S[1,1], S[4,4]\}$ , //

$\{S[1,2], S[3,4]\}$ ,

$\{S[2,2], S[3,3]\}$ ,

$\{S[1,3], S[2,4]\}$

Notice that substring can be selected by first char of string, choice of  $n-m$ ,  $n$  is string length,  $m$  is substring length.

substrings can be overlapped, but still are anagrammatic pairs.

$S[1,3]$  and  $S[2,4]$  are overlapped, but are the anagrammatic pair.

Sample test case "abba", output should be 4, but Julia got 3. Spent time to fix index error.

2. sample case:

**ifailuhkqq**

should be: 3

Julia got 4

Actually, Julia, you should

**simplify**

**the test case first**

.

it is the same as ifailqq,

also it is the same as ifilqq

How many anagrammatic pairs in "ifilqq"

"i", "i" - S[1, 1] and S[3,3]

"if", "fi" - S[1, 2] and S[2,3] <- warm up anagram definition: same chars with same counts

"q", "q"

but 2 'i' char in the string "ifilq",

1 'i' char in the string "filqq",

Actually, the test case can be simplified using:

**"abacdd", the pairs are (a,a), (ab, ba), (d, d). That is very simple and understandable!**

**"abacd" and "bacdd" are not anagrams, because two 'a' in first string, but 1 a in second string .**

so two strings are

**not**

anagram.

Julia, what you spent time on:

1. List<int> constructor issue - 10 minutes, List<int>(i)
2. 10 minutes to figure out that you should work on anagrams strings
3. 10 minutes to write, 15 minutes to debug - Wrote a wrong anagram function

"ifilq" is not an anagram of "filqq", sample test case No. 2 should be 3

Julia is not very strong on testing software, she writes the software and puts it on. The software she writes can be improved tremendously, but she needs to find out what to improve.

To be continued.

1. <https://www.hackerrank.com/challenges/sherlock-and-anagrams>
2. <https://gist.github.com/jianminchen/dd2290f3a7b67e5168c9>

---

## **HackerRank: Sherlocks and Anagram (III) (2016-03-27 11:26)**

March 27, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/sherlock-and-anagrams>

Difficulty: Moderate

This problem solving gets hot. Julia found something she struggled a lot. When Julia spent more than 2 hours on a problem in the Saturday evening, she knew that she is in trouble. She needs to be trained, and she needs a mentor.

Solution to study:

[2]<https://gist.github.com/jianminchen/68453786a6ea03774a16>

Julia, you should warm up with C # **Dictionary<string, int>** , and also **StringBuilder** class, **AppendFormat** function.

Let us review how to design anagram key more efficient, more understandable way:

/\*

Think about how smart and easy it is to design this key for anagram string.

Precondition:

if two strings are anagram, the key should be same.

if two strings are not anagram, the key should be different.

Test case:

ab, the key is {0}-1 {1}-1

ba, the key is {0}-1 {1}-1,

but

bc, the key is {0}-0 {1}-1 {2}-1

compare to the design of anagram key using integer, this one is much easy to understand and follow.

[3]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-string-sherlock-and-anagrams.html>  
/

```
static string GiveKey(int[] arr){
    StringBuilder sb = new StringBuilder();
    for(int i = 0 ; i < 26 ; i++){
        sb.AppendFormat("{0}-",arr[i]);
    }
    return sb.ToString();
}
```

1. <https://www.hackerrank.com/challenges/sherlock-and-anagrams>

2. <https://gist.github.com/jianminchen/68453786a6ea03774a16>

3. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-string-sherlock-and-anagrams.html>

---

## HackerRank: Sherlock and anagrams (II) (2016-03-27 13:04)

March 27, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/sherlock-and-anagrams>

Difficulty: Moderate

This problem solving gets hot. Julia found something she struggled a lot. When Julia spent more than 2 hours on a problem in the Saturday evening, she knew that she is in trouble. She needs to be trained, and she needs a mentor.

Time Spent: March 26, 2016 Saturday evening 9:30 - 11:30

Sunday morning 9:00 - 12:00

Julia's practice:

[2]<https://gist.github.com/jianminchen/dd2290f3a7b67e5168c9>

Let us get ideas how other people solve the problems, study the code. Julia is training herself thinking in C # using HackerRank:

1. Use Dictionary<int, int>

code source is provided by a person 19 Gold, unbelievable smart and quick/ fast / great expressive code

[3]<https://www.hackerrank.com/azukun>

[4]<https://gist.github.com/jianminchen/90c0eec6d8c3c5cd7f18>

The anagram string function is composed to the design of key in the Dictionary.

/\*

precondition:

if two string are anagram, then key of these two string should be the same

"ab" and "ba" are the anagram, key should be the same

"ab" and "bc" are not the anagram, so key should not be the same.

**Julia's comment: 701 is confusing, why it has to be this big number?**

\*/

```
int Fun(string s, int l, int r)
{
    var ret = new int[26];
    for (int i = l; i <= r; i++)
        ret[s[i] - 'a']++;
    int x = 0; //
```

Julia's comment:

should be 1

286

```

        for (int i = 0; i < 26; i++)
            x = x * 701 + ret[i];
        return x;
    }

```

**Go over the detail to check:**

Key is designed using math formula polynomial expression:

string a -> key is integer: 0

string b -> key: 1

string ab -> key:  $x = 1$

$x = 1 * 701 + 1$

string ba -> 11 -> key:  $x = 1 * 701 + 1$

string bc -> 011 -> key:  $x = 0$ , count of a is 0

$x = 1$ , count of b is 1

$x = 1 * 701 + 1$

"ba" and "bc" are not anagram, so the key should be different: both are  $1 * 701 + 1$

string ad -> 1001 -> key  $x = 1$ , count of a is 1

$x = 701 + 0$ , count of b is 0

$x = 701 * 701 + 0$

key =  $701^3 + 1$

Julia found out that the idea can save a lot of time, she likes to work hard. She is lazy and likes to write less code.

Julia changed the key design, and ran the code in HackerRank, it also passed the test cases. In Julia's opinion, the code has a bug in theory but pass the HackerRank test; so, Julia fixed the code anyway. Just practice! It is not a

science of math, it is computer science.

[5]<https://gist.github.com/jianminchen/47f0265f49a2ddc62a8e>

Julia is still interested in writing loops, more expressive. Let us review how the code does:

```
public object Solve()
{
    for (int tt = ReadInt(); tt > 0; tt--)
// Julia's comment: put ReadInt() into a loop

    {
        string s = ReadToken();
        int n = s.Length;
        int ans = 0;
        var count = new Dictionary<int, int>();
        for (int i = 1; i < n; i++) // Julia's comment:
substring length from 1 to n-1,

            for (int j = 0; j + i <= n; j++) //
substring start position - j, end position: j+i-1, and check j+i <=n, easy to
reason - avoid bug

                {
                    var key = Fun(s, j, j + i - 1);
                    if (!count.ContainsKey(key))
                        count[key] = 0;
                    count[key]++;
                }
                foreach (var p in count)
                    ans += p.Value * (p.Value - 1) / 2;
                writer.WriteLine(ans);
            }

        return null;
    }
}
```

**One more step, improvement: Failed. Number from 701 to 26, it does not work. It depends on the length of string, which is <=100. Julia tried to figure out some math, algebra, but she is sure that the number should be coefficient, so,**

**at least >100.**

Julia, the key design for anagram string can be modified:

/\*



precondition:

if two string are anagram, then key of these two string should be the same

"ab" and "ba" are the anagram, key should be the same

"ab" and "bc" are not the anagram, so key should not be the same.

\*/

```
int keyForAnagramString(string s, int l, int r)
{
    var ret = new int[26];
    for (int i = l; i <= r; i++)
        ret[s[i] - 'a']++;
    int x = 1; //
    Julia's comment:
```

should be 1

```
        for (int i = 0; i < 26; i++)
            x = x * 26 + ret[i];
        return x;
    }
```

Julia spent 5 hacko to buy the test case input/ output

:

One more try:

Because the string length is  $\leq 100$ , so that coefficient is less than 100.

Key design can be changed to a small number 701 to

101

, it passes the HackerRank test:

```
/*
```

precondition:

if two string are anagram, then key of these two string should be the same

"ab" and "ba" are the anagram, key should be the same

"ab" and "bc" are not the anagram, so key should not be the same.

```
*/
```

```
int keyForAnagramString(string s, int l, int r)
{
    var ret = new int[26];
    for (int i = l; i <= r; i++)
        ret[s[i] - 'a']++;
    int x = 1;                //
```

Julia's comment:

should be 1

```
    for (int i = 0; i < 26; i++)
        x = x * 101 + ret[i];
    return x;
}
```

1. <https://www.hackerrank.com/challenges/sherlock-and-anagrams>

2. <https://gist.github.com/jianminchen/dd2290f3a7b67e5168c9>

3. <https://www.hackerrank.com/azukun>

4. <https://gist.github.com/jianminchen/c4d9186f45c47daf9a43>

5. <https://gist.github.com/jianminchen/47f0265f49a2ddc62a8e>

---

## HackerRank: Sherlock and Anagrams IV (2016-03-27 13:19)

March 27, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/sherlock-and-anagrams>

Difficulty: Moderate

More C # solution:

Solution 1:

Julia, here is code you should study; more advanced than yours.

a person works for Box Inc.

[2][https://www.hackerrank.com/\\_ \\_run](https://www.hackerrank.com/_ _run)

[3]<https://gist.github.com/jianminchen/576ecf2cd127a703cb7a>

Learn C # coding: readonly, Equals, override, constructor, use byte instead of int.

Solution 2:

use Dictionary class, string key for anagram string, use GetHashCode() call to turn key as Int.

[4]<https://gist.github.com/jianminchen/ffcca0582b5f0d1d6a9b>

Read about GetHashCode() webpage:

[5][https://msdn.microsoft.com/en-us/library/system.object.gethashcode\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.object.gethashcode(v=vs.110).aspx)

Solution 3:

[6]<https://gist.github.com/jianminchen/8f6bd4631f0b5f0bdee7>

Solution 4.

use Dictionary class, sort the key string, then anagram strings will be the same.

[7]<https://gist.github.com/jianminchen/59e326cbd1d8910c01c7>

solution 5:

Excellent code, written by a programmer in salesforce.com

[8][https://www.hackerrank.com/rest/contests/w13/challenges/sherlock-and-anagrams/hackers/rosharyg/download\\_solution](https://www.hackerrank.com/rest/contests/w13/challenges/sherlock-and-anagrams/hackers/rosharyg/download_solution)

Julia likes the code:

[9]<https://gist.github.com/jianminchen/d2ccf6532524d8751c73>

Solution 6: <- simple and quick, it can be written in less than 20 minutes. But not time efficient!  $O(n^2 * \text{string length})$

Use brute force, 3 loops, and then define anagramChecking function, just basic array, simple and quick.

[10]<https://gist.github.com/jianminchen/9f381875942d468ccb00>

1. <https://www.hackerrank.com/challenges/sherlock-and-anagrams>
  2. [https://www.hackerrank.com/\\_run](https://www.hackerrank.com/_run)
  3. <https://gist.github.com/jianminchen/576ecf2cd127a703cb7a>
  4. <https://gist.github.com/jianminchen/ffcca0582b5f0d1d6a9b>
  5. [https://msdn.microsoft.com/en-us/library/system.object.gethashcode\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.object.gethashcode(v=vs.110).aspx)
  6. <https://gist.github.com/jianminchen/8f6bd4631f0b5f0bdee7>
  7. <https://gist.github.com/jianminchen/59e326cbd1d8910c01c7>
  8. [https://www.hackerrank.com/rest/contests/w13/challenges/sherlock-and-anagrams/hackers/rosharyg/download\\_solution](https://www.hackerrank.com/rest/contests/w13/challenges/sherlock-and-anagrams/hackers/rosharyg/download_solution)
  9. <https://gist.github.com/jianminchen/d2ccf6532524d8751c73>
  10. <https://gist.github.com/jianminchen/9f381875942d468ccb00>
- 

## HackerRank: Sherlock and anagrams (V) (2016-03-27 14:52)

March 27, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/sherlock-and-anagrams>

Difficulty: Moderate

More C # solution:

Solution 1:

Julia, here is code you should study; more advanced than yours.

a person works for Box Inc.

[2][https://www.hackerrank.com/\\_run](https://www.hackerrank.com/_run)

[3]<https://gist.github.com/jianminchen/576ecf2cd127a703cb7a>

Learn C # coding: readonly, Equals, override, constructor, use byte instead of int. Take some time off, learn C #, OO design basics:

Here are the code, make some comments to read some articles to catch up:

```

using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace SherlockAndAnagrams
{
    class CharCount
    {
        protected bool Equals(CharCount other)
        {
            return Equals(Array, other.Array);
        }
        /*
        Design concern:
        hashCode for anagram strings - same
        use unchecked function
        figure out this design:
        a b c ... y z
        3 1 1 1
        3*(26*13)^25 1*(26*13)^24 1*(26*13) 1
        */
        public override int GetHashCode()
        {
            int hc = Array.Length;
            for (int i = 0; i < Array.Length; ++i)
            {
                hc = unchecked(hc * 13 + Array[i]); //
                Julia, figure out how this hashCode is working for anagram
            }
            return hc;
        }
        public readonly byte[] Array; //
        Julia, readonly, why to use byte[]

```

```

public CharCount()
{
    Array = new byte[26];
}

public CharCount(CharCount charCount)
{
    Array = new byte[26];
    for (int i = 0; i < 26; i++)
    {
        Array[i] = charCount.Array[i];
    }
}

public void AddChar(char ch) //
{
    Array[ch - 'a']++;
}

public override bool Equals(object obj) //
override Equals function
{
    CharCount other = obj as CharCount;
    if (obj == null)
    {
        return false;
    }
    for (int i = 0; i < 26; i++)
    {
        int val = Array[i].CompareTo(other.Array[i]); // byte.CompareTo
        if (val != 0)
        {
            return false;
        }
    }
    return true;
}
}

```

```

class Program
{
    static void Main(string[] args)
    {
        int t = int.Parse(Console.ReadLine());
        for (int i = 0; i < t; i++)
        {
            HandleTestCase();
        }
    }

    private static void HandleTestCase()
    {
        IDictionary<CharCount, int> dictionary = new Dictionary<CharCount, int>();
        string str = Console.ReadLine();
        for (int i = 0; i < str.Length; i++)
        {
            CharCount charCount = new CharCount();
            for (int j = i; j < str.Length; j++)
            {
                charCount.AddChar(str[j]);
                if (!dictionary.ContainsKey(charCount))
                {
                    dictionary.Add(new CharCount(charCount), 1);
                }
                else
                {
                    dictionary[charCount] = dictionary[charCount] + 1;
                }
            }
        }

        Console.WriteLine(dictionary.Values.Sum(value => ((value * (value - 1)) / 2)));
    }
}

```

1. <https://www.hackerrank.com/challenges/sherlock-and-anagrams>
2. [https://www.hackerrank.com/\\_\\_run](https://www.hackerrank.com/__run)
3. <https://gist.github.com/jianminchen/576ecf2cd127a703cb7a>

## HackerRank: Sherlock And Anagram (VI) (2016-03-27 16:21)

March 27, 2016

Julia was surprised to have a workout on this moderate difficult string problem from 9:00am - 4:00pm. She did some code study, and then, read so many codes from Microsoft, box, and saleforce, Amazon, and then, she read linkin profile, and blogs. She is getting connected to all other programmers in the world. HackerRank is young, all the coders are in charge of business this world, right now! No complaint.

Just learn one good code a time. Pay attention to some details. Follow up with a revisit once a while. Julia, you will make your programmer life easy, just relax, and see how people are creative to solve problems. You should do so, just copy the idea. Make sure that think by yourself first, do not be a copycat.

This code is her favorite. She is still learning, never use SortedDictionary before,

code reference:

[1]<https://www.hackerrank.com/Relentless>

[2]<http://anothercasualcoder.blogspot.ca/> #!

Julia is too busy to work on coding, so she chooses on easy to moderate questions. She waited until 7 - 10 string questions, and then, finally, she worked on her first moderate question on HackerRank after 1 - 2 months. But, she likes to read other people's code, and then, she just needs ideas to solve problems.

Here is the code gist:

[3]<https://gist.github.com/jianminchen/22f8e0de115cf656995e>

/\*

**Julia likes to talk about design of the function, through debugging, she knows a few things:**

**For string "abba",**

**First, go over string with length 1,**

**then, SortedDictionary - key 'a', 'b', values are 2, 2**

**then, Hashtable htPairs - key: a2, value:**

**then, go over string with length 2,**

**then, SortedDictionary - key 'a', value 1; key 'b', value '1'**

**"abba" go through a loop, to get substring with length 2, in the order:**

**^ ->**

**|**

**"ab",**

**"bb",**

**"ba"**

**1. string "ab", , key "a1b1", htPairs["a1b1"] = 1**



2. string "bb", key "b2", htPairs["b2"] = 1
  3. string "ba", hashTable contains the key, so the value htPairs["a1b1"] = 2
- Hashtable key "a1b1", sortedDictionary, value 2

```

/
static BigInteger UnorderedAnagrams(string str)
{
    Hashtable htPairs = new Hashtable();

    for (int len = 1; len <= str.Length; len++)
    {
        for (int i = 0; i + len <= str.Length; i++)
        {
            SortedDictionary<char, int> anagram = new SortedDictionary<char, int>();
            for (int j = i; j < i + len; j++)
            {
                if (anagram.ContainsKey(str[j]))
                {
                    anagram[str[j]] = (int)anagram[str[j]] + 1;
                }
                else
                {
                    anagram.Add(str[j], 1);
                }
            }

            string finalKey = "";
            foreach (char key in anagram.Keys)
            {
                finalKey += key.ToString() + ((int)anagram[key]).ToString();
            }

            if (!htPairs.ContainsKey(finalKey))
            {
                htPairs.Add(finalKey, 1);
            }
            else
            {
                htPairs[finalKey] = (int)htPairs[finalKey] + 1;
            }
        }
    }

    BigInteger finalResult = 0;
    foreach (string k in htPairs.Keys)
    {
        finalResult += Combinatorial((int)htPairs[k], 2);
    }

    return finalResult;
}

```

}

Blogs:

[4]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-string-sherlock-and-anagrams.html>

1. <https://www.hackerrank.com/Relentless>

2. <http://anothercasualcoder.blogspot.ca/#!>

3. <https://gist.github.com/jianminchen/22f8e0de115cf656995e>

4. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-string-sherlock-and-anagrams.html>

---

## Video watching: a guide to object-oriented practice (2016-03-27 21:25)

March 27, 2016

Spent 2 hours to watch the video:

[1][https://mva.microsoft.com/en-us/training-courses/a-guide-to-objectoriented-practices-14329?l=PLMOEi2hB\\_904668937](https://mva.microsoft.com/en-us/training-courses/a-guide-to-objectoriented-practices-14329?l=PLMOEi2hB_904668937)

Take some notes, and also write down tips helpful.

Look up those terms:

internal

readonly

expression-bodied member

SRP - Single Responsibility Principle

S.O.L.I.D. - OO principles

Read some blogs:

[2]<http://stackoverflow.com/questions/28411335/expression-bodied-function-members-efficiency-and-performance-in-c-sharp-6-0>

[3]<http://yqcmmd.com/2014/06/12/%E6%95%B0%E4%BD%8Ddp%E4%B8%93%E9%A2%98/>

1. [https://mva.microsoft.com/en-us/training-courses/a-guide-to-objectoriented-practices-14329?l=PLMOEi2hB\\_904668937](https://mva.microsoft.com/en-us/training-courses/a-guide-to-objectoriented-practices-14329?l=PLMOEi2hB_904668937)

2. <http://stackoverflow.com/questions/28411335/expression-bodied-function-members-efficiency-and-performance-in-c-sharp-6-0>

3. <http://yqcmmd.com/2014/06/12/%E6%95%B0%E4%BD%8Ddp%E4%B8%93%E9%A2%98/>

## HackerRank: String algorithm - Palindrome index (2016-03-30 23:08)

March 30, 2016

HackerRank, Easy questions, **1 hours 3 questions** . More practice, please! Get some momentum!

Things are looking for:

1. Tips to cut time to write code;
2. Make the problem a simple problem - read the hint !
3. Avoid writing too many lines code in less than 10 minutes.
4. More tips !!!

Problem statement:

[1]<https://www.hackerrank.com/challenges/palindrome-index>

Julia worked on this solution in 15 minutes, but she only scored 23 out of 25. She tried to figure out how to score 25. She missed the statement: " **There will always be a valid solution.** "

Her C # solution (Time out last test case): Time complexity  $O(N^2)$ , N is the length of string

[2]<https://gist.github.com/jianminchen/db16371dda11652a7dfee6d577f2221f>

So, Julia goes over those two cases in 25 minutes. It takes time to find great ideas:

### Solution 2:

use recursive function, while loop. Time complexity is  $O(N)$ .

[3]<https://gist.github.com/jianminchen/3817befd0be48a396c1998a776bf76b5>

### Solution 3 :

use iterative solution, once a possible index is found, stop. You do not need to verify that the string is palindrome. Assuming that there is a valid solution. Time complexity is  $O(N)$ .

[4]<https://gist.github.com/jianminchen/83d3d1a71ff80d6b209f0d6f9aaa435a>

### Solution 4:

[5]<https://gist.github.com/jianminchen/44ee14c6a45f25f94e11e892baad857c>

1. <https://www.hackerrank.com/challenges/palindrome-index>
  2. <https://gist.github.com/jianminchen/db16371dda11652a7dfee6d577f2221f>
  3. <https://gist.github.com/jianminchen/3817befd0be48a396c1998a776bf76b5>
  4. <https://gist.github.com/jianminchen/83d3d1a71ff80d6b209f0d6f9aaa435a>
  5. <https://gist.github.com/jianminchen/44ee14c6a45f25f94e11e892baad857c>
- 

## HackerRank: string algorithm - Reverse Shuffle Merge (I) - First step: Failed twice (2016-03-31 08:47)

March 31, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/reverse-shuffle-merge>

The algorithm is in advanced category. 8:17am - 10:08am. Document 2 hours work as a hacker, Julia. First time, Julia thinks that hacker is a good name!

Start from 8:19am, 9:18, tried twice, passed one test case, failed others with wrong answer;  
Then, need to pay attention to reverse string word.

Spent more than 20 minutes to think, but could not figure out the solution. Have to stop here. Write down the analysis first. Come back later.

The naive solution is to count string in each char in "abc...z", and then, half of count will construct a new string.

For example, if the string counting: a2b2c2

then, the string A is one of strings {abc, bac, cba, bac, aca, cab }

of course, "abc" is the smallest one in lexicographical order, but the possible string formats:

abc\*\*\*

abc\*\*

\*abc\*

\*\*abc

or, Go through each possible string - find the minimum one, smart way to compare with previous one, keep the smallest one. Assuming that the string is matching.

Got the idea. Linear solution. From 8:19am - 8:45am, more than 20 minutes to come out the idea.

9:00am -

Two functions - one function is to count the number; determine that count for each char is even;

Another step is to go over the string, take substring(i, 3).

20 minutes to write a code, a bug to fix: wrong answer

**Need to make sure that substring count matching count first, otherwise, skip it!**

**9:15am, bug is not fixed; and then, notice that the string has to be reversed!**

10:16am, still not fixed. Julia, this is a greedy algorithm, why is the greedy part? You missed merge part in the

construction merge(reverse(A), shuffle(A))

So, you have to redesign the algorithm.

Conclusion:

1. **Design algorithm - advanced - know why it is advanced, examine the idea and see if you can make it first; Otherwise, waste time to write code**

Two hours, failed two tries! A new hacker is getting her valuable lesson using 2 hours.

Here is the C # code:

[2]<https://gist.github.com/jianminchen/37e3c65f03f62dcdeb2e9634a11e0efa>

1. <https://www.hackerrank.com/challenges/reverse-shuffle-merge>

2. <https://gist.github.com/jianminchen/37e3c65f03f62dcdeb2e9634a11e0efa>

---

## 2.4 April

**HackerRank: string algorithm - Reverse Shuffle Merge (II) - next - forming a partial correct idea (2016-04-02 13:28)**

April 2, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/reverse-shuffle-merge>

First two hours work - failed twice:

[2]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-string-algorithm-reverse.html>

Julia likes to share her experience, advance level on HackerRank. It is not bad in the weekend, spend 2 hours messing around the ideas/ hackerRank, come out a clear greedy algorithm. The hackerRank definitely helps Julia to shape her idea from start to end.

Here is the idea after 2 hours intensive work:

of course, "abc" is the smallest one in lexicographical order, but the possible string formats:

**\*a\*b\*c\*, not \*abc\*, now \* means any number of any chars .**

Let us count how many of a, b, c can be skipped when we do linear scan of a string.

Now, it should be very easily to introduce greedy algorithm.

For example,

if linear scan from left to right, visiting char b, then, we have to check how many b's left can be skipped, if it is bigger than 0, we have to check any char before b has anything left or not. For example, if a still has some number left to skip; we hold on b, just skip current b.

Otherwise, count current b into the string we are looking for, and decrease the number count of b (recording how many b can be skipped).

an example to explain:

a2b2c2 case,

ba\*, the half should be a1b1c1,

so, skip first char 'b', since greedy algorithm / let 'a' go first.

Give it a try, implement the idea:

**Some statistics: advanced algorithm 4+ hours - A mountain - "Sea to sky Gondola" to climb**

Spent 9:47am - 11:47am, wrote code, but still failed most of case; only pass "eggegg";

Hard to concentrate, and think about the issue - this reverse is tricky!

Here is the C# solution - 3rd failed try:

[3]<https://gist.github.com/jianminchen/07fe563e1d8e65dc3361d8d30eefe347>

baby hacker is crying, still score zero: 4 hours work

[4]<https://goo.gl/mLpzjb>

Now, it is 1:47 pm, another 2 hours with music, the code was submitted, now score **16.67/ 50** . Still need to work on more before Julia plan to read other people's solution:

**Some statistics: advanced algorithm 6+ hours - A mountain - Sea to sky Gondola to climb**

[5]<https://gist.github.com/jianminchen/5e3002256642454eae747da2bde26ab3>

Need to stop, go to enjoy outdoor activities!

baby hacker is showing off her baby steps:

[6]<https://goo.gl/EMRHW5>

Try to fix the error, give up - the design has another flaw,

test case:

djjcddjggbiigjfhghehhbgdigjicafgicehfhgfigadihiajgciagicdahcbajjbhifjiaajigdg dfhdiiijgaiejgegbbiigida

i=50, s[i] = 'g', the design let 'g' skip, but

i=52, s[i] = 'i', 'i' has to be added to the output, no more skip.

<- Julia, think about stack, use some data structure to do reverse work <- **such a great workout!** Release all my stress and headache, and be humble!

5:12pm, statistics: Another 3 hours, total: **9+ hours**

1. <https://www.hackerrank.com/challenges/reverse-shuffle-merge>
  2. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-string-algorithm-reverse.html>
  3. <https://gist.github.com/jianminchen/07fe563e1d8e65dc3361d8d30eefe347>
  4. <https://goo.gl/mLpzjb>
  5. <https://gist.github.com/jianminchen/5e3002256642454eae747da2bde26ab3>
  6. <https://goo.gl/EMRHW5>
- 

## **Practice difference: How top sports player practice? (2016-04-02 17:45)**

April, 2015

The best way for a software developer to take a break, is to study something else; let me propose an idea for myself today, and see if I can make a good shot or not.

See how top tennis player Male/ Female structure their practice? :-) Just kidding, I can only observe a few things as a player over 300, maybe, 500 hours.

Julia likes to study tennis teaching video, she learns a lot how to make a talk 2-3 minutes on a specific topic through coaching video.

### **Eugenie Bouchard - Fear Less**

[1]<https://www.youtube.com/watch?v=4MVb3SR2yNA>

[2]<https://www.youtube.com/watch?v=5YWdkxMQA6s>

### **Li Na**

[3]<https://www.youtube.com/watch?v=Qewi-SB5hWU>

Novak Djokovic

[4]<https://www.youtube.com/watch?v=HuWL65vrTJU>

### **Murray 's practice and his coach -**

[5]<https://www.youtube.com/watch?v=YtvQRq0FvNI>

[6][https://www.youtube.com/watch?v=zR0U7wl213I&ebc=ANyPxKpEc8T3E79\\_tAtkvdPDj8KjfbAmn05rGWxNHe3BI7RzXCazHMMmRjUmTdSxsAgyZ60CP967G21vbREU5Pd3eFahuw](https://www.youtube.com/watch?v=zR0U7wl213I&ebc=ANyPxKpEc8T3E79_tAtkvdPDj8KjfbAmn05rGWxNHe3BI7RzXCazHMMmRjUmTdSxsAgyZ60CP967G21vbREU5Pd3eFahuw)

**Overhead** , and then, from **forehand** -> **anywhere** , **receiving**, and then, **serving**.

Just watch the hitting partner - the helper, how he can follow the instruction, last minute, when Novak calls wide, how the helper can place the tennis ball wide for him to receive.

Julia, you pay more attention to how they are focusing on training, and other technical things. Julia likes Novak using hand language to communicate to switch courts.

[7]<https://www.youtube.com/watch?v=ausQH8y3nQY>

Hitting partner told Novak hitting outside - long just one inch.

Drills - Julia's favorite drills - double alley drill

#### **2014 USPTA Tennis Teacher's Conference in New York City**

[8]<https://goo.gl/bztNNE>

Julia likes to organize some activity with sports to encourage people moving/ exercise, a lot of ideas for people in office, church etc. Julia loves the teaching how to do things on the court.

[9]<https://goo.gl/KD8RHa>

Great teaching - medicine ball usage, foot work, forward to the ball.

[10]<https://www.youtube.com/watch?v=zpldQqYz2RE>

plan to watch:

Mental skills for your players

[11]<https://www.youtube.com/watch?v=E8-7LA9-UOY>

A few verses to memorize, every top tennis player loses, but still continues:





Andy Murray   
@andy\_murray

 Follow

# WHY DO I SUCCEED?

I AM WILLING TO DO THE THINGS YOU ARE NOT

I WILL FIGHT AGAINST THE ODDS

I WILL SACRIFICE

I AM NOT SHACKLED BY FEAR, INSECURITY OR DOUBT

I AM MOTIVATED BY ACCOMPLISHMENT, NOT PRIDE

IF I FALL -- I WILL GET UP

IF I AM BEATEN -- I WILL RETURN

I WILL NEVER STOP GETTING BETTER

I WILL NEVER GIVE UP -- EVER

# THAT IS WHY I SUCCEED

Genie Bouchard 2014 Montage - Good music always is welcomed! "In the Air Tonight" I want to be next version of me, not other person. What a good statement!

[12]<https://www.youtube.com/watch?v=JZJ8K5857dk>

Lyrics:

[13]<http://www.azlyrics.com/lyrics/philcollins/intheairtonight.html>

1. <https://www.youtube.com/watch?v=4MVb3SR2yNA>

2. <https://www.youtube.com/watch?v=5YWdkxMQA6s>

3. <https://www.youtube.com/watch?v=Qewi-SB5hWU>

4. <https://www.youtube.com/watch?v=HuWL65vrTJU>

5. <https://www.youtube.com/watch?v=YtvQRq0FvNI>

6. [https://www.youtube.com/watch?v=zR0U7w1213I&ebc=ANyPxKpEc8T3E79\\_tAtkvdpDj8KjfbAmn05rGWxNHe3BI7RzXCazB24KHMmRjUmTdSxsAgyZ60CP967G21vbREU5Pd3eFahuw](https://www.youtube.com/watch?v=zR0U7w1213I&ebc=ANyPxKpEc8T3E79_tAtkvdpDj8KjfbAmn05rGWxNHe3BI7RzXCazB24KHMmRjUmTdSxsAgyZ60CP967G21vbREU5Pd3eFahuw)

7. <https://www.youtube.com/watch?v=ausQH8y3nQY>
  8. <https://goo.gl/bztNNE>
  9. <https://goo.gl/KD8RHa>
  10. <https://www.youtube.com/watch?v=zpldQqYz2RE>
  11. <https://www.youtube.com/watch?v=E8-7LA9-U0Y>
  12. <https://www.youtube.com/watch?v=JZJ8K5857dk>
  13. <http://www.azlyrics.com/lyrics/philcollins/intheairtonight.html>
- 

## 10-000 hour rule (2016-04-03 08:52)

April 3, 2016

Share this article:

[1]<http://goo.gl/jnQ3qf>

10,000 hour rules - it is encouraging, keep going; learn to be mental strong; Do not get nervous when you are in trial.

To entertain this 10,000 hour rule (better grammer: "10,000-hour rule" ), Julia make a rule for herself, as a software programmer: called **a new explainable variable or function rule** - when you want to use Ctrl+C; or short name: **Ctrl+C rule** .

This rule is for fast coding, people want to be a hacker. Let hand typing catches your mind quickly.

1. <http://goo.gl/jnQ3qf>
- 

## HackerRank: string algorithm - Reverse Shuffle Merge (III) (2016-04-03 16:06)

April 3, 2016

Study the code written by some one in Google, need to get the idea how it is designed. It should not be difficult, since I already worked on this problem over 10 hours.

[1]<https://goo.gl/Rbh3B4>

up to April 3, 2016

**statistics:** 1300 score 50.00

Julia score rank: 1403

C # rank: 58

1. <https://goo.gl/Rbh3B4>
-

## HackerRank: string algorithm - Reverse Shuffle Merge (IV) (2016-04-03 16:58)

April 3, 2016

C # solution:

[1]<https://gist.github.com/jianminchen/346970ad058d64041ab7b8cbe9ecb495>

Read the code, and talk about the ideas:

Best way to explain the algorithm - greedy one, the following, is to take away reverse, and also merge two parts (these are noisy!), work on a string, simple test case, how to construct a string with minimum lexically value, using stack to do greedy design, and also allow the back and forth two ways to optimize.

Here is the simple example:

1. string a2b2c2, for example, aabbcc, so abc is shortest one.

For string b b c c a a,

first b, put in stack, second b, have to skip; and then, c is coming,

'c' > 'b', push c to stack, so stack is like this

b

c

and then, second c, skip it; now 'a' is coming,

'a' is bigger than 'b', but b does have any number to skip; need to put 'a' in anyhow.

the output will be "bca";

2. Try another one:

ba

b

c ac

First one, b, go into the stack:

b,

meet a, and then, 'a' < 'b', because b still have one to skip, so pop out b, skip the b, push 'a' in the stack;

Now, another b is coming, we have to hold one b in output, push it in the stack:

b

a

And then, c is coming, push it in c.

c

b

a

Now, a is coming, skip it; c is coming, also skip it.

the result is 'abc', smallest one in lexical sense.

3. Try third one:

work on first test case, make small change. move bbccaa, one of b, so the string is **ccbaab** ,

c - put in the stack

c - skip it.  
b - compare to stack top, b ( b has one left to skip);  
so b into stack  
b  
c

c- second c, skip it  
a- compare to the stack top: b  
'a' < 'b', still 'b' can skip one, one more is coming  
pop up b, push a in,  
a  
c  
and then, last char 'b' into stack  
b  
a  
c  
the output is "cab"

The question is to ask yourself, do you have genius to design it at the beginning, called thinking about **in stack to cover mistake in greedy approach** ! Julia has no clue right now, she spent 9 hours to take this lesson. **Backtracking** is better term for using stack in the design.

April 5, 2016

Go over a2b2c2 case, the lexical smallest substring (Julia's first practice):

bccaba,

visit first 'b', skip 'b', because a has 2 of them to visit

visit first 'c', skip 'c',

visit second 'c', cannot skip, has to take it. Now, best one is starting from c, of course it is worse than the one starting from 'b', such as "bca".

So, greedy algorithm is to take what ever you can, and then, backtracking, make it smaller iteratively.

-

code source: a box employee

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ReverseShuffleMerge
{
    class Program
    {
        static void Main(string[] args)
        {
            string input = Console.ReadLine();
```

```

int[] skip = new int[26];
int[] add = new int[26];

for (int i = 0; i < input.Length; i++)
{
    skip[input[i] - 'a']++;
}

for (int i = 0; i < 26; i++)
{
    skip[i] /= 2;
    add[i] = skip[i];
}

Stack<char> stack = new Stack<char>();
for (int i = input.Length - 1; i >= 0; i--)
{
    char ch = input[i];
    while(stack.Count > 0 && add[ch - 'a'] > 0 && stack.Peek() > ch
&& skip[stack.Peek() - 'a'] > 0)
    {
        char chTop = stack.Pop();
        add[chTop - 'a']++;
        skip[chTop - 'a']--;
    }

    if (add[ch - 'a'] > 0)
    {
        stack.Push(ch);
        add[ch - 'a']--;
    }
    else
    {
        skip[ch - 'a']--;
    }
}

string output = string.Empty;
while(stack.Count > 0)
{
    output = stack.Pop() + output;
}

Console.WriteLine(output);
}
}
}

```

1. <https://gist.github.com/jianminchen/346970ad058d64041ab7b8cbe9ecb495>

## Algorithm, Rotate matrix nxm clockwise 90 degree (2016-04-05 17:58)

April 5, 2016

Write an algorithm to rotate matrix nxm clockwise 90 degree.

Use no compiler, no Ctrl+C, and then, time complexity analysis:

Here is the code Julia wrote, she thought about the swapping strategies:

1. swap multiple times
2. leave four corners as is, swap four corners separately.

[1]<https://gist.github.com/jianminchen/e489cf7a490ac426eb50b4300890387d>

/\*

April 5, 2016

NO ctrl+C copy

matrix nxm rotate clockwise 90 degree

do it in place

/

```
public static bool rotateInplace90ClockWise(int[][] arr)
{
    if (arr == null || arr.Length == 0 || arr[0].Length == 0) return false;
    int n = arr.Length; // row
    int m = arr[0].Length; // column
```

```
    if (n != m) return false;
```

```
    // we need to start loops
```

```
    int start = 0;
    int end = n - 1;
```

```
    while (start < end && start < n / 2 && end > n / 2)
    {
```

```
        // top with right swap - left the terminal point untouched
```

```
        // left to right <- row
```

```
        // top to down <- column
```

```
        for (int i = start + 1; i < end - 1; i++)
```

```
        {
            int currx = start;
            int curry = i;
```

```
            int currx_col = i;
            int curry_col = end;
```

310

```

swap(arr, currx, curry, currx_col, curry_col);
}

// now, top down, we need to cross swap

for (int i = start + 1; i < end - 1; i++)
{
    int currx = start;
    int curry = i;

    int currx_down = end;
    int curry_down = end - i;

    swap(arr, currx, curry, currx_down, curry_down);
}

// left and top swap
for (int i = start + 1; i < end - 1; i++)
{
    // left part
    int currx = end - 1 - i;
    int curry = start;

    int currx_top = start;
    int curry_top = i;

    swap(arr, currx, curry, currx_top, curry_top);
}

// and then, rotate four corners
// 1, 2
// 4 3
swap(arr, start, start, start, end);
swap(arr, end, end, start, start);
swap(arr, start, start, end, start);

start++;
end--;
}
return true;
}

private static void swap(int[][] arr, int x, int y, int x2, int y2)
{
    int tmp = arr[x][y];
    arr[x][y] = arr[x2][y2];
    arr[x2][y2] = tmp;
}

```

[2]<https://gist.github.com/jianminchen/e489cf7a490ac426eb50b4300890387d>

time complexity: since every node in the matrix at most swaps twice, so the count of swap is  $O(n^2)$

1. <https://gist.github.com/jianminchen/e489cf7a490ac426eb50b4300890387d>

2. <https://gist.github.com/jianminchen/e489cf7a490ac426eb50b4300890387d>

---

## HackerRank articles - reading time (2016-04-05 21:35)

April 5, 2016

Julia's favorite articles on HackerRank:

Recently, Julia starts to read articles on HackerRank, and she likes to take some notes:

### 1. 8 ways to reduce bugs while coding

[1]<http://blog.hackerrank.com/8-ways-to-reduce-bugs-while-coding/>

Tools: Findbugs - Java code, GetExceptional - Ruby apps, Selenium - web browsers

unit tests & integration test / use Tools / Compiler warnings / Code Review / Logs / Use existing libraries / Pseudo code / Avoid distractions

Julia's thoughts:

1. Write a small function
2. Check variable scope - limit scope to minimum
3. Design concern - make it easy, so every one can tell there is no problem.
4. Push the coding problems to compile time not running time
5. Document the test case above function, precondition/ post condition
6. Have a lot of ideas to solve one problem, then, choose any one, you can write code right away
7. Think about maintenance, use data structure, such as Array, go through loops - use iteration

Recently, in March 2016, Julia uses a loop in JavaScript to add onclick() event - callback function; and then, finds out the variable i - iterated value is always the last one, not like C #, and then, she figured out using closure concept; she started to write her first closure variable for real work. She got so excited. You can avoid using closure to finish a project, but when the code is reviewed the expert will know your strength - you never develop the muscle! Your code is like a fat, not a muscle.

Push yourself to make the code most efficient mode first, and then, stay out of your comfortable zone; learn to be a good critic.

### 2. Great article about learning

[2]<http://blog.hackerrank.com/establish-yourself-expert/>

10 % reading -> 20 % Audiovisual -> 30 % Demonstration -> 50 % discussion -> 75 % Practice doing-> 90 % Teach others

maximum learning happens when to "Teach others"



Julia's thoughts:

In 2015, Julia spent days/ weeks/ months on JavaScript learning, books reading, code snippets practice, video watching; But, in 2016, when she reads so many JavaScript submissions on HackerRank, she suddenly noticed that people are so good at JavaScript language, use it to solve daily simple algorithm problem, maybe as a hobby. She knows that it is also urgent to catch up JavaScript.

3.

[3]<http://blog.hackerrank.com/programming-interviews-techniques-tips-by-gayle-laakmann/>

1. Simplify the question
2. Create assumptions
3. Solve for simplified case
4. Finally, generalize your answer

**Smart, write good code, who care about writing clean code**

#### 4. Promoting Coding Culture in you campus

[4]<http://blog.hackerrank.com/promote-coding-culture-campus/>

self learning - MOCCs like Udacity, Coursera, edX  
Inculcate the habit of reading

#### 5. A stronger Programming Culture for India

[5]<http://blog.hackerrank.com/stronger-programming-culture-india/>

HackerRank clubs

Self learning attitude

Rote-based education system has been detrimental

Facts: IT graduates, only 25 % are employable (as per Nasscom)

6. [6]<http://blog.hackerrank.com/how-to-crack-microsoft-interview/>

Arrive at the answer ( **not just** about arriving an answer)

How you approach the problem ( **a lot** about how you approach the problem)

Your clarity of thought

How you backtrack when you are stuck

How you use the hints given to you.

Julia's thoughts:

Do some self-evaluation:

1. Code skills - do you write a lot, read a lot of code recently? Warm up definitely helps.
2. Do you like to solve the problem or not?
3. Do you have some rituals to help you solve the problem?

For example, if you are so nervous, cannot think/ write as normal, what do you do?

**Julia's answer** : Find a simple test case, work out the design/ coding first.

4. Can you tell the problem difficulty level? Easy, medium, advanced or difficult?
5. What kind of person you are in programming style? Are you a hacker?
6. How often do you practice problem solving?
7. What kind of tools you are using? Do you use HackerRank?
8. How many peers you have - mock interview?
9. How often can you get some code screening? code assessment?
10. Do you stay on the current moment or think about consequences? mental skills.

## 7. Productivity tips for programmers

[7]<http://blog.hackerrank.com/productivity-tips-for-programmers/>

1. write Unit test for higher productivity.
2. Practice your programming.
3. Use libraries and help improve them.
4. Read code and technical stuff.

Julia's thoughts:

1. Julia did not find things exciting to do after work from 2010 - 2014, she found out that she does not develop strong interest, long term effort on somethings except tennis sports. So, she started to make change; Now, she likes to solve coding problems, and also she reduces a lot of stress as a developer. She knows a few more way to make the apps run good, easy to maintain.

1. <http://blog.hackerrank.com/8-ways-to-reduce-bugs-while-coding/>
  2. <http://blog.hackerrank.com/establish-yourself-expert/>
  3. <http://blog.hackerrank.com/programming-interviews-techniques-tips-by-gayle-laakmann/>
  4. <http://blog.hackerrank.com/promote-coding-culture-campus/>
  5. <http://blog.hackerrank.com/stronger-programming-culture-india/>
  6. <http://blog.hackerrank.com/how-to-crack-microsoft-interview/>
  7. <http://blog.hackerrank.com/productivity-tips-for-programmers/>
- 

## Distributed System - study time (2016-04-07 21:49)

April 7, 2016

Plan to study "Distributed System", get updated on this area.

[1]<http://goo.gl/5BX1ND>

Take some notes.

And read an article:

[2]<https://www.cs.cmu.edu/~dga/15-440/F12/p3.pdf>

Action item: next week

1. Spend 2+ hours to go over the lecture notes
2. Get rough ideas what are the big concerns in distributed system. (2+ hours)

1. <http://goo.gl/5BX1ND>
  2. <https://www.cs.cmu.edu/~dga/15-440/F12/p3.pdf>
-

## JavaScript - slide show case studies (2016-04-07 21:58)

April 7, 2016

Work on JavaScript, Css, html on picture slideshow.

Here is the list of sample code Julia chose to study.

1.

[1]<http://robertnyman.com/picture-slides/>

### **Mental skills tip:**

Julia read google employer's code about slideshow, how the code is structured and commented, and then, she found her issue.

Learning JavaScript, first thing, is not afraid to break things, learn the grammar first; Fix the style. She spent 3 hours to fix the style problem of her slideshow JavaScript legacy code. Make sure that the style is readable.

For example:

```
return a=1, b=1, c=1, d=1;
```

### **readable style:**

```
return
```

```
a=1,
```

```
b=1,
```

```
c=1,
```

```
d=1;
```

2. page slide show

[2]<http://pgwjs.com/pgwslider/>

[3]<https://coderwall.com/guangyi/protips#protips>

[4]<http://cssdeck.com/labs/imageslider-pager>

[5]<https://github.com/Pagawa/PgwSlider>

3. Demo -

[6]<http://devowhippit.github.io/jquery.sldr/>

source code:

[7]<https://github.com/devowhippit/jquery.sldr>

Tips learned on April 8, 2016:

about comma in the JavaScript code: return statement:

[8]<http://stackoverflow.com/questions/418799/what-does-colon-do-in-javascript>

understand return statement with multiple , commas

[9]<http://stackoverflow.com/questions/10284536/return-statement-with-multiple-comma-separated-values>

4. read more about JavaScript

[10]<http://www.adequatelygood.com/Writing-Testable-JavaScript.html>

[11]<http://speakingjs.com/>

[12]<http://www.itsalif.info/content/good-javascript-practices>

#### Action item:

Share the slideshow JavaScript as well. Make it more readable, using meaningful variables.

[13]

1. <http://robertnyman.com/picture-slides/>
  2. <http://pgwjs.com/pgwslider/>
  3. <https://coderwall.com/guangyi/protips#protips>
  4. <http://cssdeck.com/labs/imageslider-pager>
  5. <https://github.com/Pagawa/PgwSlider>
  6. <http://devowhippit.github.io/jquery.sldr/>
  7. <https://github.com/devowhippit/jquery.sldr>
  8. <http://stackoverflow.com/questions/418799/what-does-colon-do-in-javascript>
  9. <http://stackoverflow.com/questions/10284536/return-statement-with-multiple-comma-seperated-values>
  10. <http://www.adequatelygood.com/Writing-Testable-JavaScript.html>
  11. <http://speakingjs.com/>
  12. <http://www.itsalif.info/content/good-javascript-practices>
  13. <http://robertnyman.com/picture-slides/>
- 

#### Article reading: Microsoft Interview Process (2016-04-08 22:42)

April 8, 2016

Doing some research. It is fun to write down what you read, and then, come back later to review again and again.

Take some notes from the following articles:

Land that interview

[1]<http://landthatinterview.com/microsoft/the-microsoft-interview-process/>

Facts:

1. Microsoft is really structured like **many small** companies internally, each with their own recruiting teams.
2. a screening phone interview, also show that you are better than 90 % of the hundreds of others

#### Advice for phone screen :

1. Talk talk talk in the technical phone interview, keep it on topic and don't try to steer the conversation, verbalize what you are thinking.
2. Second, have fun!
3. Ask a lot of questions and don't make assumptions. If you make assumptions, explicitly state that you are doing so and get the interview to buy in to those assumptions.
4. The hiring manager likes to know that the person will be engaged and productive.

#### Advice for interview in person :

1. 1 in 10 for a particular position -
2. proactively engage the recruiter

3. business casual attire, not a 3-piece (awkward)
4. first 3 interviews. May be interviewed by a developer, a test manager, a program manager, and the hiring manager. Finish before lunch time.
5. the as-appropriate interview - last regular interview will be over at round 4:00 in the afternoon.

Julia's comment:

Julia likes to do some research on Microsoft interview process. Now, she knows that Microsoft does not practice interview as Google does, using hiring committee.

1. <http://landthatinterview.com/microsoft/the-microsoft-interview-process/>

---

### **Article reading: How to get hired at Microsoft (2016-04-08 23:08)**

April 8, 2016

Doing some research on high-tech companies interview process.

[1]<http://landthatinterview.com/microsoft/how-to-get-hired-at-microsoft/>

one of the top 100 places to work across all industries.

Understanding how to navigate the screening and hiring process is also important.

1. Apply for jobs that you are qualified for
2. Don't be intimidated by job listings  
hired for long-term potential

#### **3. The inside angle**

**Microsoft gives a lot of weight to references from current employees.  
not single monoculture, more like a conglomeration of many smaller companies.**

#### **4. Apply for multiple jobs, but keep it targeted.**

Julia's thoughts:

1. Need to study C++
2. Need to take some crash courses for distributed system, and other areas.
3. Every month spend 5-10 hours to cover some topics - study and get educated.
5. Be persistent

#### **6. Make a good first impression**

**It's not just about you - team effort, not lone cowboy coder.**

**It is about you - be specific about contributions you have personally made toward the success of previous projects.**

## 7. Understand the interview process

1. <http://landthatinterview.com/microsoft/how-to-get-hired-at-microsoft/>

---

### Article reading: Get that coveted tech interview (2016-04-08 23:19)

April 8, 2016

[1]<http://utsavized.com/getting-that-coveted-tech-interview/>

#### Revamp your resume

organize, and make it simple and clear.

list your accomplishments in brief and succinct sentences. Include facts and numbers wherever applicable.

#### Make sure you have the right things on your resume

#### List out your personal projects

If you are prepared to devote a significant amount of hours each day, sacrifice your weekends and hangouts, to almost redo your entire Bachelor's degree all by yourself, then that attitude and perseverance will undeniably get you through. And at the end, even if you don't land the job, trust me, you will become a much more competent engineer than you are right now.

Julia's thought: **It is true, practice makes difference .**

More reading:

[2]<https://www.quora.com/profile/Mohsin-Ali-19>

[3][https://drive.google.com/file/d/0B9lkfT8\\_oB3fdTRXUS1xZzBZT2s/view](https://drive.google.com/file/d/0B9lkfT8_oB3fdTRXUS1xZzBZT2s/view)

1. <http://utsavized.com/getting-that-coveted-tech-interview/>

2. <https://www.quora.com/profile/Mohsin-Ali-19>

3. [https://drive.google.com/file/d/0B9lkfT8\\_oB3fdTRXUS1xZzBZT2s/view](https://drive.google.com/file/d/0B9lkfT8_oB3fdTRXUS1xZzBZT2s/view)

---

### Elizabeth Holmes's Top 10 rules for success (2016-04-09 11:05)

April 9, 2016

Watch the video:

[1]<https://www.youtube.com/watch?v=iwKs-eoPM-Y>

Take some notes:

1. Find your way

Love things you are doing; you will get back up after you are knocked down.

Talk about concept about **10,000 rule** , you go so deep about something; you cannot be expert on it; when you love it so much, then you do not let it go. That is winning about.

In other words, fail over and over again until you succeed.

2. Focus on your mission

3. Build people

4. #Believe

5. Don't have a backup plan

6. **Embrace failure**

Open to failure, fail over a thousand times to get it work at 1001th time. Determination to make it work.

7. Surround yourself with smart people

8. Don't do it for the money

9. **Weather the storm**

Big waves come and go, you are there for the reason.

10. Make a difference

June 1, 2016

Billionaire to nothing.

[2][http://time.com/money/4353658/elizabeth-holmes-theranos-broke/?xid=frommoney\\_soc\\_socialflow\\_facebook\\_money](http://time.com/money/4353658/elizabeth-holmes-theranos-broke/?xid=frommoney_soc_socialflow_facebook_money)

1. <https://www.youtube.com/watch?v=iwKs-eoPM-Y>

2. [http://time.com/money/4353658/elizabeth-holmes-theranos-broke/?xid=frommoney\\_soc\\_socialflow\\_facebook\\_money](http://time.com/money/4353658/elizabeth-holmes-theranos-broke/?xid=frommoney_soc_socialflow_facebook_money)

---

## HackerRank - String algorithm - Count Strings (I) (2016-04-09 11:15)

April 9, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/count-strings>

Category: Difficult problem

Spend 25 minutes to think about problem, and then, will get into other people's submission, and see what should be learned.

10:30am - 10:55am

Spent 25 minutes to think about the problem.

Try to express the expression using binary tree, like  $1+2*3$  problem using a tree.

So, ab can be express as a tree:

operator .

/\

a b

a|b

operator |

/\

a b

a\*

put \* as parent node, and a is left child of operator node \*

(a\*)5

put 5 as the parent node of operator \*

Structure the input as tree structure, and then, define different class - concatOp, orOp, starOp

Read the expression from left to right, and then, put it into data structure like a stack, and then, parse it, store it in the binary tree.

**April 10, 2016 2:32pm**

This is a graph problem, and also, the NFA, DFA problem. Julia, you are lucky to review the problem. Spend some time in the short future to go over code, give yourself a graph coding experience.

1. <https://www.hackerrank.com/challenges/count-strings>

---

**HackerRank: Count strings (II) (2016-04-09 11:41)**

April 9, 2016

Julia likes to have some adventure and just get into other people's solution, and quickly learn something in next 20 - 30 minutes.

Problem statement:

320



[1]<https://www.hackerrank.com/challenges/count-strings>

**Statistics** : time spent: 11:22am - 1:00pm

Here is the solution from a programmer working in Apple:

[2]<https://gist.github.com/jianminchen/0e1fe57d9f6763d91b640c2603c73a90>

Comments after reading:

1. This problem is about DFA, NDFA, those content learning in Formal Language, how to build a compiler. <- Julia, you learned formal language, and also learned twice ( second time for Ph.D. qualification in 2001 in FAU).
2. The string parsing is more complicate, '(', ') ' should be treated as operator, and then, you can build up something.
3. Read the wiki webpage to quickly refresh knowledge  
[3][https://en.wikipedia.org/wiki/Deterministic\\_finite\\_automaton](https://en.wikipedia.org/wiki/Deterministic_finite_automaton)
4. Play with Visual studio and also the code - debug and run through the test case 20-30 minutes:  
(ab)|(ba) 2  
((a|b)\*) 5
5. And put some diagram on the paper, take picture, post here and see if you can draw the state diagram for this machine. ( 20 - 30 minutes)

Julia likes to read the code, and also debug the code;

#### **Some facts:**

1. she learned Formal language in 2001 to prepare Ph.D. qualification exam, but she never did spend time to work on an concrete example, debug the code to see how it is designed.
2. the code is well designed, and also beautiful code to read;

It is hard to figure out how to construct the Ndfa, but reading the code, Julia, you should figure out the design:

/\*

**Let us work out this basic test case:**

**((ab)|(ba)) 2**

**First char, it is '(',**

**get into the switch statement for case '(',**

**and then, recursive call to get the first FiniteAutomata,**

**another '(', two times recursive calls, and then, 'a', 'b'**

**first FiniteAutomate:**

**{START: '\*' -> #2;FINISH: #2: 'a' -> #3; #3: '\*' -> #4; #4: 'b' -> #5; #5: '\*' -> FINISH; }**

**((ab) -> go to operator | ,  
first, get the second operand, and then, complete the task: CombineAsOrAndFree**

```
/
private static FiniteAutomata ParsePatternIntoNdfa(StringBuilder builder, int start, out int end)
{
    Debug.Assert(builder.Length > start);

    end = start;

    char ch = builder[end++];
    switch (ch)
    {
        case 'a':
        case 'b':
            return FiniteAutomata.CreateForSymbol(ch);

        case '(':
            FiniteAutomata first = new FiniteAutomata();
            FiniteAutomata second = new FiniteAutomata();
            FiniteAutomata result = new FiniteAutomata();

            first = ParsePatternIntoNdfa(builder, end, out end);

            ch = builder[end++];
            switch (ch)
            {
                case '|':
                    second = ParsePatternIntoNdfa(builder, end, out end);
                    result = FiniteAutomata.CombineAsOrAndFree(first, second);
                    break;

                case '*':
                    result = FiniteAutomata.StarAndFree(first);
                    break;

                default:
                    Debug.Assert(ch == 'a' || ch == 'b' || ch == '(');
                    end--;
                    second = ParsePatternIntoNdfa(builder, end, out end);
                    result = FiniteAutomata.ConcatenateAndFree(first, second);
                    break;
            }

            ch = builder[end++];
            Debug.Assert(ch == ')');

            return result;
    }
}
```

```
default:
throw new Exception();
}
}
```

1. <https://www.hackerrank.com/challenges/count-strings>
  2. <https://gist.github.com/jianminchen/0e1fe57d9f6763d91b640c2603c73a90>
  3. [https://en.wikipedia.org/wiki/Deterministic\\_finite\\_automaton](https://en.wikipedia.org/wiki/Deterministic_finite_automaton)
- 

### HackerRank: Count string (III) (2016-04-09 13:53)

April 9, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/count-strings>

Solution to study:

[2]<https://gist.github.com/jianminchen/336dd9b9bbaa9a44de2982a84405644d>

The code has time out issue, wrong answer, and it only scores **3 out of 80** .

code source:

[3]<https://www.hackerrank.com/casaro>

#### Statistics:

Time spent: 1:00pm - 3:00pm

Go through test cases

```
using System;
using System.Collections.Generic;
using System.IO;
```

```
class CountStrings
{
    static Dictionary<string, long> knownen = new Dictionary<string, long>();

    static void Main(string[] args)
    {
```

```
long countTestCases = Convert.ToInt64(System.Console.ReadLine( ));
```

```
for (long k = 0; k < countTestCases; k++)  
{
```

```
    string[] input = System.Console.ReadLine().Split(' ');  
    string regexp = input[0];  
    int n = Convert.ToInt32(input[1]);
```

```
    long result = Count(regexp, n);
```

```
    Console.WriteLine(result);  
}  
}  
/*
```

**\* Julia's comment:**

**Test case A:**

**1**

**ab 2**

**Program crashes, the form should be (ab) instead, check the definition.**

**Test case B:**

**1**

**(a|b) 1**

**Test case C:**

**1**

**(ab) 2**

**Test case D:**

**1**

**((ab)) 2**

**or operation:**

```
for (int i = 0; i <= n; i++)
```

```
{  
    result += Count(r1, i) * Count(r2, n - i);  
    result %= 1000000007;  
}
```

**go through all the cases:**

**i = 0, 1, 2, 3**

**four iterations**

**Test case D:**

**((ab)|(ba)) 2**

**Test case E:**

**324**

**(ab\*) 3**

```
*/
static long Count(string regexp, int n)
{
    if (regexp.Length == 1)
    {
        if (n == 1)
            return 1;
        else
            return 0;
    }

    if (knownen.ContainsKey(regexp + n))
    {
        return knownen[regexp + n];
    }

    regexp = regexp.Substring(1, regexp.Length - 2);

    long result;
    if (regexp[regexp.Length - 1] == '*')
    {
        string r1 = regexp.Substring(0, regexp.Length - 1);

        if (n == 0)
            return 1;

        result = 0;
        for (int i = 1; i <= n; i++)
        {
            result += Count(r1, i) * Count("(" + regexp + ")", n - i);
            result %= 1000000007;
        }
    }
    else if (isOrExpression(regexp) >= 0)
    {
        int posDivider = isOrExpression(regexp);
        string r1 = regexp.Substring(0, posDivider);
        string r2 = regexp.Substring(posDivider + 1, regexp.Length - posDivider - 1);

        result = Count(r1, n) + Count(r2, n);
        result %= 1000000007;
    }
    else
    {
        int posDivider = isAndExpression(regexp);
        string r1 = regexp.Substring(0, posDivider + 1);
        string r2 = regexp.Substring(posDivider + 1, regexp.Length - posDivider - 1);
```

```

result = 0;
for (int i = 0; i <= n; i++)
{
    result += Count(r1, i) * Count(r2, n - i);
    result %= 1000000007;
}
}

knownen.Add("(" + regexp + ")" + n, result);

return result;
}

```

```

/*
* Test case:
* (a|b) 1
* return yes
*
*/
private static int isOrExpression(string regexp)
{
    Stack<char> braces = new Stack<char>();

    for (int i = 0; i < regexp.Length; i++)
    {
        char c = regexp[i];

        if (c == '(')
        {
            braces.Push(c);
        }
        else if (c == ')')
        {
            braces.Pop();
        }
        else if (c == '|')
        {
            if (braces.Count == 0)
                return i;
        }
    }

    return -1;
}

```

```

/*
Test case:
326

```

**(ab) 2**  
**regex ab**

**Test case:**  
**((ab)) 2**

```
*/
private static int isAndExpression(string regex)
{
    Stack<char> braces = new Stack<char>();

    if (regex[0] == 'a' || regex[0] == 'b')
        return 0;

    for (int i = 0; i < regex.Length; i++)
    {
        char c = regex[i];

        if (c == '(')
        {
            braces.Push(c);
        }
        else if (c == ')')
        {
            braces.Pop();
        }

        if (braces.Count == 0)
            return i;
    }

    return -1;
}
```

**April 10, 2016**

**Action items:**

Need to think about in computer theory, NFA, DFA, and argue that the above solution in theory has flaws.

1. <https://www.hackerrank.com/challenges/count-strings>
2. <https://gist.github.com/jianminchen/336dd9b9bb9a44de2982a84405644d>
3. <https://www.hackerrank.com/casaro>

## HackerRank: count string (IV) - JavaScript (2016-04-09 16:02)

April 9, 2016

Spend some time later to go over this JavaScript solution, score 80 out of 80.

[1]<https://gist.github.com/jianminchen/4d5d2f77d0e23e424a0953aaf4e4276e>

1. <https://gist.github.com/jianminchen/4d5d2f77d0e23e424a0953aaf4e4276e>

---

## HackerRank - count string (V) - C plus plus solution (2016-04-09 16:36)

April 9, 2016

Spend 1 hour (4:00pm - 5pm) on this C++ solution, talk about the code and implementation:

[1]<https://gist.github.com/jianminchen/b1a7b92af1856a542167d4b82517bb8f>

Julia needs to figure out the design:

For test case:

1  
((ab)|(ab)) 2

The design is to construct a graph, NFA, and then, convert it to DFA, and the count how many ways.

It is hard to teach yourself through HackerRank a solution; so, Julia likes to catch up by some reading. Searching the web ...

Read some blogs to get some help:

1. Not very useful

[2][http://webcache.googleusercontent.com/search?q=cache:http://infolab.stanford.edu/~ullman/ialc/win00/projecthints.txt-&gws\\_rd=cr&ei=VpsJV4fjKYvOjwOiuaOIBg](http://webcache.googleusercontent.com/search?q=cache:http://infolab.stanford.edu/~ullman/ialc/win00/projecthints.txt-&gws_rd=cr&ei=VpsJV4fjKYvOjwOiuaOIBg)

2. Try to read it in 10 minutes

[3]<https://cseweb.ucsd.edu/classes/wi03/cse105/hw1sol.pdf>

3. **Another reading: 30 minutes reading - excellent content! Julia learns better after she invested 8 hours to try to understand a problem, by playing with hackerRank.**

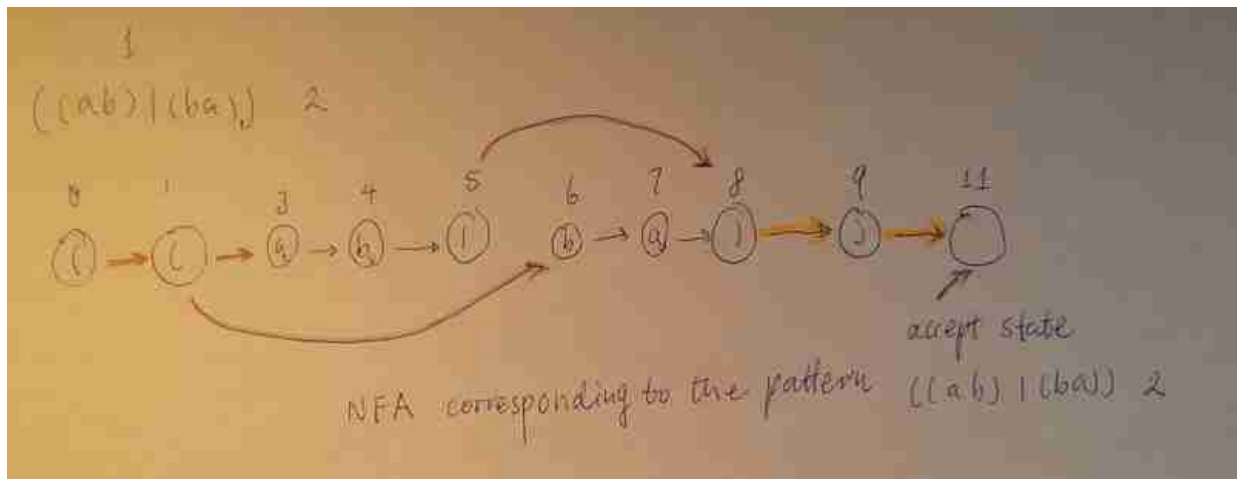
[4]<http://algs4.cs.princeton.edu/lectures/54RegularExpressions.pdf>

Julia, try to memorize content on the above slides.

Learn quickly from lecture notes, a diagram for NFA:



$((ab)|(ba))^2$



1. <https://gist.github.com/jianminchen/b1a7b92af1856a542167d4b82517bb8f>
2. [http://webcache.googleusercontent.com/search?q=cache:http://infolab.stanford.edu/~ullman/ialc/win00/projecthints.txt&gws\\_rd=cr&ei=VpsJV4fjKYv0jw0iua0IBg](http://webcache.googleusercontent.com/search?q=cache:http://infolab.stanford.edu/~ullman/ialc/win00/projecthints.txt&gws_rd=cr&ei=VpsJV4fjKYv0jw0iua0IBg)
3. <https://cseweb.ucsd.edu/classes/wi03/cse105/hw1sol.pdf>
4. <http://algs4.cs.princeton.edu/lectures/54RegularExpressions.pdf>

## Mental skills to help players (2016-04-09 22:08)

April 9, 2016

Competitive programming is like tennis sports. So, when Julia blogs her practice on problem solving on HackerRank, she also likes to journal her practice, for example, log time spent on.

Here are the notes taken:

Mental Skills to Help Your Players

[1]<https://www.youtube.com/watch?v=E8-7LA9-UOY>

## Journaling practices/ Matches (27:24/39:58)

- Date, Time, Type of Play (Practice/ Match)
- Opponent ( if applicable)
- Length of Time
- Feeling
- Emotions

- Focus/ Concentration
- Nutrition/ Hydration
- 3 things you did well, 1-3 things to improve

### Goal Setting:

- smart goals
- LT and ST goals
- Set for practice and competition
- Write them down!!!
- Reflect on motivation and you commitment to these goals
- Evaluate, assess, adjust goals as needed

### Reference:

#### USTA Mental Skills and Drills Handbook

Action item: Read the book 20 pages. Take some short notes as well.

1. <https://www.youtube.com/watch?v=E8-7LA9-U0Y>

---

### HackerRank: string function calculation - string algorithm - Brute Force Solution (2016-04-10 15:35)

April 10, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/string-function-calculation>

Time spent: 2:40pm - 3:35pm 20 minutes to think, 20 minutes to write down in C #

Understand the problem, so, here are her thoughts:

Brute force solution:

pass one test case: aaaaaa, return 12,

failed test case:

abcbcd

wrong answer - Easy fix, index error on the loop

score 8.89/ 80

one test case: wrong answer  
All other test cases time out

[2]<https://gist.github.com/jianminchen/09c77ba32b156f765b4debb2bccba0c8>

Need to work on KMP, or Boyer-Moore algorithm, see how to make HackerRank happy, score more points.

[3]<http://www.cs.tufts.edu/comp/150GEN/classpages/BoyerMoore.html>

blog:

[4][http://juliachencoding.blogspot.ca/search/label/string %20functions %20review](http://juliachencoding.blogspot.ca/search/label/string%20functions%20review)

1. <https://www.hackerrank.com/challenges/string-function-calculation>
  2. <https://gist.github.com/jianminchen/09c77ba32b156f765b4debb2bccba0c8>
  3. <http://www.cs.tufts.edu/comp/150GEN/classpages/BoyerMoore.html>
  4. <http://juliachencoding.blogspot.ca/search/label/string%20functions%20review>
- 

## **HackerRank: string function calculation (II) - string algorithm - Boyer Moore Algorithm (2016-04-10 15:44)**

April 10, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/string-function-calculation>

Time spent: 3:40pm - 4:40 pm

First hour work, score 8.9 out of 80 points.

The solution:

[2]<https://gist.github.com/jianminchen/09c77ba32b156f765b4debb2bccba0c8>

Brute force solution:

pass one test case: aaaaaa, return 12,

use Boyer Moore Algorithm, still time out:

time spent: 8:00pm - 8:45pm

copy Boyer Moore algorithm code from the webpage:

[3]<http://algs4.cs.princeton.edu/53substring/BoyerMoore.java.html>

convert the class from Java to C #.

Julia second try: ( **time out issue - Boyer Moore Algorithm does not help!!!** )

[4]<https://gist.github.com/jianminchen/43e8593146434339285a87dc38fae166>

1. <https://www.hackerrank.com/challenges/string-function-calculation>
  2. <https://gist.github.com/jianminchen/09c77ba32b156f765b4debb2bccba0c8>
  3. <http://algs4.cs.princeton.edu/53substring/BoyerMoore.java.html>
  4. <https://gist.github.com/jianminchen/43e8593146434339285a87dc38fae166>
- 

## HackerRank: String Calculate function (III) - Suffix array (2016-04-10 21:16)

April 10, 2016

**Time spent: 9:00pm - 11:40pm**

Problem statement:

[1]<https://www.hackerrank.com/challenges/string-function-calculation>

KMP Algorithm, Rabin Karp Algorithm, Finite Automate based Algorithm, Boyer Moore algorithm.

All of the above algorithms preprocess the pattern to make the pattern searching faster. The best time complexity that we could get by preprocessing pattern is  $O(n)$  where  $n$  is length of the text.

So, Julia second try still failed, no progress on time out issue.

Now, Julia read the blog and learn suffix array / LCP array in next hour 9:00pm -10:00pm:

[2]<http://www.geeksforgeeks.org/pattern-searching-set-8-suffix-tree-introduction/>

A suffix tree is built of the text. After preprocessing text (building suffix tree of text), we can search any pattern in  $O(m)$  time where  $m$  is length of the pattern.  **$O(n) \rightarrow O(m)$ , solve timeout issue** .

Read suffix array blog, play with code first:

[3]<http://www.geeksforgeeks.org/pattern-searching-set-8-suffix-tree-introduction/>

[4]<http://www.geeksforgeeks.org/suffix-array-set-1-introduction/>

here is the suffix array submission:

[5][https://www.hackerrank.com/rest/contests/master/challenges/string-function-calculation/hackers/oguzhaneker/download\\_solution](https://www.hackerrank.com/rest/contests/master/challenges/string-function-calculation/hackers/oguzhaneker/download_solution)

read C # suffixArray implementation:

[6]<https://github.com/eranmeir/Sufa-Suffix-Array-Csharp/blob/master/Sufa/SuffixArray.cs>

one more reading ( **blog #1** ):

[7]<https://blogs.msdn.microsoft.com/dhuba/2010/04/04/suffix-array/>

Use C # implementation of suffix array in blog #1, **still time out on hackerRank**

[8]<https://gist.github.com/jianminchen/0f561b8afc1cdf23ab95a8b603cd2a3e>

**Plan to work on LCP array later, solve timeout issue.**

1. <https://www.hackerrank.com/challenges/string-function-calculation>
  2. <http://www.geeksforgeeks.org/pattern-searching-set-8-suffix-tree-introduction/>
  3. <http://www.geeksforgeeks.org/pattern-searching-set-8-suffix-tree-introduction/>
  4. <http://www.geeksforgeeks.org/suffix-array-set-1-introduction/>
  5. [https://www.hackerrank.com/rest/contests/master/challenges/string-function-calculation/hackers/oguzhaneker/download\\_solution](https://www.hackerrank.com/rest/contests/master/challenges/string-function-calculation/hackers/oguzhaneker/download_solution)
  6. <https://github.com/eranmeir/Sufa-Suffix-Array-Csharp/blob/master/Sufa/SuffixArray.cs>
  7. <https://blogs.msdn.microsoft.com/dhuba/2010/04/04/suffix-array/>
  8. <https://gist.github.com/jianminchen/0f561b8afc1cdf23ab95a8b603cd2a3e>
- 

## **HackerRank: String Calculate function (III) - Suffix array (II) (2016-04-11 20:15)**

**April 11, 2016**

Problem statement:

[1]<https://www.hackerrank.com/challenges/string-function-calculation>

**Plan to work on LCP array later.**

Use C # implementation of suffix array in blog #1, **still time out on hackerRank**

[2]<https://gist.github.com/jianminchen/0f561b8afc1cdf23ab95a8b603cd2a3e>

**SuffixArray implemented in C #, by MSFT, Google employee; Julia has to catch up C #, and learn**

**how to write beautiful code like this:**

**Study C # code how to implement IEnumerable interface in suffixArray code:**

[3][https://msdn.microsoft.com/en-us/library/system.collections.ienumerable\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.collections.ienumerable(v=vs.110).aspx)

**how String.Compare api is designed**

[4][https://msdn.microsoft.com/en-us/library/x7tax739\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/x7tax739(v=vs.110).aspx)

Julia also takes some time to work on suffix array, and get some drawing about suffix tree, and play with two case:

aaaaaa,

banana,

abcbacddd,

For those two strings, what are suffix tree? Get so close to concrete examples, draw diagram, run test case, study code, and compare to other C # submission.

**Focus on how to find the range of suffix array to fit in the pattern:**

[5]<https://gist.github.com/jianminchen/700ebc3150efdd9ac33ac75f56f727d3>

**Walk through the examples how suffix array is implemented in C # code:**

public

```

SuffixArray
Search
(
string
str )
{
if
(m _lower > m _upper)
return
this
;
// Otherwise search for boundaries that enclose all
// suffixes that start with supplied string.
var
lower = Search(str, c _lower);
var
upper = Search(str, c _upper);
return
new
SuffixArray(m _text, m _pos, lower +
1
, upper);
}

```

**Example 1: string "aaaaaa", pattern string "aaa".**

**suffix array {5, 4, 3, 2, 1, 0 }**

**suffix string:**

**a**

**aa**

**aaa**

**aaaa**

**aaaaa**

**aaaaaa**

suffix array is ascending order, `string.Compare("a","aa") = -1,`

`string.Compare("aa","aa") = 0.`

`string.Compare("aa","a") = 1`

suffix array is implemented using interface IEnumerable.

we try to find 2 index, low, top,

index = 1, "aa", string.compare("aa", 0, "aaa", 0, 3) = -1,

now, we need to find top index,

assume that "abc" is in the array, string.compare("abc", 0, "aaa", 0, 3) = 1, first one is >-1, stop;

aaa, aaaa, aaaaa, aaaaa, all computed value of comparison = 0 since we only check substring with length 3.

So top index is 5.

So, count of substring "aaa" is calculate by top index - low index + 1

top index = 5,

low index = 1+1

so count = 4.

**Try to figure out how this binary search algorithm is used to calculate low index and top index, called twice, comparison value for low index = 0, for high index is -1.**

**In other words, find last one is smaller than "aaa", and first one bigger than "aaa".**

**Example 2: "banana", pattern "ban",**

**Read the content in the blog first:**

**[6]<http://www.geeksforgeeks.org/pattern-searching-set-8-suffix-tre e-introduction/>**

And then,

suffix strings:

banana

anana

nana

ana

na

a

Sort them ascending order:

a

ana

anana

banna

na

nana

pattern string "ban", first to find a string which is last one < "ban", -> "anana", index 2, "ana" < "ban"

and then, find first string which is bigger than "ban", only comparing length 3 substring,

"banna" = "ban" based on comparison (substring len = 3), so

"na" - index = 4 > "ban"

$2+1 = 3$ ,

top index = 3

so, ban pattern match is  $3-3+1 = 1$

**"Work on Examples First**

**" - Practice feels good!**

The LCP-LR array helps improve this to  $O(m+\log N)O(m+\log N)$ , in the following way

LCP array reading

<https://www.hackerrank.com/challenges/pseudo-isomorphic-substrings/top ics/lcp-array>

[https://en.wikipedia.org/wiki/LCP\\_array](https://en.wikipedia.org/wiki/LCP_array)

Follow up: May 4, 2015

Read the article:

[7]<http://stackoverflow.com/questions/2487576/trie-vs-suffix-tree-vs-suffix-array>

June 10, 2016 - Plan to spend 30 minutes to read the slides:

Julia, learn something here from the blog:

[8]<http://decomplexify.blogspot.ca/>

[9]<http://decomplexify.blogspot.ca/2014/06/suffix-array.html>

[10]<http://decomplexify.blogspot.ca/2014/07/lcp-array.html?view=classic>

1. <https://www.hackerrank.com/challenges/string-function-calculation>
2. <https://gist.github.com/jianminchen/0f561b8afc1cdf23ab95a8b603cd2a3e>
3. [https://msdn.microsoft.com/en-us/library/system.collections.ienumerable\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.collections.ienumerable(v=vs.110).aspx)
4. [https://msdn.microsoft.com/en-us/library/x7tax739\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/x7tax739(v=vs.110).aspx)
5. <https://gist.github.com/jianminchen/700ebc3150efdd9ac33ac75f56f727d3>
6. <http://www.geeksforgeeks.org/pattern-searching-set-8-suffix-tree-introduction/>
7. <http://stackoverflow.com/questions/2487576/trie-vs-suffix-tree-vs-suffix-array>
8. <http://decomplexify.blogspot.ca/>
9. <http://decomplexify.blogspot.ca/2014/06/suffix-array.html>
10. <http://decomplexify.blogspot.ca/2014/07/lcp-array.html?view=classic>



## K Index algorithm (2016-04-12 08:45)

April 12, 2016

An array, to search if there is duplicate in k steps distance

First practice, two bugs ( **Time spent: 1 hour** ):

[1]<https://gist.github.com/jianminchen/1fec656b154a70acb30b5f6d7ad509ab>

```
private static bool DFS(int[][] arr, int oriX, int oriY, int row, int col, int kIndex, int search, int MaxRow, bool[][] searchedA)
{
    if (!IsValid(row, MaxRow) || !IsValid(col, MaxRow) || kIndex < 0)
        return false;

    if (Math.Abs(oriX - row) + Math.Abs(oriY - col) > 0 && !searchedA[row][col])
    {
        if (arr[oriX][oriY] == arr[row][col])
            return true;
        //bug 001 - Julia, you need to continue do DFS here
    }
    else
    {
        searchedA[row][col] = true;

        // bug 002 - all those DFS search, you need to check search result!
        DFS(arr, oriX, oriY, row - 1, col, kIndex - 1, search, MaxRow, searchedA);
        DFS(arr, oriX, oriY, row + 1, col, kIndex - 1, search, MaxRow, searchedA);
        DFS(arr, oriX, oriY, row, col + 1, kIndex - 1, search, MaxRow, searchedA);
        DFS(arr, oriX, oriY, row, col - 1, kIndex - 1, search, MaxRow, searchedA);
    }

    return false;
}
```

Fix two bugs ( **Time spent: 20+ minutes** ):

[2]<https://gist.github.com/jianminchen/ce7ccfa5db5b57c36d6742b622e9153e>

function after bugs are fixed:

```
private static bool DFS(int[][] arr, int oriX, int oriY, int row, int col, int kIndex, int search, int MaxRow, bool[][] searchedA)
{
    if (!IsValid(row, MaxRow) || !IsValid(col, MaxRow) || kIndex < 0)
        return false;

    if (Math.Abs(oriX - row) + Math.Abs(oriY - col) > 0 && !searchedA[row][col] )
    {
        if (arr[oriX][oriY] == arr[row][col])
```

```
return true;
}
```

searchedA[row][col] = true; / / **Bug001** - check condition is wrong

```
if (DFS(arr, oriX, oriY, row - 1, col, kIndex - 1, search, MaxRow, searchedA) ||
    DFS(arr, oriX, oriY, row + 1, col, kIndex - 1, search, MaxRow, searchedA) ||
    DFS(arr, oriX, oriY, row, col + 1, kIndex - 1, search, MaxRow, searchedA) ||
    DFS(arr, oriX, oriY, row, col - 1, kIndex - 1, search, MaxRow, searchedA))
    return true; // bug002 - any of conditions are ok, then, return true
```

```
return false;
}
```

#### Actionable item:

1. Julia, you have to build up skills to do **code static analysis** . Debugging takes time, you should check your logic, run your code through yourself by thinking, criticize your code by thinking about the test case, go through virtually first.

**April 18, 2016**

Julia, you should have more than 1 idea to solve this kind of problem - thinking about using Queue to solve it.

1. <https://gist.github.com/jianminchen/1fec656b154a70acb30b5f6d7ad509ab>
  2. <https://gist.github.com/jianminchen/ce7ccfa5db5b57c36d6742b622e9153e>
- 

#### rotate array (2016-04-12 08:48)

April 12, 2016

Rotate array one element to right or down, clockwise.

For example:

```
1 2 3
8 9 4
7 6 5
```

After rotation, the matrix is the following:

```
8 1 2
7 9 3
6 5 4
```

Practice: ( **Time spent: 1 hour** )

[1]<https://gist.github.com/jianminchen/24d77970e9a58a7850f2add10dfe7c4f>

Failed test case:

```
1 2
338
```

4 3

output should be:

4 1

3 2

but my result:

3 2

4 1

Bug fix: (Time spent: 20+ minutes)

[2]<https://gist.github.com/jianminchen/6eb6244fbb1912e3a06e672f604f87e0>

```
/*
Do it in place
/
public static bool rotateArray(int[][] arr, int k)
{
    if (arr == null) return false;
    int n = arr.Length; // row
    int m = arr[0].Length; // column

    if (n != m) return false;
    if (n != k) return false;

    int start = 0;
    int end = k - 1;

    while (start <= end && start < k / 2)
    {
        // for left column
        swap(arr, start, start, start, start + 1, k);
        for (int i = start + 1; i <= end; i++)
        {
            swap(arr, i, start, i - 1, start, k);
        }

        // for down row - swap
        for (int i = start; i < end; i++)
        {
            swap(arr, end, i, end, i + 1, k);
        }

        // for right column
        // for (int i = end; i > start; i-) // bug 001 if there are only two rows, exception
        for (int i = end; i > start && (end - start) > 1; i-) // bug001 fix - do not swap if there are two rows
        {
            swap(arr, i, end, i - 1, end, k);
        }

        // for up row
        for (int i = end; i > start + 2; i-)
```

```

{
swap(arr, start, i, start, i - 1, k);
}
start++;
end--;
}
return true;
}

```

#### Action item:

Base case 1x1, 2x2, and then 3X3, you cannot skip 2x2. It doesn't matter what test case is given. Think about basics.

Also, think about how many swaps are needed for

1 2

4 3

**only 3 swaps:**

**First** one,

1 and 2

1 2

**second one:**

2 and 4

2

4

**third one:**

2 and 3

2 3

<- and then, need to filter out last 2 swaps in the while loop.

Relax, take more practice:

Smart to use debugger:

[3]<https://www.quora.com/Does-using-debugger-help-a-lot-in-competitive-programming>

[4]<https://www.quora.com/What-are-the-best-competitive-programming-debugging-tips>

#### April 15, 2016

Better design is to save arr[0][0] value, and then shift array value to left on top row. Make corner case much easy to handle.

So, good ritual is to come out more than 1 idea, and then, compare which one is better. Ask why!

Swap value cannot beat the array shift, much simpler the later one.

1. <https://gist.github.com/jianminchen/24d77970e9a58a7850f2add10dfe7c4f>

2. <https://gist.github.com/jianminchen/6eb6244fbb1912e3a06e672f604f87e0>

3. <https://www.quora.com/Does-using-debugger-help-a-lot-in-competitive-programming>

4. <https://www.quora.com/What-are-the-best-competitive-programming-debugging-tips>

## Advice for practice on coding question - easy, medium to hard questions (2016-04-12 21:25)

April 12, 2016

**Good Sharing from the following blog:  
Know new people and their success story:**

Easy, medium questions, just for coding, not much algorithm. Practice coding questions. And then, move to hard questions. And come back to easy, medium questions again, to improve speed.

[1]<https://www.quora.com/How-does-Ahmed-Aly-practice-for-competitive-programming>

**Very good story:**

[2]<https://www.quora.com/Whom-do-you-respect-the-most-in-the-world-living-Aly?srid=o3vR> -Why/answer/Ahmed-

**How to improve? Solve a lot of problems, easy problems.**

[3]<http://qr.ae/81vRga>

**Julia's thought:**

No time to write your own solutions. Then, read a lot of solution, over 100 solution for one problem. Easy problem, string, on HackerRank, read all kinds of solutions, more than 20 of them; and try to understand them first.

**Coach gave tips how to improve quickly?**

[4]<http://qr.ae/81vc1V>

**Talk about bugs:**

[5]<http://qr.ae/81rRyj>

Julia's thought:

Debug takes time.

Relax and read more coding blog:

From MSFT, and

[6]<https://www.quora.com/profile/Duncan-Smith-23>

[7]<http://www.redgreencode.com/>

weekly blog about TopCoder competitive algorithms:

[8]<http://petr-mitrichev.blogspot.com/2016/04/a-doubles-week.html>

1. <https://www.quora.com/How-does-Ahmed-Aly-practice-for-competitive-programming>

2. <https://www.quora.com/Whom-do-you-respect-the-most-in-the-world-living-Why/answer/Ahmed-Aly?srid=o3vR>

3. <http://qr.ae/81vRga>

4. <http://qr.ae/81vc1V>
  5. <http://qr.ae/81rRyj>
  6. <https://www.quora.com/profile/Duncan-Smith-23>
  7. <http://www.redgreencode.com/>
  8. <http://petr-mitrichev.blogspot.com/2016/04/a-doubles-week.html>
- 

## **HackerRank: String Calculate function (III) - LCP array (2016-04-13 20:34)**

April 13, 2016

Still work on the problem:

Problem statement:

[1]<https://www.hackerrank.com/challenges/string-function-calculation>

Put together 2-3 hours on LCP array study first, and then, work out a solution for this advanced algorithm.

The LCP-LR array helps improve this to  $O(m+\log N)$ , in the following way

**LCP array reading**

[2]<https://www.hackerrank.com/challenges/pseudo-isomorphic-substrings/topics/lcp-array>

[3][https://en.wikipedia.org/wiki/LCP\\_array](https://en.wikipedia.org/wiki/LCP_array)

**Question and Answer:**

**Before you write LCP implementation, tell me what you learn about LCP, give me an example you work on?**

1. <https://www.hackerrank.com/challenges/string-function-calculation>
  2. <https://www.hackerrank.com/challenges/pseudo-isomorphic-substrings/topics/lcp-array>
  3. [https://en.wikipedia.org/wiki/LCP\\_array](https://en.wikipedia.org/wiki/LCP_array)
- 

## **HackerRank: Matrix Rotation (2016-04-14 18:12)**

April 14, 2016

Rotate array - HackerRank

problem statement:

[1]<https://www.hackerrank.com/challenges/matrix-rotation-algo>

Spent hours to work on the solution, passed 4 basic test case, but still score 0 out of 80 points:

[2]<https://gist.github.com/jianminchen/4d719488e378aeb498d4cb363cbd1f7d>

Just be patient. Study other's code:

[3]<https://gist.github.com/jianminchen/c234a238ea331ea9a037d603c37a6649>

So, do not swap two nodes value, just save first node's value in a variable, and then, shift array kind operation.

Corner case is very simple. **Julia wasted hours to work on poor design, try to avoid overwrite the value of arr[1][0]. Just use a variable to save it.**

[4][https://www.hackerrank.com/rest/contests/master/challenges/matrix-rotation-algo/hackers/Relentless/download\\_solution](https://www.hackerrank.com/rest/contests/master/challenges/matrix-rotation-algo/hackers/Relentless/download_solution)

So, changed my own code, with bugs to fix: Int16 should be int, check Int16 max value 32767, whereas Int32 max value 2,147,484,647, less than  $3 * 10^9$

[5]<https://gist.github.com/jianminchen/6e3ca88e8633ad8bace8387f154904df>

only pass one case:

[6]<https://gist.github.com/jianminchen/12b58729c59e44a0d71d9f9c47bc937c>

**another practice: (more than 1 hour, still having bugs, score 8.89/ wrong answer)**

[7]<https://gist.github.com/jianminchen/57572227d4fe939060f7cc81b193cd9b>

**study the code here :**

[8]<https://gist.github.com/jianminchen/6e2ca0ac395913646ed012c40c283e7f>

[9]<https://gist.github.com/jianminchen/c0f91e5f5406c3dce35ce48c4c646c33>

study forum:

[10]<https://www.hackerrank.com/challenges/matrix-rotation-algo/forum>

1. <https://www.hackerrank.com/challenges/matrix-rotation-algo>
2. <https://gist.github.com/jianminchen/4d719488e378aeb498d4cb363cbd1f7d>
3. <https://gist.github.com/jianminchen/c234a238ea331ea9a037d603c37a6649>
4. [https://www.hackerrank.com/rest/contests/master/challenges/matrix-rotation-algo/hackers/Relentless/download\\_solution](https://www.hackerrank.com/rest/contests/master/challenges/matrix-rotation-algo/hackers/Relentless/download_solution)
5. <https://gist.github.com/jianminchen/6e3ca88e8633ad8bace8387f154904df>

6. <https://gist.github.com/jianminchen/12b58729c59e44a0d71d9f9c47bc937c>
  7. <https://gist.github.com/jianminchen/57572227d4fe939060f7cc81b193cd9b>
  8. <https://gist.github.com/jianminchen/6e2ca0ac395913646ed012c40c283e7f>
  9. <https://gist.github.com/jianminchen/c0f91e5f5406c3dce35ce48c4c646c33>
  10. <https://www.hackerrank.com/challenges/matrix-rotation-algo/forum>
- 

## HackerRank: Connected Cell in a Grid - C# solution - using DFS (2016-04-16 15:03)

**April 16, 2016**

Practice one DFS algorithm this Saturday.

problem statement:

[1]<https://www.hackerrank.com/challenges/connected-cell-in-a-grid>

C # solution written by Julia:

[2]<https://gist.github.com/jianminchen/4f4d499599c015144e77e5016bd86912>

**Time spent: 4:48pm - 5:41pm**

**20 minutes to think about design,**

**30 minutes to write the code**

**Feeling:** nervous when thinking about solution - DFS, it took more than 15 minutes

**Study other submissions, and see if I can learn something, also figure out how to cut time to write the code.**

**C # solution:**

**Time spent: 60 minutes**

**1. Use Queue , instead of using recursive calls, using 2 dimension array, excellent code to study:**

[3]<https://gist.github.com/jianminchen/9776d3d341d80c742c6c751f5ef00cb6>

**tips to learn:**

**1. use integer as key:  $i*10+j$**

**2. mark the visited node using value 2**



## 2. Design the function to return a value

[4]<https://gist.github.com/jianminchen/d6e2994723452aef951a860e8de2def8>

Julia, people are smart, no need for extra bool array[row][col], just mark the visited node using a new value:

1. mark visited node using 'X'

3. use C # Tuple class, declare a shift array to get more organized. 8 neighbors nodes - go through array twice.

[5]<https://gist.github.com/jianminchen/c22abefe625ca4201d18841112a1a99a>

4. study the code

[6]<https://gist.github.com/jianminchen/bc2c399407a30181ca68292459a3c791>

5. declare offset array for x and y - Julia likes the idea.

[7]<https://gist.github.com/jianminchen/737138aeb946961aded556181c06502e>

### Statistics :

1. 10 submissions, there are at least 2 of them using queue - Julia, you have to catch up this using queue! <- write queue version, practice it 20 minutes, post your version down here!

2. 10 submission, 2-3 DFS algorithm return count <- easy way to track the count

3. Different ways to mark the node is visited.

1. <https://www.hackerrank.com/challenges/connected-cell-in-a-grid>

2. <https://gist.github.com/jianminchen/4f4d499599c015144e77e5016bd86912>

3. <https://gist.github.com/jianminchen/9776d3d341d80c742c6c751f5ef00cb6>

4. <https://gist.github.com/jianminchen/d6e2994723452aef951a860e8de2def8>

5. <https://gist.github.com/jianminchen/c22abefe625ca4201d18841112a1a99a>

6. <https://gist.github.com/jianminchen/bc2c399407a30181ca68292459a3c791>

7. <https://gist.github.com/jianminchen/737138aeb946961aded556181c06502e>

---

## HackerRank - Connected Cell In A Grid (II) - C# solution (II) using Queue (2016-04-17 16:19)

April 17, 2:50 - 3:25

problem statement:

[1]<https://www.hackerrank.com/challenges/connected-cell-in-a-grid>

**First practice** using C #, solution written by Julia:

[2]<https://gist.github.com/jianminchen/4f4d499599c015144e77e5016bd86912>

**Study the code:**

1. Use Queue, instead of using recursive calls, using 2 dimension array, excellent code to study:

[3]<https://gist.github.com/jianminchen/9776d3d341d80c742c6c751f5ef00cb6>

So, Julia did one more practice, and just wrote second implementation using idea in the above blog.

Write C # code again using idea - 2 dimension array, queue, and also, mark the visited node using '2'. C # solution 1

### Second Practice:

[4]<https://gist.github.com/jianminchen/b84a84ad3d4b69e153290e8fa528c288>

It takes 30 minutes to write and fix bug, log interesting things happening in the practice:

1. First, fix the issue to read a row to a string, and then go over one char a time to get each node for the row.

use Console.ReadKey() first <- fatal error

2. use wrong local variable about key

3. Count should be 5 but 11 for one test case,

**count same thing more than once**

! - > add extra checking before counting. Or, set node is visited just before adding to the queue.

Compare to the solution #1, two dimension array uses integer 1 or 0, my copy is using char '1', '0';

And also, the solution #1, every node sets visited true before adding to queue.

And my design, do not set, which causes problem. Same node is added to the queue more than once.

It is better to add unvisited node to the queue once.

Julia, you have to make sure that no bug in the code, it does not matter what idea you use.

### Third practice:

Write in 20 minutes, no bug:

[5]<https://gist.github.com/jianminchen/574e77039f303cbbe3b72940236d5526>

Again, 3 practices:

#1

[6]<https://gist.github.com/jianminchen/4f4d499599c015144e77e5016bd86912>

#2

[7]<https://gist.github.com/jianminchen/b84a84ad3d4b69e153290e8fa528c288>

#3:

[8]<https://gist.github.com/jianminchen/574e77039f303cbb3b72940236d5526>

1. <https://www.hackerrank.com/challenges/connected-cell-in-a-grid>
  2. <https://gist.github.com/jianminchen/4f4d499599c015144e77e5016bd86912>
  3. <https://gist.github.com/jianminchen/9776d3d341d80c742c6c751f5ef00cb6>
  4. <https://gist.github.com/jianminchen/b84a84ad3d4b69e153290e8fa528c288>
  5. <https://gist.github.com/jianminchen/574e77039f303cbb3b72940236d5526>
  6. <https://gist.github.com/jianminchen/4f4d499599c015144e77e5016bd86912>
  7. <https://gist.github.com/jianminchen/b84a84ad3d4b69e153290e8fa528c288>
  8. <https://gist.github.com/jianminchen/574e77039f303cbb3b72940236d5526>
- 

## **HackerRank: Connected Cell in a Grid (III) - C# solution (III) - using recursive function (2016-04-17 17:08)**

April 17, 2016

problem statement:

[1]<https://www.hackerrank.com/challenges/connected-cell-in-a-grid>

Here is the code studied using DFS function with return value.

[2]<https://gist.github.com/jianminchen/bc2c399407a30181ca68292459a3c791>

Julia likes to practice using same idea, write her own implementation.

### **Practice #1:**

[3]<https://gist.github.com/jianminchen/fd909c3545e2081cf2dd2b6daea5900f>

### **Time spent:**

Copy main function from previous implementation, and write recursive function - it takes less than **15 minutes**.

First time write, no bug!

**April 18, 2018**

Here is another writing over **30 minutes** :

**Practice #2:**

[4]<https://gist.github.com/jianminchen/d434bcc59ebc8eb7c2cbb20c6f48ac06>

The practice goal is to see if I can shorten the time to 10 minutes in writing. How good I can write.

But, surprisingly, the practice takes 30 minutes; and I found out the several issues:

1. Timeout - recursive calls, if/ while checking
2. Array.GetLength api

First, time out issue; read a row of '0' or '1', should be if, I put a while. <- it takes more than 10 minutes to find out. I pinpoint recursive call, because I have doubt, not 100 % sure.

And then, I tried to avoid the recursive call to itself, and then, (dr, dc), mistakenly I put (dr, dr); HackerRank shows wrong answers for 2 test cases.

Make one change a time, and see if I can shorten the time to 10 minutes.

Study Array.getLength()

[5][https://msdn.microsoft.com/en-us/library/system.array.getlength\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.array.getlength(v=vs.110).aspx)

Another version, it takes **14 minutes** to write, without a bug:

**Practice #3:**

[6]<https://gist.github.com/jianminchen/bd155e574b32ebb163455dad02fa76b1>

Julia's performance is up and down, from 15 minutes to 30 minutes.

Again, all practices links:

#1

[7]<https://gist.github.com/jianminchen/fd909c3545e2081cf2dd2b6daea5900f>

#2

[8]<https://gist.github.com/jianminchen/d434bcc59ebc8eb7c2cbb20c6f48ac06>

#3

[9]<https://gist.github.com/jianminchen/bd155e574b32ebb163455dad02fa76b1>

Work on speed and accuracy. Practice, more concentrated on writing. More alert on bug code. More knowledge about api, and basic things.

Learn Jagged Array -

[10]<http://stackoverflow.com/questions/597720/what-are-the-differences-between-a-multidimensional-array-and-an-array-of-arrays>

1. <https://www.hackerrank.com/challenges/connected-cell-in-a-grid>
2. <https://gist.github.com/jianminchen/bc2c399407a30181ca68292459a3c791>
3. <https://gist.github.com/jianminchen/fd909c3545e2081cf2dd2b6daea5900f>
4. <https://gist.github.com/jianminchen/d434bcc59ebc8eb7c2cbb20c6f48ac06>
5. [https://msdn.microsoft.com/en-us/library/system.array.getlength\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.array.getlength(v=vs.110).aspx)
6. <https://gist.github.com/jianminchen/bd155e574b32ebb163455dad02fa76b1>

7. <https://gist.github.com/jianminchen/fd909c3545e2081cf2dd2b6daea5900f>
  8. <https://gist.github.com/jianminchen/d434bcc59ebc8eb7c2cbb20c6f48ac06>
  9. <https://gist.github.com/jianminchen/bd155e574b32ebb163455dad02fa76b1>
  10. <http://stackoverflow.com/questions/597720/what-are-the-differences-between-a-multidimensional-array-and-a-n-array-of-arrays>
- 

## reading time - reading articles (2016-04-17 18:23)

April 17, 2016

1. Read this blog about **C++ study** :

[1]<http://qr.ae/8NvjCa>

2. **sports programmer** :

[2][https://en.wikipedia.org/wiki/Gennady\\_Korotkevich](https://en.wikipedia.org/wiki/Gennady_Korotkevich)

3. **How to perform at work?**

[3]<http://qr.ae/8NYfUf>

Julia's thought:

Finally, I found some article to read, and then, try to express my ideas and concerns, relieve some stress.

First take notes from the talk:

1. **Constant learning** is everything in our line of work and **the perpetual hunger for knowledge** will keep you sharp throughout your years.
2. Like tennis sports, find a good tennis coach, maybe, it is also a solution to find a therapist that can guide you along the path of improvement of your self-esteem and self-evaluation.
3. If you are dealing with serious companies who will value you for **your strong CS fundamentals** and the ability to think in math and algorithms, they really don't give a damn if you don't speak their lingo yet, because they will see in you **an innate ability to learn**, even quicker than others and know that you'll ultimately be able to chew whatever they throw at you.

**Julia's thoughts:**

Confidence, and play nice, always be calm under stress at work. It takes a lot of practice, and learn through the experience. At work, you cannot play like sports; In sports, you make mistakes, and then, you try to fix them. Learn from sports activities's mistakes is much cheaper.

Still remembered that first day showing up in Burnaby central park tennis court in summer of 2012, I told a Chinese that I played tennis over 10 year from 2001 - 2010, one hour every 2 weeks; After 10 minutes to rally with him, he told me that I have to go to the wall and then practice against the wall. I found out that people over the park

may have play hundreds of hour tennis.

Later, people ask me how many hour I put on tennis sports already. 100 hours, 200 hours, 300 hours. I started to measure my hours spent, I did spend summer time every day 2 hours after work, 7:00pm - 10:00pm; the whole summer, over 100 hours, what it means, I lost at least 20 lb, first time, I built muscle around my stomach.

#### **Mistakes made in tennis sports :**

1. Hurt my **wrist** , wrist pain <- put a lot of hours on practice, do not have muscle memory to protect wrist; In my first 100 hours practice.
2. Hurt my **back** , fall on the ground in a double match <- could not walk after standing up at office after falling accident. In my first 200 - 300 hours practice, do not have muscle memory, knowledge of balance. To maintain body balanced is most important, it does not worth to fall in order to save a point.
3. Swing forehand, racket hitting my head, my legs/ arms a few times. <- after 200 - 300 hours practice

But I learned from tennis sports, how important it is to love the sports; build up relationship to your hitting partners, help each other to practice, and get improved; How important it is to find new players, help them improve, and then, you have a new hitting partner on the court.

So, people will feed me tennis balls; And invite me to play matches.

Certainly, I also learn to play for fun, not too cheap; give people some decency, not cheating on points when playing games, give people credit when in doubt. Show respect and encourage people to play more sports through the game activity.

Right now, I am totally going to extreme; To play social games, when the team I am in to win 2 sets in double games, my partner and I will let the opponent to win 3rd game.

Summarize what I say, in tennis sports, I learn a lot of people skills from my practice. People can help you achieve more on sports activities as well. You learn quickly how to read people, and get workout done.

#### **So, too far away from topic - how work is related to sports?**

When in trouble at work, sometimes, it is ok to be panic, get frustrated; but, tell the boss that "I am still learning". Ask for fair chance to learn, catch up, recover.

Recently, I found out that I broke the principles when I wrote code, If I have followed " **Do not repeat yourself** " principle closely, or " **Single Responsibility Principle** ", I should be less stressful to maintain products. I can exhaust all the test cases for a simple function/ class quickly, but I could not handle a function with more than 200 lines of code, just by human eye, not using debugging. If you play good on design principles, you find yourself in better situation.

Stop complaining. Just work hard. It takes more than 1 year to figure out how to excel at a job. My case, I think that it takes more than 5 years.

1. <http://qr.ae/8NvjCa>
  2. [https://en.wikipedia.org/wiki/Gennady\\_Korotkevich](https://en.wikipedia.org/wiki/Gennady_Korotkevich)
  3. <http://qr.ae/8NYfUf>
- 

## **HackerRank: Connected Cells in a grid (IV) - JavaScript code (2016-04-17 20:37)**

**April 16, 2016**

**problem statement:**

[1]<https://www.hackerrank.com/challenges/connected-cell-in-a-grid>

**JavaScript submission:**

Time spent: 30 minutes

Read 5 - 10 submissions, write down good tips:

[2]<https://www.hackerrank.com/challenges/connected-cell-in-a-grid/leaderboard/filter/language=javascript>

1. Favorite JavaScript implementation:

[3]<https://gist.github.com/jianminchen/eed134d2642090603865a44009d1e67a>

2. Very structured JavaScript code, using prototype, ===, neighbor nodes - an array - [], define function acting like class Node.

[4]<https://gist.github.com/jianminchen/8f17d6582ce6335af33e952450d255a3>

1. <https://www.hackerrank.com/challenges/connected-cell-in-a-grid>
2. <https://www.hackerrank.com/challenges/connected-cell-in-a-grid/leaderboard/filter/language=javascript>

3. <https://gist.github.com/jianminchen/eed134d2642090603865a44009d1e67a>
  4. <https://gist.github.com/jianminchen/8f17d6582ce6335af33e952450d255a3>
- 

## HackerRank: Connected Cells in a grid (V) - C++ solution (2016-04-17 20:39)

April 17, 2016

problem statement:

[1]<https://www.hackerrank.com/challenges/connected-cell-in-a-grid>

**C++ solution:**

**spent time: 40 minutes**

1. use `pair<int, int>`, make `_pair`, and queue; very short code - excellent!

[2]<https://gist.github.com/jianminchen/a1d7ac03bcaea4e1def43f61f41f53cb>

pair, make `_pair`, understand the standard library function template - make `_pair`:

[3][http://www.cplusplus.com/reference/utility/make\\_pair/](http://www.cplusplus.com/reference/utility/make_pair/)

2. Another solution:

[4]<https://gist.github.com/jianminchen/93839295fb15ab1a7d2226213f15dcd0>

3. use vector

[5]<https://gist.github.com/jianminchen/3e489592beaa267c10540302bf08745b>

4. interesting solution, no queue, no recursive calls; just store the values:

```
4
4
1 1 0 0
0 1 1 0
0 0 1 0
1 0 0 0
```

```
{-5, 0, -1, -1, -1, 0, 0, -1, -1, -1, 0, -1, -1, -1, -1, -1, -1, -1, -1, -1, ... } int[111]
```

[6]<https://gist.github.com/jianminchen/be42129298ca4ec156f06ecca58e72a1>

1. <https://www.hackerrank.com/challenges/connected-cell-in-a-grid>
2. <https://gist.github.com/jianminchen/a1d7ac03bcaea4e1def43f61f41f53cb>
3. [http://www.cplusplus.com/reference/utility/make\\_pair/](http://www.cplusplus.com/reference/utility/make_pair/)
4. <https://gist.github.com/jianminchen/a43da87aff353d894a22453668a962d2>



5. <https://gist.github.com/jianminchen/3e489592beaa267c10540302bf08745b>
  6. <https://gist.github.com/jianminchen/be42129298ca4ec156f06ecca58e72a1>
- 

## K index - Algorithm (II) using Queue (2016-04-18 23:13)

April 18, 2016

More code writing. Try to use Queue to solve the problem. Write more than one solution using Queue.

Julia practiced twice, first one hour she failed some test cases using HackerRank. And then, she tried again.

Practice #2:

[1]<https://gist.github.com/jianminchen/ce7ccfa5db5b57c36d6742b622e9153e>

First blog about this algorithm - K index:

[2]<http://juliachencoding.blogspot.ca/2016/04/k-index-algorithm.html>

Julia wrote C # implementation using Queue,

[3]<https://gist.github.com/jianminchen/63c0bcc2ab476d71abbe43c8837566>

### Test cases :

1. Input

4

1 2 3 4

5 6 7 8

9 10 11 12

10 14 15 16

2

Yes, 10 is found at arr[3,0], 2 steps away from arr[2,1]

### Time spent:

More than 30 minutes

### Learned from mistakes :

1. Two dimension array arr[,], use getLength(0) and getLength(1) for first and second dimension length; But, jagged array - arr[[]], getLength(1) will throw exception, use arr.Length, arrp[0].Length to check the size.

The first practice using queue, Julia created a run-time exception - index out of range; the error is so late to catch, not in compile time. So, take it seriously. Array, two dimensional array, jagged array are very basics.

[4]<https://gist.github.com/jianminchen/93f963043c47649a496a2fccc862d801>

change search array to use two dimensional array:

[5]<https://gist.github.com/jianminchen/9346c2d2cb2e611ec2db3519dc7a879a>

Read blog:

[6]<http://stackoverflow.com/questions/597720/what-are-the-differences-between-a-multidimensional-array-and-an-array-of-arrays>

[7]<http://stackoverflow.com/questions/12567329/multidimensional-array-vs?lq=1>

[8][https://msdn.microsoft.com/en-us/library/aa288453\(v=vs.71\).aspx](https://msdn.microsoft.com/en-us/library/aa288453(v=vs.71).aspx)

April 21, 2016

Julia found something she likes - C # tutorial

read it every day when you have 20 minutes in the morning.

[9][https://msdn.microsoft.com/en-us/library/aa288436\(v=vs.71\).aspx](https://msdn.microsoft.com/en-us/library/aa288436(v=vs.71).aspx)

1. <https://gist.github.com/jianminchen/ce7ccfa5db5b57c36d6742b622e9153e>
2. <http://juliachencoding.blogspot.ca/2016/04/k-index-algorithm.html>
3. <https://gist.github.com/jianminchen/63c0bcc2ab476d71abbe43c8837566>
4. <https://gist.github.com/jianminchen/93f963043c47649a496a2fccc862d801>
5. <https://gist.github.com/jianminchen/9346c2d2cb2e611ec2db3519dc7a879a>
6. <http://stackoverflow.com/questions/597720/what-are-the-differences-between-a-multidimensional-array-and-an-array-of-arrays>
7. <http://stackoverflow.com/questions/12567329/multidimensional-array-vs?lq=1>
8. [https://msdn.microsoft.com/en-us/library/aa288453\(v=vs.71\).aspx](https://msdn.microsoft.com/en-us/library/aa288453(v=vs.71).aspx)
9. [https://msdn.microsoft.com/en-us/library/aa288436\(v=vs.71\).aspx](https://msdn.microsoft.com/en-us/library/aa288436(v=vs.71).aspx)

---

## HackerRank: Matrix Rotation (II) (2016-04-18 23:26)

April 18, 2016

Rotate array - HackerRank

problem statement:

[1]<https://www.hackerrank.com/challenges/matrix-rotation-algo>

Study other's code:

[2]<https://gist.github.com/jianminchen/c234a238ea331ea9a037d603c37a6649>

So, do not swap two nodes value, just save first node's value in a variable, and then, shift array kind operation.

**Write C # solution:**

another practice: (more than 1 hour, still having bugs, score 8.89/ wrong answer)

[3]<https://gist.github.com/jianminchen/57572227d4fe939060f7cc81b193cd9b>

**Time spent:**

**More than 2 hours**

1. <https://www.hackerrank.com/challenges/matrix-rotation-algo>
  2. <https://gist.github.com/jianminchen/c234a238ea331ea9a037d603c37a6649>
  3. <https://gist.github.com/jianminchen/57572227dafa939060f7cc81b193cd9b>
- 

### **HackerRank: Matrix Rotation (III) - using extra space - an array (2016-04-18 23:29)**

April 18, 2016

Rotate array - HackerRank

problem statement:

[1]<https://www.hackerrank.com/challenges/matrix-rotation-algo>

Study other's code:

[2]<https://gist.github.com/jianminchen/9005cbbbbb60e307759747856e3a35a>

Write a solution using the same idea. Try to practice 20 - 30 minutes.

Just be simple; Read other people's code, like it; and then, write exactly same idea, and see if you can make it work as well.

**Write same algorithm again and again, use various ideas; same idea, write again and again, see if you can improve the performance, more concentration - write the program from hours to 30 minutes, less than 10 minutes. Practice makes perfect .**

To be continued:

**Action item:**

Will write my own version of implementation.

1. <https://www.hackerrank.com/challenges/matrix-rotation-algo>
2. <https://gist.github.com/jianminchen/9005cbbbbb60e307759747856e3a35a>

---

## video: How to Keep Calm at Crunch Time - Ask Ian #31 (2016-04-19 21:35)

April 19, 2016

To write code for a competition/ assessment is to like to play a sports match. Julia still has to work on mental toughness, and improve her probability to pass any code assessment. How to keep calm in crucial moment?

Spend a lot of hours to work on techniques, practice; but in the crucial moment, allow the pressure take control, and cannot execute as normal.

**How to do your best? practice, continue to improve and be in present.** It takes a lot of discipline. A lot of experience under the situation. Be in present, problem solving.

Take notes from the talk in the video:

Key #1:

1. **Be present** . Do not think about past, do not think about future;

Do not allow yourself travel time line; What if you lose, do not take it too seriously; do not focus on the future outcome. Second guess themselves, it is not good.

Focus on present, problem needs to be solved.

2. **Learn to embrace your nerves and anxieties.** Everybody gets nervous, does not matter how good you are, how long you play. Every one has anxiety, every one has fears over winning or losing. It is natural response if you are put in the competition.

As a matter of fact, it is a great thing to feel nervous and anxiety. This shows that you care about outcome. Use it to sharpen your focus, sharpen your perform.

Do not put yourself in a downward spiral.

video link:

[1]<https://www.youtube.com/watch?v=t8KjshrxS48>

Arguments:

Amazon hiring bar: better coding than 50 % existing employees. Code assessment is just a basic skills indicator. Learn from the experience.

[2]<https://goo.gl/HkHyFV>

**Always memorize a bible verse to calm down :**

How precious to me are Your thoughts, O God! How vast is the sum of them! Were I to count them, they would outnumber the grains of sand.

Psalm139 17 -18 NIV

June 14, 2016

Study 2 hours on the blog:

[3]<https://www.quora.com/How-can-I-get-over-my-failed-interview-at-Amazon>

Study the article - July 4, 2016

[4]<http://firstround.com/review/Mechanize-Your-Hiring-Process-to-Make-Better-Decisions/>

1. <https://www.youtube.com/watch?v=t8KjshrxS48>

2. <https://goo.gl/HkHyFV>

3. <https://www.quora.com/How-can-I-get-over-my-failed-interview-at-Amazon>

4. <http://firstround.com/review/Mechanize-Your-Hiring-Process-to-Make-Better-Decisions/>

---

## HackerRank: facts to share (2016-04-19 23:28)

April 19, 2016

Read one more blog:

[1]<http://blog.hackerrank.com/dreams-silicon-valley/>

Notes:

HackerRank is a great platform — the challenges are approachable and related to real-world problems. The best part about it, though, is that there are some great coders who compete on HackerRank, so **you can evaluate how you're compared with e.g. top 1 % of hackers** . It's very motivating and inspiring — in my case it was enough to make me solve a majority of problems listed there initially.

Some people might be drawn to Silicon Valley to chase their dreams of money and fame. But for me, my dream was as simple as making it there, to be surrounded by the best talent in the industry.

More reading:

[2]<http://goo.gl/m6jcFE>

1. <http://blog.hackerrank.com/dreams-silicon-valley/>

2. <http://goo.gl/m6jcFE>

---

## C# tutorial - reading first, coding follows (2016-04-21 20:55)

April 21, 2016

Julia found something she likes - C # tutorial

read it every day when you have 20 minutes in the morning.

[1] [https://msdn.microsoft.com/en-us/library/aa288436\(v=vs.71\).aspx](https://msdn.microsoft.com/en-us/library/aa288436(v=vs.71).aspx)

Julia likes to make C # her professional language - 35 % of programmer are using C #, 40 % of programmer are using C++, 40 % are using Java; Some of them use more than 1 programming language.

Get more tips to help C # learning, help fast coding.

1. [https://msdn.microsoft.com/en-us/library/aa288436\(v=vs.71\).aspx](https://msdn.microsoft.com/en-us/library/aa288436(v=vs.71).aspx)

---

## K nearest point (2016-04-21 21:04)

April 21, 2016

Try to find similar algorithm in HackerRank about K nearest point, and get some workout on coding.

Read the blog:

[1][https://www.glassdoor.co.in/Interview/Given-a-list-of-points-in-2D-and-a-single-reference-point-find-k-nearest-neighbors-No-code-required-iirc-QT\\_N\\_296848.htm](https://www.glassdoor.co.in/Interview/Given-a-list-of-points-in-2D-and-a-single-reference-point-find-k-nearest-neighbors-No-code-required-iirc-QT_N_296848.htm)

[2]<https://www.careercup.com/question?id=4751976126480384>

[3]<https://www.quora.com/What-are-the-some-of-the-problems-on-the-heap-priority-queue-on-SPOJ-HackerRank-CodeChef-Codeforces-or-HackerEarth>

Julia, try to work on one of 4 questions using priority queue/ heap:

[4]<https://www.hackerearth.com/code-monk-heaps-and-priority-queues/judge/>

Read the tutorial: (Excellent reading time - 1 hour)

[5]<https://www.hackerearth.com/notes/heaps-and-priority-queues/>

Take some notes:

Heapify process, using array to store the heap, amortized heapify analysis -  $O(N)$  instead of  $O(n \log N)$

Many ways to implement the **priority queue** .

Naive approach to build priority queue:

1. a list, sorted them, so  $O(N \log N)$  time

Efficient approach:

using heaps to implement the priority queue. It will take  $O(\log N)$  to insert and delete each element in the priority queue.

1. [https://www.glassdoor.co.in/Interview/Given-a-list-of-points-in-2D-and-a-single-reference-point-find-k-nearest-neighbors-No-code-required-iirc-QTN\\_296848.htm](https://www.glassdoor.co.in/Interview/Given-a-list-of-points-in-2D-and-a-single-reference-point-find-k-nearest-neighbors-No-code-required-iirc-QTN_296848.htm)
  2. <https://www.careercup.com/question?id=4751976126480384>
  3. <https://www.quora.com/What-are-the-some-of-the-problems-on-the-heap-priority-queue-on-SPOJ-HackerRank-CodeChef-Codeforces-or-HackerEarth>
  4. <https://www.hackerearth.com/code-monk-heaps-and-priority-queues/judge/>
  5. <https://www.hackerearth.com/notes/heaps-and-priority-queues/>
- 

## Window minimum (2016-04-21 21:05)

April 21, 2016

Try to get some algorithm reading and then find a similar algorithm in HackerRank, and get some workout on coding.

It is called Window minimum.

problem statement from the blog:

[1]<http://www.geeksforgeeks.org/amazon-interview-set-8-2/>

Given an array of size  $N$ , a window of size  $W$  slides over it by increment of slide  $S$ . If the window reaches to the end, we should stop there. Find a formula in form of  $N, S, W$  so that we can find the number of valid windows. Write a program to find minimum in every window and print it. Optimize it.

e.g.  $\{1, 2, 3, 4, 5\}$ ,  $W=2$ ,  $S=1$

first window:  $\{1, 2\}$  min=1

second window(increment by  $S=1$ ):  $\{2, 3\}$ , min=2

...

last window:  $\{4, 5\}$ , min=4

The array might not be sorted. I have taken sorted array for simplicity.

or:

??????:

????ArrayList<Integer> arr = new ArrayList<Integer>();  
arr.add(4); arr.add(2); arr.add(12); arr.add(11); arr.add(-5);  
int size = arr.size();  
for (int i = 0; i < size; i++) {  
 System.out.print(arr.get(i) + " ");  
}

Leetcode 239: <https://leetcode.com/problems/sliding-window-maximum/>

check solution:

[2]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/239.sliding-window-maximum.java>

1. <http://www.geeksforgeeks.org/amazon-interview-set-8-2/>

2. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/239.sliding-window-maximum.java>

---

## Matrix shortest distance set - get its Maximum value (2016-04-21 21:06)

April 21, 2016

Try to work on this algorithm, including problem statement, algorithm involved, and then, get similar question on HackerRank, and do some practice.

?????????:

`int[][] matrix; path_p_im_i_m_i`

---

## Find optimal weight (2016-04-21 21:07)

April 21, 2016

Find optimal weight - an algorithm to study

---

## Two rectangles to see if they overlap (2016-04-21 21:08)

April 21, 2016

Work on algorithm called Two rectangle to see if they overlap.

[1]<http://www.geeksforgeeks.org/find-two-rectangles-overlap/>

[2]<http://stackoverflow.com/questions/306316/determine-if-two-rectangles-overlap-each-other>

1. <http://www.geeksforgeeks.org/find-two-rectangles-overlap/>

2. <http://stackoverflow.com/questions/306316/determine-if-two-rectangles-overlap-each-other>

---

## Insert value into a circle linked list (2016-04-21 21:09)

April 21, 2016

Work on an algorithm called "Insert value into a circle linked list"

[1]<http://www.geeksforgeeks.org/sorted-insert-for-circular-linked-list/>



another problem:

Insert into a cyclic sorted list:

[2]<http://articles.leetcode.com/insert-into-a-cyclic-sorted-list/>

### Question and Answer:

#### 1. Three cases:

1.  $\text{prev} \rightarrow \text{val} \leq x \leq \text{current} \rightarrow \text{val}$ :

Insert between prev and current.

2. x is the maximum or minimum value in the list:

Insert before the head. (ie, the head has the smallest value and its  $\text{prev} \rightarrow \text{val} > \text{head} \rightarrow \text{val}$ ).

3. Traverses back to the starting point:

Insert before the starting point.

2. Let us walk through the code, and put some comment:

```
void
insert
(
Node *
&
aNode
,
int
x
)
{
if
(
!
aNode
)
{
// Julia: if the node is empty, then, create one, link to itself to form a loop
```

```
aNode
=
new
Node
(
x
)
;
aNode
->
next
=
aNode
;
return
;
}
Node *
p
=
aNode
; // Julia:
two pointers, one is current, one is previous, get into a loop
Node *
prev
=
NULL
;
do
{
prev
=
p
;
p
= p->next
}
```

```
=  
p  
->  
next  
;
```

```
int[] arr = new int[2] {prev->data, p->data }; //
```

**Julia: add some explanation variable**

```
if  
(  
x  
>=  
arr[0] & &  
x  
<=  
arr[1]  
)  
break  
;  
// For case 1)  
if  
(  
(  
arr[0]  
>  
arr[1]  
)  
& &  
(  
x  
>  
arr[0] ||  
x  
<  
arr[1]
```

```

)
)
break
;
// For case 2)
}
while
(
p
!=
aNode
)
;
// when back to starting point, then stop. For case 3)
Node *
newNode
=
new
Node
(
x
)
;
newNode
->
next
=
p
;
prev
->
next
=
newNode
;
}

1. http://www.geeksforgeeks.org/sorted-insert-for-circular-linked-list/
2. http://articles.leetcode.com/insert-into-a-cyclic-sorted-list/

```

## Leetcode 17 Phone number combination - practice (II) (2016-04-21 21:10)

April 26, 2016

To master this DFS algorithm, Julia likes to do more practice. Every time she practices, she finds new issue, new concern. She also knows that to memorize a solution is not a good idea, do not focus on what you remember about the algorithm, even you have some memory about the algorithm and solution. You may have to solve a totally different problem.

### Leetcode 17 phone number combination

Review the blog:

[1] <http://juliachencoding.blogspot.ca/2016/01/leetcode-17-letter-combinations-of.html>

One more practice - write the code using the same idea, still makes a bug, and learn how to do fast coding, add more explanation variable, and temporary variable.

[2] <https://gist.github.com/jianminchen/acbbf5fbd5fefa1a9fb2f7b3664bf4ed>

[3] <https://gist.github.com/jianminchen/90ac4390929704ce439d0a1f255f4546>

Make function more flat, close to left side, more readable - **remove if/ else statement in the function, more flat; also, more readable code, less chance to have a bug** .

[4] <https://gist.github.com/jianminchen/b866a5c331556ea8f36aecfe123bfeea>

### Reading:

[5] <https://msdn.microsoft.com/en-us/library/zcbcf4ta.aspx>

[6] <http://geekswithblogs.net/robp/archive/2008/08/13/speedy-c-part-3-understanding-memory-references-pinned-objects-and.aspx>

### Question and Answer:

Julia has some concerns about recursive function, she will find some articles to read:

1. General about stack, recursive function.
2. How to check using C # - recursive argument one copy or its own copy?
3. Play safe, use extra variable to save the value.

1. <http://juliachencoding.blogspot.ca/2016/01/leetcode-17-letter-combinations-of.html>

2. <https://gist.github.com/jianminchen/acbbf5fbd5fefa1a9fb2f7b3664bf4ed>

3. <https://gist.github.com/jianminchen/90ac4390929704ce439d0a1f255f4546>

4. <https://gist.github.com/jianminchen/b866a5c331556ea8f36aecfe123bfeea>

5. <https://msdn.microsoft.com/en-us/library/zcbcf4ta.aspx>

6. <http://geekswithblogs.net/robp/archive/2008/08/13/speedy-c-part-3-understanding-memory-references-pinned-objects-and.aspx>

## Search in Matrix (2016-04-21 21:11)

April 21, 2016

Work on algorithm called "Search in Matrix".

?????:

<https://leetcode.com/problems/search-a-2d-matrix-ii/>

Read the blog:

[1]<http://noalgo.info/397.html>

1. <http://noalgo.info/397.html>

---

## Two sum pair (2016-04-21 21:11)

April 21, 2016

Work on algorithm called two sum pair.

Problem statement:

[1]<https://www.careercup.com/question?id=12887674>

Given an array. Find pairs of numbers that add up to a given value 'x'. with time complexity less than  $O(n^2)$  and use no additional space.

Also, Array Pair Sum, look up the google:

[2]<http://www.ardendertat.com/2011/09/17/programming-interview-questions-1-array-pair-sum/>

work on this hackerRank problem to entertain the time:

[3]<https://www.hackerrank.com/contests/infinitem8/challenges/pairwise-sum-and-divide>

Moderate level

[4]<https://www.hackerrank.com/challenges/sherlock-and-pairs>

1. <https://www.careercup.com/question?id=12887674>

2. <http://www.ardendertat.com/2011/09/17/programming-interview-questions-1-array-pair-sum/>

3. <https://www.hackerrank.com/contests/infinitem8/challenges/pairwise-sum-and-divide>

4. <https://www.hackerrank.com/challenges/sherlock-and-pairs>

---

## Merge two sorted linked list (2016-04-21 21:12)

April 21, 2016

Work on algorithm called "Merge two sorted linked list".

Recursive call:

[1]<http://stackoverflow.com/questions/10707352/interview-merging-two-sorted-singly-linked-list>

[2]<http://www.geeksforgeeks.org/merge-two-sorted-linked-lists/>

1. <http://stackoverflow.com/questions/10707352/interview-merging-two-sorted-singly-linked-list>

2. <http://www.geeksforgeeks.org/merge-two-sorted-linked-lists/>

---

## longest palindrome substring (2016-04-21 21:13)

April 21, 2016

Work on algorithm called "longest palindrome substring".

Review the blog:

[1]<http://juliachencoding.blogspot.ca/2015/06/leetcode-longest-palndromic-substring.html>

August 8, 2016

Java practice:

[2]<https://gist.github.com/jianminchen/ee7e6512cd8fbffb1895e51ebaa4487c>

1. <http://juliachencoding.blogspot.ca/2015/06/leetcode-longest-palndromic-substring.html>

2. <https://gist.github.com/jianminchen/ee7e6512cd8fbffb1895e51ebaa4487c>

---

## A binary tree is subtree of another binary tree (2016-04-21 21:14)

April 21, 2016

Work on algorithm called "Is subtree".

problem statement:

[1]<http://www.geeksforgeeks.org/check-if-a-binary-tree-is-subtree-of-another-binary-tree/>

Time Complexity: Time worst case complexity of above solution is  $O(mn)$  where  $m$  and  $n$  are number of nodes in given two trees.

Another solution:

[2]<http://www.geeksforgeeks.org/check-binary-tree-subtree-another-binary-tree-set-2/>

$O(n)$  time

Special case handling - for leaf node of tree - append null

Need to review KMP algorithm - strstr -  $O(N)$  algorithm:

[3]<http://www.geeksforgeeks.org/searching-for-patterns-set-2-kmp-algorithm-m/>

### Question and answers:

1. What do you learn through this study? How long does it take you to figure out things?

Answer: First, it is about the subtree definition: it should be uniquely defined; for any node in the tree, the subtree starting from the node is only one. In other words, the node is the start, and all leaf nodes underneath should all be included.

2. The very good way to think recursively; do not repeat the work, do not do the extra work; only work on root node, since every node can be root node; get in the loop or recursive function.

3. Let us walk through the code and add some comment:

/\*

**Function is designed to see if S is a subtree of T, S stands for subtree .**

\*/

boolean isSubtree(Node T, Node S) {

/\* base cases \*/

if

(S == null) { // Julia: subtree is null, assume that it is null is true for subtree.

return true;

}

if (T == null) {

return

false; // T is empty, then, return false.

}

if

(areIdentical(T, S)) { // Now, let us check if T and S are identical; if it is, then return true.

return true;

}

return

isSubtree(T.left, S) // otherwise, continue to check S with T's left and right child, ask same question.



```

|| isSubtree(T.right, S);
}
/*
Two trees are identical - check every node in the tree
/

boolean areIdentical(Node node1, Node node2) {
if
(node1 == null
& & node2 == null) { // base case: both trees are null, then true;
return true;
}
if
(node1 == null
|| node2 == null) { // afterwards, if any one is null, then, return false;
return false;
}
// check 3 things: first, two nodes are having same value;
// and then, check both left child;
// and then, check both right child.

return (node1.data == node2.data
& & areIdentical(node1.left, node2.left)
& & areIdentical(node1.right, node2.right));
}

1. http://www.geeksforgeeks.org/check-if-a-binary-tree-is-subtree-of-another-binary-tree/
2. http://www.geeksforgeeks.org/check-binary-tree-subtree-another-binary-tree-set-2/
3. http://www.geeksforgeeks.org/searching-for-patterns-set-2-kmp-algorithm/

```

---

## valid parenthesis pairs (2016-04-21 21:14)

April 21, 2016

Work on algorithm called "Valid parenthesis pairs".

Review blog:

[1]<http://juliachencoding.blogspot.ca/2016/01/leetcode-questions-20-valid-parentheses.html>

1. <http://juliachencoding.blogspot.ca/2016/01/leetcode-questions-20-valid-parentheses.html>

---

## gray code (2016-04-21 21:15)

April 21, 2016

Work on algorithm called "gray code".

1) Given two words, find if second word is the round rotation of first word.

For example: abc, cab

return 1

since cab is round rotation of abc

Example2: ab, aa

return -1

since aa is not round rotation for aa

2) Given two hexadecimal numbers find if they can be consecutive in gray code

For example: 10001000, 10001001

return 1

since they are successive in gray code

Example2: 10001000, 10011001

return -1

since they are not successive in gray code.

problem statement source:

[1]<http://www.geeksforgeeks.org/amazon-interview-experience-set-137-assessment-test-sde/>

Read the blog:

[2]<http://www.cnblogs.com/lautsie/p/3909927.html>

choose some questions to work on, practice!

[3]<http://www.cnblogs.com/lautsie/tag/hackerrank/>

try this one - see if the content is related to gray code

[4]<https://www.hackerrank.com/contests/w6/challenges/consecutive-subsequences>

Leetcode gray code:

[5]<https://leetcode.com/problems/gray-code/>

Two solutions:

[6]<http://bangbingsyb.blogspot.ca/2014/11/leetcode-gray-code.html>

[7]<http://fisherlei.blogspot.ca/2012/12/leetcode-gray-code.html>

### Question and answer time:

Q: Julia, tell me what you know about gray code.

Answer:

1. Gray code is very specific for the one to build up using previous results.

To extend, append all the number in the set in reverse order, with number adding  $2^n$ ,  $n$  is the number in binary format length.

2. Let us work on some coding:

$n = 0$ : 0

$n = 1$ :

0

, 1

$n = 2$ : 00, 01, 11, 10 ( 0, 1, 3, 2)

$n = 3$ : 000, 001, 011, 010, 110, 111, 101, 100 ( 0, 1, 3, 2, 6, 7, 5, 4)

C++:

```
vector
```

```
<
```

```
int
```

```
>
```

```
grayCode(
```

```
int
```

```
    n) {
```

```
vector
```

```
<
```

```

int
>
    greySeq;

if
(n
<
0
)
return
    greySeq; //

julia's comment:

    greySeq.push_back(
0
); //

first, start from n=0, 0

int
    inc
=

1

; //

incremental value is 1, 2^n, n =0;
372

```

```

for
(
int
    i
=
1
; i
<=
n; i
++

) { //
loop starting value: 1

```

```

for
(
int
    j
=
greySeq.size()
-
1
; j

```

```

>=

0

; j

--

) //

reverse order iteration

    greySeq.push_back(greySeq[j]

+

inc); //

add incremental value for each one

    inc

{\textless}{\textless}=

1

; //

inc = inc *2 or inc {\textless}{\textless}=1;

}

return

```

1. <http://www.geeksforgeeks.org/amazon-interview-experience-set-137-assessment-test-sde/>
2. <http://www.cnblogs.com/lautsie/p/3909927.html>
3. <http://www.cnblogs.com/lautsie/tag/hackerrank/>
4. <https://www.hackerrank.com/contests/w6/challenges/consecutive-subsequences>
5. <https://leetcode.com/problems/gray-code/>

6. <http://bangbingsyb.blogspot.ca/2014/11/leetcode-gray-code.html>
  7. <http://fisherlei.blogspot.ca/2012/12/leetcode-gray-code.html>
- 

## Remove Vowels From A String in Place (2016-04-21 21:15)

April 21, 2016

Work on algorithm called "Remove Vowels From A String in Place".

problem statement see the blog:

[1]<https://www.careercup.com/question?id=2698>

Take any word, remove vowels and place them in reverse order by untouched consonants and place them back. Word is AIRPLANE.

HackerRank has one algorithm, but only 4 submissions:

[2]<https://www.hackerrank.com/contests/knockout-coding/challenges/pair-odd>

Read the blog about "Remove vowels":

[3]<http://shashank7s.blogspot.ca/2011/03/wap-to-remove-remove-vowels-from-a-string.html>

[4]<https://www.quora.com/How-do-I-remove-vowels-from-a-string>

### Question and Answer:

Walk through the code, add comments:

```
1. const
   string vowels
   =
   "aeiou"
   ; // Julia: all vowels in a string

2.

3. void
   remVowels
   (
   string
   &
   s
   ){
```

```
4. int
   k
   =
   0
   ; // pointer: int k, mark the next available position for non vowel char
```

```
5. for
   (
   int
   i
   =
   0
   ;
   i
   <
   s
   .
   length
   ( );
   i
   ++ ) { // Julia: go through the string once, in a loop
```

```
6. if
   (
   vowels
   .
   find
   (
   s
   [
   i
   ])
   ==
   string
   ::
   npos
   ) // if current char is not a vowel
```



```

7. s
   [
   k
   ++]
   =
   s
   [
   i
   ];

8. // skip vowel, do nothing.

9. }

10. s
    .
    resize
    (
    k
    ); // mark the end of the string removing all vowels.

11. }

```

Read api how resize is defined in C++:

[5]<http://www.cplusplus.com/reference/string/string/resize/>

1. <https://www.careercup.com/question?id=2698>
2. <https://www.hackerrank.com/contests/knockout-coding/challenges/pair-odd>
3. <http://shashank7s.blogspot.ca/2011/03/wap-to-remove-remove-vowels-from-string.html>
4. <https://www.quora.com/How-do-I-remove-vowels-from-a-string>
5. <http://www.cplusplus.com/reference/string/string/resize/>

---

### Largest continuous sub array problem (2016-04-22 21:39)

April 22, 2016

Buy and sell stock - Leetcode question

[1]<http://www.geeksforgeeks.org/largest-sum-contiguous-subarray/>

It is Dynamic programming problem.

Check Kadane's algorithm:

**Question and Answer Time:**

Question: What do you learn through the study?

Answer: This is a dynamic programming problem. Use two variables to track the statistics:

One is called maximum value ending here. `max_ending_here`;

Another one is called maximum value so far. `max_so_far`.

Let us walk through the code:

```
int maxSubArraySum(int a[], int size)
{
    int max_so_far = 0, max_ending_here = 0;
    for (int i = 0; i < size; i++)
    {
        max_ending_here = max_ending_here + a[i];
        if (max_ending_here < 0)
            max_ending_here = 0;
        /* Do not compare for all elements. Compare only
        when max_ending_here > 0 */
        else if (max_so_far < max_ending_here)
            max_so_far = max_ending_here;
    }
    return max_so_far;
}
```

Handle the case when all numbers in the array are negative:

```
int maxSubArraySum(int a[], int size)
{
    int max_so_far = a[0];
    int curr_max = a[0];
    for (int i = 1; i < size; i++)
    {
        curr_max = max(a[i], curr_max+a[i]);
        max_so_far = max(max_so_far, curr_max);
    }
    return max_so_far;
}
```

1. <http://www.geeksforgeeks.org/largest-sum-contiguous-subarray/>

## Fisher-Yates algorithm (2016-04-22 22:45)

April 22, 2016

A deck of cards is represented by an integer array of size 52, with possible value of 1 - 13 for each position. Design an algorithm to shuffle the deck of cards in linear time, in-place.

[1][https://en.wikipedia.org/wiki/Fisher-Yates\\_shuffle](https://en.wikipedia.org/wiki/Fisher-Yates_shuffle)

April 26, 2016

[2]<http://www.geeksforgeeks.org/shuffle-a-given-array/>

“shuffle a deck of cards” or “randomize a given array”

### Question and answer:

1. Understand the naive approach, extra space - an array, and also time  $O(n^2)$ .
2. Came cross this blog <- So, use **Fisher-Yates algorithm**

????

???array??-H??n?????,?????-????-????index??-Я????????index??

???array?[5,1,3,3],

????n?2?

???array??-H??num????[5,1] or [5,3] or [1,3] or [3,3] ?????[5,5]

???[5,1,3,3], 1 ==> [5] or [1] or [3]

????n>array.length?-0??array??,????array

????loop????-?R??random?index????-P??hashset????index????random index????get????-R??random? index?

loop n????array??

.????-e??-????

????-4????boolean array????-£

????unit test????[5,1]???[5,1,3,3]??

Julia's comment:

3 issues:

1. In place - no extra array

## 2. Hashset is also extra memory

### 3. Work on array index - random number selection using % operator , not on value in the array.

3. Let us walk through the code and then add comments - make it fun memory:

```
// A function to generate a random permutation of arr[]
```

```
void randomize ( int arr[], int n )
```

```
{
```

```
// Use a different seed value so that we don't get same
```

```
// result each time we run this program
```

```
srand
```

```
( time (NULL) ); //
```

**Julia, Good to know, C++, look up srand API**

```
// Start from the last element and swap one by one. We don't
```

```
// need to run for the first element that's why i > 0
```

```
for (int i = n-1; i > 0; i--)
```

```
{
```

```
// Pick a random index from 0 to i
```

```
int
```

```
j = rand () % (i+1); // Julia: random number -> Range (0, i) using module
```

```
// Swap arr[i] with the element at random index
```

```
swap( &arr[i], &arr[j]); // swap two nodes
```

```
}
```

```
}
```

Read C++ API:

[3]<http://www.cplusplus.com/reference/cstdlib/srand/>

[4] <http://www.cplusplus.com/reference/cstdlib/rand/>

1. [https://en.wikipedia.org/wiki/Fisher%E2%80%93Yates\\_shuffle](https://en.wikipedia.org/wiki/Fisher%E2%80%93Yates_shuffle)

2. <http://www.geeksforgeeks.org/shuffle-a-given-array>

3. <http://www.cplusplus.com/reference/cstdlib/srand/>

4. <http://www.cplusplus.com/reference/cstdlib/rand/>

## **Tortoise-hare algorithm (2016-04-22 22:47)**

April 22, 2016

Find if a linked list has a cycle. Expectation  $O(n)$  time;  $O(1)$  space.

[1][http://codingfreak.blogspot.com/2012/09/detecting-loop-in-singly-linked-list\\_22.html](http://codingfreak.blogspot.com/2012/09/detecting-loop-in-singly-linked-list_22.html)

[2]<https://www.quora.com/How-does-Floyds-cycle-finding-algorithm-work>

1. [http://codingfreak.blogspot.com/2012/09/detecting-loop-in-singly-linked-list\\_22.html](http://codingfreak.blogspot.com/2012/09/detecting-loop-in-singly-linked-list_22.html)

2. <https://www.quora.com/How-does-Floyds-cycle-finding-algorithm-work>

---

## **Find the lowest common ancestor of two nodes in a Binary Tree. (2016-04-22 22:48)**

April 22, 2016

Find the lowest common ancestor of two nodes in a Binary Tree. The tree does not have a parent pointer. Your algorithm should run in linear time, without any extra space.

Bottom-up approach.

[1]<http://www.geeksforgeeks.org/lowest-common-ancestor-binary-tree-set-1/>

HackerRank problem:

[2]<https://www.hackerrank.com/challenges/binary-search-tree-lowest-common-ancestor>

[3]<https://www.topcoder.com/community/data-science/data-science-tutorials/range-minimum-query-and-lowest-common-ancestor/>

Review blog:

[4]<http://juliachencoding.blogspot.ca/2015/07/leetcode-lowest-common-ancestor-in.html>

1. <http://www.geeksforgeeks.org/lowest-common-ancestor-binary-tree-set-1/>

2. <https://www.hackerrank.com/challenges/binary-search-tree-lowest-common-ancestor>

3. <https://www.topcoder.com/community/data-science/data-science-tutorials/range-minimum-query-and-lowest-common-ancestor/>

4. <http://juliachencoding.blogspot.ca/2015/07/leetcode-lowest-common-ancestor-in.html>

---

## Find if a Directed Acyclic Graph has a cycle. (2016-04-22 22:50)

April 22, 2016

Find if a Directed Acyclic Graph has a cycle. Use DFS with coloring to find if cycle exists.

Given a directed graph of 10 cities, each of which may or may not be connected to each other, represented by an adjacency list, write an algorithm to find if there is a write from a city that eventually cycles back to the same city. What is the time complexity of your algorithm? You may use reasonable extra storage, or modify the structure of the graph node class.

### Question and Answers:

Julia, read a few blogs, and then quickly go over first time. 30 minutes:

1. [1]<https://www.quora.com/How-can-DFS-and-BFS-be-used-to-find-out-if-there-are-cycles-in-a-graph>

2. [2]<http://www.geeksforgeeks.org/detect-cycle-in-a-graph/>

Take notes here:

Depth First Traversal can be used to detect cycle in a Graph. DFS for a connected graph produces a tree. There is a cycle in a graph only if there is a [3]back edge present in the graph. A back edge is an edge that is from a node to itself (selfloop) or one of its ancestor in the tree produced by DFS.

In other words, Julia, using DFS, to detect the cycle;

DFS for a connected graph produces a tree; So, from **graph -> DFS -> Tree** ;

Then, work on Tree, to see if there is a back edge present in the tree;

What is back edge? How to track? ancestor node in a stack - using array.

### Question and Answer:

Walk through the code, and add the comment:

[4]<https://gist.github.com/jianminchen/2eea7cc7cca69f296fc105c3fc3faafa>

Read another blog:

[5]<http://www.geeksforgeeks.org/depth-first-traversal-for-a-graph/>

Read another blog from HackerRank:

[6]<https://www.hackerrank.com/topics/topological-sorting>

### Actionable item:

Work on HackerRank graph problem today:

[7]<https://www.hackerrank.com/challenges/even-tree>

And then, study as many as possible solution about this graph problem. Get ideas how people are talented on graph problem solving.

1. <https://www.quora.com/How-can-DFS-and-BFS-be-used-to-find-out-if-there-are-cycles-in-a-graph>
  2. <http://www.geeksforgeeks.org/detect-cycle-in-a-graph/>
  3. [http://en.wikipedia.org/wiki/Depth-first\\_search#Output\\_of\\_a\\_depth-first\\_search](http://en.wikipedia.org/wiki/Depth-first_search#Output_of_a_depth-first_search)
  4. <https://gist.github.com/jianminchen/2eea7cc7cca69f296fc105c3fc3faafa>
  5. <http://www.geeksforgeeks.org/depth-first-traversal-for-a-graph/>
  6. <https://www.hackerrank.com/topics/topological-sorting>
  7. <https://www.hackerrank.com/challenges/even-tree>
- 

## HackerRank: Even Tree - Graph Problem (I) - Just thinking (2016-04-23 14:46)

April 23, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/even-tree>

Motivation to work on graph problem:

1. There are over 10,000 submission on this problem, definitely, Julia likes to give it a try.

### Statistics:

1:20 - 2:30pm more than 1 hour wild thinking.

Think about the graph problem, before Julia writes any code, she think about how the problem is developed:

1. Have to remove as many edges from the tree as possible; <- kind of greedy algorithm
2. Each connected component of the forest should contain an even number of vertices.  
<- Otherwise, odd number, 1 node itself, cannot be called forest; at least, 3.
3. Think about the how to remove one edge:  
both ends should be ok; if one end only has one edge, then, no for removing the edge.  
so, the node checks its edge count:  
= 0 <- not possible - > "even number of vertices"  
= 1 <- impossible  
= 2 <- possible, can keep one, remove one

But think about forest with even number 2, 4,

for number 2, only one case: 3 connects to 4 in the test case

for number 4, only one case: one node in the center, all other 3 are connected to the same one.

for number 6, only one case: one node in the center, all other 5 are connected to the same one;

otherwise, one node connects to other 4 nodes, the 6th one will connect to the one of the node (other 4 nodes), then, it can be break into 2 and 4.

So, it should be two checking:

1. if the node only has one edge, keep it; do nothing;
2. if the node has more than one edge, remove all edges with more than 1 count.

### **Add comment on 10:00pm on April 23, 2016**

1. Julia, you took distributed system in Florida Atlantic University around 2001, talking about spanning tree, ad hoc routing protocol; one thing is about spanning tree. Use DFS, and then, discuss spanning tree; if the root node's child has a tree with even length, then, edge can be removed.

1. <https://www.hackerrank.com/challenges/even-tree>

---

### **HackerRank: Even Tree - Graph Problem (II) - Coding first try (2016-04-23 14:59)**

April 23, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/even-tree>

**Time spent:**

over 1 hour,

score 10 out of 50

pass two basic cases, but with run time error after submission on other test cases

[2]<https://gist.github.com/jianminchen/455978d2fd4b1ebafb3a5aa7d4761fed>

1. <https://www.hackerrank.com/challenges/even-tree>

2. <https://gist.github.com/jianminchen/455978d2fd4b1ebafb3a5aa7d4761fed>

---

### **HackerRank: Even Tree - C# solutions to study (III) (2016-04-23 17:23)**

April 23, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/even-tree>



C # solutions to study:

[2]<https://www.hackerrank.com/challenges/even-tree/leaderboard/filter/language=csharp>

1.

[3]<https://gist.github.com/jianminchen/1858f0b4994efc65c99701290fb951ec>

2. written by an engineer in Microsoft

[4]<https://gist.github.com/jianminchen/5c6b6cb4de9c4a2faf6237b7ac331a41>

3.

[5]<https://gist.github.com/jianminchen/3722227eb8c3df4085c10e34779763bf>

4. By an engineer in Amazon

[6]<https://gist.github.com/jianminchen/8359a5d4f2567db387c0dd80a0f70513>

Julia, practice the above 4 solutions. Write it by yourself one by one, and count the time. Work on speed.

1. <https://www.hackerrank.com/challenges/even-tree>

2. <https://www.hackerrank.com/challenges/even-tree/leaderboard/filter/language=csharp>

3. <https://gist.github.com/jianminchen/1858f0b4994efc65c99701290fb951ec>

4. <https://gist.github.com/jianminchen/5c6b6cb4de9c4a2faf6237b7ac331a41>

5. <https://gist.github.com/jianminchen/3722227eb8c3df4085c10e34779763bf>

6. <https://gist.github.com/jianminchen/8359a5d4f2567db387c0dd80a0f70513>

---

## HackerRank: Even Tree (V) C# solution - use queue to help counting spanning tree's node (2016-04-23 22:05)

April 23, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/even-tree>

C # solutions to study:

Code to study:

written by an engineer in Microsoft

[2]<https://gist.github.com/jianminchen/5c6b6cb4de9c4a2faf6237b7ac331a41>

Julia modified the C # code to use DFS, spanning tree's node count, but still got the run time error/ wrong answer.

[3]<https://gist.github.com/jianminchen/fe4da18247e4ad712c8e0bcb846983eb>

Later, find out what is wrong. ( Probably, here is the reason:

**The edge is directed.**

**Do not duplicate the edge.**

**For example, 2 1, just 2->1, do not add 1->2;**

**but**

**vertex[1].add(2),**

**vertex[2].add(1),**

**edge just add Tuple(2,1) )**

Now, write exactly same code using study code, and see what I can learn.

The following code passes all the test cases.

[4]<https://gist.github.com/jianminchen/0e40bb76011e60f8aaf4683ed9c9c3d6>

**Julia's comment:**

The edge is directed. Do not duplicate the edge and save two copy of the edge. But, vertex list is added for both vertexes of one edge.

**Follow up** - on April 24, 2016 8:40pm

Read Graph Topological Sorting:

[5]<http://www.geeksforgeeks.org/topological-sorting/>

**Follow up** - on April 30, 2016 11:52pm

Read graph representation

Use Adjacency Matrices

Use Adjacency Lists

[6]<https://www.khanacademy.org/computing/computer-science/algorithms/graph-representation/a/representing-graphs>

Notes from the blog:

How much space do adjacency lists take? We have

[MATH:  $|V|$  :MATH]

?

V

?

lists, and although each list could have as many as

[MATH:  $|V|$  :MATH]

?

V

?

-

1

vertices, in total the adjacency lists for an undirected graph contain

[MATH: <semantics><annotation encoding="application/x-tex">2|E| </annotation></semantics> :MATH]

2

?

E

?

elements. Why

[MATH: <semantics><annotation encoding="application/x-tex">2|E| </annotation></semantics> :MATH]

2

?

E

?

? Each edge

[MATH: <semantics><annotation encoding="application/x-tex">(i,j)</annotation></semantics> :MATH]

(

i

,

j

)

appears exactly twice in the adjacency lists, once in

[MATH: <semantics><annotation encoding="application/x-tex">i</annotation></semantics> :MATH]

i

's list and once in

[MATH: <semantics><annotation encoding="application/x-tex">j</annotation></semantics> :MATH]

j

's list, and there are

[MATH: <semantics><annotation encoding="application/x-tex">|E| </annotation></semantics> :MATH]

?

E

?

edges. For a directed graph, the adjacency lists contain a total of

[MATH: <semantics><annotation encoding="application/x-tex">|E| </annotation></semantics> :MATH]

?

E

?

elements, one element per directed edge.

1. <https://www.hackerrank.com/challenges/even-tree>
  2. <https://gist.github.com/jianminchen/5c6b6cb4de9c4a2faf6237b7ac331a41>
  3. <https://gist.github.com/jianminchen/fe4da18247e4ad712c8e0bcb846983eb>
  4. <https://gist.github.com/jianminchen/0e40bb76011e60f8aaf4683ed9c9c3d6>
  5. <http://www.geeksforgeeks.org/topological-sorting/>
  6. <https://www.khanacademy.org/computing/computer-science/algorithms/graph-representation/a/representing-graphs>
- 

## Back tracking - N Queen problem (2016-04-24 09:50)

April 24, 2016

Review N Queen problem, this one makes me easy to follow:

[1]<http://www.geeksforgeeks.org/backtracking-set-3-n-queen-problem/>

My own blog:

[2][http://juliachencoding.blogspot.ca/search/label/N-Queens %20problem](http://juliachencoding.blogspot.ca/search/label/N-Queens%20problem)

Find one problem on HackerRank:

[3]<https://www.hackerrank.com/challenges/queens-on-board>

1. <http://www.geeksforgeeks.org/backtracking-set-3-n-queen-problem/>
  2. <http://juliachencoding.blogspot.ca/search/label/N-Queens%20problem>
  3. <https://www.hackerrank.com/challenges/queens-on-board>
- 

## Leetcode 314: Binary Tree Vertical Order Traversal (2016-04-24 09:57)

April 24, 2016

Review the algorithm:

Leetcode 314: Binary Tree Vertical Order Traversal

blog:

388

[1]<http://juliachencoding.blogspot.ca/2016/01/leetcode-314-binary-tree-vertical-order.html>

Read the blog:

[2]<http://www.geeksforgeeks.org/print-binary-tree-vertical-order/>

1. <http://juliachencoding.blogspot.ca/2016/01/leetcode-314-binary-tree-vertical-order.html>
  2. <http://www.geeksforgeeks.org/print-binary-tree-vertical-order/>
- 

## Articles to read - big O cheat sheet (2016-04-24 21:51)

April 24, 2016

[1]<http://bigocheatsheet.com/>

[2]<https://gist.github.com/jianminchen/f99280129501be2a702d95882cd62459>

[3]<http://goo.gl/OJYsZW>

1. <http://bigocheatsheet.com/>
  2. <https://gist.github.com/jianminchen/f99280129501be2a702d95882cd62459>
  3. <http://goo.gl/OJYsZW>
- 

## Skip List (2016-04-24 22:05)

April 24, 2016

[1]<http://www.geeksforgeeks.org/skip-list/>

Plan to spend next 2 weeks to go over the solution provided by Temple university Ph.D. Dawei Li

[2]<https://github.com/jianminchen/LeetCode-Java-Solutions>

Some leetcode questions to work on first:

1. reverse half linked list
2. priority queue
3. number of island
4. flying ticket
5. unsorted slots with numbers
6. Sum query 2nd mutable

7. word search in matrix
8. course schedule

1. <http://www.geeksforgeeks.org/skip-list/>
  2. <https://github.com/jianminchen/LeetCode-Java-Solutions>
- 

## Leetcode 235: Lowest common ancestor in binary search tree (2016-04-25 18:21)

April 25, 2016

Study the code:

[1][https://github.com/douglasleer/LeetCode-Java-Solutions/blob/master/235\\_lowest-common-ancestor-of-a-binary-search-tree.java](https://github.com/douglasleer/LeetCode-Java-Solutions/blob/master/235_lowest-common-ancestor-of-a-binary-search-tree.java)

Walk through less than 10 lines of code, and then add some comment:

```
public
TreeNode
lowestCommonAncestor
(
    TreeNode
    root
    ,
    TreeNode
    p
    ,
    TreeNode
    q
) {
    if
    (p
    .
    val
    >
    390
```

```

q
.
val)
return
lowestCommonAncestor(root, q, p); // Julia, reduce 2 cases to 1 case: p.val <= q.val
if
(q
.
val
<
root
.
val)
return
lowestCommonAncestor(root
.
left, p, q); // two nodes are in the left side of root
if
(p
.
val
>
root
.
val)
return
lowestCommonAncestor(root
.
right, p, q); // two nodes are in the right side of root
return
root; //
otherwise, one node is left side, another node is right side: root is the parent.
}

```

1. <https://github.com/douglasleer/LeetCode-Java-Solutions/blob/master/235.lowest-common-ancestor-of-a-binary-search-tree.java>

## Leetcode 236: Lowest common ancestor in binary tree (2016-04-25 18:23)

April 25, 2016

Study the code:

[1][https://github.com/douglasleer/LeetCode-Java-Solutions/blob/master/236\\_lowest-common-ancestor-of-a-binary-tree.java](https://github.com/douglasleer/LeetCode-Java-Solutions/blob/master/236_lowest-common-ancestor-of-a-binary-tree.java)

Julia's C # warm up practice:

1. using Stack class ToArray API, keep the iterator order - stack output order: (5/21/2016)

[2]<https://gist.github.com/jianminchen/2bec8a70ef520e715240a226c01c7371>

Read Stack class APIs:

[3][https://msdn.microsoft.com/en-us/library/415129w1\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/415129w1(v=vs.110).aspx)

2. Use List.AddRange(Stack), still keep the stack iterator order

[4]<https://gist.github.com/jianminchen/754f45023f60992211cb34bef6d6254f>

Compared to Java Stack class APIs later.

### Question and answer:

1. What do you learn through the study of the code?

The code is using Binary Tree post order traversal method, and then, to find node p and q common ancestor, work on the stack, get list of nodes from root node to p, same to q, and then, find the common ancestor.

Blog:

[5]<http://juliachencoding.blogspot.ca/2015/08/tree-algorithms-review.html>

Review:

Post order traversal iterative

[6]<https://github.com/jianminchen/leetcode-tree/blob/master/TreeDemo.cs>

Write a C # version using exactly same idea.

1. [https://github.com/douglasleer/LeetCode-Java-Solutions/blob/master/236\\_lowest-common-ancestor-of-a-binary-tree.java](https://github.com/douglasleer/LeetCode-Java-Solutions/blob/master/236_lowest-common-ancestor-of-a-binary-tree.java)
2. <https://gist.github.com/jianminchen/2bec8a70ef520e715240a226c01c7371>
3. [https://msdn.microsoft.com/en-us/library/415129w1\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/415129w1(v=vs.110).aspx)
4. <https://gist.github.com/jianminchen/754f45023f60992211cb34bef6d6254f>
5. <http://juliachencoding.blogspot.ca/2015/08/tree-algorithms-review.html>
6. <https://github.com/jianminchen/leetcode-tree/blob/master/TreeDemo.cs>



## Leetcode 266: Palindrome permutation (2016-04-25 18:32)

April 25, 2016

Study the problem/ solution:

[1]<https://github.com/douglasleer/LeetCode-Java-Solutions/blob/master/266.palindrome-permutation.java>

[2]<https://leetcode.site.wordpress.com/2016/02/08/266-palindrome-permutation/>

1. <https://github.com/douglasleer/LeetCode-Java-Solutions/blob/master/266.palindrome-permutation.java>

2. <https://leetcode.site.wordpress.com/2016/02/08/266-palindrome-permutation>

---

## Leetcode - 300 algorithms - Learning by Reading Code First (2016-04-26 21:35)

April 26, 2016

Plan to spend next 2 weeks to go over the solution provided by Temple university Ph.D. Dawei Li

[1]<https://github.com/jianminchen/LeetCode-Java-Solutions>

Some leetcode questions to work on first:

1. reverse half linked list
2. priority queue
3. number of island
4. flying ticket
5. unsorted slots with numbers
6. Sum query 2nd mutable
7. word search in matrix
8. course schedule
9. reverse linked list
10. Leetcode 138: deep copy linked list with random number
11. Leetcode 169 Majority Element I
12. Leetcode 229 Majority Element II

13. Leetcode 218 Skyline

**Work on those two algorithms:**

insert a node into a sorted linked list

shortest job first.

Leetcode 138:

[2][http://fisherlei.blogspot.ca/2013/11/leetcode-copy-list-with-random-po inter.html](http://fisherlei.blogspot.ca/2013/11/leetcode-copy-list-with-random-pointer.html)

Leetcode 229

[3]<http://www.cnblogs.com/EdwardLiu/p/4179345.html>

Boyer-Moore Voting algorithm

[4][https://en.wikipedia.org/wiki/Boyer %E2 %80 %93Moore \\_majority \\_vote \\_algorithm](https://en.wikipedia.org/wiki/Boyer%E2%80%93Moore_majority_vote_algorithm)

3 solutions:

[5]<http://yuanhsh.iteye.com/blog/2185974>

Code Test:

[6]<http://www.cnblogs.com/EdwardLiu/category/614910.html>

1. <https://github.com/jianminchen/LeetCode-Java-Solutions>
  2. <http://fisherlei.blogspot.ca/2013/11/leetcode-copy-list-with-random-pointer.html>
  3. <http://www.cnblogs.com/EdwardLiu/p/4179345.html>
  4. [https://en.wikipedia.org/wiki/Boyer%E2%80%93Moore\\_majority\\_vote\\_algorithm](https://en.wikipedia.org/wiki/Boyer%E2%80%93Moore_majority_vote_algorithm)
  5. <http://yuanhsh.iteye.com/blog/2185974>
  6. <http://www.cnblogs.com/EdwardLiu/category/614910.html>
- 

## Leetcode 146: LRU - Cache (2016-04-26 21:36)

April 26, 2016

Design Last Recently Used Cache

[1]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/146.lru-cache.java>

C # implementation with one test case:

[2]<https://gist.github.com/jianminchen/3dc7da7e0465819e6aa8c10c71901723>

### Question and answer:

1. What do you learn through the practice?

Use dummy head and dummy tail to help maintain double linked list as cache.

2. Do not forget to set the node to last one when it is visited.

3. Talk about API design:

AddToLast,

set

get - get method is confusing, since it is also to do repositioning of visited node in the linked list - the cache - better to add another function to let get() call, function - **getKeyAndThenMoveToLast**  
use Dictionary to hold the key value pair,

4. when a node is added to last, with two dummy node in this double linked list, no temp variable needed to set up new connections. Just be patient, dummy tail's prev point: copy, and break, and set a new one.

May 17, 2016

Julia loves this article talking about cache system design; once she does some code review on basic LRU, she starts to enjoy the computer science, she just loves the engineering, good ideas to solve problems.

[3]<http://goo.gl/pL5ee4>

classical reader/ writer problem / lock / multiple shards/ commit logs/ memcached

June 2, 2016

## 5. How to design LRU? Data structure? What for?

copy from blog:[4][http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/?utm\\_source=email&utm\\_medium=email&utm\\_campaign=email](http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/?utm_source=email&utm_medium=email&utm_campaign=email)

LRU

One of the most common cache systems is [5]LRU (least recently used). In fact, another common interview question is to discuss data structures and design of an LRU cache. Let's start with this approach.

The way LRU cache works is quite simple. When the client requests resource A, it happens as follow:

- If A exists in the cache, we just return immediately.
- If not and the cache has extra storage slots, we fetch resource A and return to the client. In addition, insert A into the cache.
- If the cache is full, we kick out the resource that is least recently used and replace it with resource A.

The strategy here is to maximum the chance that the requesting resource exists in the cache. So how can we implement a simple LRU?

LRU design

An LRU cache should support the operations: lookup, insert and delete. Apparently, in order to achieve fast lookup, we need to use hash. By the same token, if we want to make insert/delete fast, something like linked list should come to your mind. Since we need to locate the least recently used item efficiently, we need something in order like queue, stack or sorted array.

To combine all these analyses, we can use queue implemented by a doubly linked list to store all the resources. Also, a hash table with resource identifier as key and address of the corresponding queue node as value is needed.

Here's how it works. when resource A is requested, we check the hash table to see if A exists in the cache. If exists, we can immediately locate the corresponding queue node and return the resource. If not, we'll add A into the cache. If there are enough space, we just add a to the end of the queue and update the hash table. Otherwise, we need to

delete the least recently used entry. To do that, we can easily remove the head of the queue and the corresponding entry from the hash table.

## Eviction policy

When the cache is full, we need to remove existing items for new resources. In fact, deleting the least recently used item is just one of the most common approaches. So are there other ways to do that?

As mentioned above, The strategy is to maximum the chance that the requesting resource exists in the cache. I'll briefly mention several approaches here:

- Random Replacement (RR) – As the term suggests, we can just randomly delete an entry.
- **Least frequently used (LFU)** – We keep the count of how frequent each item is requested and delete the one least frequently used.
- W-TinyLFU – I'd also like to talk about this modern eviction policy. In a nutshell, the problem of LFU is that sometimes an item is only used frequently in the past, but LFU will still keep this item for a long while. W-TinyLFU solves this problem by calculating frequency within a time window. It also has various optimizations of storage.

Skip concurrency and distributed cache in this design.

Next, talk about coding part - data structure and algorithm:

C # implementation.

[6]<https://gist.github.com/jianminchen/3dc7da7e0465819e6aa8c10c71901723>

int **capacity** - specify the size of cache, cache should be with limited size, since resource is limited, specially for high speed access. source code on line 24.

int **size** - current size of cache - track current size of cache to determine if eviction is needed or not.  
source code on line 24.

Design a double linked list, so ListNode class is defined with two pointers, **prev**, **next**  
source code from line 10 - line 22.

every entry has key, value. Use int to simplify the coding. Source code, line 12.

**line 12 public int key, val;**

Also, we need to add two more variables: dummy head, dummy tail to help maintain the double linked list.  
source code on line 25.

Also, we need to be able to find the key in O(1) since it is in cache. Extra space is used, maintain a hash map using Dictionary class in C #:

**line 27 Dictionary(int key, ListNode)**

Let us count how many variables inside the class LRUCache:

6 variables: - memorize the variable count - 6 - Try to recall.

```
private int capacity, size;
```

```
private ListNode dummyHead, dummyTail;
```

```
private Dictionary<int, ListNode> map;
```

ListNode class as a node in a double linked list:

```
public int key, value;
```

```
public ListNode prev, next;
```

4 variables.

June 2, 2016

Warm up the algorithm: (a lot of hurdles, just read source code.)

[7]<https://gist.github.com/jianminchen/207e20632f7837285ee9b913b0d34f3a>

Second warm up the algorithm:

[8]<https://gist.github.com/jianminchen/3bd8cdab7a31662d402c62fff9c0b597>

1. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/146.lru-cache.java>
  2. <https://gist.github.com/jianminchen/3dc7da7e0465819e6aa8c10c71901723>
  3. <http://goo.gl/pL5ee4>
  4. [http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/?utm\\_source=email&utm\\_medium=email&utm\\_campaign=email](http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/?utm_source=email&utm_medium=email&utm_campaign=email)
  5. [https://en.wikipedia.org/wiki/Cache\\_algorithms#LRU](https://en.wikipedia.org/wiki/Cache_algorithms#LRU)
  6. <https://gist.github.com/jianminchen/3dc7da7e0465819e6aa8c10c71901723>
  7. <https://gist.github.com/jianminchen/207e20632f7837285ee9b913b0d34f3a>
  8. <https://gist.github.com/jianminchen/3bd8cdab7a31662d402c62fff9c0b597>
- 

**HackerRank: Bear Steady Gene (II) - Better Code (2016-04-27 22:00)**

March 4, 2016

Problem statement:

[1]<https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>

A gene is represented as a string of length

$n$

(where

$n$

is divisible by

4

), composed of the letters

A

,

C

,

T

, and

G

. It is considered to be steady if each of the four letters occurs exactly

$n/4$

times. For example,

GACT

and

AAGTCCT

are both steady genes.

**Julia's 1st practice :**

[2]<https://gist.github.com/jianminchen/80723bae951328a690bb>

score 15 out of 50, there are 2 run time error, failed a few of test cases.

The algorithm ends up in time complexity  $O(n^2)$ , close to brute force solution -  $O(n^2)$

**C # code** implementation to study:

Readable code, with some analysis.

[3]<https://gist.github.com/jianminchen/fae9142eff9a6f9643fc>

Work on two pointers, sliding window, so time complexity is  $O(2n) = O(n)$ .

Let us go over two pointers algorithm here using example:

GAAATAAA,

A - count of A, denoted as  $cA = 6$ ,  $n/4 = 2$ , so we have to change 4 of A to other thing.

A -  $6 - 2 = 4$ , 4 of change

C -  $0 - 2 = -2$

T -  $1 - 2 = -1$

G -  $1 - 2 = -1$

So, at least minimum is 4, but a substring containing 4 of A, shortest one is AAAA. But "AAAA" is not a substring of "GAAATAAA"

G A A A T A A A

0

start

end

Let us find the substring starting from 0, but will include all 4 of A, and then, rest of string will not include any char of "ACGT" more than  $n/4$ .

GAAATA, string length is 6.

start - 0, index of 0

end - A, index of 5

continue to move start to next one, 1, then, G is moved out from substring, adjust count of chars.

AAATA can be the substring, so the length is  $\min(6, 5) = 5$ . Do not need to move end pointer.

next step, start = 2, missing one of A, and then, end has to move to next one until the string fits into requirement.

Every thing should be covered in 20 minutes, problem reading, the design of algorithm, the coding.

So, **Julia had second practice** , and the code scores 50 out of 50 this time.

[4]<https://gist.github.com/jianminchen/61dfe437f82edb9793fc>

**April 27, 2016**

Change C # program to make variables more meaningful, matching the design:

```
code -> GENES <- global string array
```

function name matches design of two pointers moving.

Improvement 1:

**Third practice:**

[5]<https://gist.github.com/jianminchen/b8263048c297473319c23836e9468c14>

Improvement 2:

**Fourth practice:**

searchStrArray -> searchStrNumbers, more meaningful.

[6]<https://gist.github.com/jianminchen/b23c4f606a101b9aeec71eff3268db32>

**Comment: (April 27, 2016)**

Spend some time to read "clean code", get some ideas to write better code.

1. <https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>
2. <https://gist.github.com/jianminchen/80723bae951328a690bb>
3. <https://gist.github.com/jianminchen/fae9142eff9a6f9643fc>
4. <https://gist.github.com/jianminchen/61dfe437f82edb9793fc>
5. <https://gist.github.com/jianminchen/b8263048c297473319c23836e9468c14>
6. <https://gist.github.com/jianminchen/b23c4f606a101b9aeec71eff3268db32>



## Clean Code - book reading (2016-04-27 22:06)

April 27, 2016

Julia was recommended to read this book "Clean Code". So, plan to read the book in next 2 weeks.

Put some notes together on this blog.

[1]<http://goo.gl/hHBZPS>

Short slides about the book: clean code

1. <http://goo.gl/hHBZPS>

---

## The art of readable code - book review second time (2016-04-27 22:12)

April 27, 2016

Julia spent one month to go over the book, use it to do some code review in 2014. So, write down things learned from her favorite book:

[1]<http://www.amazon.com/Art-Readable-Code-Theory-Practice/dp/0596802293>

chapter 5: Knowing what to comment

chapter 6: Making comments precise and compact

chapter 7: Make control flow easy to read <- 9 out of 10

chapter 8: **Break Down giant expression** <- 10 out of 10

chapter 10:

Extracting unrelated subproblems

<- Favorite (8/ 1-10)

chapter 11: One task a time <- Should follow the idea more closely

Later, compare to "Clean Code" book, write a few words to share.

A blog about naming variable: (10-15 minutes reading)

[2]<http://a-nickels-worth.blogspot.ca/2016/04/a-guide-to-naming-variables.html>

1. <http://www.amazon.com/Art-Readable-Code-Theory-Practice/dp/0596802293>

2. <http://a-nickels-worth.blogspot.ca/2016/04/a-guide-to-naming-variables.html>

---

## HackerRank: Bear And Steady Gene - binary search algorithm (VI) (2016-04-28 21:10)

April 28, 2016

Previous blog on binary search on Bear and Gene algorithm:

[1]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-bear-and-steady-gene-binary.html>

Come back to the problem on binary search solution, figure out the design:

Problem statement:

[2] <https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>

A gene is represented as a string of length

$n$

(where

$n$

is divisible by

4

), composed of the letters

A

,

C

,

T

, and

G

. It is considered to be steady if each of the four letters occurs exactly

$n/4$

times. For example,

GACT

and

AAGTGCCT

are both steady genes.

Study code using binary search:

[3]<https://gist.github.com/jianminchen/d01faa03ca9b06696db3>

[4]<https://gist.github.com/jianminchen/395eb9e76fe19cc9338f>

Comment:

Binary search is better than linear search using two pointers.

Brute force  $O(n^2)$  -> linear search  $O(n)$  -> Binary search  $O(\log n)$  <- **Not True!**

**Java code:**

[5]<https://gist.github.com/jianminchen/d01faa03ca9b06696db3>

**C # code**

[6]<https://gist.github.com/jianminchen/76dffc51880a80279f25>

First, go through two test cases: "GTTCCAAA" and "GAAATTCC"

1. "GTTCCAAA",

Binary search is doing this way:

Biggest value of search string length is 8.

First, divide range from 0 to 8 into half, 4

Find first string with length 4,

one is "GTTC", one is "CAAA",

and then, remove count of first half, rest string (two parts, separated) "CAAA", since A is repeated 3 times, not in the range;

instead of stopping, wrongly conclude that 4 is not high value, go through all the possible substring with length 4, by sliding window of search string: "GTTC" forward from left to right, but keep the same window size

"GTTC" -> "TTCC" -> "TCCA" -> stop here, since "TCCA" is removed from counting of GENES="ACGT", fit into the requirement.

Next, low=0, high=4, mid = 2,

Because both test cases with one 'A' to replace, Julia figured out through the debugging:

The slide window of fixed length technique,

How to slide?

Which direction to slide?

Kind of clever in design.

**Time complexity analysis** : length search using binary search,  $n$  - length of string,  $O(\log n)$  times; each search for length  $m$ , go over each string once, since using calculated counting array, only do add one/ remove one at a time, one char only does the work once. So, it is  $O(n)$  on this.

**Total time complexity** :  $O(n \log n)$

Conclusion: this binary search is not better than linear search is previous blog (IV).

1. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-bear-and-steady-gene-binary.html>
  2. <https://www.hackerrank.com/contests/hourrank-6/challenges/bear-and-steady-gene>
  3. <https://gist.github.com/jianminchen/d01faa03ca9b06696db3>
  4. <https://gist.github.com/jianminchen/395eb9e76fe19cc9338f>
  5. <https://gist.github.com/jianminchen/d01faa03ca9b06696db3>
  6. <https://gist.github.com/jianminchen/76dfffc51880a80279f25>
- 

## Testable JavaScript - Book Reading (2016-04-28 21:15)

April 28, 2016

One of Julia's new favorite is to read JavaScript coding submissions on HackerRank, she loves to read how people solve problem using JavaScript. How creative it is, help her expand her thinking power in problem solving, also help her to expedite on JavaScript learning.

You never know, sometimes, best players write code using JavaScript on HackerRank only; if you do not read it, you miss great chance to learn a few things in problem solving.

However, back in 2014, 2015, Julia learned JavaScript by book reading.

Julia spent over 40 hours to read "Testable JavaScript" in 2015.

She also likes to blog her favorite book: "Testable JavaScript" here. She likes a few things about unit test, how to set up test case through book reading.

Will come back.

---

## What are the best programming blogs? (2016-04-28 22:57)

April 28, 2016

Julia likes to do some research on best programming blogs, so she starts to read the article:

[1]<https://www.quora.com/What-are-the-best-programming-blogs>

She may find something to share - blogs to read.

[2]<http://www.hanselman.com/blog/GreatestHits.aspx>

Look into the blog later:

[3]<http://sportprogramming.blogspot.ca/2014/07/getting-started-with-sport-of.html>

1. <https://www.quora.com/What-are-the-best-programming-blogs>
  2. <http://www.hanselman.com/blog/GreatestHits.aspx>
  3. <http://sportprogramming.blogspot.ca/2014/07/getting-started-with-sport-of.html>
- 

### **Leetcode 200: Number of Island (2016-04-30 11:19)**

April 29, 2016

problem statement:

[1]<http://blog.csdn.net/ljiabin/article/details/44975717> (DFS, recursive, the code is very readable)

[2]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/200.number-of-islands.java>

1. <http://blog.csdn.net/ljiabin/article/details/44975717>
  2. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/200.number-of-islands.java>
- 

### **Leetcode 207: course schedule (2016-04-30 11:22)**

April 30, 2016

study the code later,

[1]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/207.course-schedule.java>

Read blog about this problem:

[2]<http://www.tangjikai.com/algorithms/leetcode-207-208-course-schedule-i-ii>

C # code:

[3]<https://gist.github.com/jianminchen/ad604224b53e80013d6ed147556f13fe>

1. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/207.course-schedule.java>
  2. <http://www.tangjikai.com/algorithms/leetcode-207-208-course-schedule-i-ii>
  3. <https://gist.github.com/jianminchen/ad604224b53e80013d6ed147556f13fe>
- 

## Leetcode 210: course schedule (2016-04-30 11:28)

April 30, 2016

Read the problem and analysis, read python code 10 - 20 minutes.

[1]<http://www.tangjikai.com/algorithms/leetcode-207-208-course-schedule-i-ii>

Read Java code later.

[2]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/210.course-schedule-ii.java>

Understand the problem and solution quickly through this blog: (10 minutes to read)

[3]<http://www.voidcn.com/blog/qq508618087/article/p-5039267.html>

Another 10 minutes to read this blog:

[4]<http://www.cnblogs.com/grandyang/p/4504793.html>

Read 10-20 minutes about graph representation: (11:40am - 12:00pm)

[5]<https://www.khanacademy.org/computing/computer-science/algorithms/graph-representation/a/representing-graphs>

**Spent 20 minutes to watch the video :**

[6]<https://class.coursera.org/algo-003/lecture>

Take some notes:

1. Sink Vertex - concept
2. **To compute topological ordering:**

- Let  $v$  be a sink vertex of  $G$
- set  $f(v) = n$
- recurse on  $G - \{v\}$

**why does it work ?** when  $v$  is assigned to position  $i$

Topological Sort vis DFS

DFS-Loop (graph  $G$ )

- mark all nodes unexplored
- current\_label =  $n$  [to keep track of ordering]
- for each vertex  $v$  in set  $G$ :
- if  $v$  not yet explored [ in some previous DFS call]
- DFS(  $G, v$ )

DFS( graph  $G$ , start vertex  $s$ )

- mark  $s$  explored
- for every edge  $(s, v)$ :
- if  $v$  not yet explored
- DFS(  $G, v$ )
- **Set  $f(s) = \text{current\_label}$**
- **current\_label -**

Read an article: (1:04pm - 1:14pm)

[7]<http://www.geeksforgeeks.org/topological-sorting/>

**Fun part later:**

play with visual presentation through the link:

[8]<https://www.cs.usfca.edu/galles/visualization/TopoSortDFS.html>

C # code for Leetcode 210:

[9]<https://gist.github.com/jianminchen/5873fc014e806d626807917959e05ea2>

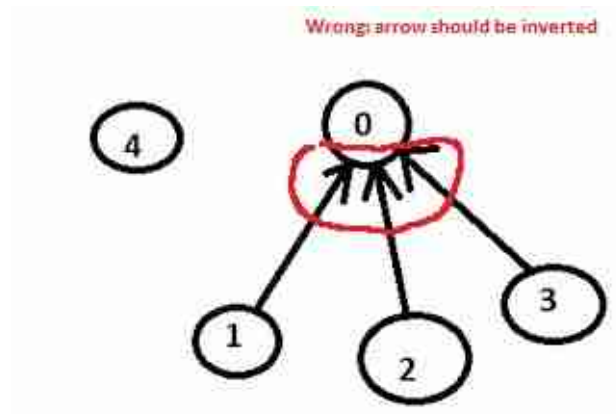
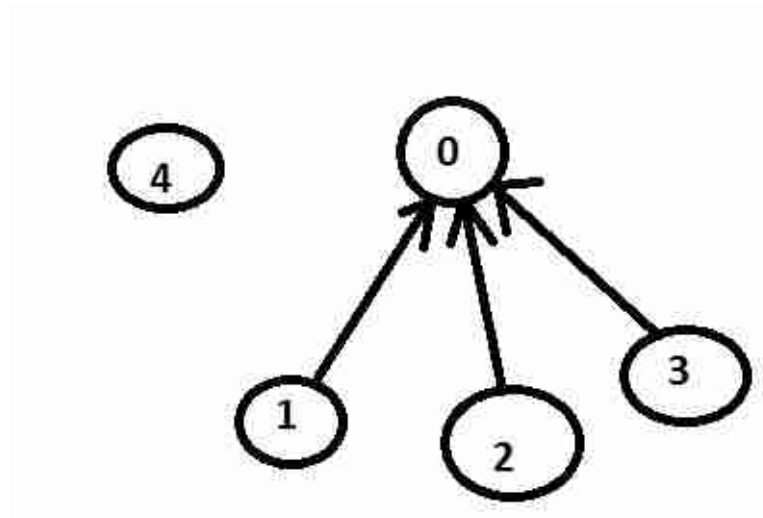
update C # code to add the example in the comment, and then, update code to match the example:

[10]<https://gist.github.com/jianminchen/85478d1bed0faf4410b76a7af3a48fc3>

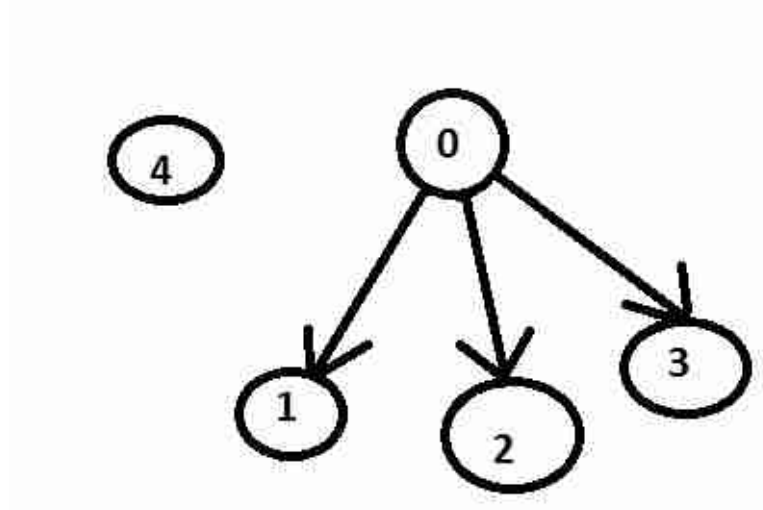
**Question and answer:**

**What do you learn today? And what is idea to solve this course schedule problem?**

**1. First, Julia learned how to do the topological sort, here is an example with its diagram:**



The image should be the following: directed edge  $0 \rightarrow 1$ , which means that 0 is prerequisite course.



Double check with geekforgeeks article:

[11]<http://www.geeksforgeeks.org/topological-sorting/>

Confused since it is the first time to draw the diagram.

**So, always start from nodes with 0 indegree (node 0, node 4).**

**So, the ordering can be 0, 1, 2, 3, 4; or 4, 0, 1, 2, 3**



2. So, Julia also practices to do some counting for vertex.

vertex 0: indegree 0, but dependency list: 1, 2, 3

vertex 1: indegree 1, but dependency list: empty set

...

So, Julia learns to manage the graph using indegree array for the graph, and dependency list for each vertex.

3. Julia also learns the easy way to do topological sorting using the above graph, and also make the code easy to read, once you remember the graph, you can read the code in 5 minutes.

So, I updated the version of C # code to match this test case:

[12] <https://gist.github.com/jianminchen/85478d1bed0faf4410b76a7af3a48fc3>

**add explanation variable - line 51:**

```
int[] tmpDep = new int[2] { prerequisites[i][1] , prerequisites[i][0] };
```

tmpDep[0] - 0 is the index, similar to the above diagram node 0, add dependency list: 1, 2, 3

tmpDep[1] - 1 is the index, similar to the above diagram node 1.

4. Now, understanding one example. Ready to talk about idea of problem solving.

The idea to solve this course schedule problem is first for each vertex to get indegree value and also dependency list.

Do not get confused with dependency.

In above diagram, the directed graph, course 1 should take after course 0, so course 1 is depending on course 0; course 1 has indegree 1 related from course 0. course 0 has no indegree. If course 2, 3 also has to take after course 0, so course 0 has dependency list: 1, 2, 3. If course 0 is dequeue, then, course 0's dependency list: 1, 2, 3, 3 courses' indegree value has to be decremented by one.

Secondly, to find all nodes in the graph with indegree's value 0; put them in the queue {4, 0 }, and then, dequeue one by one, add to the result list; and then, each node in the dependency list will decrease the indegree value by 1; and then, if any node is found with 0 indegree value, then, it will be added to the queue as well.

In other words, here are steps:

1. Add nodes with indegree value 0 to the queue;

2. if queue is not empty, dequeue one in the front,

add one to the result;

check dependency list one by one,

decrement one on indegree value,

if the node's indegree value is 0, then add to the queue

More reading:

[13] <http://baike.baidu.com/view/4573323.htm>

[14] <http://baike.baidu.com/view/288212.htm>

statistics:

Time spent: 5+ hours

1. <http://www.tangjikai.com/algorithms/leetcode-207-208-course-schedule-i-ii>

2. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/210.course-schedule-ii.java>

3. <http://www.voidcn.com/blog/qq508618087/article/p-5039267.html>
  4. <http://www.cnblogs.com/grandyang/p/4504793.html>
  5. <https://www.khanacademy.org/computing/computer-science/algorithms/graph-representation/a/representing-graphs>
  6. <https://class.coursera.org/algo-003/lecture>
  7. <http://www.geeksforgeeks.org/topological-sorting/>
  8. <https://www.cs.usfca.edu/~galles/visualization/TopoSortDFS.html>
  9. <https://gist.github.com/jianminchen/5873fc014e806d626807917959e05ea2>
  10. <https://gist.github.com/jianminchen/85478d1bed0faf4410b76a7af3a48fc3>
  11. <http://www.geeksforgeeks.org/topological-sorting/>
  12. <https://gist.github.com/jianminchen/85478d1bed0faf4410b76a7af3a48fc3>
  13. <http://baike.baidu.com/view/4573323.htm>
  14. <http://baike.baidu.com/view/288212.htm>
- 

## **coursera course: Algorithm (2016-04-30 12:31)**

April 30, 2016

Come cross this course, and then, Julia likes to use it more often in the study of algorithm:

[1]<https://class.coursera.org/algo-003/lecture>

Take notes the videos watched:

April 30, 2016 20 minutes:

[2]<https://class.coursera.org/algo-003/lecture/52>

May 1, 2016 1 -2 hours

graph

1. <https://class.coursera.org/algo-003/lecture>
  2. <https://class.coursera.org/algo-003/lecture/52>
- 

## **2.5 May**

### **Leetcode 133: clone graph (2016-05-01 13:56)**

May 1, 2016

Problem statement and solutions:

[1]<http://bangbingsyb.blogspot.ca/2014/11/leetcode-clone-graph.html>

[2]<http://www.cnblogs.com/springfor/p/3874591.html>

C # code:

Easy to read - Node class

[3]<https://gist.github.com/jianminchen/c328dbc4391cb03a9ab8665b3ab54966>

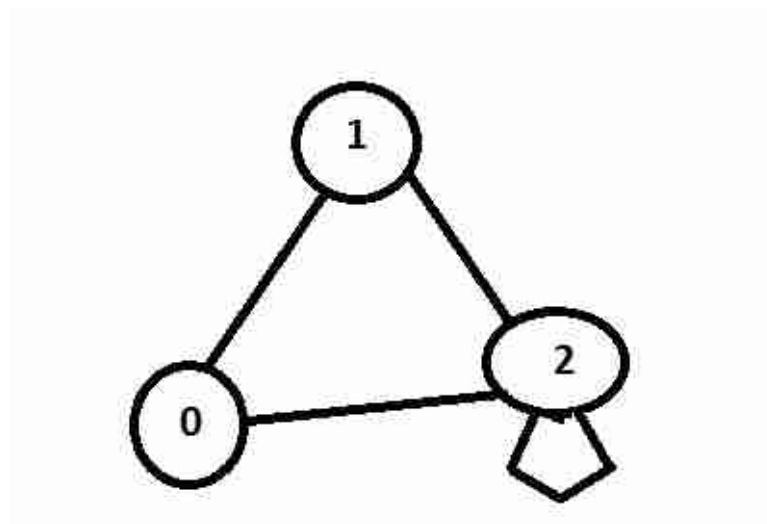
Fit into leetcode online judge

[4]<https://gist.github.com/jianminchen/bf0821320af0e549b486997faf11b358>

### Question and answer:

1. What do you learn through the practice? Can you write down your own words with your experience of learning?

1. Let us talk about a small test case, undirected graph {0,1,2 #1,2 #2,2 }, here is the graph:



So, in Julia's practice, undirected graph node is defined in class Node:

```
class Node
{
public int label;
public List<node> neighbors;

public Node(int x)
{
label = x;
neighbors = new List<Node>();
}
}
```

The above graph, 3 nodes are declared, node0, node1, node2,  
3 edges in the graph:

0 - 1,

0 - 2

the above 2 edges are saved in node0's neighbors - a list;

edge 1-2

the third edge is saved in node1's neighbors;

edge 2-2 - a loop to itself

the fourth edge is saved in node2's neighbors - a list containing node2 itself.

Here is the expression to parse the graph: {0,1,2 #1,2 #2,2 }

**Any edge in the graph is only saved once in the list named in neighbors. <- It is good!**

Julia likes to catch up some reading about graph, so she spent 10 minutes to read:

[5]<http://algs4.cs.princeton.edu/41graph/>

2. Need to review graph search, DFS vs. BFS, in the above C # practice, queue is used, so it should be BFS - breadth first search.

So, cloneGraph(Node node) function is designed to do BFS - breadth first search. If node0 is passed, then we can go over the steps what should do:

Main idea is to put input argument - a node into queue, copy the node to a first node in cloned graph, denoted as newHead.

Also, in the Dictionary - a map, add one entry - node, newHead.

Next, get into a loop with queue length checking > 0:

dequeue the node from the queue, and then,

check its neighbors, go through the iteration loop one by one.

If the neighbor node is not in the dictionary, then,

it is not in cloned graph, what we need to do, is to add a new node in cloned graph,

and also add a new entry in the dictionary,

and update cloned graph neighbors list as well.

If the neighbor node is in the dictionary, then

update cloned graph with current node's mapping - add one entry in neighbor list. (\*)

<- (\*) **Need to figure out when this executable path will be executed!**

**So, node0 is passed in as an argument, walk through steps:**

**node0 has two neighbors, node1 and node2,**

**go through one by one,**

**node1 is not in the dictionary, create a copy for node1,**

**add an entry {node1, copy } into the dictionary,**

**also bind copy to newHead's neighbor.**

**add node1 into queue.**

node2 is visited, and then, ...  
add node2 into queue

when node1 is dequeued from queue, node1's neighbors are examined:  
only one neighbor, node2.  
But node2 is already in the dictionary, <-  
need to debug the code to verify

when node0 is dequeued from queue,  
node1 and node2 are added to queue,  
two edges are added to clonedGraph,  
dictionary is updated with node1 and node2.

when node1 is dequeued, need to add edge 1->2.

change C # code:

[6]<https://gist.github.com/jianminchen/35f3efda39c30d1e993a16a89d14308d>

Make the code more flat, remove if/else in previous version.

```
foreach (Node aNeighbor in currNeighbors)
{
    bool containing = map.ContainsKey(aNeighbor);
```

```
// 1.update queue
if (!containing)
    queue.Enqueue(aNeighbor);
```

```
// 2. update map
if (!containing)
{
    Node copy = new Node(aNeighbor.label);
    map.Add(aNeighbor, copy);
}
```

```
// 3. update neighbors for cloned graph
// definitely, map.ContainsKey[aNeighbor]
map[curr].neighbors.Add(map[aNeighbor]);

}
```

Comment:

1. The explanation is kind of tricky, not so clear! Improve it later.

2. Look into issues about else statement, Julia makes mistakes when she write if/else, and do not pay attention to else condition. <- basic logic checking...

Next time, try to put all true case in one statement. Your cognitive ability is in a standard level, only thing to improve is to discipline yourself, write code with more disciplines, simplify!

1. <http://bangbingsyb.blogspot.ca/2014/11/leetcode-clone-graph.html>
  2. <http://www.cnblogs.com/springfor/p/3874591.html>
  3. <https://gist.github.com/jianminchen/c328dbc4391cb03a9ab8665b3ab54966>
  4. <https://gist.github.com/jianminchen/bf0821320af0e549b486997faf11b358>
  5. <http://algs4.cs.princeton.edu/41graph/>
  6. <https://gist.github.com/jianminchen/35f3efda39c30d1e993a16a89d14308d>
- 

## Leetcode 261: graph-valid-tree (2016-05-01 14:59)

May 1, 2016

[1]<http://happycoding2010.blogspot.ca/2015/11/leetcode-261-graph-valid-tree.html>

[2]<http://www.cnblogs.com/yrbbest/p/5018217.html>

Read the blog to entertain, figure out how to improve practice:

[3]<http://home.cnblogs.com/u/yrbbest/>

1. <http://happycoding2010.blogspot.ca/2015/11/leetcode-261-graph-valid-tree.html>
  2. <http://www.cnblogs.com/yrbbest/p/5018217.html>
  3. <http://home.cnblogs.com/u/yrbbest/>
- 

## Leetcode 323: Number of connected components (2016-05-01 15:16)

May 1, 2016

[1]<https://asanchina.wordpress.com/2015/12/29/323-number-of-connected-components-in-an-undirected-graph/>

[2]<http://massivealgorithms.blogspot.ca/2015/12/leetcode-323-number-of-connected-components.html>

[3]<http://leetcode0.blogspot.ca/2016/02/323-number-of-connected-components-in.html>

1. <https://asanchina.wordpress.com/2015/12/29/323-number-of-connected-components-in-an-undirected-graph/>
  2. <http://massivealgorithms.blogspot.ca/2015/12/leetcode-323-number-of-connected.html>
  3. <http://leetcode0.blogspot.ca/2016/02/323-number-of-connected-components-in.html>
- 

## **HackerRank: article reading (2016-05-03 18:32)**

May 3, 2016

[1]<http://blog.hackerrank.com/step-0-before-you-do-anything/>

very good article - favorite:

**Designing Challenges that Gauge Depth of Thinking, and also, talk about hurdles, how senior people can overcome a few of them once.**

[2]<http://blog.hackerrank.com/design-impactful-code-challenges/>

[3]<http://blog.hackerrank.com/setting-expectations-warming-your-candidates/>

Julia's favorite quote:

"Gauging not only their intelligence but also how much they value fundamentals through algorithm and data structure questions are strong instruments to find the best engineering talent"

[4]<http://blog.hackerrank.com/how-amazon-web-services-surged-out-of-nowhere/>

1. <http://blog.hackerrank.com/step-0-before-you-do-anything/>
  2. <http://blog.hackerrank.com/design-impactful-code-challenges/>
  3. <http://blog.hackerrank.com/setting-expectations-warming-your-candidates/>
  4. <http://blog.hackerrank.com/how-amazon-web-services-surged-out-of-nowhere/>
- 

## **HackerRank: Delete duplicate value nodes from a sorted linked list (2016-05-06 21:47)**

May 6, 2016

## Easy question - Linked List

Problem statement:

[1]<https://www.hackerrank.com/challenges/delete-duplicate-value-nodes-from-a-sorted-linked-list>

Julia's solution:

[2]<https://gist.github.com/jianminchen/9013539e71764a018f3745c514da9d91>

Most favorite solution:

[3]<https://gist.github.com/jianminchen/a13738b4ab32decb8601f29777172209>

```
/*
Remove all duplicate elements from a sorted linked list
Node is defined as
struct Node
{
    int data;
    struct Node *next;
}
/
Node* RemoveDuplicates(Node *head)
{
    Node *cur = head;
    int last_seen = cur->data;
    while(cur->next) {
        if(cur->next->data == last_seen) {
            cur->next = cur->next->next;
        }else {
            cur = cur->next;
            last_seen = cur->data;
        }
    }
    return head;
}
```

### Question and Answer:

1. What do you learn through the practice? How long do you take?

Julia spent more than 30 minutes to work on this question; She wrote a two nested loops, and then, came cross time out issue; She tried to direct a pointer to correct position, like 1->1->1->2, she likes to set up first node with value 1's next point to 2. She was in hurry, tried to do it once to make it work in nested second loop. Actually, she figured out the solution just be lazy to let first node with value 1 to point to third node with value 1 first.

Then, she came out this solution using one loop only:

[4]<https://gist.github.com/jianminchen/9013539e71764a018f3745c514da9d91>



## 2. What do you learn through studying other submissions?

Julia learned that through the problem solving, she read first 20 submissions, 80 % of them should get rid of two loops, or reduce code length to less than 10 lines; one loops is enough to take care of the business; But, people stop when the code works.

**Julia knows that value of excellent code. She has to push herself, train herself to discipline herself.**

**She found the favorite solution,**

[5]<https://gist.github.com/jianminchen/a13738b4ab32decb8601f29777172209>

**Her new strategy:**

Write the code, finish the coding; and then, delete the code, write again. Stop on the best one. And then, try to sort out if there is a bug in the code, test cases etc. And then, discuss or present the code.

Because when the code is short and clean, it shortens time to do code review, bug fixes.

Another concern is that there are so many solutions to solve a problem, only if the solution is short and concise, people can easily follow and tell the correctness.

**Statistics and Comment:**

1. Read 200 submissions, around 5-10 using recursive function, 80 % more than 10 lines of code, over 10 of them use two nested loops.

2. Enjoy the reading; thinking about myself as interviewer, and see how many of them are impressive; I do not see more than 10 of them.

[6]<https://www.hackerrank.com/challenges/delete-duplicate-value-nodes-from-a-sorted-linked-list/leaderboard>

one more readable code to follow:

[7]<https://gist.github.com/jianminchen/b46755065e4d17c82ad793889e2c911b>

3. Time spent: reading code submissions - 2+ hours

**Look into those:**

[8]<https://www.hackerrank.com/contests/programming-interviews-practice-session/challenges>

[9]<https://www.hackerrank.com/contests/algorithms-practice-match-2/challenges>

[10]<https://www.hackerrank.com/contests/regex-practice-2/challenges>

[11]<https://www.hackerrank.com/contests/basic-ds-quiz-2/challenges>

[12]<https://www.hackerrank.com/basic-ds-quiz-2>

[13]<https://www.hackerrank.com/countercode>  
[14]<https://www.hackerrank.com/accel-contest>

Courses to check:

[15]<https://www.coursera.org/learn/algorithmic-thinking-1>

[16][#">https://www.coursera.org/learn/algorithmic-thinking-2 #](https://www.coursera.org/learn/algorithmic-thinking-2)

[17][https://www.coursera.org/maestro/api/certificate/get\\_certificate?verify-code=V5LZ37UU7P](https://www.coursera.org/maestro/api/certificate/get_certificate?verify-code=V5LZ37UU7P)

Quizes:

[18]<https://www.hackerrank.com/challenges/basic-algo-quiz-1>

[19]<https://www.hackerrank.com/challenges/data-structures-quiz-2>

Find some interesting problems on the link - hackerRank:

[20]<https://www.hackerrank.com/roaclark>

[21]<https://www.hackerrank.com/epiccode>

[22]<https://www.hackerrank.com/contests/codesprint-practice/challenges>

[23][https://www.hackerrank.com/cherry\\_su](https://www.hackerrank.com/cherry_su)

[24]<https://www.hackerrank.com/dotgc>

Try quick sort in place on HackerRank - 3 questions

1. <https://www.hackerrank.com/challenges/delete-duplicate-value-nodes-from-a-sorted-linked-list>
2. <https://gist.github.com/jianminchen/9013539e71764a018f3745c514da9d91>
3. <https://gist.github.com/jianminchen/a13738b4ab32dec8601f29777172209>
4. <https://gist.github.com/jianminchen/9013539e71764a018f3745c514da9d91>
5. <https://gist.github.com/jianminchen/a13738b4ab32dec8601f29777172209>
6. <https://www.hackerrank.com/challenges/delete-duplicate-value-nodes-from-a-sorted-linked-list/leaderboard>
7. <https://gist.github.com/jianminchen/b46755065e4d17c82ad793889e2c911b>

8. <https://www.hackerrank.com/contests/programming-interviews-practice-session/challenges>
  9. <https://www.hackerrank.com/contests/algorithms-practice-match-2/challenges>
  10. <https://www.hackerrank.com/contests/regex-practice-2/challenges>
  11. <https://www.hackerrank.com/contests/basic-ds-quiz-2/challenges>
  12. <https://www.hackerrank.com/basic-ds-quiz-2>
  13. <https://www.hackerrank.com/countercode>
  14. <https://www.hackerrank.com/accel-contest>
  15. <https://www.coursera.org/learn/algorithmic-thinking-1>
  16. <https://www.coursera.org/learn/algorithmic-thinking-2>
  17. [https://www.coursera.org/maestro/api/certificate/get\\_certificate?verify-code=V5LZ37UU7P](https://www.coursera.org/maestro/api/certificate/get_certificate?verify-code=V5LZ37UU7P)
  18. <https://www.hackerrank.com/challenges/basic-algo-quiz-1>
  19. <https://www.hackerrank.com/challenges/data-structures-quiz-2>
  20. <https://www.hackerrank.com/roaclark>
  21. <https://www.hackerrank.com/epiccode>
  22. <https://www.hackerrank.com/contests/codesprint-practice/challenges>
  23. [https://www.hackerrank.com/cherry\\_su](https://www.hackerrank.com/cherry_su)
  24. <https://www.hackerrank.com/dotgc>
- 

## **HackerRank: Linked List - Find Merge Point of two linked list (2016-05-08 11:34)**

May 8, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/find-the-merge-point-of-two-joined-linked-lists>

C++ solution:

[2]<https://gist.github.com/jianminchen/ce99505020ee58f2fd23ecd78bb2f19f>

Value simple, easy question, solve the problem, learn from other submissions. From the simple ones, build up skills, feel more comfortable to solve moderate, difficult one later.

Submissions:

[3]<https://www.hackerrank.com/challenges/find-the-merge-point-of-two-joined-linked-lists/leaderboard>

Julia's favorite code:

1. define a few functions, very well written:

length, position, common - use recursive function, very short function - 3 lines of code each function

[4]<https://gist.github.com/jianminchen/2ab387360a0ff2ef41c10c67219fd1c1>

source:

[5]<https://goo.gl/J5WK7y>

2. unbelievable short - only use for loop to do things:

[6]<https://gist.github.com/jianminchen/9d14ff515146af193c4bee660b0a6fb8>

**Question and answer :**

1. What do you learn through the problem solving? Code submission study?

Julia likes to document the problems she found when she reads first 100 solutions, in order to improve the speed, accuracy, and avoid bugs - avoid repetition of same logic:

1. Best code she found so far:

[7]<https://gist.github.com/jianminchen/2ab387360a0ff2ef41c10c67219fd1c1>

2. One is to use brute force, go through each node in the first list, compare to each node in the second list; The algorithm is not optimal.

3. Shortest time to write the code, in less than 20 lines:

[8]<https://gist.github.com/jianminchen/9d14ff515146af193c4bee660b0a6fb8>

3. Repeat same code twice - if/else statement - avoid two cases, simplify to one case.

4. Same code as Julia does, but better code:

[9]<https://goo.gl/NXTurm>

1. <https://www.hackerrank.com/challenges/find-the-merge-point-of-two-joined-linked-lists>
2. <https://gist.github.com/jianminchen/ce99505020ee58f2fd23ecd78bb2f19f>
3. <https://www.hackerrank.com/challenges/find-the-merge-point-of-two-joined-linked-lists/leaderboard>
4. <https://gist.github.com/jianminchen/2ab387360a0ff2ef41c10c67219fd1c1>
5. <https://goo.gl/J5WK7y>
6. <https://gist.github.com/jianminchen/9d14ff515146af193c4bee660b0a6fb8>
7. <https://gist.github.com/jianminchen/2ab387360a0ff2ef41c10c67219fd1c1>
8. <https://gist.github.com/jianminchen/9d14ff515146af193c4bee660b0a6fb8>
9. <https://goo.gl/NXTurm>

---

## HackerRank: Insert a node in a double linked list (2016-05-08 15:42)

May 8, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/insert-a-node-into-a-sorted-doubly-linked-list>

Julia spent over 30 minutes to work out a solution:

[2]<https://gist.github.com/jianminchen/a9794202ddc66d25cc1d416e04adc7a9>

A few of mistakes in first writing:

1. If the list is empty, add a new node, forget to return the node;
2. Insert a node, 3 possible positions:

before the head,  
in the middle of list,  
at the end of list <- edge case -> forget the edge case first time

Review other submissions:

### **Question and answer :**

#### **1. What do you learn through the study?**

Julia learns to write an iterative solution for the problem. And also she fixed all the bugs and then complete the task.

The solution Julia takes is not the best one, she uses dummy head node to help, and then, use iterative solution to do work. Need more practice.

She learned that she needs more ideas, in order to complete it in less than 10 minutes.

Also, the linked list - the simple problem is also very helpful to learn how to think recursively.

#### **2. This is a solution to study - use recursive function, much short code, less time to write:**

[3]<https://gist.github.com/jianminchen/e1e528eda506c185182a68a0df3be544>

Before Julia reads the code in detail, she wrote her own recursive solution. It is fast, no dummy head need, and also much easy to read:

[4]<https://gist.github.com/jianminchen/fc0d0fc2171ed891330d5cf773c45206>

Forget to add one more line: after line 30, before line 31.

```
(head->next)->prev=head;
```

Let us talk about the idea using recursive function - how to design the function?

Test case: 2->4->6, insert a value 1,

First, deal with the case that the list is empty: create a new node, and then return;

Second, if the inserted value is less than head's value, then create a new head with value 1,

link it to the head;

Otherwise, the first node is processed, and the next node of it is the value of recursive call.

It saves time to write recursively here.

#### **3. What are the common issues you find in the submissions?**

[5]<https://www.hackerrank.com/challenges/insert-a-node-into-a-sorted-doubly-linked-list/leaderboard>

-

More practice on May 9, 2016

**Linked List** on HackerRank:

problem statements:

[6]<https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>

Get Node value:

[7]<https://gist.github.com/jianminchen/d4da2ec82876d93bcd1920927054702d>

Reverse a doubly linked list:

[8]<https://gist.github.com/jianminchen/5bdf999c91964e87788e0d010eb52e53>

Merge two sorted linked list

[9]<https://gist.github.com/jianminchen/c09df676896e938d7f6c3819e301545c>

Compare two linked list:

[10]<https://gist.github.com/jianminchen/904a955b78a56ece032f5b9a681f91fc>

Reverse a linked list:

[11]<https://gist.github.com/jianminchen/2c0203cebd076ae329a10e870c414219>

Reverse print:

[12]<https://gist.github.com/jianminchen/91e8dfeac0e962307b30f4afd0bb6f98>

Delete a node:

[13]<https://gist.github.com/jianminchen/d0a360f5a5a4be3b6d94e09d6bd0f266>

Insert a node at a specific position:

[14]<https://gist.github.com/jianminchen/5b357be38d6bad427e885c7b7343c19b>

Insert a node at the head of a linked list

[15]<https://gist.github.com/jianminchen/79dd121f3135769d57088c7d2e9e51fe>

Insert a node at tail of a linked list

[16]<https://gist.github.com/jianminchen/957d2025fd310a9e59fb88fe8766e0de>

print list

[17]<https://gist.github.com/jianminchen/cd219738d812605e0c6ced926d2e488c>

**Trees:**

Problem statements:

[18]<https://www.hackerrank.com/domains/data-structures/trees>

height of tree:

[19]<https://gist.github.com/jianminchen/18de7257b29bf6ccbab63cdc93d4077f>

Top view:

problem statement:

[20]<https://www.hackerrank.com/challenges/tree-top-view>

[21]<https://gist.github.com/jianminchen/f7742a123f3524c1d0fbd98e71c8f516>

Level Order Traversal:

[22]<https://gist.github.com/jianminchen/a3194f885fc542a7f5827cbcdc92a3f5>

1. <https://www.hackerrank.com/challenges/insert-a-node-into-a-sorted-doubly-linked-list>
2. <https://gist.github.com/jianminchen/a9794202ddc66d25cc1d416e04adc7a9>
3. <https://gist.github.com/jianminchen/e1e528eda506c185182a68a0df3be544>
4. <https://gist.github.com/jianminchen/fc0d0fc2171ed891330d5cf773c45206>
5. <https://www.hackerrank.com/challenges/insert-a-node-into-a-sorted-doubly-linked-list/leaderboard>
6. <https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>
7. <https://gist.github.com/jianminchen/d4da2ec82876d93bcd1920927054702d>
8. <https://gist.github.com/jianminchen/5bdf999c91964e87788e0d010eb52e53>
9. <https://gist.github.com/jianminchen/c09df676896e938d7f6c3819e301545c>
10. <https://gist.github.com/jianminchen/904a955b78a56ece032f5b9a681f91fc>
11. <https://gist.github.com/jianminchen/2c0203cebd076ae329a10e870c414219>
12. <https://gist.github.com/jianminchen/91e8dfeac0e962307b30f4afd0bb6f98>
13. <https://gist.github.com/jianminchen/d0a360f5a5a4be3b6d94e09d6bd0f266>
14. <https://gist.github.com/jianminchen/5b357be38d6bad427e885c7b7343c19b>
15. <https://gist.github.com/jianminchen/79dd121f3135769d57088c7d2e9e51fe>
16. <https://gist.github.com/jianminchen/957d2025fd310a9e59fb88fe8766e0de>
17. <https://gist.github.com/jianminchen/cd219738d812605e0c6ced926d2e488c>
18. <https://www.hackerrank.com/domains/data-structures/trees>
19. <https://gist.github.com/jianminchen/18de7257b29bf6ccbab63cdc93d4077f>
20. <https://www.hackerrank.com/challenges/tree-top-view>
21. <https://gist.github.com/jianminchen/f7742a123f3524c1d0fbd98e71c8f516>
22. <https://gist.github.com/jianminchen/a3194f885fc542a7f5827cbcdc92a3f5>

---

## Reverse a linked list (2016-05-09 21:01)

More practice on May 9, 2016

**Linked List** on HackerRank:

problem statements:

[1]<https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>

Reverse a linked list:

[2]<https://gist.github.com/jianminchen/2c0203cebd076ae329a10e870c414219>

1. <https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>
2. <https://gist.github.com/jianminchen/2c0203cebd076ae329a10e870c414219>

## HackerRank: Linked List - Get Node value (2016-05-09 21:01)

More practice on May 9, 2016

**Linked List** on HackerRank:

problem statements:

[1]<https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>

Get Node value:

[2]<https://gist.github.com/jianminchen/d4da2ec82876d93bcd1920927054702d>

1. <https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>

2. <https://gist.github.com/jianminchen/d4da2ec82876d93bcd1920927054702d>

---

## Linked List: Delete a node (2016-05-09 21:02)

More practice on May 9, 2016

**Linked List** on HackerRank:

problem statements:

[1]<https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>

Delete a node:

[2]<https://gist.github.com/jianminchen/d0a360f5a5a4be3b6d94e09d6bd0f266>

1. <https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>

2. <https://gist.github.com/jianminchen/d0a360f5a5a4be3b6d94e09d6bd0f266>

---

## Insert a node at tail of a linked list (2016-05-09 21:03)

More practice on May 9, 2016

**Linked List** on HackerRank:

problem statements:

[1]<https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>



Insert a node at tail of a linked list

[2]<https://gist.github.com/jianminchen/957d2025fd310a9e59fb88fe8766e0de>

1. <https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>

2. <https://gist.github.com/jianminchen/957d2025fd310a9e59fb88fe8766e0de>

---

### Insert a node at the head of a linked list (2016-05-09 21:03)

More practice on May 9, 2016

**Linked List** on HackerRank:

problem statements:

[1]<https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>

Insert a node at the head of a linked list

[2]<https://gist.github.com/jianminchen/79dd121f3135769d57088c7d2e9e51fe>

1. <https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>

2. <https://gist.github.com/jianminchen/79dd121f3135769d57088c7d2e9e51fe>

---

### Insert a node at a specific position in a linked list (2016-05-09 21:03)

More practice on May 9, 2016

**Linked List** on HackerRank:

problem statements:

[1]<https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>

Insert a node at a specific position:

[2]<https://gist.github.com/jianminchen/5b357be38d6bad427e885c7b7343c19b>

1. <https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>
  2. <https://gist.github.com/jianminchen/5b357be38d6bad427e885c7b7343c19b>
- 

## **Linked List: print list (2016-05-09 21:04)**

More practice on May 9, 2016

**Linked List** on HackerRank:

problem statements:

[1]<https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>

print list

[2]<https://gist.github.com/jianminchen/cd219738d812605e0c6ced926d2e488c>

1. <https://www.hackerrank.com/domains/data-structures/linked-lists/difficulty/all/page/1>
  2. <https://gist.github.com/jianminchen/cd219738d812605e0c6ced926d2e488c>
- 

## **Merge two sorted linked list (2016-05-09 21:08)**

May 9, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/merge-two-sorted-linked-lists>

Merge two sorted linked list

[2]<https://gist.github.com/jianminchen/c09df676896e938d7f6c3819e301545c>

1. <https://www.hackerrank.com/challenges/merge-two-sorted-linked-lists>
  2. <https://gist.github.com/jianminchen/c09df676896e938d7f6c3819e301545c>
- 

## **Reverse a doubly linked list: (2016-05-09 21:08)**

May 9, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/reverse-a-doubly-linked-list>

Reverse a doubly linked list:

[2]<https://gist.github.com/jianminchen/5bdf999c91964e87788e0d010eb52e53>

1. <https://www.hackerrank.com/challenges/reverse-a-doubly-linked-list>
  2. <https://gist.github.com/jianminchen/5bdf999c91964e87788e0d010eb52e53>
- 

### **Compare two linked list (2016-05-09 21:09)**

May 9, 2016

Problem statement:

[1]<https://www.hackerrank.com/challenges/compare-two-linked-lists>

Compare two linked list:

[2]<https://gist.github.com/jianminchen/904a955b78a56ece032f5b9a681f91fc>

Submission code to study:

[3]<https://www.hackerrank.com/challenges/compare-two-linked-lists/leaderboard>

Iterative solution:

[4]<https://goo.gl/vyGJkW>

1. <https://www.hackerrank.com/challenges/compare-two-linked-lists>
  2. <https://gist.github.com/jianminchen/904a955b78a56ece032f5b9a681f91fc>
  3. <https://www.hackerrank.com/challenges/compare-two-linked-lists/leaderboard>
  4. <https://goo.gl/vyGJkW>
- 

### **Reverse print (2016-05-09 21:10)**

Problem statement:

[1]<https://www.hackerrank.com/challenges/print-the-elements-of-a-linked-list-in-reverse>

Reverse print:

[2]<https://gist.github.com/jianminchen/91e8dfeac0e962307b30f4afd0bb6f98>

Submission reading: 10 - 20 minutes.

[3]<https://www.hackerrank.com/challenges/print-the-elements-of-a-linked-list-in-reverse/leaderboard>

1. <https://www.hackerrank.com/challenges/print-the-elements-of-a-linked-list-in-reverse>
  2. <https://gist.github.com/jianminchen/91e8dfeac0e962307b30f4afd0bb6f98>
  3. <https://www.hackerrank.com/challenges/print-the-elements-of-a-linked-list-in-reverse/leaderboard>
- 

## Height of a tree (2016-05-09 22:01)

May 9, 2016

problem statement:

[1]<https://www.hackerrank.com/challenges/tree-height-of-a-binary-tree>

Julia's practice:

[2]<https://gist.github.com/jianminchen/18de7257b29bf6ccbab63cdc93d4077f>

1. <https://www.hackerrank.com/challenges/tree-height-of-a-binary-tree>
  2. <https://gist.github.com/jianminchen/18de7257b29bf6ccbab63cdc93d4077f>
- 

## HackerRank: Algorithms > Sorting (2016-05-09 22:33)

May 9, 2016

Work on Sorting Challenges:

Value the easy question. Write your own code, and then, study other's submission. Rewrite the code again and again, learn how to stay focus, and avoid common mistakes in writing.

1. Insertion Sort - Part 1 (Easy)
2. Insertion Sort - Part 2 (Easy)
3. Correctness and the Loop Invariant
4. Running Time of Algorithms
5. Quicksort 1 - Partition
6. Quicksort 2 - Sorting

## 7. Quicksort In-Place

---

### **HackerRank: Search algorithms (2016-05-09 22:35)**

May 9, 2016

Work on a few of search algorithms:

1. Ice Cream Parlor
  2. Maximise Sum
  3. Missing Numbers
- 

### **HackerRank: Greedy Algorithms (2016-05-09 22:37)**

May 9, 2016

Work on easy algorithms in Greedy category:

1. Grid Challenge
  2. Beautiful Pairs
  3. Jim and the Orders
  4. Mark and Toys
- 

### **Common mistakes in code writing (2016-05-09 22:40)**

May 9, 2016

Julia likes to do some research on common mistakes in code writing.

Here are the articles she chooses to read:

1. [1]<https://www.quora.com/What-are-the-common-mistakes-made-by-beginner-competitive-programmers>
2. [2]<https://www.quora.com/How-do-you-debug-your-code-quickly-in-programming-contest-environments>
3. [3]<https://www.quora.com/What-are-some-macros-that-are-used-in-programming-contests>
4. [4]<https://www.quora.com/What-was-the-biggest-mistake-you-made-while-practicing-for-a-programming-contest>
5. [5]<https://www.quora.com/What-are-some-cool-C++-tricks-to-use-in-a-programming-contest>
6. [6]<http://www.cprogramming.com/tips/>  
favorite tips:
  1. be aware of loop invariants
  2. passing array elements to functions

Common mistakes in interviews:

6. [7]<http://www.geeksforgeeks.org/top-5-common-mistakes-in-technical-on-site-interviews/>
  7. [8]<http://warrington.ufl.edu/graduate/gbcs/docs/50jobinterviewmistakes.pdf>
  8. [9]<http://skillcrush.com/2016/02/16/common-tech-job-interview-mistakes/>
  9. [10]<http://www.thegeekstuff.com/2013/11/interview-mistakes/>
1. <https://www.quora.com/What-are-the-common-mistakes-made-by-beginner-competitive-programmers>
  2. <https://www.quora.com/How-do-you-debug-your-code-quickly-in-programming-contest-environments>
  3. <https://www.quora.com/What-are-some-macros-that-are-used-in-programming-contests>
  4. <https://www.quora.com/What-was-the-biggest-mistake-you-made-while-practicing-for-a-programming-contest>
  5. <https://www.quora.com/What-are-some-cool-C++-tricks-to-use-in-a-programming-contest>
  6. <http://www.cprogramming.com/tips/>
  7. <http://www.geeksforgeeks.org/top-5-common-mistakes-in-technical-on-site-interviews/>
  8. <http://warrington.ufl.edu/graduate/gbcs/docs/50jobinterviewmistakes.pdf>
  9. <http://skillcrush.com/2016/02/16/common-tech-job-interview-mistakes/>
  10. <http://www.thegeekstuff.com/2013/11/interview-mistakes/>

---

## External merge sort - Duplicate Elements of An Array (2016-05-10 20:24)

March 10, 2016

External merge sort, distributed algorithm discussion. Very good topic, and enjoy the reading.

External merge sort article:

- [1]<http://goo.gl/wbwAi8>

More reading:

[2][https://en.wikipedia.org/wiki/External\\_sorting](https://en.wikipedia.org/wiki/External_sorting)

[3]<http://www-inst.eecs.berkeley.edu/cs186/sp08/notes/08-Sorting.pdf>

Write down some notes, and tell what you learn through the study.

1. <http://goo.gl/wbWAi8>

2. [https://en.wikipedia.org/wiki/External\\_sorting](https://en.wikipedia.org/wiki/External_sorting)

3. <http://www-inst.eecs.berkeley.edu/~cs186/sp08/notes/08-Sorting.pdf>

---

## HackerRank - Connected Cell in a Grid - Warm up with Five Practices (2016-05-11 19:07)

May 11, 2016

Being a software programmer, it is easy to spend hours to read and catch up technologies, work on [1]new algorithm, but no coding, one day, or one week, even half month/ month.

So, warm up like sports. Julia chose the algorithm - Connected Cell In a Grid to warm up for a few hours.

Last time, less than 1 month ago, Julia did work on this algorithm - connected cell in a grid. And then, she started to warm up again.

Here is one of blogs last practice on April 16, 2016:

[2]<http://juliachencoding.blogspot.ca/2016/04/hackerrank-connected-cell-in-a-grid.html>

**First practice** , it takes her close to 60 minutes to write, fix issues. Use dimensional array, use queue to do BFS - breadth first search.

[3]<https://gist.github.com/jianminchen/41ac81c6ab6dc1345c88a8db2185e93f>

Here are mistakes:

1

Forget to add boundary check function, do boundary check (source code: line 108)

2

Forget to introduce neighbor\_X, neighbor\_Y (source code: line 90, 91)

3 N

ighbor\_X is mistakenly written as neighbor\_Y, so wrong answer; Debug the code and find the issue. It takes more than 20 minutes, a lot of stress. (source code: line 96)

So, it is excellent chance to learn and improve the performance.

Write a small function to debug the code, figure out the wrong answer issue –  
testRoutine, source code: line 36.

[4]<https://gist.github.com/jianminchen/41ac81c6ab6dc1345c88a8db2185e93f>

**Second practice** , using queue, but use jagged array: (20 minutes to write)

[5]<https://gist.github.com/jianminchen/9cf4702b56dc7a57ec6b699449b89593>

**Third practice** , use DFS – recursive function, which also returns the count.

[6]<https://gist.github.com/jianminchen/2279967c64a712fb138355b69a6201dd>

**Fourth practice** , using DFS – recursive function, but use an argument – reference int to track value

[7]<https://gist.github.com/jianminchen/589f05f5c87bb6e018de0a472d8a3b6f>

**Fifth practice** , using stack instead of recursive function, implement the DFS algorithm:

[8]<https://gist.github.com/jianminchen/11725341e9474e0503950616c9ad1848>

**Question and answer :**

1 1.

What do you learn through the warm up? Do you learn some better ways to fix the bugs?

It is better to write down the functions need **ed** to help the task, this way, you will be more efficient:

-

1. Calculate the key 2. Boundary check 3. Maximum value search 4. Using queue to do search

2 2.

Why do you do warm up this time? What are the advantages?

Julia still remembers the favorite tip to work on the tasks:

1.

Just mark the visited node as 0 from value 1

2.

Update node value from 1 to 0 before it is added to the queue

3.

Use key = row \* 10 + col, since row < 10, col < 10 to track each node in the queue

Julia likes to write code and do some warm up, therefore, she can get more experience; she tries to improve performance to 20 minutes for this kind of DFS, BFS, matrix, search algorithm.

3. Do you reproduce the experience of high stress to trouble shooting and work on bug fix?

Julia reproduced the issue of high stress, she could not fix the bug on her first practice. So, she wrote a small debug function try to figure out; actually, it is a mistake in writing.

Next time, reexamine every line of code, every variable, every executable path, when the code is executed. Do not depend on debugging, running the code, because stress level is high.

1. <https://www.blogger.com/null>

2. <http://juliachencoding.blogspot.ca/2016/04/hackerrank-connected-cell-in-grid.html>



3. <https://gist.github.com/jianminchen/41ac81c6ab6dc1345c88a8db2185e93f>
  4. <https://gist.github.com/jianminchen/41ac81c6ab6dc1345c88a8db2185e93f>
  5. <https://gist.github.com/jianminchen/9cf4702b56dc7a57ec6b699449b89593>
  6. <https://gist.github.com/jianminchen/2279967c64a712fb138355b69a6201dd>
  7. <https://gist.github.com/jianminchen/589f05f5c87bb6e018de0a472d8a3b6f>
  8. <https://gist.github.com/jianminchen/11725341e9474e0503950616c9ad1848>
- 

## **HackerRank – Connected Cell in a Grid - Warm up with Five Practices (II) (2016-05-11 19:15)**

May 11, 2016

Being a software programmer, it is easy to spend hours to read and catch up technologies, work on [1]new algorithm, but no coding, one day, or one week, even half month/ month.

So, warm up like sports. Julia chose the algorithm - Connected Cell In a Grid to warm up for a few hours.

Last time, less than 1 month ago, Julia did work on this algorithm – connected cell in a grid. And then, she started to warm up again.

Here is one of blogs last practice on April 16, 2016:

[2]<http://juliachencoding.blogspot.ca/2016/04/hackerrank-connected-cell-i-n-grid.html>

### **Warmup coding:**

**Her practice** , using queue, but use jagged array: (20 minutes to write), using queue, jagged array,

[3]<https://gist.github.com/jianminchen/9cf4702b56dc7a57ec6b699449b89593>

### **Question and answer :**

1. How is the warmup experience?

Because this is the second one, after 1 hour writing, debugging the code in first practice using dimensional array/ queue, this one is much easy. Just replace the dimension array using jagged array.

1. <https://www.blogger.com/null>

2. <http://juliachencoding.blogspot.ca/2016/04/hackerrank-connected-cell-in-grid.html>

3. <https://gist.github.com/jianminchen/9cf4702b56dc7a57ec6b699449b89593>

---

## **HackerRank – Connected Cell in a Grid - Warm up with Five Practices (III) (2016-05-11 19:19)**

May 11, 2016

Being a software programmer, it is easy to spend hours to read and catch up technologies, work on [1]new algorithm, but no coding, one day, or one week, even half month/ month.

So, warm up like sports. Julia chose the algorithm - Connected Cell In a Grid to warm up for a few hours.

Last time, less than 1 month ago, Julia did work on this algorithm – connected cell in a grid. And then, she started to warm up again.

Here is one of blogs last practice on April 16, 2016:

[2]<http://juliachencoding.blogspot.ca/2016/04/hackerrank-connected-cell-i-n-grid.html>

**Third practice** , use DFS – recursive function, which also returns the count.

[3]<https://gist.github.com/jianminchen/2279967c64a712fb138355b69a6201dd>

Question and Answer:

1. What do you like this approach? DFS using recursive function, which returns the count?

It is easy to write, less error prone; most efficient in time consumption.

1. <https://www.blogger.com/null>

2. <http://juliachencoding.blogspot.ca/2016/04/hackerrank-connected-cell-in-grid.html>

3. <https://gist.github.com/jianminchen/2279967c64a712fb138355b69a6201dd>

---

## HackerRank – Connected Cell in a Grid - Warm up with Five Practices (IV) (2016-05-11 19:23)

May 11, 2016

Warm up an algorithm like tennis sports, work on different strokes before she plays matches. Julia chose the algorithm - Connected Cell In a Grid to warm up for a few hours.

Last time, less than 1 month ago, Julia did work on this algorithm – connected cell in a grid. And then, she started to warm up again.

Here is one of blogs last practice on April 16, 2016:

[1]<http://juliachencoding.blogspot.ca/2016/04/hackerrank-connected-cell-in-grid.html>

**Fourth practice** , using DFS – recursive function, but use an argument – reference int to track value

[2]<https://gist.github.com/jianminchen/589f05f5c87bb6e018de0a472d8a3b6f>

Question and Answer:

1. What do you like the approach - DFS, recursive function, use an argument - reference to track the count?

Julia likes to use an argument to track the count, and let the recursive function return void.

1. <http://juliachencoding.blogspot.ca/2016/04/hackerrank-connected-cell-in-grid.html>

2. <https://gist.github.com/jianminchen/589f05f5c87bb6e018de0a472d8a3b6f>

---

## HackerRank – Connected Cell in a Grid - Warm up with Five Practices (V) (2016-05-11 19:30)

May 11, 2016

Being a software programmer, it is easy to spend hours to read and catch up technologies, work on [1]new algorithm, but no coding, one day, or one week, even half month/ month.

So, warm up like sports. Julia chose the algorithm - Connected Cell In a Grid to warm up for a few hours.

Last time, less than 1 month ago, Julia did work on this algorithm – connected cell in a grid. And then, she started to warm up again.

Here is one of blogs last practice on April 16, 2016:

[2]<http://juliachencoding.blogspot.ca/2016/04/hackerrank-connected-cell-in-grid.html>

**Fifth practice** , using stack instead of recursive function, implement the DFS algorithm:

[3]<https://gist.github.com/jianminchen/11725341e9474e0503950616c9ad1848>

Question and Answer:

1. What do you like the approach - using stack, DFS algorithm?

To write using stack is similar to using queue, but the search is DFS instead of BFS. Julia does not have time to build a test case to compare the order of visited nodes this time.

1. <https://www.blogger.com/null>

2. <http://juliachencoding.blogspot.ca/2016/04/hackerrank-connected-cell-in-grid.html>

3. <https://gist.github.com/jianminchen/11725341e9474e0503950616c9ad1848>

---

### **Leetcode 317 - shortest distance from all building - a warm up practice (2016-05-11 19:33)**

May 11, 2016

Julia spent time to rewrite the algorithm on Leetcode 317.

Here is her last practice on January, 2016. The blog is top 3 most visited blog - 138 visit up to May 11, 2016. So, she is encouraged to rewrite the code.

[1]<http://juliachencoding.blogspot.ca/2016/01/leetcode-317.html>

Her practice:

[2]<https://gist.github.com/jianminchen/f06a6a25e382bb6efabc43049dc69ecc>

She likes to do more writing on this algorithm, try to figure out ways to improve the performance.

1. <http://juliachencoding.blogspot.ca/2016/01/leetcode-317.html>

2. <https://gist.github.com/jianminchen/f06a6a25e382bb6efabc43049dc69ecc>

---

### **Leetcode 239: sliding window maximum (2016-05-11 19:44)**

May 11, 2016

[1]<http://blog.csdn.net/xudli/article/details/46955257>

[2]<http://blog.csdn.net/xudli/article/details/46955257>

[3]<http://yuanhsh.iteye.com/blog/2190852>

Read the blog:

[4]<http://www.cnblogs.com/yrbbest/p/5004596.html>

Julia, please write down your C # practice:

**Question and Answer :**

1. How long do you study the problem? What do you learn?

Julia spent over 30 minutes on this question. And she learns that deque is excellent data structure to achieve optimal time complexity and complete the task.

2. Can you tell the concrete the example and show how to solve the problem?

Just think like a greedy algorithm. Make the special data structure like deque, every node is added to the deque once and also removed once. Keep the deque as small as possible, in other words, if the element cannot be the maximum of sliding window, then it should be removed from deque right away.

So, time complexity is  $O(N)$ .

Also, make it greedy, inside the deque, all elements are sorted by descending order from left to right.

For example, windows size with 4, [1, 3, -1, 2], no matter what next numbers are, 1 and -1 are never going to be a maximal as the window moving. The queue should like [3,2].

So, to maintain the queue in order.

add node in queue from right side only; but remove nodes from both end, just before a node is added.

3. Time complexity analysis:

For those solutions - you can come out:

A. Use heap, or other solutions, time complexity can up to  $O(n \log n)$ .

w - window size

n - array size

Building a heap, time complexity  $O(w \log W)$

...

so, if  $w \ll n$ , close to  $O(n)$ , but if  $w = 3/n$  or  $4/n$ , the running time goes up to  $O(n \log n)$ .

B. ?

4. Learn Java Dequeue class, and C # linkedList:

API:

**First :**

getFirst

RemoveFirst

**Last:**

addLast

getLast

removeLast

isEmpty

1. <http://blog.csdn.net/xudli/article/details/46955257>
  2. <http://blog.csdn.net/xudli/article/details/46955257>
  3. <http://yuanhsh.iteye.com/blog/2190852>
  4. <http://www.cnblogs.com/yrbbest/p/5004596.html>
- 

## Leetcode 48: rotate image (2016-05-11 21:49)

May 11, 2016

Julia likes to study the algorithm: rotate image

[1]<http://fisherlei.blogspot.ca/2013/01/leetcode-rotate-image.html>

[2]<http://shunrang.blogspot.ca/2015/10/rotate-image-and-spiral-matrix.html>

1. <http://fisherlei.blogspot.ca/2013/01/leetcode-rotate-image.html>
  2. <http://shunrang.blogspot.ca/2015/10/rotate-image-and-spiral-matrix.html>
- 

## Algorithm: Rotate array by one element (2016-05-11 21:52)

May 11, 2016

Quickly find out the blogs related to "Rotate array by one element", look for optimal solution. Julia, it is not so important to show optimal solution, but at least, never waste the opportunity to show that you are a hacker, willing to solve any problem.

Anything about rotate two dimensional array:

1.  
[1]<http://stackoverflow.com/questions/42519/how-do-you-rotate-a-two-dimensional-array>
2.  
[2]<https://blogs.msdn.microsoft.com/oldnewthing/20080902-00/?p=21003>
3.  
[3]<http://geekswithblogs.net/cwilliams/archive/2008/06/16/122906.aspx>
4. Matrix transpose  
[4]<https://en.wikipedia.org/wiki/Transpose>

Question and answer:

1. How many hours do you spend to work on this problem?

Reading 2 hours +.

1. <http://stackoverflow.com/questions/42519/how-do-you-rotate-a-two-dimensional-array>
  2. <https://blogs.msdn.microsoft.com/oldnewthing/20080902-00/?p=21003>
  3. <http://geekswithblogs.net/cwilliams/archive/2008/06/16/122906.aspx>
  4. <https://en.wikipedia.org/wiki/Transpose>
- 

## Leetcode 17: Phone Number - Practice using DFS/ Stack and BFS/ Queue (2016-05-12 19:07)

May 12, 2016

Write two more practice using Queue/ BFS and DFS/ stack on this leetcode question:

Using Queue/ BFS

[1]<https://gist.github.com/jianminchen/5691a3488a2ef4f0660194502ae15f68>

Using Stack/ DFS

[2]<https://gist.github.com/jianminchen/872bf70039fa8c61ff208b34a591c8ec>

Previous blogs about practice:

April 21, 2016 using recursive function, DFS

[3]<http://juliachencoding.blogspot.ca/2016/04/window-sum.html>

January 24, 2016 using recursive function, DFS

[4]<http://juliachencoding.blogspot.ca/2016/01/leetcode-17-letter-combinations-of.html>

### Question and Answer:

1. What is the difference using Queue and Stack on space usage?

For example, using Queue/ BFS, source code line:

Line 31: `IList<string> list = letterCombination("234");` // test result: "abc", "def", so  $3 \times 3 = 9$  cases. If the string is "2345678", 7 digits string, then, when first string is added to the list, on line 86, the queue's count is around  $3^7 = 9 * 9 * 9 * 3$ , more than 2000 records in the queue. However, using Stack/ DFS, source code line 31, the string is "2345678", when the first phone number is added to the list, the stack size is around  $3 * 7 = 20$ , around 20 records in the stack. So, it takes more space to use queue/ BFS on this problem solving, not efficient on space usage.

2. Please look into string, stringBuilder on this problem solving.

Read more blogs on this discussion:

1. <https://gist.github.com/jianminchen/5691a3488a2ef4f0660194502ae15f68>
  2. <https://gist.github.com/jianminchen/872bf70039fa8c61ff208b34a591c8ec>
  3. <http://juliachencoding.blogspot.ca/2016/04/window-sum.html>
  4. <http://juliachencoding.blogspot.ca/2016/01/leetcode-17-letter-combinations-of.html>
- 

## Some algorithms to study (2016-05-13 22:56)

May 13, 2016

Here are some algorithm problems Julia likes to review, what she did is to spend 2 hours to go over Glassdoor.com and go over the interview questions on companies (A, F, L, M, G) from 2016 backward to January 2015.

As a software programmer, Julia knows the value of good thinker in algorithm, problem solving; and write readable, clean code to implement the idea using Java, C++, C #, JavaScript, learn through simple problem solving every day.

Put problems in the groups to help study:

### Array :

1.

Leetcode 23 - Merge K sorted array

[1]<http://www.geeksforgeeks.org/merge-k-sorted-arrays/>

- read stack, queue, priority queue - definition - May 13, 2016

[2]<https://msdn.microsoft.com/en-us/library/4ef4dae9.aspx>

read through priority queue API document, and understand more if I can.

2. Leetcode 215: Find the kth largest element in an array

[3]<http://www.geeksforgeeks.org/kth-smallestlargest-element- unsorted-array/>

### Tree problems:

1. Check if given binary tree is a mirror.

2. **Serialize and deserialize the tree**

3. Get mirror image of a BST

4. Connect nodes at the same level in a binary tree

[4]<http://www.geeksforgeeks.org/connect-nodes-at-same-level/> (? bug)

better one:

[5]<http://javabypatel.blogspot.ca/2015/08/connect-nodes-at-same-level-in-binary-tree-using-constant-extra-space.html>

**Leetcode:**

1. Leetcode – 3 sum
2. Leetcode 215 - Kth largest element in the array (quick select)

**Linked List:**

1. If we have a linked list, and if I give you a number n, then first n node of the linked list should get reversed next n node should as it is, next n should get reversed like wise.

For example, n=3

**String :**

1. Given an array of strings, need to check every string in the array is a palindrome or not. If it is, we have to print it.

[6]<http://codereview.stackexchange.com/questions/73542/hashtable-implementation>

[7]<http://www.geeksforgeeks.org/k-largestor-smallest-elements- in-an-array/>

2. Write a function to determine the longest palindromic substring of a given string.
3. Find the minimum number of palindromes in a string.

**Sorting:**

1. Write a merge sort

**Hashtable :**

1. How do you implement a hash table?

**Dynamic Programming:****1. optimal coins**

2. Ladder of height 100, u can use jumps {1,2,3 } how many different ways can u reach 100.

**DFS/ BFS/ Search:**

1. Transform a [1,0] matrix grid into matrix grid of manhattan distance between closest 1's
2. Given a certain number of tasks with specific start and end times, find the maximum number of jobs that can be done

**Misc:**

1. Check whether rectangle overlap
2. Reverse a matrix in a given sequence

[8]<http://iwillgetthatjobatgoogle.tumblr.com/post/11352414963/ implement-a-hash-table>

3. There is a 9 digit number, you need to rearrange the number and make just bigger number then this one  
pivot element - google to find a solution

4. There are 9 buckets, Each bucket contains some chocolates (number of chocolates labeled on the bucket), a kid is there, He takes 1 second to eat all the chocolates of 1 bucket and as puts the bucket back, bucket gets filled again with the half chocolate then the original



bucket had. kid has n second, find a way in which he can eat max chocolate in n seconds.

5. Find the next largest number.

6. Solving the jigsaw puzzle. Input is the pieces of the puzzle and a method taking two pieces as input and returning true if they fit.

1. <http://www.geeksforgeeks.org/merge-k-sorted-arrays/>

2. <https://msdn.microsoft.com/en-us/library/4ef4dae9.aspx>

3. <http://www.geeksforgeeks.org/kth-smallestlargest-element-unsorted-array/>

4. <http://www.geeksforgeeks.org/connect-nodes-at-same-level/>

5. <http://javabypatel.blogspot.ca/2015/08/connect-nodes-at-same-level-in-binary-tree-using-constant-extra-space.html>

6. <http://codereview.stackexchange.com/questions/73542/hashtable-implementation>

7. <http://www.geeksforgeeks.org/k-largestor-smallest-elements-in-an-array/>

8. <http://iwillgetthatjobatgoogle.tumblr.com/post/11352414963/implement-a-hash-table>

---

## **Top 10 strategies Work Hard for Entrepreneurs (2016-05-14 12:59)**

May 14, 2016

Watch the video:

[1][https://www.youtube.com/watch?v=pkH5EwEI\\_LU](https://www.youtube.com/watch?v=pkH5EwEI_LU)

Here are some notes about work hard:

1. Donald Trump: He heard the meaningful way from golf player: about work hard and luck.

Great golf player: Gary player, "The hard you work, the lucky you get.", all I have to know that I am working very hard, the hard you work, the lucky you get.

2. Bill Gates: 2 weeks in a year, work and think: think week. Think about things, reading book, do we have right priorities.

3. Oprah: Talk to girls, real work: Figure out where your power bases. Alignment of personality, gifts you have to give, work on yourself. Fulfill self up, keep your cup full.

Do not afraid. Embrace the full of yourself. Honor yourself.

4. Sean Combs (American Rapper): Hard work

5. Jason Calacanis (Internet entrepreneur and blogger): Work 3 jobs to go to college, father did not pay tax, and then restaurant was taken.

6. Elon Musk: work on super hard. 7 days a week, programming all the time. Every waking hour. 50 hours/ week, you work 100 hours/ week.

7. Derek Jeter (baseball shortstop): never satisfied. Some one is better. You have to work hard than other people.

8. Grant Cardone: work ethics so high, unbelievable , know you work hard, therefore, you can be top 10 - 20

%.

People know you because you have a good ethnics - He works, he produces, vibrate the level, tremendous work ethnics.

Ask you a question:

" **Do people know you because of your unbelievable work ethics ?**" If you have to get top 10 percent.

How good your skills are. Discipline yourself even if you have billionaire dollars.

American way - work ethics.

9. Alex Morgan (soccer player): practice more than your opponent and teammate, mental prepare every day, not sucking off, put a lot of hours. No secret why you are succeed.

10. Cristiano Ronaldo (football player): You have to work. Work every day.

Put in mind, body can be improved. Work, is Part of me, not the attitude of "I have to do the work". As an athlete, small detail like drills, conditioning helps.

1. [https://www.youtube.com/watch?v=pkH5EwEI\\_LU](https://www.youtube.com/watch?v=pkH5EwEI_LU)

---

### **Les Brown's motivation talk (2016-05-14 14:08)**

May 14, 2016

Julia likes the motivation speech and she found a favorite talk. She likes the talk from Les Brown.

Let Julia start to write this blog using one of sentences in the talk:

"Some people know what is happening, some people make things happen, some people do not know what is happening."

Julia likes to make things happen, work on building confidence, and she thinks that Les Brown can be a great motivation for her.

[1]<https://www.youtube.com/watch?v=5zEJGsvXhXg>

1. Believe in yourself

2. Amaze your customers

3. Take full responsibility for your life

4. Stand up to yourself

5. Go all out

Mind likes garden. Weeds can grow everywhere, but exotic flower has to be grown in certain conditions.

6. Stay busy

Sometimes life is in slump, continue to execute, stay busy.

7. **Give more than you are paid for**

Courage to work on - Start to work early - Develop a habit to work for more than you are paid for.

Work on the path and leave a trail.

8. Someone's opinion is not your reality

9. You are different

10. Don't stop running towards your dream

1. <https://www.youtube.com/watch?v=5zEJGsvXhxg>

---

## **Roger Federer's Top 10 Rules For Success (2016-05-15 00:02)**

May 14, 2016

Julia had some bad behavior one time on tennis court in 2014, she hit the tennis racket on the net when she made the mistake on the court. The opponent told her that it was just a game, relax and enjoy it. Now, she is getting better, she learns to celebrate when she wins a point instead of showing disappointment when she loses a point. Power fist is her favorite.

[1][https://www.youtube.com/watch?v=rAdn\\_JGq5bQ](https://www.youtube.com/watch?v=rAdn_JGq5bQ)

1. Question yourself

2. Work on your strengths

3. Enjoy what you're doing

4. Have structured goals

Short term goal

Long term goal - without goal and target, cannot compete totally.

Do it the Roger Federer Way

Question yourself.

5. Build a great team

Have a team, good friends around you. Some people question me, really care about me.

6. Be prepared

7. Learn from your mistakes

Do not smash rackets, do not complain. No more commentary on his own points. Mental focus.

8. Separate work from life

9. Chase your dreams

10. Have fun

Roger Federer CNN Talk Asia 2013 Interview Part 1

[2]<https://www.youtube.com/watch?v=ixesWSqfJx0> &feature=youtu.be

"Enjoy yourself. Train hard, no regrets, all I can do is to do my best. No matter the outcome. Be honest. Losing is fine. Everything about win is a bonus."

1. [https://www.youtube.com/watch?v=rAdn\\_JGq5bQ](https://www.youtube.com/watch?v=rAdn_JGq5bQ)

2. <https://www.youtube.com/watch?v=ixesWSqfJx0> &feature=youtu.be

---

#### **Leetcode 4: Median of two sorted array - a warmup practice (2016-05-15 14:28)**

May 15, 2015

Review the leetcode 4: Median of two sorted array, warm up the algorithm.

Here is the last practice:

[1][https://github.com/jianminchen/Leetcode\\_C/blob/master/4MedianOfTwoSortedArrays.cs](https://github.com/jianminchen/Leetcode_C/blob/master/4MedianOfTwoSortedArrays.cs)

Step 1:

Read the last practice 10 - 20 minutes on May 15, 2016

- write down the ideas:

1. Brute force solution:  $O(n \log n)$

2. Linear solution:  $O(n)$

3. Transform to the kth element problem first, and then:

Try to get rid of  $k/2$  elements once, so the algorithm will go to  $\log k$ ,  $k = (m+n)/2$

Some analysis and reasoning:

Assuming that A and B both arrays are with length  $> k/2$ , and then, compare  $A[k/2-1]$  and  $B[k/2-1]$ .

Read the blog again:

[2]<http://blog.csdn.net/yutianzuijin/article/details/11499917>

[3]<http://blog.csdn.net/zxzy1988/article/details/8587244>

Step 2:

Spend 30 minutes to read and then write some code.

Related to Leetcode 215: Find kth largest element in the array.

### Question and Answer :

1. Write down what you learn through this practice.

Answer: The algorithm is very well defined, is a special case of find kth element in two sorted arrays.

2. Algorithms learning and important thing to learn:

Answer:

1. Always know how to solve the problem using naive solution, brute force one first.

$O(n \log n) \rightarrow O(n) \rightarrow O(\log(m+n)/2)$

2. And then, discuss the improvement.

1. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/4MedianOfTwoSortedArrays.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/4MedianOfTwoSortedArrays.cs)

2. <http://blog.csdn.net/yutianzuijin/article/details/11499917>

3. <http://blog.csdn.net/zxzy1988/article/details/8587244>

---

## Leetcode 215: Find kth largest element in the array (2016-05-15 15:46)

May 15, 2016

Problem statement:

[1]<https://leetcode.com/problems/kth-largest-element-in-an-array/>

Find the kth largest element in an unsorted array. Note that it is the kth largest element in the sorted order, not the kth distinct element.

Read some blogs about this problem:

[2]<http://www.jianshu.com/p/f52a88550588>

Julia likes to spend 10 - 20 minutes to review "Quick Select" - the idea, similar to Quick Sort, most important part is to partition.

Time complexity:  $O(n \log n)$  in quicksort, but  $O(n)$  in quick select.

1. Quick sort: Time complexity:  $O(n \log n)$

2. Use Heap sort,  $O(k \log N)$

Java Solution, using priority queue, code to study:

[3]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/215.kth-largest-element-in-an-array.java>

Spend 20 minutes to read the blog: 4:23pm - 4:43 pm

1. [4]<http://www.cnblogs.com/yuzhangcmu/p/4164807.html>

2. [5]<https://en.wikipedia.org/wiki/Introsselect> (read 10 minutes - 4:30pm - 4:40pm)

3. [6]<http://stackoverflow.com/questions/7559608/median-of-three-values-strategy>

4. [7]<http://www.quora.com/What-is-the-most-efficient-algorithm-to-find-the-kth-smallest-element-in-an-array-having-n-elements>

5. [8]<http://www.geeksforgeeks.org/k-largestor-smallest-elements-in-an-array/> (4:50pm - 5:20pm)

30 minutes to review the solutions, write down short notes:

6 methods:

1. Use bubble k times -  $O(nk)$

Modify bubble sort to run the outer loop at most k times

Like bubble sort, other sorting algorithms like selection sort can also be modified to get the k largest element.

2. Use temporary array

Time complexity:  $O((n-k) * k)$

3. Use sorting

1. Sort the elements in descending order in  $O(n \log n)$

4. Print the first k numbers of the sorted array  $O(k)$

Time complexity:  $O(n \log n)$

4. Use Max Heap

1. Build a Max Heap tree in  $O(n)$

2. Use Extract Max k times to get k maximum elements from the Max Heap  $O(k \log n)$

Time complexity:  $O(n + k \log n)$

5. Use order statistics:

1) use order statistics algorithm to find the kth largest element.

Julia, take some time to review the article: see [9]the topic selection in worst-case linear time  $O(n)$ . (10 - 15 minutes)  
Write down main ideas using your own words:  
Use a deterministic algorithm that runs in  $O(n)$  in the worst case.

2)

1. <https://leetcode.com/problems/kth-largest-element-in-an-array/>
  2. <http://www.jianshu.com/p/f52a88550588>
  3. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/215.kth-largest-element-in-an-array.java>
  4. <http://www.cnblogs.com/yuzhangcmu/p/4164807.html>
  5. <https://en.wikipedia.org/wiki/Introselect>
  6. <http://stackoverflow.com/questions/7559608/median-of-three-values-strategy>
  7. <http://www.quora.com/What-is-the-most-efficient-algorithm-to-find-the-kth-smallest-element-in-an-array-having-n-elements>
  8. <http://www.geeksforgeeks.org/k-largestor-smallest-elements-in-an-array/>
  9. <http://www.cse.ust.hk/~dekai/271/notes/L05/L05.pdf>
- 

## Connect nodes at the same level in a binary tree (2016-05-16 22:44)

May 16, 2016

Connect nodes at the same level in a binary tree

[1]<http://www.geeksforgeeks.org/connect-nodes-at-same-level/> (? bug)

**better one:**

[2]<http://javabypatel.blogspot.ca/2015/08/connect-nodes-at-same-level-in-binary-tree-using-constant-extra-space.html>

Using Queue -

[3]<http://javabypatel.blogspot.ca/2015/08/connect-nodes-at-same-level-in-binary-tree.html>

Question and answer:

1. How long do you study the problem? What do you learn?

Julia spent more than 1 hour to study the blogs above. She will write down her own notes about the test case, analysis and solution, and post them in the blog. Encourage herself to write, to share and to improve her confidence on problem solving.

Will come back soon.

1. <http://www.geeksforgeeks.org/connect-nodes-at-same-level/>
  2. <http://javabypatel.blogspot.ca/2015/08/connect-nodes-at-same-level-in-binary-tree-using-constant-extra-space.html>
  3. <http://javabypatel.blogspot.ca/2015/08/connect-nodes-at-same-level-in-binary-tree.html>
-

## web technology - a short research in Vancouver technology industry (2016-05-16 23:21)

May 16, 2016

As a software programmer, Julia noticed that she was out of date from 2010 - 2015. She started to do a little research here and there, try to catch up small details. 10 - 20 minutes a time, "What is hot in Vancouver area - IT skills".

Starting from August 2015, she started to learn OO principle: S.O.L.I.D.. She came cross the topic by a small talk with a manager from Microsoft in 2015. She started to follow the principles and write a lot of small classes and functions. She also feels less stressful to handle frequent code change in live website.

She noticed that in the city of Vancouver, the principle is also very popular this year 2016.

### 1. Hootsuite - May 16 - 2016 Junior Software Developer

- Automated testing using tools like Casper.js, Nightwatch.js or Selenium
- Javascript, CSS & HTML
- JS libraries and frameworks like jQuery, Backbone, Angular, React, Underscore
- CSS pre-processor(s) like Sass, Less, etc...
- PHP
- Using relational (MySQL) or NoSQL (Mongo) data stores
- Working with REST APIs
- Agile philosophies and continuous delivery

Julia, you are learning MVC, Angular JS, also, check out REST APIs.

### 2. Simba - May 16, 2016 Senior C++ software developer

superlative code, top quality code reviews, and comprehensive automated tests.

BI tools: Tableau, Microsoft power BI and Lumira - Limira?

using C++ memory management and performance analysis tools - what tools? Google it!

Visual Studio and Xcode - (Julia's comment: Xcode? )

3+ years of C++, STL, smart pointers, RAIL, and concurrent (multi-threaded) programming

### 3. Global Relay - automation developer

Automated testing frameworks

REST and JSON

GIT and SVN

Scrum and Kanban

Build tools: Maven

Continuous integration system: Bamboo



Network protocols: HTTP, TLS and TCP

Jbehave and Selenium Web Driver

Thrift API

Julia, can you name 2 service oriented architectures? REST and JSON

---

## Human resource analytical articles (2016-05-17 08:36)

May 17, 2016

As a software programmer, Julia likes every effort she put in to help her grow, and also every company she had chance to explore, get educated by reading, talking.

One of her interest is about human resource analytics. Her favorite reading blogs:

[1]<https://www.linkedin.com/company/1360576?trk=prof-exp-company-name>

one of articles:

[2]<http://www.miamiherald.com/news/business/biz-columns-blogs/cindy-krischer-goodman/article64841942.html>

She likes to explore more in the city of Vancouver.

1. <https://www.linkedin.com/company/1360576?trk=prof-exp-company-name>

2. <http://www.miamiherald.com/news/business/biz-columns-blogs/cindy-krischer-goodman/article64841942.html>

---

## Leetcode 16 - 3 sum closest (2016-05-17 19:45)

May 17, 2016

Julia likes to write some code after she reads the blog:

[1]<http://www.cnblogs.com/TenosDolt/p/3649607.html>

Write C # practice on Leetcode 16 - 3 sum closest solution:

[2]<https://gist.github.com/jianminchen/d23598296e7ba665b6d45ae4acfeb600>

First practice on Leetcode 16 - 3 sum closest solution on

[3][https://github.com/jianminchen/Leetcode\\_C-/blob/master/3sumCloset.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/3sumCloset.cs)

Previous blog:

[4]<http://juliachencoding.blogspot.ca/2016/01/leetcode-1-15-15-two-sum-3-sum-3-sum.html>

Question and Answer:

1. What do you learn through the practice this time?

Julia learns the importance to do static analysis on the code. Do not rush, make sure every variable/ name is making sense, every line of code is best she can present, every executable path should be examined. Every scope of variable is close to minimum as possible. Walk through the examination steps loudly.

**2. What common steps Julia likes to build a ritual after her first writing - C # code - after she failed to present a two sum algorithm on May 4, 2016?**

Answer:

1. Julia likes to examine every line of code, see if she can improve the presentation;
2. Check every variable, scope, meaningful name
3. Check every executable path, make sure that no bug
4. What test case will be executed on the line.
5. Avoid early return error, other common errors.
6. Read the code, speak out what she is doing on reviewing of each line, each variable. Talk about the change she likes to make, thing found needs to be taken care of.

**3. What is most important to solve the problem?**

Using two sum problem solving technique, extend the method to solve the 3 sum closest one.

The idea is most important and also know how to do time complexity analysis -  $O(n^2)$  solution - best one.

Since  $O(n^2)$  is the best time complexity we can solve the problem, sorting takes less than  $O(n^2)$ ; certainly, array can be sorted first.

So, here is the analysis Julia does for the problem - 3 sum closest.

Idea:

- \* 1. Sorting takes  $O(n \log n)$  for an array
- \* 2. And then, choose  $(i, j, k)$ , assuming  $i < j < k$ , iterate on variable  $i$ ,
- \* we need to find two sum problem for each  $i$ , new target = target - nums[i]
- \* since the array is sorted, two pointer solution can be used. One is the beginning
- \* of array, another one is the end of the array.
- \*
- \* So, this step 2 process takes  $O(n^2)$  time
- \*
- \* Overall, the algorithm will take  $O(n^2)$  time complexity
- \*
- \*
- \* read the C # Array API - 10 minutes
- \* [5][https://msdn.microsoft.com/en-us/library/6tf1f0bc\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/6tf1f0bc(v=vs.110).aspx)

1. <http://www.cnblogs.com/TenosDoIt/p/3649607.html>

2. <https://gist.github.com/jianminchen/d23598296e7ba665b6d45ae4acfeb600>

3. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/3sumClosest.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/3sumClosest.cs)
  4. <http://juliachencoding.blogspot.ca/2016/01/leetcode-1-15-15-two-sum-3-sum-3-sum.html>
  5. [https://msdn.microsoft.com/en-us/library/6tf1f0bc\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/6tf1f0bc(v=vs.110).aspx)
- 

## System Design (2016-05-17 19:46)

May 17, 2016

System Design:

[1]<http://blog.gainlo.co/index.php/category/system-design-interview-questions/>

tiny URL system design:

<http://blog.gainlo.co/index.php/2016/03/08/system-design-interview-question-create-tinyurl-system/>

Amazon dynamo:

<http://www.allthingsdistributed.com/files/amazon-dynamo-sosp2007.pdf>

Julia loves this article talking about cache system design; once she does some code review on basic LRU, she starts to enjoy the computer science, she just loves the engineering, good ideas to solve problems.

[2]<http://goo.gl/pL5ee4>

classical reader/ writer problem / lock / multiple shards/ commit logs/ memcached

Reference:

1. LRU - C # practice

[3][http://juliachencoding.blogspot.ca/2016/04/leetcode-146-lru-cache\\_26.html](http://juliachencoding.blogspot.ca/2016/04/leetcode-146-lru-cache_26.html)

1. <http://blog.gainlo.co/index.php/category/system-design-interview-questions/>

2. <http://goo.gl/pL5ee4>

3. [http://juliachencoding.blogspot.ca/2016/04/leetcode-146-lru-cache\\_26.html](http://juliachencoding.blogspot.ca/2016/04/leetcode-146-lru-cache_26.html)

---

## Leetcode 15: 3 Sum (2016-05-17 22:20)

May 17, 2016

problem statement:

[1]<https://leetcode.com/problems/3sum/>

Given an array  $S$  of  $n$  integers, are there elements  $a, b, c$  in  $S$  such that  $a + b + c = 0$ ? Find all unique triplets in the array which gives the sum of zero.

Note:

- Elements in a triplet  $(a, b, c)$  must be in non-descending order. (ie,  $a \leq b \leq c$ )
- The solution set must not contain duplicate triplets.

Julia likes to write some code after she reads the blog:

[2]<http://www.cnblogs.com/TenosDolt/p/3649607.html>

Write her own practice using C #:

[3]<https://gist.github.com/jianminchen/9ca3f9982c1011ed4a658f83ca203549>

[4][https://github.com/jianminchen/Leetcode\\_C-/blob/master/15\\_3Sum.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/15_3Sum.cs)

Pass Leetcode online Judge

Line 62, Change the variable name from newTarget to twoSumTarget, since newTarget is not so clear, confusing.

[5]<https://gist.github.com/jianminchen/61c848a30b1e10364359d2d73c0338fe>

### Question and Answer:

1. How is your practice?

Answer: Julia tried to practice guidelines from "Clean Code", she learned a few things:

name a list variable to avoid using "list", HashSet variable does not call "set", force her to think about meaningful name.

For example, line 47 - keys, variable is declared as HashSet<string>.

1. She tried to put "two sum target" function (line 62 - line 103) as a while loop instead of standalone function;

2. Confused with target - 3 sum target with 2 sum target, two variables - naming change:

newTarget -> twoSumTarget (line 62)

3. 3 variables are replaced by an array: line 58,

trialTriplet

4. line 71 twoSumValue variable is named to explicitly tell the sum of 2 values, not 3 values.

2. Fun time?

2.1 Julia spent more than 45 minutes to go over C # Array class and all API,

Array.Sum(), use C # Array.Sum() instead of writing yourself.

2.2. 3 variables related to target value,

target, twoSumTarget, twoSumValue

Good time to read the blog: (10 - 20 minutes)

[6]<http://www.sigmainfy.com/blog/summary-of-ksum-problems.html>

3. Can you improve the code to get rid of using HashSet to check duplicate?

Answer:

Read the blog:

[7]<http://fisherlei.blogspot.ca/2013/01/leetcode-3-sum-solution.html>

line 21, 22: quick solution to avoid duplicate set:

```
while(start<end && num[start] == num[start-1]) start++;
```

```
while(start<end & & num[end] == num[end+1]) end-;
```

1. <https://leetcode.com/problems/3sum/>
  2. <http://www.cnblogs.com/TenosDoIt/p/3649607.html>
  3. <https://gist.github.com/jianminchen/9ca3f9982c1011ed4a658f83ca203549>
  4. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/15\\_3Sum.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/15_3Sum.cs)
  5. <https://gist.github.com/jianminchen/61c848a30b1e10364359d2d73c0338fe>
  6. <http://www.sigmainfy.com/blog/summary-of-ksum-problems.html>
  7. <http://fisherlei.blogspot.ca/2013/01/leetcode-3-sum-solution.html>
- 

## Leetcode 18: 4 Sum (2016-05-17 22:22)

May 17, 2016

Julia likes to write some code after she reads the blog:

[1]<http://www.cnblogs.com/TenosDoIt/p/3649607.html>

Write her own practice using C #:

Will come back.

1. <http://www.cnblogs.com/TenosDoIt/p/3649607.html>
- 

## Leetcode 236: Lowest common ancestor (2016-05-18 23:30)

May 18, 2016

Lowest common ancestor:

[1]<http://juliachencoding.blogspot.ca/2016/04/leetcode-236-lowest-common-ancestor-in.html>

Julia's C # practice on May 21, 2016

1. User Stack class ToArray API, keep stack iterator order

code has a bug - index out of range - line 93

[2]<https://gist.github.com/jianminchen/2bec8a70ef520e715240a226c01c7371>

[3][https://msdn.microsoft.com/en-us/library/415129w1\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/415129w1(v=vs.110).aspx)

2. Second warm up, just use List.AddRange(Stack), still keep stack iterator order

code has a bug - index out of range - line 93

[4]<https://gist.github.com/jianminchen/754f45023f60992211cb34bef6d6254f>

3. Third warm up, use the test case in the diagram 1, 2, 3, 4, 5, 6, 7, 8, 9, and then, find out the bug! Big surprise! Every line of code should be optimized, and then examined.

Fix the bug - index out of range - line 93

[5]<https://gist.github.com/jianminchen/99585d73e4196136fd87274174dec3b9>

4. Fourth warm up practice, change code while loop code, make it explicitly - loop variable  
[6]<https://gist.github.com/jianminchen/388114d99020972413464a072e5e9a9b>

Make the code more readable about line 93.

Review blog about post order traversal,

[7]<http://juliachencoding.blogspot.ca/2015/09/morris-order-post-order-traversal.html>

And then, we can easily find out that the order of output can be implemented using stack, and then, the stack - the root node is always the last one. We can find two nodes p, q through post order traversal, once any one of those two nodes is visited, the path from root to p or q can be retrieved from the stack. Use stack API, move stack content to a list.

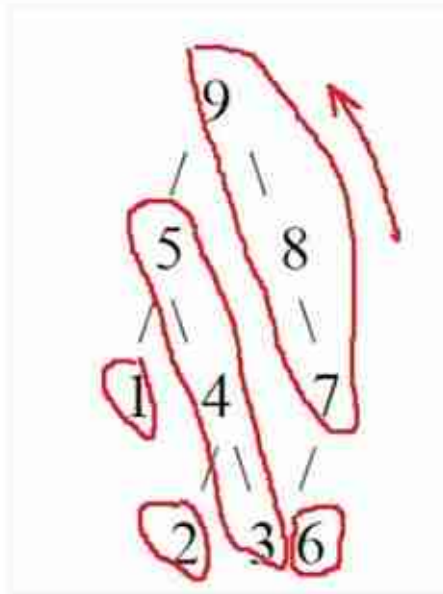
So clever, let us recall those images used in the above blog:

### Post order traversal - May 18, 2016



就是从最左边开始, 遍历五次,

1,  
2,  
3, 4, 5  
6,  
7, 8, 9,



The node may be visited more than once, it is better to check if it is one of two input nodes when first time to visit. Also, the stack is kind of tricky, you can peek, if the node is ready to add into post order traversal output, then pop it.

#### Questions and Answers:

1. Please write down what you learn through this practice, warmup? How long do you take?

It takes over 6 hours to get back in tree traversal algorithm.

The most important to remember that the tree's post order traversal only has 3 parts:

1. Push root node and its left child to the stack, repeat to the end of leftmost node.
2. And then, peek the node has right child or not. In the above diagram, N1 has no right child.

if there is one, denoted as N3, then, treat the node N3 as step 1 - push node and its left's child to the end of leftmost node.

else, it is ready to pop the node, and add it to traversal result.

2. Save a variable as a previous visited node, check the current node relationship to previous node, in the above diagram, 3 -> 4 -> 5, upward traversal through the stack, how do we know?  
previous node = current node's right child, in other words, right child is just visited, it is root node's turn to pop out.

2. Please write down steps of post order traversal in the above diagram:

In the above diagram,

step 1: 9 -> stack

2: 5 -> stack

3. 1 -> stack

4. check stack is empty or not,

5. not empty, peek the stack, the node N, see if previous is N's right child or not  
not true

6. Pop node with value 1

7. Peek stack's top node, and see if previous visited node is current node's right child or not  
it is not

8. Check its right child existing or not

9. It is existing, then, add current node to stack, repeat while loop...

Julia could not explain it very clearly, she spent over 2 hours to get lost in the code, played with simple test cases:

Null tree, tree with one node, tree with two nodes, tree with only left child, tree with only right child.

She just noticed that it is easy to get complicated while loop, much more complicated code, and then, spin inside and waste time.

It is better to delete everything and then start it from beginning.

3. Go through the code and explain what you learn through the practice:

Answer:

To make solution simple as possible, three things need to be taken care of:

1. First, downward, push nodes to stack, like 9->5->1, leftmost nodes to the stack, continuously push them to the stack.

2. Second, need to know when to pop out to add traversal list

1 to the list,

then, 5 is the top of stack, smart enough to know that it is turn to visit right child.

3. Third, need to take care of upward pop up, like 7->8->9

When you design the process, leave "set stack's top node's right child to runner" as last.

Then, take care of upward pop up process, it is a while loop, not a if checking - one time deal.

Like the above diagram, let 7 -> 8 -> 9 three continuously pop out.

And then, check the stack is empty or not, if it is empty, break the while loop.

For example, the last one, root node 9 is popped out, then, break the outside while loop, terminate the loop, otherwise, it is a dead loop.



Last, stack is not empty, we need to set stack's top node's right node as runner node, let iteration repeats.

Julia learned that in order to produce working code with no bug, take a lot of practice.

line 68: `while (stack.Count > 0 && (stack.Peek().right == null || stack.Peek().right == runner))`

```
{
runner = stack.Pop();
}
if (stack.Count == 0)
{
break;
}
else
{
runner = stack.Peek().right;
}
```

#### 4. What are most difficult to learn through your practice?

Answer:

When the node is added to post order traversal list, in the above diagram,

Node with value 1 (denoted as short hand N1) is added to post order traversal output list, then, stop output, add more nodes (N4, N2) into stack;

N2 is added to post order traversal, and then, stop output, add N3 in the stack;

N3 is added to post order traversal output list, then N4, then N5, 3 in a row to add output traversal list.

N1, N2, N3 can generalized to upward output process, N1, N2 are special case, only 1 in row, N3 starts a 3 in a row.

Every node is added to stack, N9, N5, N1, 3 in a row to add into the stack; then, N4, N3, two in a row, so adding nodes to the stack is always in a process - downward, left most nodes in a row process.

So, the design of function now becomes the following 4 steps:

1. Set up a while loop, work on runner node, use extra variable called previous visited node, prev
2. Take care downward push into stack in a row process.
3. Take care upward pop out from stack in a row process.
4. If the stack is empty, terminate the while loop.
5. If the stack is not empty, set runner as stack's top node's right child, let while loop takes care next iteration.

One more note, the upward pop out from stack in a row process, 3 conditions to check, first, **the stack is not empty, and then, stack top node's right child is null (For node N1), or stack top node's right child is previous node .**

Based on the above analysis - 3 conditions, let us write down in one line of code:

`while(stack.Count > 0 && (stack.Peek().right == null || stack.Peek().right == prev )` <- first try, it is wrong!

**second writing:**

`while(stack.Count > 0 && (stack.Peek().right == null || stack.Peek().right == runner)`

The study code Julia chose is so good, even previous node variable is not needed to be declared!

[8]<https://github.com/douglasleer/LeetCode-Java-Solutions/blob/master/236> .lowest-common-ancestor-of-a-binary-tree.java

Julia spent 2 days to work on this algorithm, felt very down struggling with complicated while loop code, gave up after 3+ hours. Come back again with more warm ups, with a bug - out-of-index, and then, continuously test the code, write down ideas.

### 5. What are fun part?

Julia learns to teach herself algorithm very struggling on this algorithm, take more than 8 hours, she likes to focus on code; always work on code, play with code, get hands dirty, frustrated, add more test cases. She finds out playing part is fun like sports playing.

1. <http://juliachencoding.blogspot.ca/2016/04/leetcode-236-lowest-common-ancestor-in.html>
  2. <https://gist.github.com/jianminchen/2bec8a70ef520e715240a226c01c7371>
  3. [https://msdn.microsoft.com/en-us/library/415129w1\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/415129w1(v=vs.110).aspx)
  4. <https://gist.github.com/jianminchen/754f45023f60992211cb34bef6d6254f>
  5. <https://gist.github.com/jianminchen/99585d73e4196136fd87274174dec3b9>
  6. <https://gist.github.com/jianminchen/388114d99020972413464a072e5e9a9b>
  7. <http://juliachencoding.blogspot.ca/2015/09/morris-order-post-order-traversal.html>
  8. <https://github.com/douglasleer/LeetCode-Java-Solutions/blob/master/236.lowest-common-ancestor-of-a-binary-tree.java>
- 

## Leetcode 236: Lowest Common Ancestor - Warm up practice (2016-05-21 21:18)

May 21, 2016

Julia took more than 10 hours to play with the tree traversal algorithm. Then, she likes to write down the algorithm bug free within 20 - 30 minutes. It is a warm up practice. Just enjoy the adventure and see how many issues are not resolved.

Here is the second blog she worked on more than 10 hours on May 22, 2016:

[1]<http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor.html>

Previous blogs: (April 22, 2016)

[2]<http://juliachencoding.blogspot.ca/2016/04/find-lowest-common-ancestor-of-two.html>

[3]<http://juliachencoding.blogspot.ca/2015/07/leetcode-lowest-common-ancestor-in.html>

Review solutions again:

[4]<https://github.com/jianminchen/LowestCommonAncestorInBinaryTree/blob/master/LowestCommonAncestorB.cs> aster-

Now, start her warm up practice:

May 21, 2016 9:18pm - 9:48pm - 30 minutes taken

[5]<https://gist.github.com/jianminchen/5aa398b8c6e9f13c5a11213ef03cb5bc>

Julia found out a few things:

1. First, a new variable is added: stopTraversal on line 49.

When both two nodes p and q are found, stop to traversal the binary tree.

Julia also learned how to break two while loop:

C # does not allow two "break; ", complaining that the second "break; " unreachable.

4 lines are added:

line 49, line 69, line 76, line 77.

set StopTraversal = true, and then, break the inside while loop, next to the while loop, check if(StopTraversal == true) then, break outside loop.

The code Julia studied and practiced before just break inside while loop only. So, here is the change. line 74 in the following version needs some extra work:

[6]<https://gist.github.com/jianminchen/388114d99020972413464a072e5e9a9b>

2. Julia forget to declare the stack at the beginning, and also, declare two lists for p and q.

Question and answer:

**1. Why do the warm up practice right away?**

Answer:

Julia knows the importance to learn an algorithm a time. Keep working on the code, until it works perfectly, until she makes all mistakes and learns lessons. Afterwards, she may start to learn to analyze the algorithm, even memorize the algorithm easily.

The code is executable, optimized, every line is executed and she pays some attention on them how to optimize.

Practice makes perfect. Show progress, that is mental toughness, that is value of sharing - document progression to excellence.

2. Write C # version based on the blog:

[7]<http://www.geeksforgeeks.org/lowest-common-ancestor-binary-tree-set-1/>

1. <http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor.html>

2. <http://juliachencoding.blogspot.ca/2016/04/find-lowest-common-ancestor-of-two.html>

3. <http://juliachencoding.blogspot.ca/2015/07/leetcode-lowest-common-ancestor-in.html>

4. <https://github.com/jianminchen/LowestCommonAncestorInBinaryTree/blob/master/LowestCommonAncestorB.cs>
  5. <https://gist.github.com/jianminchen/5aa398b8c6e9f13c5a11213ef03cb5bc>
  6. <https://gist.github.com/jianminchen/388114d99020972413464a072e5e9a9b>
  7. <http://www.geeksforgeeks.org/lowest-common-ancestor-binary-tree-set-1/>
- 

## Leetcode 37: Sudoku - a warm up practice (2016-05-21 22:17)

May 21, 2016

Problem statement:

[1]<https://leetcode.com/problems/sudoku-solver/>

Review the blog:

[2]<http://juliachencoding.blogspot.ca/2015/07/leetcode-sudoku-solver.html>

Choose the implementation in those two blogs to do one more practice:

[3]<http://blog.csdn.net/fightforyourdream/article/details/16916985>

And then, start to implement the solution using C # code:

[4]<https://github.com/jianminchen/sudokuSolver/blob/master/Program.cs>

May 21, 2016 practice - Work on practice, executable code, not pseudo code,  
Try to finish in 30 minutes

[5]<https://gist.github.com/jianminchen/90b8455b01347aaf3a091810219c1a8d>

1. <https://leetcode.com/problems/sudoku-solver/>
  2. <http://juliachencoding.blogspot.ca/2015/07/leetcode-sudoku-solver.html>
  3. <http://blog.csdn.net/fightforyourdream/article/details/16916985>
  4. <https://github.com/jianminchen/sudokuSolver/blob/master/Program.cs>
  5. <https://gist.github.com/jianminchen/90b8455b01347aaf3a091810219c1a8d>
- 

## Leetcode 331: Verify Preorder serialization of a Binary Tree (2016-05-22 00:41)

May 22, 2016

Review the blog, and write some code using C # related to serialization of tree:

[1]<http://juliachencoding.blogspot.ca/2016/02/leetcode-331-verify-preorder.html>

Read the blog: (10 - 15 minutes)

[2]<https://www.hrwhisper.me/leetcode-verify-preorder-serialization-of-a-binary-tree/>

C # code practice:

[3]<https://gist.github.com/jianminchen/a24a6f88b7d66721695032ac9cb8373a>

Also, check out this one:

[4]<https://www.hackerrank.com/challenges/uds-echo-server>

Review blogs:

[5]<http://blog.hackerrank.com/step-0-before-you-do-anything/>

1. <http://juliachencoding.blogspot.ca/2016/02/leetcode-331-verify-preorder.html>

2. <https://www.hrwhisper.me/leetcode-verify-preorder-serialization-of-a-binary-tree/>

3. <https://gist.github.com/jianminchen/a24a6f88b7d66721695032ac9cb8373a>

4. <https://www.hackerrank.com/challenges/uds-echo-server>

5. <http://blog.hackerrank.com/step-0-before-you-do-anything/>

---

## **Radix Sort - a distribution sort (2016-05-22 17:48)**

May 22, 2016

Read the blog, have some code using C # in short future:

[1]<http://www.geeksforgeeks.org/radix-sort/>

C # practice based on the above blog:

comment: Line 47, change array name from int[] count -> int[] position, but still get confused; This variable does more than one task.

[2]<https://gist.github.com/jianminchen/ba73ca1550b88eae99c1617c1636180d>

Make some changes:

Change the array's name to helper, helper serves three functions (line 47):

1. First, get count for each digit
2. Second, add sum from 0 to up
3. Third, decrease one by one to track index of next available position for i.

comment: Line 47, change array name to helper, and add comment to list tasks for helper. Feel more control, there is a term called "express the intent."

[3]<https://gist.github.com/jianminchen/09f12e539fce1b267e75d808e13c4ff6>

Prepare for Leetcode 164: Maximum Gap

[4]<http://juliachencoding.blogspot.ca/2015/06/leetcode-maximum-gap-no-164.html>

**Statistics:**

**Time Spent:**

## 4 hours +

1. <http://www.geeksforgeeks.org/radix-sort/>
  2. <https://gist.github.com/jianminchen/ba73ca1550b88eae99c1617c1636180d>
  3. <https://gist.github.com/jianminchen/09f12e539fce1b267e75d808e13c4ff6>
  4. <http://juliachencoding.blogspot.ca/2015/06/leetcode-maximum-gap-no-164.html>
- 

## Binary Tree Post Order Traversal Iterative solution (2016-05-22 20:56)

May 22, 2016

Warm up the post order traversal after 10 hours intensive study of binary tree lowest common ancestor.

First version, there is a bug in the code: out of memory on line 59

[1]<https://gist.github.com/jianminchen/4092880aef47a501ce2c3097531744fa>

Need to reset the runner node as `stack.Pop()`.

Correct version:

[2]<https://gist.github.com/jianminchen/9eaa168499642430baa463fef6471c76>

Review solutions:

1. User two stacks:

[3][https://github.com/jianminchen/leetcode-tree/blob/master/TreePostOrder Iterative.cs](https://github.com/jianminchen/leetcode-tree/blob/master/TreePostOrder%20Iterative.cs)

2. Review previous blog:

[4][http://juliachencoding.blogspot.ca/2015/06/leetcode-post-order-binary- tree.html](http://juliachencoding.blogspot.ca/2015/06/leetcode-post-order-binary-tree.html)

3. Previous blog: Leetcode 106: binary tree serialization - Leetcode

[5][http://juliachencoding.blogspot.ca/search/label/post %20order %20traversal](http://juliachencoding.blogspot.ca/search/label/post%20order%20traversal)

4. Review blog:

[6][http://juliachencoding.blogspot.ca/2016/01/divide-and-conquer-preorder -traversal.html](http://juliachencoding.blogspot.ca/2016/01/divide-and-conquer-preorder-traversal.html)

5. The solution can be optimized - redundant code.

[7][https://github.com/jianminchen/leetcode-tree/blob/master/TreePostOrder Iterative \\_PrevVarirable.cs](https://github.com/jianminchen/leetcode-tree/blob/master/TreePostOrder%20Iterative_PrevVariable.cs)

## Questions and answers:

1. How is the practice?

Julia made a bug in first time writing, she did not notice that she needs to maintain runner when nodes in a row are popped out, so only one node with value 3 (test case: a tree with post order traversal 1 - 9) will be popped out, node with value 4 cannot be popped out.

First, wrong answer.

Second, the code runs on the test case - tree 1-9 will stop on node with value 4, and then, keep adding node with value 4 forever.

```
line 57: while(stack.Count > 0 && (stack.Peek().right == null || stack.Peek().right == runner))
{
    TreeNode node = (TreeNode)stack.Pop(); // bug - out of memory - should be runner = (TreeNode)stack.Pop()
    postTraversal.Add(node.val);
}

line 63: if(stack.Count == 0)
break;
else
runner = stack.Peek().right;
```

Julia was not sure if runner should be reset. She does not have strong clues at that time.

**Lessons :** Need to stop, and take 3-5 minutes to do some reasoning and analysis. Do not rush to finish coding even if you have ideas to implement.

Actually, while loop - line 57 is maintained by runner variable, previous's node is using runner variable.

## 2. Tips about practice?

Do not rush to compile/ running the code, ask yourself, if you can make the code no compiler error, no run time error, with correct answer, without compiling and running the code, you start to use reasoning and analysis to poke every place possible going wrong.

**one of practice goals:** reasoning and analysis.

1. <https://gist.github.com/jianminchen/4092880aef47a501ce2c3097531744fa>
  2. <https://gist.github.com/jianminchen/9eaa168499642430baa463fef6471c76>
  3. <https://github.com/jianminchen/leetcode-tree/blob/master/TreePostOrderIterative.cs>
  4. <http://juliachencoding.blogspot.ca/2015/06/leetcode-post-order-binary-tree.html>
  5. <http://juliachencoding.blogspot.ca/search/label/post%20order%20traversal>
  6. <http://juliachencoding.blogspot.ca/2016/01/divide-and-conquer-preorder-traversal.html>
  7. [https://github.com/jianminchen/leetcode-tree/blob/master/TreePostOrderIterative\\_PrevVarirable.cs](https://github.com/jianminchen/leetcode-tree/blob/master/TreePostOrderIterative_PrevVarirable.cs)
- 

## Leetcode 236: Lowest Common Ancestor - practice (III) (2016-05-22 22:58)

May 22, 2016

Leetcode 236: Lowest common ancestor

Based on the blog:

[1]<http://www.geeksforgeeks.org/lowest-common-ancestor-binary-tree-set-1/>

Julia practices using C #:

user recursive function, use preorder traversal to do search:

[2]<https://gist.github.com/jianminchen/3a9931ecb7818498cbbb47152ed66d89>

Reference:

1. Use post order traversal to find the lowest common ancestor in binary tree:

[3][http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor\\_21.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor_21.html)

1. <http://www.geeksforgeeks.org/lowest-common-ancestor-binary-tree-set-1/>

2. <https://gist.github.com/jianminchen/3a9931ecb7818498cbbb47152ed66d89>

3. [http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor\\_21.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor_21.html)

---

## Leetcode 236: Binary Tree Lowest Common Ancestor - warm up practice (IV) (2016-05-23 09:57)

May 23, 2016

First thing in Canadian Victoria long weekend morning, 9:00am, warm up an algorithm. Write the code using ~~in~~order preorder traversal, using recursive function, find binary tree lowest common ancestor.

Latest practice on this problem (Less than 24 hours ago):

[1][http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor\\_22.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor_22.html)

C # practice on May 23, 2016

[2]<https://gist.github.com/jianminchen/c9941b8daf6afe51d049ecb5f5a17e19>

### Questions and answers:

1. How is your warm up practice?

Answer: Julia likes to train herself, discipline herself to write bug free code, ready to production code if she knows the idea to solve the problem, and also has strong analysis of problem solving already through multiple practice, over 5+ hour practice.

May 22, 2016 practice is based on geeksforgeeks.com blog, and then, May 23, 2016 practice is based on her own analysis and reasoning, the code has more than 2 differences. Good ones.

First one, on line 43, return early if p==null or q==null.

line 43: if (root == null || p == null || q == null)

Second one: line 56

line 56: if (findPath(root.left, search, path) || findPath(root.right, search, path))  
return true;

old version:

line 67: if ( (root.left != null && findPath(root.left, path, searchNode)) ||  
(root.right != null && findPath(root.right, path, searchNode)) )  
return true;

old version line 67, the condition checking is a giant expression, root.left != null is not necessary, let it fall through to allow findPath to do the checking, code is much more clean.



**2. You already practiced preorder and post order traversal to solve binary tree least common ancestor problem. How about inorder traversal?**

Answer: Inorder traversal does not work out this algorithm. Since we are looking for paths from root to search nodes, in the inorder traversal the root node's position is unknown. It is neither the first one as preorder traversal does, nor the last one as post order traversal does. It is somewhere in the middle.

1. [http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor\\_22.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor_22.html)
  2. <https://gist.github.com/jianminchen/c9941b8daf6afe51d049ecb5f5a17e19>
- 

**Leetcode 297: Serialize and deserialize a binary tree - 2 study cases (2016-05-23 14:31)**

May 23, 2016

Problem statement:

[1]<https://leetcode.com/problems/serialize-and-deserialize-binary-tree/>

Study solutions:

1. Study code:

Java Solution:

[2][https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/297\\_serialize-and-deserialize-binary-tree.java](https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/297_serialize-and-deserialize-binary-tree.java)

C # code:

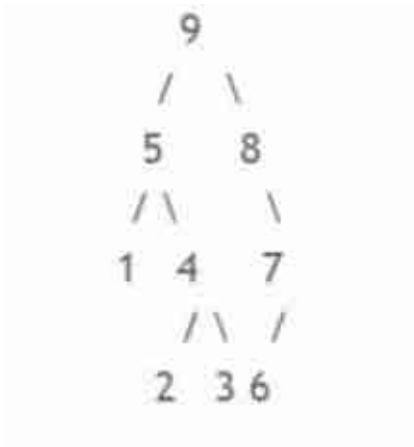
[3]<https://gist.github.com/jianminchen/1f88954d31d1ea7c18a2e5f17ed4ab97>

Spent 60 minutes (9:30pm - 10:30pm) to set up a test case, modify code to make it readable, debug the code and then understand the design, code works.

[4]<https://gist.github.com/jianminchen/da26aa9dbe2a74ea6f641298e57fd289>

Test Case:

Tree diagram:



Tree preorder traversal:

9 5 1 4 2 3 8 7 6

Index: 0 1 2 3 4 5 6 7 8

So, node 9 is serialized:

step 1:

int[3] {9, 1, 6 },

9 is the value of node value,

1 is the left child's index position in List<int[]>, value is 5.

6 is the right child's index position in List<int[]>, value is 8.

The tree is serialized as a string:

" 9 1 6

5 2 3

1 -1 -1

4 4 5

2 -1 -1

3 -1 -1

8 -1 7

7 8 -1

6 -1 -1 "

2. Study code:

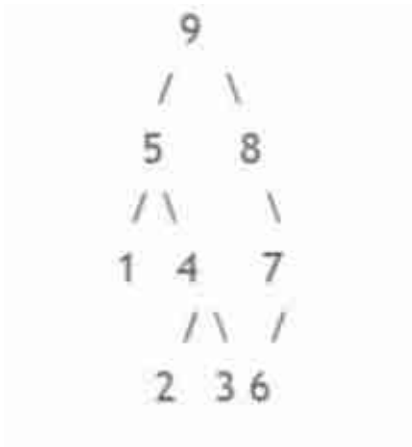
[5]<http://www.cnblogs.com/yrbbest/p/5047035.html>

C # practice based on the above blog: (output limit exceeded - Leetcode Online Judge)

[6]<https://gist.github.com/jianminchen/f4d6d81f300a0c7c26cc4bcdbe99b3db>

Test case:

Tree: Same tree from 1 - 9



Tree serialization string:

9,5,1, #, #,4,2, #, #,3, #, #,8, #,7,6, #, #, #,

Tree preorder traversal:

9 5 1 4 2 3 8 7 6

So, the binary tree can not be uniquely identified just by preorder traversal of binary tree - a string, only if it is a complete binary search tree or other special tree. The serialization string does not include all the information needed. The design has flaws.

## Questions and Answers:

### 1. How is the practice?

Answer: Try to get more experience on preorder traversal iterative solution, and see how the preorder traversal helps to solve the problem. Debug the code step by step and then understand the design.

C # code:

[7]<https://gist.github.com/jianminchen/da26aa9dbe2a74ea6f641298e57fd289>

based on study of code:

[8][https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/297\\_serialize-and-deserialize-binary-tree.java](https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/297_serialize-and-deserialize-binary-tree.java)

Also, study the design of serialization of binary tree using the following ideas:

1. First, use an array to store a tree node;  
array[0] = value;  
array[1] = li, // left child index, if null, -1  
array[2] = ri, // right child index, if null, -1

2. nodes will be added to a list of arrays; then, convert the array list to a string.

Come back later to play more with the solution!

**More reading about Leetcode problems and solutions:**

1. Read the blog - Amazon SDE II - Leetcode solutions

2. Review Leetcode solutions (2nd practice):

[9]<http://www.cnblogs.com/yrbbest/tag/2%E5%88%B7/>

3. 3rd practice:

[10]<http://www.cnblogs.com/yrbbest/tag/3%E5%88%B7/>

### Actionable items:

1. Need to warm up the algorithm, code writing; it is hard to write bug-free code under stress.

Fun part:

10 ways to check if the number is power of 2

[11]<http://www.exploringbinary.com/ten-ways-to-check-if-an-integer-is-a-power-of-two-in-c/>

1. <https://leetcode.com/problems/serialize-and-deserialize-binary-tree/>

2. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/297.serialize-and-deserialize-binary-tree.java>

3. <https://gist.github.com/jianminchen/1f88954d31d1ea7c18a2e5f17ed4ab97>

4. <https://gist.github.com/jianminchen/da26aa9dbe2a74ea6f641298e57fd289>

5. <http://www.cnblogs.com/yrbbest/p/5047035.html>

6. <https://gist.github.com/jianminchen/f4d6d81f300a0c7c26cc4bcdbe99b3db>

7. <https://gist.github.com/jianminchen/da26aa9dbe2a74ea6f641298e57fd289>

8. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/297.serialize-and-deserialize-binary-tree.java>

9. <http://www.cnblogs.com/yrbbest/tag/2%E5%88%B7/>

10. <http://www.cnblogs.com/yrbbest/tag/3%E5%88%B7/>

11. <http://www.exploringbinary.com/ten-ways-to-check-if-an-integer-is-a-power-of-two-in-c/>

---

### Algorithm: check if the number is power of 2 (2016-05-24 22:43)

May 24, 2016

Fun part:

10 ways to check if the number is power of 2

[1]<http://www.exploringbinary.com/ten-ways-to-check-if-an-integer-is-a-power-of-two-in-c/>

10 solutions:

[2]<https://gist.github.com/jianminchen/410d19acf623a7e2a4a349ccd8767c1e>

add comment and test case for solution 7: count ones:

[3]<https://gist.github.com/jianminchen/1035e23fd6117ba7daa38476596e2784>

Plan to spend hours to go over the blog:

1. [4]<http://graphics.stanford.edu/~seander/bithacks.html>
2. [5]<http://graphics.stanford.edu/~seander/bithacks.html#DetermineIfPowerOf2>
3. [6]<http://www.catonmat.net/blog/low-level-bit-hacks-you-absolutely-must-know/>

Review following algorithms:

1. LeetCode 29 Divide Two Integer
2. LeetCode 124: Binary tree max path sum (Hard)
3. LeetCode 126: word ladder II (Hard)
4. LeetCode 127: word ladder (Medium)
5. LeetCode 159: Longest Substring with At Most Two Distinct Characters
6. LeetCode 252: meeting room
7. LeetCode 253: meeting room 2
8. find first repeated word in a string
9. LeetCode 88: merge 2 sorted arrays

[7]<http://fisherlei.blogspot.ca/2012/12/leetcode-merge-sorted-array.html>

C # practice:

[8]<https://gist.github.com/jianminchen/0acddf70788fce10b9660dc16b2e3b71>

10. Given a dictionary of words and a word, return the word if it exists in dict, else return the top 5 words in the dict that are closest to the given word

11. [9]<http://www.geeksforgeeks.org/dynamic-programming-set-32-word-break-problem/>

12. [10]<http://articles.leetcode.com/double-square-problem-analysis/>

13. Design a parking Lot.

1. <http://www.exploringbinary.com/ten-ways-to-check-if-an-integer-is-a-power-of-two-in-c/>
2. <https://gist.github.com/jianminchen/410d19acf623a7e2a4a349ccd8767c1e>
3. <https://gist.github.com/jianminchen/1035e23fd6117ba7daa38476596e2784>
4. <http://graphics.stanford.edu/~seander/bithacks.html>
5. <http://graphics.stanford.edu/~seander/bithacks.html#DetermineIfPowerOf2>
6. <http://www.catonmat.net/blog/low-level-bit-hacks-you-absolutely-must-know/>
7. <http://fisherlei.blogspot.ca/2012/12/leetcode-merge-sorted-array.html>
8. <https://gist.github.com/jianminchen/0acddf70788fce10b9660dc16b2e3b71>
9. <http://www.geeksforgeeks.org/dynamic-programming-set-32-word-break-problem/>
10. <http://articles.leetcode.com/double-square-problem-analysis/>

## LeetCode 159: Longest Substring with At Most Two Distinct Characters (2016-05-24 23:32)

May 24, 2016

Work on the algorithm. The string "eoeab", the longest substring with at most two distinct characters is "eoe", and the length is 3.

1. Read the blog about solution:

[1]<http://yuanhsh.iteye.com/blog/2188917>

Brute force solution:  $O(n^3)$

Sliding window - better solution  $O(n)$  solution in time complexity

2. Study the code:

[2]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/159.longest-substring-with-at-most-two-distinct-characters.java>

Write C # code:

[3]<https://gist.github.com/jianminchen/9061c12fee5e050e56a98cb3ffcc6f57>

1. <http://yuanhsh.iteye.com/blog/2188917>

2. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/159.longest-substring-with-at-most-two-distinct-characters.java>

3. <https://gist.github.com/jianminchen/9061c12fee5e050e56a98cb3ffcc6f57>

---

## Leetcode 252: meeting room (2016-05-24 23:33)

May 24, 2016

Review the algorithm. Will come back very soon.

---

## Leetcode 126: word ladder II (Hard) (2016-05-24 23:34)

May 24, 2016

Will work on algorithm very soon.

May 26, 2016

Read the blog:

[1]<http://www.jiuzhang.com/solutions/word-ladder-ii/>

Great blog:

[2]<http://blog.csdn.net/kenden23/article/details/17611675>

1. Spent over 1 hour to study Java Code:

[3]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/126.word-ladder-ii.java>

and then prepared the solution using C #:

the code runs ok, but need more changes:

[4]<https://gist.github.com/jianminchen/dc64ad0cf06220d293278f874ac07ad4>

2. Read more blogs:

[5]<http://yucoding.blogspot.ca/2014/01/leetcode-question-word-ladder-ii.html>

[6]<http://siyang2leetcode.blogspot.ca/2015/01/word-ladder-ii.html>

1. <http://www.jiuzhang.com/solutions/word-ladder-ii/>
  2. <http://blog.csdn.net/kenden23/article/details/17611675>
  3. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/126.word-ladder-ii.java>
  4. <https://gist.github.com/jianminchen/dc64ad0cf06220d293278f874ac07ad4>
  5. <http://yucoding.blogspot.ca/2014/01/leetcode-question-word-ladder-ii.html>
  6. <http://siyang2leetcode.blogspot.ca/2015/01/word-ladder-ii.html>
- 

## Leetcode 127: word ladder (Medium) (2016-05-24 23:34)

May 24, 2016

Work on the algorithm. Will come back very soon.

Review previous practice:

[1][https://github.com/jianminchen/Leetcode\\_C-/blob/master/127wordLadder.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/127wordLadder.cs)

blog:

[2]<http://juliachencoding.blogspot.ca/2015/06/leetcode-word-ladder.html>

1. Study Java code:

[3]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/127.word-ladder.java>

and then, spend 30 minutes to write C # version.

[4]<https://gist.github.com/jianminchen/8d9aeaaa8ad98fd1bc4a78237118506c>

change a string variable name `anb` to `s`, `Queue q` -> `queue`, `qLen`->`queueLen`, `int length` -> `ladderLength`

[5]<https://gist.github.com/jianminchen/8f1081b25010c572b0a674e43ffba1e5>

use Tuple class to replace two queues with one queue only

[6]<https://gist.github.com/jianminchen/60211d6ef2ae9c6b3142570b45525e21>

1. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/127wordLadder.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/127wordLadder.cs)

2. <http://juliachencoding.blogspot.ca/2015/06/leetcode-word-ladder.html>

3. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/127.word-ladder.java>

4. <https://gist.github.com/jianminchen/8d9aeaaa8ad98fd1bc4a78237118506c>

5. <https://gist.github.com/jianminchen/8f1081b25010c572b0a674e43ffba1e5>

6. <https://gist.github.com/jianminchen/60211d6ef2ae9c6b3142570b45525e21>

---

## Binary Tree Inorder traversal - Iterative solution - warm up practice (2016-05-25 20:55)

May 25, 2016

Review binary tree inorder iterative solution, add some test cases.

[1]<https://github.com/jianminchen/leetcode-tree/blob/master/TreeDemo.cs>

C # practice:

[2]<https://gist.github.com/jianminchen/7a8e54d34d52940d2356a4308efd4739>

### Questions and Answers:

1. How is the practice of inorder traversal iterative solution?

Julia compared to the other solution, line 354 - line 372, `inOrderIterative _B`

[3]<https://github.com/jianminchen/leetcode-tree/blob/master/TreeDemo.cs>

and chose this one to practice:

[4]<https://gist.github.com/jianminchen/7a8e54d34d52940d2356a4308efd4739>

She wrote some comment to argue that the solution is very efficient and no redundant code. She likes to use reasoning and analysis to help her structure the function.



```

68     public static List<TreeNode> inorderIterative(TreeNode root)
69     {
70         List<TreeNode> inorderNodes = new List<TreeNode>();
71
72         if (root == null)
73             return null;
74
75         Stack<TreeNode> stack = new Stack<TreeNode>();
76
77         TreeNode runner = root;
78
79         while (true)
80         {
81             while (runner != null) // line 81
82             {
83                 stack.Push(runner);
84                 runner = runner.left;
85             }
86
87
88             if (stack.Count == 0) // line 88
89                 break;
90
91             runner = stack.Pop();
92             inorderNodes.Add(runner);
93
94             runner = runner.right; // line 94
95         }
96
97         return inorderNodes;
98     }
99 }

```

Here are her analysis:

- \* 1. First, argue that line 88, why to check "if stack is empty, break the while loop"
- \* before outputting any node in the stack.
- \* otherwise, if the stack is empty, then, line 91: execution run time error
- \* 2. Second, argue that output of inorder traversal nodes is not in inner loop.
- \* If there are more than 1 in a row to output, then the outer while loop (line 79)
- \* will take care of it.
- \* 3. Third, argue that line 94: runner = runner.right;
- \* it does not matter that right child exists or not, line 81 will take care of null.
- \* **Complete one iteration** -

- \* Push nodes into stack, then first node in inorder is on the top of stack,
- \* If stack is empty, break; otherwise, ready to pop top one from the stack,
- \* and then, move runner to its right child.

1. <https://github.com/jianminchen/leetcode-tree/blob/master/TreeDemo.cs>
  2. <https://gist.github.com/jianminchen/7a8e54d34d52940d2356a4308efd4739>
  3. <https://github.com/jianminchen/leetcode-tree/blob/master/TreeDemo.cs>
  4. <https://gist.github.com/jianminchen/7a8e54d34d52940d2356a4308efd4739>
- 

## string algorithms - C# practice (2016-05-25 21:14)

May 25, 2016

study string algorithms:

[1]<http://www.cnblogs.com/luxiaoxun/archive/2012/11/12/2766095.html>

1. Find most frequent char in a string:

C # practice

[2]<https://gist.github.com/jianminchen/d530d47982262de8a5e1cf5b49f7b54a>

2. Find a string can be a substring of another string's rotation:

Given

s1 = "ABCD" and s2 = "CDAA", return true;

ABCD -> rotate to left -> BCDA -> contains substring "CDAA"

Given s1 = "ABCD" and s2 = "ACBD", return false.

[3]<http://kenby.iteye.com/blog/1451910>

[4]<http://www.cnblogs.com/kaituorensheg/archive/2013/06/01/3105042.html>

BBC documentary: Amazon - Lead principles study:

[5]<https://www.youtube.com/watch?v=RXLA1ziEzAE>

1. <http://www.cnblogs.com/luxiaoxun/archive/2012/11/12/2766095.html>
  2. <https://gist.github.com/jianminchen/d530d47982262de8a5e1cf5b49f7b54a>
  3. <http://kenby.iteye.com/blog/1451910>
  4. <http://www.cnblogs.com/kaituorensheg/archive/2013/06/01/3105042.html>
  5. <https://www.youtube.com/watch?v=RXLA1ziEzAE>
-

## Leetcode 88: Merge two sorted arrays (2016-05-25 21:16)

May 25, 2016

Leetcode 88: merge 2 sorted arrays

Study the code:

[1]<http://fisherlei.blogspot.ca/2012/12/leetcode-merge-sorted-array.html>

C # practice:

[2]<https://gist.github.com/jianminchen/0acddf70788fce10b9660dc16b2e3b71>

Edge case handling:

line 42 - 47.

1. <http://fisherlei.blogspot.ca/2012/12/leetcode-merge-sorted-array.html>

2. <https://gist.github.com/jianminchen/0acddf70788fce10b9660dc16b2e3b71>

---

## Binary Tree Preorder Traversal - Iterative Solution - Warm up Practice (2016-05-25 23:24)

May 25, 2016

Review blogs:

[1]<http://juliachencoding.blogspot.ca/2015/06/leetcode-binary-tree-preorder-traversal.html>

C # practice:

use stack, and also know the order to push child nodes: right child first, and then left child next.

[2]<https://gist.github.com/jianminchen/0a157cdf8eab4b88d40eb8079c877f99>

### Question and Answer :

1. How is the practice?

Julia first thought about using queue to implement the solution, left child first, and then right child next. But, it does not work, since left child's left child should go to traversal first before right child. So, she had to stop.

Then, she tried to look into iterative solutions in general through Google, and also checked her previous practice.

Next time, go through a test case, do some analysis on the test case. Draw some diagram, help yourself to analyze, at least behaves like a teacher.

2. Things to work on through the practice?

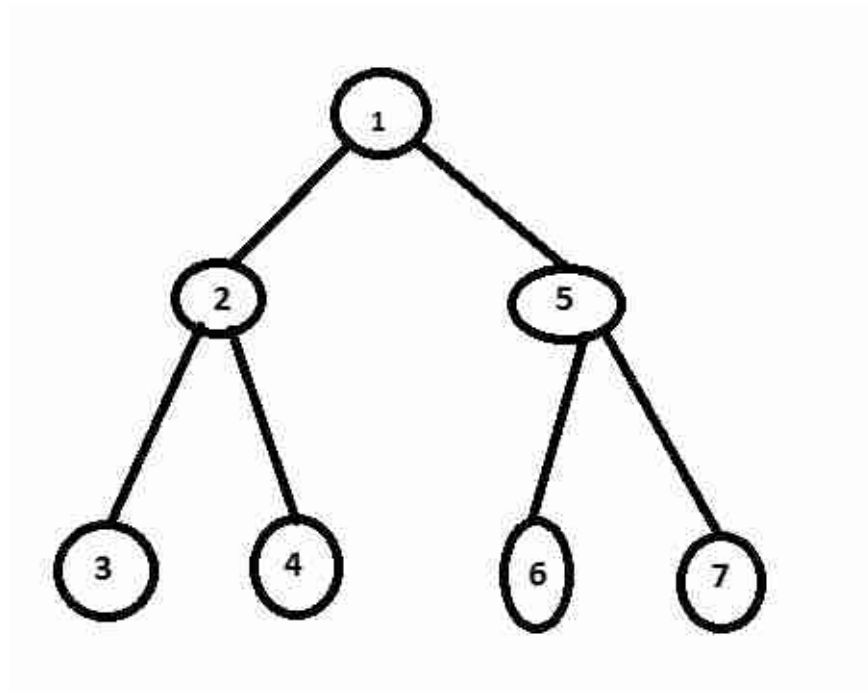
queue -> stack -> opposite order to push into stack

3. Can you describe the process using your own words, a few of drawings to help the thinking process?

First, read the blog:

[3][https://en.wikipedia.org/wiki/Stack\\_\(abstract\\_data\\_type\)](https://en.wikipedia.org/wiki/Stack_(abstract_data_type))

Let us work on a simple test case:

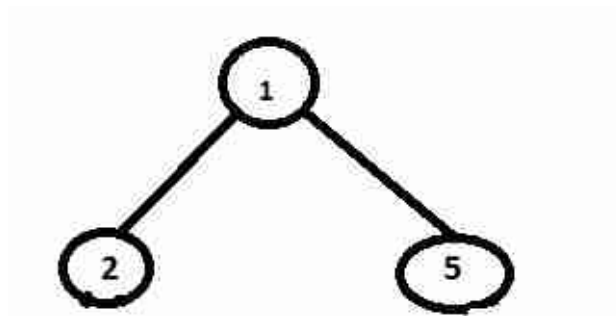


The preorder traversal of tree: 1 2 3 4 5 6 7

When the root node 1 is visited, 2 and 5 should be added to some data structure, 3 and 4 will be added after 5, but output of 3 and 4 should be before 5.

In other words, the data structure should accommodate last in first out feature. So, it is stack!

Once stack is chosen to use, then work out the simple tree first with 3 nodes:



preorder traversal: 1 2 5

so, push 1 into stack, and then, pop out. 5 is pushed in stack first, and then it is turn of 2.

Next, work on the test case: tree - preorder traversal 1 2 3 4 5 6 7

1 is pushed into stack,

1 is on the top of stack, 1 is popped out,

1's right child 5 is pushed into stack,

1's left child 2 is pushed into stack,

2 is popped out from stack,

2's right child 4 is pushed into stack,

2's left child 3 is pushed into stack,

3 is popped out from stack,

4 is popped out from stack,

5 is popped out from stack,

5's right child 7 is pushed into stack,

5's left child 6 is pushed into stack.

6 is popped out from stack,

7 is popped out from stack.

### 3. Most favorite problem solving using stack?

HackerRank: string algorithm - Reverse Shuffle Merge (IV)

[4]<http://juliachencoding.blogspot.ca/2016/04/april-3-2016-httpsgist.html>

warmup practice:

[5]<https://gist.github.com/jianminchen/f3c4b8249b8233771183f1a7f5d01c12>

#### Second practice :

[6]<https://gist.github.com/jianminchen/c695be78345751a5f389ba3332c33342>

so many bugs in second practice:

1. confused on skip, add array -, ++; line 71, line 80 did the opposite.

2. string reverse, char[], string, Array.Reverse etc. lookup

3. add one more test case: "abcacb",

b in stack, c in stack, then, run into a, pop up c, and pop up b, let 'a' in the stack. Two in row pop up in stack. While loop is tested on line 64 - 67.

4. while statement from line 64 - 67, fix compile error.

both are ok to compile:

```
(  
char  
)stack.Peek() > runner)
```

```
(char)stack.Peek() - runner > 0)
```

#### Third practice :

The code passes HackerRank online test cases as well.

A few good changes:

1. add comment from line 67 - line 72, test case: "abcacb",

use test case, stack top -'c' is removed, help to ensure the code is correct.

2. line 73, avoid bug to overwrite the variable runner, create a new variable called backTracked.

[7]<https://gist.github.com/jianminchen/8e2c28262dd3d13c4db856feaed5603e>

**August 8, 2016**

Work on facebook code lab, preorder iterative solution. Chicken out! Forget to enforce the rule, right child goes to stack first, and then, left child goes to stack afterwards. And then, instead, nervousness kicked in, gave up in 5 minutes, looked up blogs. Need more practice!

3 smart choices:

stack vs queue ->

left, right who goes first ->

enforce rule, null pointer will not go into stack, save time ->

one node ->

3 node tree - complete binary tree ->

7 node tree - complete binary tree

1. <http://juliachencoding.blogspot.ca/2015/06/leetcode-binary-tree-preorder-traversal.html>

2. <https://gist.github.com/jianminchen/0a157cdf8eab4b88d40eb8079c877f99>

3. [https://en.wikipedia.org/wiki/Stack\\_\(abstract\\_data\\_type\)](https://en.wikipedia.org/wiki/Stack_(abstract_data_type))

4. <http://juliachencoding.blogspot.ca/2016/04/april-3-2016-httpsgist.html>

5. <https://gist.github.com/jianminchen/f3c4b8249b8233771183f1a7f5d01c12>

6. <https://gist.github.com/jianminchen/c695be78345751a5f389ba3332c33342>

7. <https://gist.github.com/jianminchen/8e2c28262dd3d13c4db856feaed5603e>

---

**HackerRank: string algorithm - Reverse Shuffle Merge - Stack, backtracking techniques (2016-05-26 21:05)**

May 26, 2016

Reverse Shuffle Merge - Problem statement:

[1]<https://www.hackerrank.com/challenges/reverse-shuffle-merge>

**Algorithm analysis :**

Use test case: "abcacb" to explain the solution:

int[] add = new int[26],

int[] skip = new int[26]

a - 0, b - 1, c - 2

add[0] = 1, add[1] = 1, add[2] = 1,

skip[0] = 1, skip[1] = 1, skip[2] = 1,

Find smallest lexical string.

reverse input char one by one:

push b into stack,

'c' > 'b', then, push c into stack

stack:

c

b

Now, a is scanned, 'a' < stack.peek() = 'c', skip[2] = 1 > 0,

backtracking, pop c out of stack, skip[2] = 0

Now, 'a' < stack.peek() = 'b', skip[1] = 1 > 0,

backtracking, pop b out of stack, then adjust skip[1] = 0

add[0] > 0, push a into stack, add[0] = 0,

next, c is scanned, cannot skip, push into stack,

next, b is scanned, cannot skip, push into stack,

next, a is scanned, skip a.

stack.ToArray(), keep stack iterator order, "bca",

Array.Reverse(stack.ToArray()) -> "acb"

Lexical smallest string.

**previous blog :**

[2]<http://juliachencoding.blogspot.ca/2016/04/april-3-2016-httpsgist.html>

warmup practice:

[3]<https://gist.github.com/jianminchen/f3c4b8249b8233771183f1a7f5d01c12>

**Second practice :**

[4]<https://gist.github.com/jianminchen/c695be78345751a5f389ba3332c33342>

so many bugs in second practice:

1. confused on skip, add array -, ++; line 71, line 80 did the opposite.

2. string reverse, char[], string, Array.Reverse etc. lookup

3. add one more test case: "abcacb",

b in stack, c in stack, then, run into a, pop up c, and pop up b, let 'a' in the stack. Two in row pop up in stack. While loop is tested on line 64 - 67.

4. while statement from line 64 - 67, fix compile error.

both are ok to compile:

(

char

)stack.Peek() > runner)

(char)stack.Peek() - runner > 0)

### Third practice :

The code passes HackerRank online test cases as well.

A few good changes:

1. add comment from line 67 - line 72, test case: "abcacb",

use test case, stack top -'c' is removed, help to ensure the code is correct.

2. line 73, avoid bug to overwrite the variable runner, create a new variable called backTracked.

[5]<https://gist.github.com/jianminchen/8e2c28262dd3d13c4db856feaed5603e>

### Fourth practice :

1. create a new function getIndex - on line 100 - 103

2. after reading through the code, still missed a bug in writing - on line 55; debug the code and find the result of first test case "abcacb" should be "acb", but it was "bca".

Julia examined line by line, but still missed the bug on line 55.

[6]<https://gist.github.com/jianminchen/7572e92ea48211d3d05c557d97601dbf>

Read C # string constructor char[]: spend 10 - 20 minutes to read all constructors of string in C #.

[7][https://msdn.microsoft.com/en-us/library/aa331865\(v=vs.71\).aspx](https://msdn.microsoft.com/en-us/library/aa331865(v=vs.71).aspx)

1. <https://www.hackerrank.com/challenges/reverse-shuffle-merge>
  2. <http://juliachencoding.blogspot.ca/2016/04/april-3-2016-httpsgist.html>
  3. <https://gist.github.com/jianminchen/f3c4b8249b8233771183f1a7f5d01c12>
  4. <https://gist.github.com/jianminchen/c695be78345751a5f389ba3332c33342>
  5. <https://gist.github.com/jianminchen/8e2c28262dd3d13c4db856feaed5603e>
  6. <https://gist.github.com/jianminchen/7572e92ea48211d3d05c557d97601dbf>
  7. [https://msdn.microsoft.com/en-us/library/aa331865\(v=vs.71\).aspx](https://msdn.microsoft.com/en-us/library/aa331865(v=vs.71).aspx)
- 

### Design patterns - Quick Study (2016-05-26 23:06)

May 26, 2016

Choose the blog to read, at least 2 hours to study:

[1]<http://blog.csdn.net/kenden23/article/category/2256833>



Review blog:

[2]<http://juliachencoding.blogspot.ca/2015/12/book-reading-head-first-design-pattern.html>

1. MongoDB (20 minutes reading)

[3]<https://en.wikipedia.org/wiki/MongoDB>

craigslist -> MySQL -> MongoDB

2. Amazon DynamoDB vs MongoDB

3. [4]<https://plus.google.com/104439038177484907272>

1. <http://blog.csdn.net/kenden23/article/category/2256833>

2. <http://juliachencoding.blogspot.ca/2015/12/book-reading-head-first-design-pattern.html>

3. <https://en.wikipedia.org/wiki/MongoDB>

4. <https://plus.google.com/104439038177484907272>

---

## Leetcode 126: word ladder II - warm up practice (2016-05-28 22:18)

May 28, 2016

Share professional tennis player Kristina Mladenovic tweet:

[1]<https://twitter.com/kikimladenovic/status/732966170375114752>

I know what it takes to win. Forget everyone else and put the work in.

Spend one hour to work on the word ladder II, Leetcode 126.

Previous blog:

[2][http://juliachencoding.blogspot.ca/2016/05/leetcode-127-word-ladder-medium\\_24.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-127-word-ladder-medium_24.html)

work on one test case:

hit-> cog

Dictionary<string, int> has entries:

Let us talk about a test case to help understand the work:

hit -> cog

Two transformation paths:

dot -> dog

hit -> hot -> -> cog

lot -> log

dist value is 5

Dictionary<string, int>

key value new value

hit 0 4

hot 1 3

dot 2 2

lot 2 2

dog 3 1

log 3 1

cog 4 0

Extract a function called resetDistanceFromEnd

[3]<https://gist.github.com/jianminchen/ca720a93cb8e8fd12610ba58fef9889f>

1. <https://twitter.com/kikimladenovic/status/732966170375114752>

2. [http://juliachencoding.blogspot.ca/2016/05/leetcode-127-word-ladder-medium\\_24.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-127-word-ladder-medium_24.html)

3. <https://gist.github.com/jianminchen/ca720a93cb8e8fd12610ba58fef9889f>

---

## Leetcode 126: Word Ladder II - warm up practice (II) (2016-05-29 13:49)

May 29, 2016

Share one tweet from profession tennis play Angelique Kerber (2016 Australian Grand Slam champion):

[1]<https://twitter.com/AngeliqueKerber/status/734019444868059136>

" Everyone asks me what's next. I'm here to stay, taking it one game at a time. "

One algorithm at a time. Still work on Leetcode 126: word ladder II,

Here is the practice version on May 28, 2016, less than 15 hours ago.

[2]<http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii -warm-up.html>

And then, she spent over 2 hour to make changes on the code, here is the new version:

[3]<https://gist.github.com/jianminchen/90075ee13a6ad0d9d59c8843d17e3a18>

Here are the list of things she made the change:

1. line 21, class member variable is commented out. ladders.

It is replaced by an argument in the function:

line 189 getLadders \_DFS \_Backtracking, 8th argument - ladderHelper

2. line 27, comment out member variable ladderHelper, pass an argument in function

getLadders \_DFS \_Backtracking line 189, last argument: List<string> ladderHelper

3. line 60, change function name to

getLadders \_DFS \_Backtracking , remind myself two things:

1. it is a DFS algorithm

2. Do not forget to do backtracking

4. line 102, change function name to

getLadderLengthAndDictionary \_BFS . BFS stands for

breadth first search.

5. line 203, line 204, add two more explanation variable, make code more readable.

isEndWord,

isBeforeEndWord

6. line 220, add explanation variable, backtracking \_char , helps user to understand

the backtracking process.

7. line 221, add explanation variable replace , the char will be replaced by any one of from 'a' to 'z'.

8. line 225, 226

if(j == replace)

continue;

Make the code more flat, no nested two if statements, only one if statement.

9. line 229, ij \_word , ij prefix helps to track index i and index j.

## 10. line 245, line 249, line 251 3 backtracking statements.

1. <https://twitter.com/AngeliqueKerber/status/734019444868059136>
  2. <http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-warm-up.html>
  3. <https://gist.github.com/jianminchen/90075ee13a6ad0d9d59c8843d17e3a18>
- 

### Leetcode 126: Word Ladder II - warm up practice (III) (2016-05-29 23:34)

May 29, 2016

Share one tweet from profession tennis play Angelique Kerber (2016 Australian Grand Slam champion):

[1]<https://twitter.com/AngeliqueKerber/status/734019444868059136>

" Everyone asks me what's next. I'm here to stay, taking it one game at a time. "

One algorithm at a time. Still work on Leetcode 126: word ladder II,

Here is the practice version on May 28, 2016, less than 15 hours ago.

And then, she spent over 2 hour to make changes on the code, here is the 2nd version:

[2]<https://gist.github.com/jianminchen/90075ee13a6ad0d9d59c8843d17e3a18>

And then, she continued to spend over 2+ to write the code again, here is the third version:

[3]<https://gist.github.com/jianminchen/5570fdb038f5d5d2fd2d64af09fdb643>

Question and Answer:

#### 1. What do you learn through the practice?

**Julia created three bugs:**

1. First, function call on line 59, return dist = 0.

Because endWord, such as "cog" should be added to HashSet<string> and then use it in the function call on line 59, instead of original HashSet<string> wordList.

line 144, wordList.Contains(trial), last word "cog" is not in wordList, therefore, function can not function normally.

2. Second, function call on line 73, wordListExtended should be used instead of wordList,

otherwise, the endWord is not in HashSet<string>, cannot find ladders.

line 247 cannot return true when ij\_word is endWord "cog":

wordList.Contains(ij\_word)

3. Third, in function getLadder\_DFS\_Backtracking on line 204 - 271, line 219 is commented out,

dictionary[runner] < dist, cannot be =, otherwise, ladders with distance 5 and 6 both are included. Instead of 2 list, there are 6 lists.

2. What improvement does this practice make?

2nd version:

[4]<https://gist.github.com/jianminchen/90075ee13a6ad0d9d59c8843d17e3a18>

3rd version:

[5]<https://gist.github.com/jianminchen/5570fdb038f5d5d2fd2d64af09fdb643>

- a. In 3rd version, remove 2nd version from line 37 - 44.
- b. In 3rd version, line 71, variable name is changed from visited to visitedHelper.
- c. In 3rd version, function **findDistAndPrepareDictionary \_BFS \_UsingQueue** , last line,
- d. line 153, return 0; in 2nd version, length is return.
- e. In 3rd version, line 106, int length - variable, can be removed, no use at all.
- f. In 3rd version, removed 2nd version nested if - line 141.
- g. In 3rd version, Tuple class is used to avoid using two queues.

#### **Actionable items:**

- 1. More practice, goal is to write executable code with correct result.
- 2. Study more solutions.

- 1. <https://twitter.com/AngeliqueKerber/status/734019444868059136>
  - 2. <https://gist.github.com/jianminchen/90075ee13a6ad0d9d59c8843d17e3a18>
  - 3. <https://gist.github.com/jianminchen/5570fdb038f5d5d2fd2d64af09fdb643>
  - 4. <https://gist.github.com/jianminchen/90075ee13a6ad0d9d59c8843d17e3a18>
  - 5. <https://gist.github.com/jianminchen/5570fdb038f5d5d2fd2d64af09fdb643>
- 

### **Amazon Leadership principle study (2016-05-30 20:39)**

May 27, 2016

Julia likes to do small and quick research about Amazon leadership principle.

Here is the public link about Amazon leadership principles:

[1]<https://www.amazon.jobs/principles>

Videos she watched and she understands the basic things - things can be controllable, long term sustainable - customer central culture, not on competitor etc.. Do not feel good when stock go up 10 %, feel 10 % smarter; vice versa.

1. Interview: Amazon CEO Jeff Bezos (1 hour video)

[2]<https://www.youtube.com/watch?v=Xx92bUw7WX8>

1. Amazon headquarter campus is in downtown vs. remote area - environment concern.
2. Senior manager stress level - More control, and should be less stress.
3. AWS - tremendous transaction of distributed computing, resource etc.

2. [3]<https://www.youtube.com/watch?v=56uxnKbvbJ4>

1. big selection 2. fast and quick delivery, 3 price - customer concerns, repeated customers.

**offer wider selection, lower prices and fast, reliable delivery .**

Intense hard working, high IQ; can hire great people.

3. Building Amazon One Box at a Time

[4]<https://www.youtube.com/watch?v=tfAhTtBlb2Q>

4.

BBC documentary: Amazon - Lead principles study:

[5]<https://www.youtube.com/watch?v=RXLAIziEzAE>

5. Jeff Bezo's Top 10 Rules For Success

[6]<https://www.youtube.com/watch?v=pAdjNuE6EZQ>

6. Read the article:

[7]<http://www.forbes.com/forbes/2012/0423/ceo-compensation-12-amazon-tech-nology-jeff-bezos-gets-it.html>

1. In 2006, Amazon Web Services as a standalone business. AWS story - processing 150,000 picture in 2 hours, cloud service, in-house takes 15 days, cost 200 dollars.
2. TV ads about customer of Kindle reader gets hurt, Jeff asked to replace the funny video. Care about customers.
3. Two pizzas feeds a team, team is too big.
4. Even the tiniest delay in loading a Web page isn't trivial. Amazon has metrics showing that a 0.1 second delay in page rendering can translate into a 1 % drop in customer activity.

The respect for that ethic explains why Amazon screens its job candidates for a strong bias to action and an ability to work through ambiguity. Both help identify people who can **innovate fast** and **do right by the customer** . One popular interviewing tack: asking candidates to create an action plan as brand managers in an area where they lack any direct knowledge and then being told they have no budget.

Stumped candidates will find their path into Amazon slipping away. Those who cobble together guerrilla answers informal polls through free online tools such as SurveyMonkey tend to thrive at Amazon. They are the same people who might have challenged Bezos in math class and also succeeded on Grandpa's ranch.

To attend two days of call-center training each year. **The payoff: humility and empathy for the customer.**

7. **2001 Interview "How Did Jeff Bezos Start Amazon? His Background, the Internet & the Future of E-Commerce (2001)"**

[8]<https://www.youtube.com/watch?v=og8B45GV-Is>

Stock market is a voting machine in short term, in long term it is a weighing machine.

More reading - Chinese articles: (June 1, 2016)

[9][http://www.weixinyidu.com/n\\_1766932](http://www.weixinyidu.com/n_1766932)

[10]<http://www.knowledger.info/2015/08/27/starbucks-vs-amazon-a-tale-of-two-cultures/>

8. Stanford university artificial intelligent graduate certificate

[11]<http://scpd.stanford.edu/public/category/courseCategoryCertificatePro> file.do?method=load &certificateId=1226717

9. Amazon student program

[12]<https://www.amazon.ca/gp/help/customer/display.html?nodeId=201552950>

June 12, 2016

10. UTC 2012 Hall of Fame - Jeff Bezos Keynote

Amazon is looking for people thinking out-of-box, creative, be able to invent something.

June 19, 2016

2005 Entrepreneurship Conference - Taking on the Challenge. Jeffrey Bezos, Amazon

[13]<https://www.youtube.com/watch?v=WhnDvvNS8zQ>

[14]<https://www.linkedin.com/pulse/ace-leadership-interview-prakash-krishnan?trk=hp-feed-article-title-like>

August 30, 2016

Microsoft research - Amazon 14 leadership principles

[15]<https://www.microsoft.com/en-us/research/video/the-amazon-way-14-leadership-principles-behind-the-worlds-most-disruptive-company/>

Take notes on Sept. 5, 2016 10:17pm

1. AWS - how to get there?

Separate application team from infrastructure team, force infrastructure team to make external customer's happy, high-reliability product

2. Future release - concept?

1. <https://www.amazon.jobs/principles>

2. <https://www.youtube.com/watch?v=Xx92bUw7WX8>

3. <https://www.youtube.com/watch?v=56uxnKbvbJ4>

4. <https://www.youtube.com/watch?v=tfAhTtBlb2Q>

5. <https://www.youtube.com/watch?v=RXLA1ziEzAE>

6. <https://www.youtube.com/watch?v=pAdjNuE6EZQ>

7. <http://www.forbes.com/forbes/2012/0423/ceo-compensation-12-amazon-technology-jeff-bezos-gets-it.html>

8. <https://www.youtube.com/watch?v=og8B45GV-ls>

9. [http://www.weixinyidu.com/n\\_1766932](http://www.weixinyidu.com/n_1766932)

10. <http://www.knowledger.info/2015/08/27/starbucks-vs-amazon-a-tale-of-two-cultures/>

11. <http://scpd.stanford.edu/public/category/courseCategoryCertificateProfile.do?method=load&certificateId=1226717>
  12. <https://www.amazon.ca/gp/help/customer/display.html?nodeId=201552950>
  13. <https://www.youtube.com/watch?v=WhnDvvNS8zQ>
  14. <https://www.linkedin.com/pulse/ace-leadership-interview-prakash-krishnan?trk=hp-feed-article-title-like>
  15. <https://www.microsoft.com/en-us/research/video/the-amazon-way-14-leadership-principles-behind-the-worlds-most-disruptive-company/>
- 

## Leetcode 126: word ladder - a practice (IV) (2016-05-30 20:41)

May 30, 2016

Study the Java code shown in the blog:

[1]<http://www.programcreek.com/2014/06/leetcode-word-ladder-ii-java/>

Here is the C # code to use the same ideas in the above blog (programcreek.com)

[2]<https://gist.github.com/jianminchen/3d8e5ad6042dce7e019cf11c0b1eee22>

Previous blogs on Leetcode 126: word ladder

Study Java Code,

[3]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/126.word-ladder-ii.java>

and then, write C # code, and then, improve code readability, play with styles etc. Time spent: more than 8 hours +.

the code runs ok, but need more changes:

1. [4]<https://gist.github.com/jianminchen/dc64ad0cf06220d293278f874ac07ad4>
2. [5]<http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-warm-up.html>
3. [6][http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-warm-up\\_29.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-warm-up_29.html)
4. [7][http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-warm-up\\_52.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-warm-up_52.html)

Questions and Answers:

### 1. How about the practice?

The practice is with better ideas, short code.

idea 1: To track the actual ladder, we need to add a pointer that points to the previous node in the WordNode class.

For example, hit -> cog, each hop with distance 5:

**hit->hot->dot->dog->cog** ,

hit, prev = null,



hot, prev = hit,

dot, prev = hot

dog, prev = dot

cog, prev = dog

So, using BFS - breadth first search, queue, once the WordNode with "cog" is found, then, whole path can be retrieved. This is a singly linked list.

Same applies to the path: hit->hot->lot->log->cog

idea 2: In addition, the used word can not directly removed from the dictionary. The used word is only removed when steps change.

2. C # practice vs. Java practice?

Java code - use LinkedList as queue, so Julia tried to use C # LinkedList to implement the queue as well.

**C # LinkedList** API used in the practice:

line 48, line 115: AddLast() <- queue.enqueue, force to add at the end

line 63: First() <- queue.Peek(), just look up the front node's value, but do not remove the node

line 64: RemoveFirst() <- queue.Dequeue()

C # practice:

line 101: string.ToCharArray()

line 112: new string(char[]) - constructor

1. <http://www.programcreek.com/2014/06/leetcode-word-ladder-ii-java/>
2. <https://gist.github.com/jianminchen/3d8e5ad6042dce7e019cf11c0b1eee22>
3. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/126.word-ladder-ii.java>
4. <https://gist.github.com/jianminchen/dc64ad0cf06220d293278f874ac07ad4>
5. <http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-warm-up.html>
6. [http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-warm-up\\_29.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-warm-up_29.html)
7. [http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-warm-up\\_52.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-warm-up_52.html)

---

## Leetcode 126: word ladder II - study C++ code using BFS - Queue (Practice V) (2016-05-30 21:13)

May 30, 2016

Study the code:

[1]<http://mrsuyi.com/2016/01/31/leetcode-126/>

C # code:

[2]<https://gist.github.com/jianminchen/3e99a564341995e2dd93e2fd3580aaef>

Questions and Answers:

### 1. How is the practice?

Julia spent more than 1 hour to work on practice.

Test case:

hit -> hot -> dot -> dog -> log

hit -> hot -> lot -> log -> cog

1. Line 40 - 43, add beginWord into Dictionary. The code can be extracted to one standalone function.
  2. Line 52, inside while loop, HashSet, try to figure out the purpose, named: **bfs \_nextNodes** same distance nodes to startWord will stay in the same HashSet.
  3. Line 70, arr.ToString(), debug runtime error, the string will be "new String()". Bug is fixed, using "new string(char[])"
  4. Line 84 - 87, add discussion of ladders.ContainsKey(newstring), otherwise, ladders[newstring].AddRange(newList) through runtime error. ladders[newstring] is null pointer.
- Test case:
1. First time try, only one path is added; miss the second path (hit -> hot -> lot -> log -> cog)
  5. line 99, Dictionary API ContainsKey, not Contains, vs. HashMap
  2. What is the idea to implement the solution using your own words?

The idea is much clever. Explain the idea using the test case:

dot -> dog

hit->hot -> -> cog

lot -> log

so, the queue process is (breadth first search)

first layer:

hit

second layer:

hot

third layer:

dot, lot

forth layer:

dog, log

fifth layer:

490

cog

build ladders step by step, from startWord to current.

Dictionary<string, List<List<string>>> ladders

each string will be the key in ladders

hit -> hit

hot -> one list, {"hit", "hot" }

dot -> one list, {"hit", "hot", "dot" }

lot -> one list, {"hit", "hot", "lot" }

dog -> one list, {"hit", "hot", "dot", "dog" }

log -> one list, {"hit", "hot", "lot", "log" }

cog -> two lists,

{"hit", "hot", "dot", "dog", "cog" }

{"hit", "hot", "lot", "log", "cog" }

1. <http://mrsuyi.com/2016/01/31/leetcode-126/>

2. <https://gist.github.com/jianminchen/3e99a564341995e2dd93e2fd3580aaef>

---

## Leetcode 126: word ladder II - using BFS - Queue, Set Paths (Practice VI) (2016-05-30 21:36)

May 30, 2016

Study the blog:

[1]<http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-study-c.html>

And continue to write C # solution - focus on coding, design ideas.

1. First practice, with a bug on just after line 111 - missing to add tmpList to newList.

[2]<https://gist.github.com/jianminchen/9e45848c86ebaaa8f0bc0d881cfd1ad5>

2. Fix the bug and pass the test case, add line 112:

[3]<https://gist.github.com/jianminchen/e3466a9f1319f62f869aa53487c05536>

Questions and Answers:

### 1. How is the practice?

Here are some sharing:

1. Write down the comment about the idea to solve the problem, from line 23 - line 58

Good idea is to solve 50 % of problem. Repeat the idea in the following:

Explain the idea using the test case:

dot -> dog

hit->hot -> -> cog

lot -> log

so, the queue process is (breadth first search)

first layer:

hit

second layer:

hot

third layer:

dot, lot

forth layer:

dog, log

fifth layer:

cog

build ladders step by step, from startWord to current.

Dictionary<string, List<List<string>>> ladders

each string will be the key in ladders

hit -> hit

hot -> one list, {"hit", "hot" }

dot -> one list, {"hit", "hot", "dot" }

lot -> one list, {"hit", "hot", "lot" }

dog -> one list, {"hit", "hot", "dot", "dog" }

log -> one list, {"hit", "hot", "lot", "log" }

cog -> two lists,

{"hit", "hot", "dot", "dog", "cog" }

{"hit", "hot", "lot", "log", "cog" }

2. Some highlights?

First of all, first writing is wrong. There are 5 levels, 4 loops and then one if, Julia got confused the position of code.

So, marks are added in the comment just after the loop - start statement and also close }.

**Work on indentation - Better to write down your own marks.**

line 75 // level 1, while loop

line 88 // level 2, for loop

line 93 // level 3, for loop

line 97 // level 4, for loop

line 104 // level 5, if statement

line 121 // end of level 5, }

line 122 // end of level 4, }

line 125 // end of level 3, - end for loop for hop string length  
line 128 // end of level 2, - processing - all nodes in the same distance in the queue  
line 139 // level 1 - processing queue - queue.Count > 0

line 95, char variable name: char **backtracking\_char** , just friendly remind to do backtracking later on line 124.

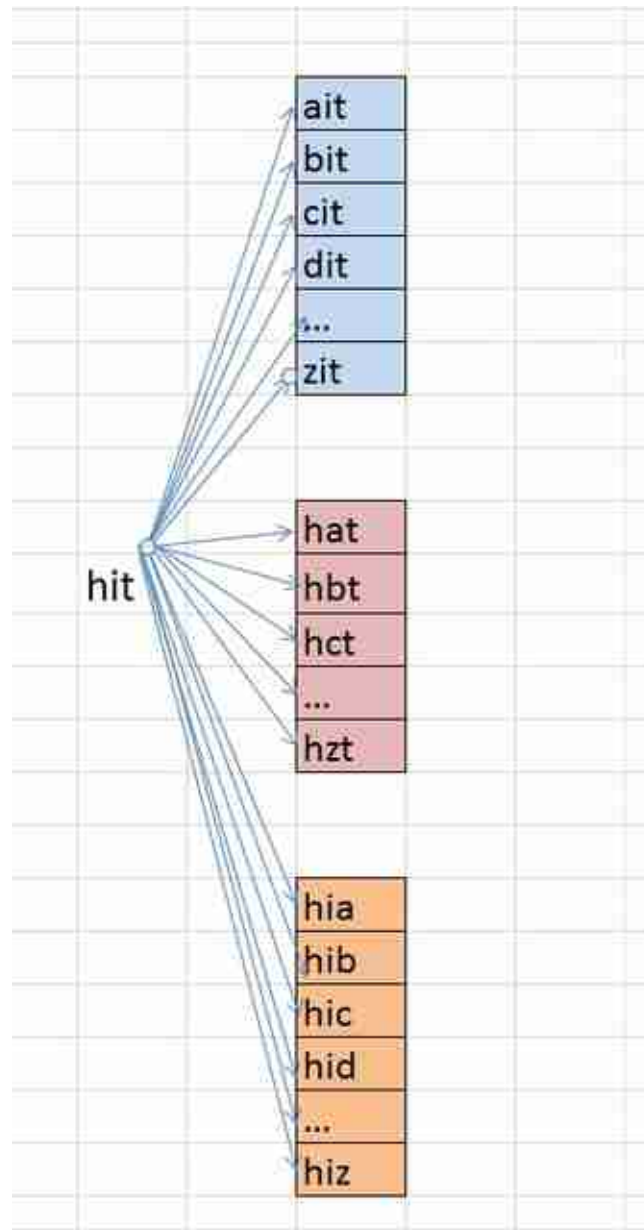
line 102, string variable name change: newstring -> **nextHop** , hopefully the current one is more meaningful.

### 3. Actionable items?

Write again a few times, try to finish coding in 30 minutes, and write down issues not resolved.  
Coding is like sports, **just do it!**

### 4. More to share about the algorithm?

Share one graph to illustrate the graph problem of word ladder:



5. A lot of people complained the problem is too tough. Julia, you miss the most important things in the design? Time out is a big issue. And also, we need to discuss how to avoid a cycle in the list; multiple paths can be found from start word to end word.

**Challenge 1 :** The visited word must be removed sometime to avoid a cycle;

**Challenge 2 :** When to remove visited words from dictionary properly? Same distance, one word can repeat multiple times.

Sure, let us talk about it!

on line 84, inside the while loop, `HashSet<string> nextHops_BFS` are defined. We need to examine this HashSet: on line 120, nextHop is added one by one to the HashSet, but no action is taken on the HashSet. Lazy processing here!

Until all nodes in the current layer are processed, dequeue from the queue. From line 133 - 137, add next layer hops into the queue, where to find the nodes - `nextHops_BFS` . Remove the words from wordList just before adding to the queue.

Put line 135 just after line 120, and see what is difference.  
The program will generate no output for test case:

```
dot -> dog
hit->hot -> -> cog
lot -> log
```

log word's next layer is cog, and dog word's next layer is cog as well. Both routes have same word, terminate word. cog can not be removed from wordDist HashSet until two routes are processed both. Actually, the graph should look likes the following:

```
dot -> dog -> cog
hit->hot ->
lot -> log -> cog
```

In fifth layer, cog word shows up on line 102 twice:

```
line 102 if
(wordList.Contains(nextHop))
```

So, 2 lists can be added with the same word "cog" as last one.

It is important to keep wordList untouched until the whole layer is processed, then, remove

**words in HashSet nextHops \_BFS** from wordList.

Rules to obey in the design:

1. The same word does not repeat twice in the same list. For example, hit->hot->lot->log->lot-log..., it may go on forever.
2. One word does not show up in more than one list, except start word and end word. For example, test case, hit, cog are only exceptions.

**Not True** . "hot" shows up in two lists.

3. Words in the distance n should not be any word in distance 0 to n-1.

```
dot -> dog
hit->hot -> -> cog
lot -> log
```

So, when dog or log are processed in the queue, all words less than 4, hit, hot, dot, lot should be removed from HashSet wordList.

Need to go through those cases one by one.

#### Actionable items:

Read the blog and then implement the solutions as well.

[4][http://www.cnblogs.com/shawnhue/archive/2013/06/05/leetcode\\_126.html](http://www.cnblogs.com/shawnhue/archive/2013/06/05/leetcode_126.html)

1. <http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-study-c.html>
  2. <https://gist.github.com/jianminchen/9e45848c86ebaaa8f0bc0d881cfd1ad5>
  3. <https://gist.github.com/jianminchen/e3466a9f1319f62f869aa53487c05536>
  4. [http://www.cnblogs.com/shawnhue/archive/2013/06/05/leetcode\\_126.html](http://www.cnblogs.com/shawnhue/archive/2013/06/05/leetcode_126.html)
- 

## **Distributed System, noSQL, operating system Quick Review (2016-05-30 22:23)**

March 30, 2016

### **1. Operating System - Review**

Spend one hour to study:

[1]<http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-828-operating-system-engineering-fall-2012/>

Review operating System:

1. [2]<http://blog.csdn.net/u012290414/article/details/48782375>

### **2. Distributed System**

Network service and app framework

Load balance, caching, replication, rest, nosql

Reading time:

[3]<https://www.khanacademy.org/computing/computer-science/algorithms>

[4]<https://blog.dev mastery.com/how-to-win-the-coding-interview-71ae7102d685#.drghie8as>

### **3. JQuery github open source project - source code style.**

[5]<https://github.com/jquery/jquery/tree/master/src>

4. Visit this Leetcode blog - most popular blog I have seen - over 10,000 visitor

[6]<http://blog.csdn.net/linhuanmars/article/details/23071455>

### **5. NoSQL**

[7]<https://en.wikipedia.org/wiki/NoSQL>

Learn a few key features:

1. "non SQL" or "non relational"



2. Compared to relational database, tabular relations -

3. Motivation of NoSQL:

Data structures in NoSQL are different.

name of data structures: key-value, wide column, graph, or document

4. ACID - Atomic, Consistency, I - Isolation, D - Durability

NoSQL - "eventual consistency"

1. <http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-828-operating-system-engineering-fall-2012/>

2. <http://blog.csdn.net/u012290414/article/details/48782375>

3. <https://www.khanacademy.org/computing/computer-science/algorithms>

4. <https://blog.dev mastery.com/how-to-win-the-coding-interview-71ae7102d685#.drghie8as>

5. <https://github.com/jquery/jquery/tree/master/src>

6. <http://blog.csdn.net/linhuanmars/article/details/23071455>

7. <https://en.wikipedia.org/wiki/NoSQL>

---

## Leetcode 103: Binary Tree ZigZag traversal - a practice (2016-05-31 22:21)

May 31, 2016

Review the blog:

[1]<http://juliachencoding.blogspot.ca/2015/06/leetcode-zigzag-order-traversal-of.html>

practice in 2015:

[2]<https://github.com/jianminchen/zigzagOrderTraversal/blob/master/Program.cs>

Warmup practice on May 31, 2016, write a C# solution again:

[3]<https://gist.github.com/jianminchen/3723020cd9757f85d0ceb383da425c6c>

### Practice notes:

1. source code line 107, Stack, use ref, pass reference.

Lookup why Stack is different from Array. Array reference is passed automatically.

Read the blog:

[4]<http://stackoverflow.com/questions/967402/are-arrays-or-lists-passed-by-default-by-reference-in-c>

2. Extract a function named swap(), actually, the last practice:

[5]<https://github.com/jianminchen/zigzagOrderTraversal/blob/master/Program.cs>

line 57, Stack<int> tmp is declared, but line 92 tmp is used. The scope is too large to pay attention. So, extract it to a function.

3. The last practice first while loop on line 59:

[6]<https://github.com/jianminchen/zigzagOrderTraversal/blob/master/Program.cs>

It is confusing, so in current practice, line 72, while(true),  
line 72: while(true)

later, on line 112, break the while loop.

```
line 112: if  
(currLevel ==  
null  
|| currLevel.Count ==  
0  
)
```

```
113: break;
```

4. line 61, if (root == null), return empty list instead of null pointer.

1. <http://juliachencoding.blogspot.ca/2015/06/leetcode-zigzag-order-traversal-of.html>
  2. <https://github.com/jianminchen/zigzagOrderTraversal/blob/master/Program.cs>
  3. <https://gist.github.com/jianminchen/3723020cd9757f85d0ceb383da425c6c>
  4. <http://stackoverflow.com/questions/967402/are-arrays-or-lists-passed-by-default-by-reference-in-c>
  5. <https://github.com/jianminchen/zigzagOrderTraversal/blob/master/Program.cs>
  6. <https://github.com/jianminchen/zigzagOrderTraversal/blob/master/Program.cs>
- 

## Leetcode 124: Binary Tree Maximum Path Sum - reasoning and analysis (I, II, III) (2016-05-31 22:40)

May 31, 2016

Review the blog:

[1]<http://juliachencoding.blogspot.ca/2015/07/leetcode-maximum-binary-tree-path-sum.html>

[2]<http://juliachencoding.blogspot.ca/2015/07/itint5-tree-maximum-path-sum.html>

Practice on May 31, 2016, first version:

[3]<https://gist.github.com/jianminchen/432bdb1af38a1ec6f4a0741420891ebf>

Second version: add more comment:

[4]<https://gist.github.com/jianminchen/578656e1079e8c58b08dd19f5b027e68>

The above second version line 133 - line 137 can be simplified as one statement:

```
int value = root.val + ((left>0)?left:0) + ((right>0)?right:0);
```

Third version: line 133 - line 137 -> one line

[5]<https://gist.github.com/jianminchen/9fc060ae74290637f732a0fc820f44>

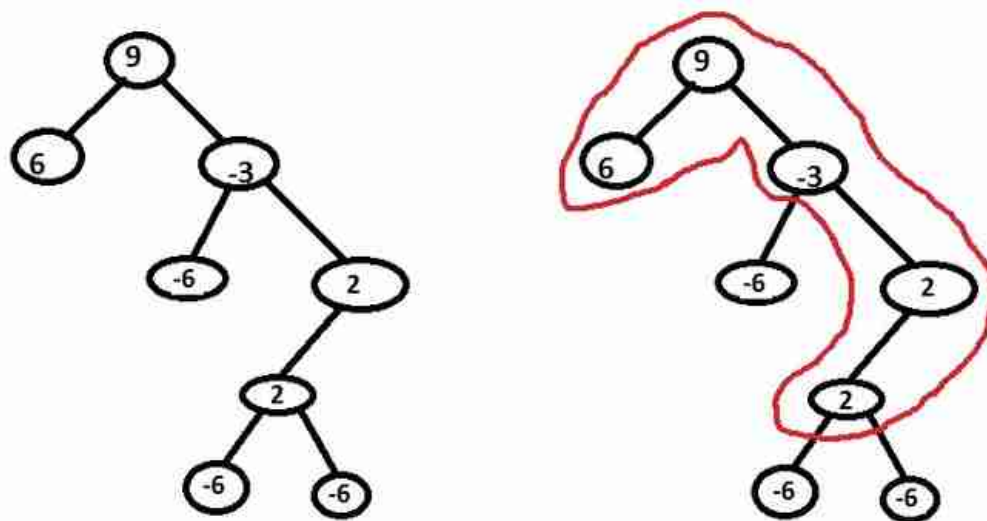
## Reasoning and analysis (I):

Write some reasoning and analysis.

Let us talk about an example first:

In the practice on May 31, 2016, line 60 - 79, test case 3:

[6]<https://gist.github.com/jianminchen/578656e1079e8c58b08dd19f5b027e68>



The maximum path sum in the above tree is highlighted using red color, node 6->9->-3->2->2. Any two nodes in the tree can form a unique path, and the choice of path is  $N^2$ ,  $N$  is the total nodes in the tree.

If we do not use recursive solution, try to use brute force solution:

1. Find any two nodes in the tree, and then find the path, and calculate the sum of path.

( June 24, 2016 but least common ancestor is much more difficult problem to solve! )

For example, maximum path, we have to find the common ancestor of those two nodes with value 6 (root->left) and node with value 2 (root->right->right->left) first, and then, the path can be formed.

So, the recursive solution can be based on the following facts:

1. Any node in the tree can be visited from top to down, preorder traversal, once. And the node is treated as root node of one subtree.

2. So, just consider the maximum path starting from root node, this way, the recursive function can be set up.

2. Any two nodes in the tree -> the root to any node in the tree (simplify the case)

3. the maximum path in the above diagram: 6->9->-3->2->2 -> construct by the root to any node in the tree.

value1: Root node with value 9,

value2: Root-> left node's maximum value of "the root to any node in the tree"

value3: Root-> right node's maximum value of "the root to any node in the tree"

**the maximum path = value1 +(value2>0?value2:0) + (value3>0?value3:0)**

So, the recursive solution can be formed quickly.

**August 4, 2016**

Come back to review the previous work, but it is still not easy to tell the ideas to solve the problem.

### **Reasoning and analysis (II):**

Add one more explanation to walk through the example:

First talk about " **maximum value cross root** " - variable: `maxValueCrossRoot`,

each node is the root of its subtree, so the maximum one will be maximum one from the following list:

1. n1: root node (9): need to calculate `maximumEndByRoot` on right child (-3) first.
2. n2: left child (6):
3. n3: right child (-3):
4. n4: right->right child(-6): -6
5. n5: right->right->right (2): 4
6. n6: right->right->right->left(2) : 2
7. n7: right->right->right->left->left(-6): -6
8. n8: right->right->right->left->right(-6): -6

It is comparison by values.

**Tip** : 1.The idea to get the maximum value is to pass a reference int to any recursive function. Any subtree will have one value, and compare with the global variable's value.

2. Use preorder traversal to travel tree once.

And " **maximum value end by root** " - variable: `maximumEndByRoot`

the above node n8: -6

n7: -6

n6: 2

n5: 4

n3: +1

n2: 6

So, the root node `maxValueCrossRoot` =  $9 + 6 + 1 = 16$

`maximumByEnd` = 15.

**Tip** : 1. use recursive function with return value - `maxValueEndByRoot`, the formula is easy to recall.

2. use preorder traversal to travel tree once.

The above case is coincident, the `maxValueCrossRoot` is ended at the root node n1 (value 9), which may be anywhere ( $n_i$ , i is one value from 1 to 8) in the tree.

### **Reasoning and analysis (III):**

#### **Creative way to solve the problem:**

##### **1. Work on a simple problem first:**

Maximum value end by root in a binary tree - in other words, maximum value from the root node to any node in binary tree

[7]<https://gist.github.com/jianminchen/8f3ec942e90bcdca5a1569d1a70e92df>

Goal: be able to write the function in 10 minutes, verify code with static analysis.

## 2. Add one more function in the above program -

Based on the simple problem - maximum value end by the root node, add one more task to the function: maxValCrossRoot calculation. (bottom up solution)

[8]<https://gist.github.com/jianminchen/5eab22189f0fd7a58aa4fbc56b725dd8>

Add 2 more lines of code: line 104, 105, add one more input argument - ref int maxCrossRoot, 3 places update

Goal: make it 10 minutes to add

So, overall, it should be less than 20 minutes code writing. Follow the function - do one task only: a time.

1. **Step 1:** write a simple recursive function - preorder traversal. Pay attention to negative value node, just ignore them if it is. Avoid if statement, just get minimum value through 3 values, make it one line statement - no if/else discussion of left/right child value > 0.

Only 5 lines of code. Short and concise.

```
89     private static int getMaximumEndByRoot(TreeNode root )
90     {
91         if (root == null)
92             return 0;
93
94         int left  = getMaximumEndByRoot(root.left);
95         int right = getMaximumEndByRoot(root.right);
96
97         // 3 values, get maximum one.
98         return Math.Max(root.val, Math.Max(root.val + left, root.val + right));
99     }
```

2. **Step 2:** add 2 lines code (line 104, line 105), 3 changes - add one more argument (line 96, line 101, line 102):

```

96     private static int getMaximumEndByRoot(TreeNode root, ref int maxCrossRoot)
97     {
98         if (root == null)
99             return 0;
100
101         int left = getMaximumEndByRoot(root.left, ref maxCrossRoot);
102         int right = getMaximumEndByRoot(root.right, ref maxCrossRoot);
103
104         int value = root.val + ((left > 0) ? left : 0) + ((right > 0) ? right : 0);
105         maxCrossRoot = (value > maxCrossRoot) ? value : maxCrossRoot;
106
107         // 3 values, get maximum one.
108         return Math.Max(root.val, Math.Max(root.val + left, root.val + right));
109     }

```

1. <http://juliachencoding.blogspot.ca/2015/07/leetcode-maximum-binary-tree-path-sum.html>
2. <http://juliachencoding.blogspot.ca/2015/07/itint5-tree-maximum-path-sum.html>
3. <https://gist.github.com/jianminchen/432bdb1af38a1ec6f4a0741420891ebf>
4. <https://gist.github.com/jianminchen/578656e1079e8c58b08dd19f5b027e68>
5. <https://gist.github.com/jianminchen/9fc060ae74290637f732a0fc820f44>
6. <https://gist.github.com/jianminchen/578656e1079e8c58b08dd19f5b027e68>
7. <https://gist.github.com/jianminchen/8f3ec942e90bcdca5a1569d1a70e92df>
8. <https://gist.github.com/jianminchen/5eab22189f0fd7a58aa4fbc56b725dd8>

## 2.6 June

### Leetcode 146: LRU Cache - a practice makes difference (III) (2016-06-02 21:11)

June 2, 2016

First two practices:

[1][http://juliachencoding.blogspot.ca/2016/04/leetcode-146-lru-cache\\_26.html](http://juliachencoding.blogspot.ca/2016/04/leetcode-146-lru-cache_26.html)

Warm up the algorithm: (a lot of hurdles, just read source code and then add a lot of comments)

[2]<https://gist.github.com/jianminchen/207e20632f7837285ee9b913b0d34f3a>

The third practice:

[3]<https://gist.github.com/jianminchen/3bd8cdab7a31662d402c62fff9c0b597>

**Statistics:**

Time spent: 3+ hours

1. [http://juliachencoding.blogspot.ca/2016/04/leetcode-146-lru-cache\\_26.html](http://juliachencoding.blogspot.ca/2016/04/leetcode-146-lru-cache_26.html)
2. <https://gist.github.com/jianminchen/207e20632f7837285ee9b913b0d34f3a>
3. <https://gist.github.com/jianminchen/3bd8cdab7a31662d402c62fff9c0b597>

## Leetcode 146: LRU cache - a practice makes difference (II) (2016-06-02 21:11)

June 2, 2016

Previous blog:

[1][http://juliachencoding.blogspot.ca/2016/04/leetcode-146-lru-cache\\_26.html](http://juliachencoding.blogspot.ca/2016/04/leetcode-146-lru-cache_26.html)

Warm up the algorithm: (a lot of hurdles to go through)

[2]<https://gist.github.com/jianminchen/207e20632f7837285ee9b913b0d34f3a>

Questions and answers:

### How to design LRU? Data structure? What for?

copy from blog:[3][http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/?utm\\_source=email&utm\\_medium=email&utm\\_campaign=email](http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/?utm_source=email&utm_medium=email&utm_campaign=email)

LRU

One of the most common cache systems is [4]LRU (least recently used). In fact, another common interview question is to discuss data structures and design of an LRU cache. Let's start with this approach.

The way LRU cache works is quite simple. When the client requests resource A, it happens as follow:

- If A exists in the cache, we just return immediately.
- If not and the cache has extra storage slots, we fetch resource A and return to the client. In addition, insert A into the cache.
- If the cache is full, we kick out the resource that is least recently used and replace it with resource A.

The strategy here is to maximum the chance that the requesting resource exists in the cache. So how can we implement a simple LRU?

### LRU design

An LRU cache should support the operations: lookup, insert and delete. Apparently, in order to achieve fast lookup, we need to use hash. By the same token, if we want to make insert/delete fast, something like linked list should come to your mind. Since we need to locate the least recently used item efficiently, we need something in order like queue, stack or sorted array.

To combine all these analyses, we can use queue implemented by a doubly linked list to store all the resources. Also, a hash table with resource identifier as key and address of the corresponding queue node as value is needed.

Here's how it works. when resource A is requested, we check the hash table to see if A exists in the cache. If exists, we can immediately locate the corresponding queue node and return the resource. If not, we'll add A into the cache. If there are enough space, we just add a to the end of the queue and update the hash table. Otherwise, we need to delete the least recently used entry. To do that, we can easily remove the head of the queue and the corresponding entry from the hash table.

### Eviction policy

When the cache is full, we need to remove existing items for new resources. In fact, deleting the least recently used item is just one of the most common approaches. So are there other ways to do that?

As mentioned above, The strategy is to maximum the chance that the requesting resource exists in the cache. I'll briefly mention several approaches here:

- Random Replacement (RR) – As the term suggests, we can just randomly delete an entry.
- **Least frequently used (LFU)** – We keep the count of how frequent each item is requested and delete the one least frequently used.
- W-TinyLFU – I'd also like to talk about this modern eviction policy. In a nutshell, the problem of LFU is that sometimes an item is only used frequently in the past, but LFU will still keep this item for a long while. W-TinyLFU solves this problem by calculating frequency within a time window. It also has various optimizations of storage.

Skip concurrency and distributed cache in this design.

end copy from blog:[5][http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/?utm\\_source=email&utm\\_medium=email&utm\\_campaign=email](http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/?utm_source=email&utm_medium=email&utm_campaign=email)

**Next, talk about coding part - data structure and algorithm using her own words:**

C # implementation.

[6]<https://gist.github.com/jianminchen/3dc7da7e0465819e6aa8c10c71901723>

int **capacity** - specify the size of cache, cache should be with limited size, since resource is limited, specially for high speed access. source code on line 24.

int **size** - current size of cache - track current size of cache to determine if eviction is needed or not.

source code on line 24.

Design a double linked list, so ListNode class is defined with two pointers, **prev**, **next**

source code from line 10 - line 22.

every entry has key, value. Use int to simplify the coding. Source code, line 12.

**line 12 public int key, val;**

Also, we need to add two more variables: dummy head, dummy tail to help maintain the double linked list. source code on line 25.

Also, we need to be able to find the key in O(1) since it is in cache. Extra space is used, maintain a hash map using Dictionary class in C #:

**line 27 Dictionary(int key, ListNode)**

**Let us count how many variables inside the class LRUCache:**

**6 variables: - memorize the variable count - 6 - Try to recall.**

**private int capacity, size;**

**private ListNode dummyHead, dummyTail;**

**private Dictionary<int, ListNode> map;**

**ListNode class as a node in a double linked list:**

**public int key, value;**

**public ListNode prev, next;**

**4 variables.**



## Statistics:

Time spent: 2 hours +

Research paper: feel so good to read a research paper after some coding

[7]<http://www.cs.cmu.edu/~pavlo/courses/fall2013/static/papers/11730-atc13-bronson.pdf>

[8]<http://blog.csdn.net/whuwangyi/article/details/42801293>

Reading blog: (June 30, 2016)

[9]<http://dl.acm.org/citation.cfm?id=2522722>

Reading blog: (July 27, 2016)

[10]<http://cenalulu.github.io/linux/all-about-cpu-cache/>

1. [http://juliachencoding.blogspot.ca/2016/04/leetcode-146-lru-cache\\_26.html](http://juliachencoding.blogspot.ca/2016/04/leetcode-146-lru-cache_26.html)
  2. <https://gist.github.com/jianminchen/207e20632f7837285ee9b913b0d34f3a>
  3. [http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/?utm\\_source=email&utm\\_medium=email&utm\\_campaign=email](http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/?utm_source=email&utm_medium=email&utm_campaign=email)
  4. [https://en.wikipedia.org/wiki/Cache\\_algorithms#LRU](https://en.wikipedia.org/wiki/Cache_algorithms#LRU)
  5. [http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/?utm\\_source=email&utm\\_medium=email&utm\\_campaign=email](http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/?utm_source=email&utm_medium=email&utm_campaign=email)
  6. <https://gist.github.com/jianminchen/3dc7da7e0465819e6aa8c10c71901723>
  7. <http://www.cs.cmu.edu/~pavlo/courses/fall2013/static/papers/11730-atc13-bronson.pdf>
  8. <http://blog.csdn.net/whuwangyi/article/details/42801293>
  9. <http://dl.acm.org/citation.cfm?id=2522722>
  10. <http://cenalulu.github.io/linux/all-about-cpu-cache/>
- 

## small talk on code interview (2016-06-02 21:28)

June 2, 2016

Still try to learn something about white board/ code interview.

Study articles, and then write down some notes:

1. [1]<https://blog.dev mastery.com/how-to-win-the-coding-interview-71ae7102d685#.3ilg3h6vb>

Whiteboard and Coding:

Look for a developer who can think on her feet, under pressure, in a room with others.

### 1. Verbalize your assumptions and seek to confirm them .

Think about assumptions you might be making, and ask me about them.

### 2. Think out loud .

Show your thought process.

Knowing that you understand how to reason about a problem is far more valuable to me than knowing that you've memorize the name of some built-in function.

Julia's comment:

Draw a diagram to prepare a test case for the problem, at least show some complexity of problem.  
Like brute force discussion - how to solve them.

### 3. **Don't be afraid to ask for help**

It is very expensive to hire someone who refuses to ask for help when he is stuck?

If you are stuck or don't know something, ask me.

#### **Julia's comment:**

Julia has to learn how to communicate, learn a new word: Speak concise with some detail. Stop, and then ask if I answer your question. Do not speak more, showing no confidence. People will ask you more if you stop. Do not volunteer information, do not change topic.

### 4. **Represent your skills and experience honestly**

There is a threshold for questions and commentary.

#### **Julia's comment:**

Always be humble. And open to learn.

Part 2 - Coding on a computer

#### 1. **Code to spec** -

Ask good questions -

#### 2. **How well you can follow instructions.**

#### 3. **Assume that this code will be put into a real production system and write accordingly.**

Your code should be commented.

You should have error handling or at least logging.

Your code should avoid breaking at all costs.

You should have a test harness.

Your code should be easy-to-read and self-explanatory. (clear variable names, good formatting, ideally "lint free" code).

suggestions: check JavaScript, jQuery codebase on GitHub.

"efficient" in production:

Run fast/ Doesn't take up more memory than it needs to / is stable and easy to maintain.

### **Part 3: Algorithms**

Khan Academy - [2]<https://www.khanacademy.org/computing/computer-science/algorithms>

### **Part 4 - Passing without solving the problem**

#### 1. **Do not give up too easily.**

Put in a real effort.

Julia's comment: Usually there are over 10 solutions working out for a problem. Julia saw hackerRank submit-

sions, so many creative ideas out there.

## 2. Pseudo-code it.

**Julia's comment:** call your own function, write a few small functions to support your algorithm. Do not interrupt your main function, write small functions afterwards.

## 3. List your Know Unknowns

## Part 5 - Practice, Practice, Practice

**If you practice enough, and really work at it, you'll even be able to handle a question you've never seen before. You'll have confidence and you'll be able to relate it to something else you've probably tried.**

2. [3]<https://medium.com/@evnowandforever/f-you-i-quit-hiring-is-broken-bb8f3a48d324#.y3j7n2m>

How the hashtable is implemented?

[4][https://en.wikipedia.org/wiki/Hash\\_table](https://en.wikipedia.org/wiki/Hash_table)

Questions in interview

1. Hash function, key -> index value in associated array

Load factor

Collision

Open addressing

2. Two sum

3. Palindrome

1. <https://blog.dev mastery.com/how-to-win-the-coding-interview-71ae7102d685#.3ilg3h6vb>

2. <https://www.khanacademy.org/computing/computer-science/algorithms>

3. <https://medium.com/@evnowandforever/f-you-i-quit-hiring-is-broken-bb8f3a48d324#.y3j7n2m>

4. [https://en.wikipedia.org/wiki/Hash\\_table](https://en.wikipedia.org/wiki/Hash_table)

---

## Top 10 Ranking Algorithm problems - June 2016 (2016-06-02 23:30)

June 2, 2016

Julia likes to rank her favorite algorithms and also her practice, then she has to chance to dig into more and develop some skills through the practice.

Here are top 10 practice she likes most:

1. LRU cache - Leetcode 146 - third practice:

[1][http://juliachencoding.blogspot.ca/2016/06/leetcode-146-lru-cache-practice-makes\\_2.html](http://juliachencoding.blogspot.ca/2016/06/leetcode-146-lru-cache-practice-makes_2.html)

[2][http://juliachencoding.blogspot.ca/2016/06/leetcode-146-lru-cache-practice-makes\\_45.html](http://juliachencoding.blogspot.ca/2016/06/leetcode-146-lru-cache-practice-makes_45.html)

Practice a few more times.

2. Leetcode 126 - Word Ladder II (hard) - fifth practice

[3]<http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-using-bfs.html>

3. HackerRank: String - Reverse Shuffle Merge (advanced) 5 practices

[4]<https://gist.github.com/jianminchen/7572e92ea48211d3d05c557d97601dbf>

4. Serialize and Deserialize Tree - 2 study cases

[5]<http://juliachencoding.blogspot.ca/2016/05/leetcode-297-serialize-and-deserialize.html>

5. Binary Tree Maximum Path sum - second practice

[6]<http://juliachencoding.blogspot.ca/2016/05/leetcode-124-binary-tree-maximum-path.html>

6. Binary Tree Preorder Traversal - Iterative solution -

[7]<http://juliachencoding.blogspot.ca/2016/05/binary-tree-preorder-traversal.html>

7. Leetcode 159 - longest substring with at most two distinct characters

[8]<http://juliachencoding.blogspot.ca/2016/05/leetcode-159-longest-substring-with-at.html>

8. Leetcode 236 - Binary Tree Lowest Common Ancestor - **Recursive, Backtracking**

**First blog: (5 practices)**

[9]<http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor.html>

**Second blog:**

[10][http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor\\_21.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor_21.html)

**Third blog:**

[11][http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor\\_22.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor_22.html)

**Fourth blog:**

[12]<http://juliachencoding.blogspot.ca/2016/05/leetcode-236-binary-tree-lowest-common.html>

9. Binary Tree Post Order Traversal - Iterative solution

[13]<http://juliachencoding.blogspot.ca/2016/05/binary-tree-post-order-traversal.html>

10. Leetcode 17: Phone Number Combination

[14]<http://juliachencoding.blogspot.ca/2016/04/window-sum.html>

There are so many algorithms to list here. But just keep 10 a time.

Forget what everybody else is doing. [15]@kikimladenovic is the first on court and the last one off. [16]  
#CreateYourMark [17]pic.twitter.com/3E7n6psKdF — adidas tennis (@adidastennis) [18]May 18, 2016

1. [http://juliachencoding.blogspot.ca/2016/06/leetcode-146-lru-cache-practice-makes\\_2.html](http://juliachencoding.blogspot.ca/2016/06/leetcode-146-lru-cache-practice-makes_2.html)
2. [http://juliachencoding.blogspot.ca/2016/06/leetcode-146-lru-cache-practice-makes\\_45.html](http://juliachencoding.blogspot.ca/2016/06/leetcode-146-lru-cache-practice-makes_45.html)
3. <http://juliachencoding.blogspot.ca/2016/05/leetcode-126-word-ladder-ii-using-bfs.html>
4. <https://gist.github.com/jianminchen/7572e92ea48211d3d05c557d97601dbf>
5. <http://juliachencoding.blogspot.ca/2016/05/leetcode-297-serialize-and-deserialize.html>
6. <http://juliachencoding.blogspot.ca/2016/05/leetcode-124-binary-tree-maximum-path.html>
7. <http://juliachencoding.blogspot.ca/2016/05/binary-tree-preorder-traversal.html>
8. <http://juliachencoding.blogspot.ca/2016/05/leetcode-159-longest-substring-with-at.htm>
9. <http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor.html>
10. [http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor\\_21.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor_21.html)
11. [http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor\\_22.html](http://juliachencoding.blogspot.ca/2016/05/leetcode-236-lowest-common-ancestor_22.html)
12. <http://juliachencoding.blogspot.ca/2016/05/leetcode-236-binary-tree-lowest-common.html>
13. <http://juliachencoding.blogspot.ca/2016/05/binary-tree-post-order-traversal.html>
14. <http://juliachencoding.blogspot.ca/2016/04/window-sum.html>
15. <https://twitter.com/KikiMladenovic>
16. <https://twitter.com/hashtag/CreateYourMark?src=hash>
17. <https://t.co/3E7n6psKdF>
18. <https://twitter.com/adidastennis/status/732933865476591617>

---

## Study on failing - a small research starts a learning Sunday (2016-06-05 10:18)

June 5, 2016

Julia thought about coding on this Sunday, and then, she decided to do some research first. The topic is called "study on failing".

### 1. Kevin O'Lery - Failing is Good

[1]<https://www.youtube.com/watch?v=1c6OoOHXz0k>

1. Fail once, learn and move on
  2. 36 months does not make money, then shoot it. It is not a business, it is a hobby.
  3. Lose job - Mistakes
  2. Kids in school, remove grades system, allow to fail? Kevin: Fail in early age, learn quickly. Let them prepare early. Life is tough out there. Do not make life too easy for kids, let them learn.
  3. Learn from Shark Tank TV show,
- ask about Marks for failure:

Fall apart in talking about numbers.

4. Relationship failure - tell the truth.

## 2. Kevin O'Leary on how to get ahead in the workplace

[2]<https://www.youtube.com/watch?v=bG6L5z7fMxE>

1. Complete job, and this is a must.
  2. Great employee gets the job done; and do not make too much noise.
  3. Make money, not make friends.
  4. Work alone vs work in team.
  5. Dress for work you want?
- Code ethics - People do not remember what you wear yesterday, that is good.
6. Woman in leadership role?
- Kevin: focus, do not take huge risk; Canada is very open.

## 3. Kevin O'Leary tips on Negotiating

[3]<https://www.youtube.com/watch?v=0TUvuHRRLLCA>

1. Hesitate to negotiate: tell the truth, take the heat, do not lie.
  2. Set goals to live with, have some flexibility
  3. How to teach kids to negotiate?
- Kevin: teach kids starting from 5 years old. Give them everything they want, you never teach kids anything about negotiate.

Remember a saying: you do not get your deserving, you get what you are negotiating.

## 4. Process of negotiating:

3 quotes of changing cars, some one will tell the truth. Spend time to get 3 quotes.

## 5. Kevin O'Leary's Story

[4][https://www.youtube.com/watch?v=mnCmmHs\\_XO8](https://www.youtube.com/watch?v=mnCmmHs_XO8)

Strict advice: put down guitar, pick up books.  
Step father tried to help him out in his teenager.  
Play music, take picture, doing nothing. But, step father does not like him do that.

A story, the owner asked him to take off a gum stick on the floor.

## 7. ABC 20/20 says Kevin O'Leary is a Bosshole!

[5]<https://www.youtube.com/watch?v=Y-SDadRCUug>

Respect the boss and trust from the employer.  
Sell software billions, at the beginning from her mom's money \$10,000 level.  
Dad is not sharing his wealth. Only pay his children's education up to their need. Go figure out, go fix it.

## 8. Kevin O'Leary: Seteve Jobs was the "toughest bastard" ever

[6]<https://www.youtube.com/watch?v=eIYDUhLdF3I>

Questions:

Respectable vs Fearful

Concept: Work with people never being friends.

Great sales people - a little of bit arts, science

Both are unhappy -> good deal

Fear vs greed  
Stock trade -  
Transparency of stock trade  
Sales are born or made - great sales - Sales manager is different from sales people  
Never invest in salesman to run sales -

9. Book:

**Kevin O'Leary Book:**

Cold hard truth on business money and life.

2 minutes video - the process to write the book, funny and so funny:

[7][https://www.youtube.com/watch?v=u\\_-ppODHfSk](https://www.youtube.com/watch?v=u_-ppODHfSk)

See if I can take down the words to a note correctly 80 % in 20 minutes, again and again. English, vocabulary:

**First try:**

It is dark and contamin (?) night, the enemy is everywhere.

How to turn a few m

And also I

And also my softside

By my book, ..., a lot of books. Why? Because it is about money, cold hard truth.

mm. Beautiful

**Second try:**

It is a dk and strm nigt. compt everyhwer, tk out. Too soft.

Compt evh, probly kill them all.

writ bk -> a b dollar empire

Scratch . I am not doing that.

Where was I ? Buy book, a lot, mom, even your dog? Why? Money.

Eat that cold h truth.

**Third try:**

buddy, kitchen, heat

book worms

typing..

it is dk strmy nigt. compt looks evhe, take them out. Too soft?

Crush them, anngli them, ptu boiling water,

Few tho -> bill, soft side?

scrtach, tearning paper.

Where was I ?

Buy book, for evehwehre, for money, eat that for cold hard truth.

beautiful at end with wine.

**Fourth try: speed typing.**

Buddy-> kitchen -> heat

book worms, with music

typing:

It is dk and stmy nt; Cpt evy, tkae them out -> too soft!

Crusht hem , ana, boiling wta, that is what i like.  
Too may , also revealing soft side. a lot of per on flr  
buy my bks, a lot of bks, cat even u dog. Why? money  
Cold hard truth.

#### **Here is the notes:**

So, it is the dark and stormy night, competitors are everywhere. Take them out. Too soft?  
Crush them like croches, annihilate them, put boiling water, ..., kill them all.  
Two thousand to a billionaire dollar empire.  
Where was I ?  
Buy my books, buy a lot of books, for ...

#### **Cold hard truth.**

The research just starts from "study on failing", and then, catch up more on outside world.

Video:

Eben Pagan's Top 10 Rules For Success

[8]<https://www.youtube.com/watch?v=IOjfqU-ngwM>

Michael Gerber's Top 10 Rules For Success

[9]<https://www.youtube.com/watch?v=cjy0shpSh60>

1. <https://www.youtube.com/watch?v=1c60o0HXz0k>
  2. <https://www.youtube.com/watch?v=bG6L5z7fMxE>
  3. <https://www.youtube.com/watch?v=OTUvuHRRLCA>
  4. [https://www.youtube.com/watch?v=mnCmmHs\\_X08](https://www.youtube.com/watch?v=mnCmmHs_X08)
  5. <https://www.youtube.com/watch?v=Y-SDadRCUug>
  6. <https://www.youtube.com/watch?v=eIYDUhLdF3I>
  7. [https://www.youtube.com/watch?v=u\\_-ppODHfSk](https://www.youtube.com/watch?v=u_-ppODHfSk)
  8. <https://www.youtube.com/watch?v=IOjfqU-ngwM>
  9. <https://www.youtube.com/watch?v=cjy0shpSh60>
- 

## **Top 10 ranking algorithm and practice in June, 2016 (II) (2016-06-07 21:44)**

June 7, 2016

Put together her favorite 10 algorithm and practice in the following:

1. HackerRank: Sherlock and anagram - 7 blogs with practices

**Practice on HackerRank is like to play tennis sports, you have to experience various hitting partners, good workout!**

No. 1.



[1]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-string-sherlock-and-anagrams.html>

No. 2

[2]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-ii.html>

No. 3

[3]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlocks-and-anagram-ii.html>

No. 4 **Study 6 solutions - Julia chose from over 200 solutions**

[4]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-iv.html>

**Julia, work on this code, and put test case in, write your own C # practice:  
study the code, order by values, not by key, try it!**

[5]<https://gist.github.com/jianminchen/ffca0582b5f0d1d6a9b>

study the blog: Dictionary OrderByDescending

[6]<https://goo.gl/6ZbTPz>

baby step to learn C # Dictionary class API - Order by and distinct

[7]<https://gist.github.com/jianminchen/eff03bea08a95061deb4185af74fea18>

No. 5

[8][http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-iv\\_27.html](http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-iv_27.html)

No. 6

[9]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagram-vii.html>

**Warm up 12 solutions - "Sherlock and anagram" one by one. Each one for a blog.** Focus on speed, correctness.

**Back to top 10 Ranking - top 2 ...**

2. Leetcode 208 course schedule - warm up the algorithm, practice again in short future.

[10]<http://juliachencoding.blogspot.ca/2016/04/leetcode-208-course-schedule.html>

3. HackerRank: Two strings - C # 15 solutions

[11]<http://juliachencoding.blogspot.ca/2016/03/hacker-rank-two-strings.html>

4. Leetcode 312 - Burst Balloons - **Julia, please practice it using C # and post here later.**

[12]<http://juliachencoding.blogspot.ca/2016/02/leetcode-312-burst-balloons.html>

5. Binary Tree Post Order Traversal - Iterative solution

[13]<http://juliachencoding.blogspot.ca/2016/05/binary-tree-post-order-traversal.html>

6. Leetcode 215: find kth largest element - **Julia, please practice it using C # and post here later.**

<http://juliachencoding.blogspot.ca/2016/05/leetcode-215-find-kth-largest-element.html>

Study the solution implemented in Java:

[14]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/21>

5.kth-largest-element-in-an-array.java

7. Connect nodes at same level in binary tree - **Julia, please practice it using C # and post here later.**

[15]<http://juliachencoding.blogspot.ca/2016/05/connect-nodes-at-same-level-in-binary.html>

8. Leetcode 15: 3 sum

[16]<http://juliachencoding.blogspot.ca/2016/05/leetcode-15-3-sum.html>

9. Check number is power of 2

[17]<http://juliachencoding.blogspot.ca/2016/05/algorithm-check-if-number-is-power-of-2.html>

10. Radix sort and practice:

[18]<http://juliachencoding.blogspot.ca/2016/05/radix-sort-distribution-sort.html>

11. Leetcode: Maximum Gap

[19]<http://juliachencoding.blogspot.ca/2015/06/leetcode-distribution-sort-algorithm.html>

[20]<http://juliachencoding.blogspot.ca/2015/06/leetcode-maximum-gap-no-164.html>

I know what it takes to win, first arrive, last to leave.

Forget what everybody else is doing. [21]@kikimladenovic is the first on court and the last one off. [22]

#CreateYourMark [23][pic.twitter.com/3E7n6psKdF](http://pic.twitter.com/3E7n6psKdF)

— adidas tennis (@adidastennis) [24]May 18, 2016

1. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-string-sherlock-and-anagrams.html>

2. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-ii.html>

3. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlocks-and-anagram-ii.html>

4. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-iv.html>

5. <https://gist.github.com/jianminchen/ffcca0582b5f0d1d6a9b>

6. <https://goo.gl/6ZbTPz>

7. <https://gist.github.com/jianminchen/eff03bea08a95061deb4185af74fea18>

8. [http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-iv\\_27.html](http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-iv_27.html)

9. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagram-vii.html>

10. <http://juliachencoding.blogspot.ca/2016/04/leetcode-208-course-schedule.html>

11. <http://juliachencoding.blogspot.ca/2016/03/hacker-rank-two-strings.html>

12. <http://juliachencoding.blogspot.ca/2016/02/leetcode-312-burst-balloons.html>

13. <http://juliachencoding.blogspot.ca/2016/05/binary-tree-post-order-traversal.html>

14. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/215.kth-largest-element-in-an-array.java>

15. <http://juliachencoding.blogspot.ca/2016/05/connect-nodes-at-same-level-in-binary.html>

16. <http://juliachencoding.blogspot.ca/2016/05/leetcode-15-3-sum.html>

17. <http://juliachencoding.blogspot.ca/2016/05/algorithm-check-if-number-is-power-of-2.html>
18. <http://juliachencoding.blogspot.ca/2016/05/radix-sort-distribution-sort.html>
19. <http://juliachencoding.blogspot.ca/2015/06/leetcode-distribution-sort-algorithm.html>
20. <http://juliachencoding.blogspot.ca/2015/06/leetcode-maximum-gap-no-164.html>
21. <https://twitter.com/KikiMladenovic>
22. <https://twitter.com/hashtag/CreateYourMark?src=hash>
23. <https://t.co/3E7n6psKdF>
24. <https://twitter.com/adidastennis/status/732933865476591617>

---

Giri Mani 2 (2016-06-09 05:11:32)

This comment has been removed by a blog administrator.

## **A small research on RestAPI, SOA, AWS, AZURE (2016-06-07 22:48)**

June 7, 2016

Read the article:

[1]<https://www.linkedin.com/pulse/rest-vs-rpc-soa-showdown-joshua-hartman?trk=hp-feed-article-title-hpm>

About complaints:

[2][http://www.nytimes.com/2015/07/29/business/linkedin-notorious-for-sending-too-many-emails-cuts-back.html?\\_r=0](http://www.nytimes.com/2015/07/29/business/linkedin-notorious-for-sending-too-many-emails-cuts-back.html?_r=0)

[3]<https://azure.microsoft.com/en-us/campaigns/azure-vs-aws/>

Amazon AWS:

[4]<http://blog.hackerrank.com/how-amazon-web-services-surged-out-of-nowhere/>

[5]<https://aws.amazon.com/resources/gartner-storage-mq-2014-learn-more/?tag=viglink121898-20>

AWS provides durable, low cost cloud storage solutions for their backup, storages and achiving needs. (Pinterest, Dropbox, Adobe, ASANA, etc. )

**Actionable items:** plan to study 2 hours

Study **Amazon Kinesis** , most innovative service. **Julia, can you name 3 services in AWS?**

[6]<https://aws.amazon.com/kinesis/streams/>

[7]<https://aws.amazon.com/kinesis/streams/details/>

Leveling up your JavaScript: - plan to spend at least 30 minutes to read

[8]<http://goo.gl/OmDASd>

plan to read the article 30 minutes

[9]<http://www.cs.cmu.edu/~pavlo/courses/fall2013/static/papers/11730-atc13-bronson.pdf>

1. <https://www.linkedin.com/pulse/rest-vs-rpc-soa-showdown-joshua-hartman?trk=hp-feed-article-title-hpm>
  2. [http://www.nytimes.com/2015/07/29/business/linkedin-notorious-for-sending-too-many-emails-cuts-back.html?\\_r=0](http://www.nytimes.com/2015/07/29/business/linkedin-notorious-for-sending-too-many-emails-cuts-back.html?_r=0)
  3. <https://azure.microsoft.com/en-us/campaigns/azure-vs-aws/>
  4. <http://blog.hackerrank.com/how-amazon-web-services-surged-out-of-nowhere/>
  5. <https://aws.amazon.com/resources/gartner-storage-mq-2014-learn-more/?tag=viglink121898-20>
  6. <https://aws.amazon.com/kinesis/streams/>
  7. <https://aws.amazon.com/kinesis/streams/details/>
  8. <http://goo.gl/0mDASd>
  9. <http://www.cs.cmu.edu/~pavlo/courses/fall2013/static/papers/11730-atc13-bronson.pdf>
- 

## Content marketing - ideas to write blogs (2016-06-10 20:13)

June 10, 2016

Spend 1 - 2 hours to do some research on content marketing. It takes some preparation to be able to write a good coding blog, the basic content marketing techniques are helpful.

[1]<http://www.forbes.com/sites/jaysondemers/2014/04/16/50-content-marketing-ideas-for-your-website-or-blog/#23f88b677211>

Mark the ideas Julia likes:

16. Look through your analytics to see your top three blog posts, then write a follow up post for each one. (rank 8 of 10)

28. Create category pages on your website or blog that can make finding your content easier for your visitors. (7 out of 10)

32. Try using Quora to find questions people are asking in your niche or industry.

33. Compile Top 10 resource lists for your niche: Top 10 blogs; top 10 companies; top 10 tools, etc. (

**10 out of 10** ).

Some blogs:

1. [2]<http://www.slideshare.net/AxonnMedia/full-report-axonn-research-content-marketing-trends-in-2013>

2. [3]<http://www.audiencebloom.com/2013/04/how-to-build-a-kickass-content-strategy/>

user-focused content strategies that will stand the test of time, while winning approval from users as well as Google.

**Julia's comment:**

**Learning takes time; share personal coding experience, and encourage others to work hard as well.**

2. [4]<http://www.codeofhonor.com/blog/marketing-yourself-as-a-programmer>

3. [5]<http://www.kalzumeus.com/2011/10/28/dont-call-yourself-a-programmer/>

4. [6]<https://successfulsoftware.net/2015/06/10/7-reasons-software-developers-should-learn-marketing/>

1. <http://www.forbes.com/sites/jaysondemers/2014/04/16/50-content-marketing-ideas-for-your-website-or-blog/#23f88b677211>
  2. <http://www.slideshare.net/AxonMedia/full-report-axon-research-content-marketing-trends-in-2013>
  3. <http://www.audiencebloom.com/2013/04/how-to-build-a-kickass-content-strategy/>
  4. <http://www.codeofhonor.com/blog/marketing-yourself-as-a-programmer>
  5. <http://www.kalzumeus.com/2011/10/28/dont-call-yourself-a-programmer/>
  6. <https://successfulsoftware.net/2015/06/10/7-reasons-software-developers-should-learn-marketing/>
- 

## **C# Hashtable vs Dictionary (2016-06-10 20:36)**

June 10, 2016

Spend 2 - 3 hours to look into this topic: **C # Hashmap Hashtable vs Dictionary class**

**Reading always helps to think deeper, and expand the knowledge. Write down notes:**

[1]<http://stackoverflow.com/questions/301371/why-is-dictionary-preferred-over-hashtable>

[2] [http://www.codeproject.com/](http://www.codeproject.com/Tips/602537/Hashtable-vs-Dictionary)

Tips/602537/Hashtable-vs-

Dictionary

Plan to go over each function in Dictionary class in C #, understand them; get familiar with Dictionary class like SQL statement, slice and dice gets easy to use Dictionary class APIs.

**Code to study:**

**study the code, use Dictionary class, "Sherlock and anagram" - hackerRank,**

[3]<https://gist.github.com/jianminchen/ffcca0582b5f0d1d6a9b>

Solution 2:

use Dictionary class, string key for anagram string, use GetHashCode() call to turn key as Int.

[4]<https://gist.github.com/jianminchen/ffcca0582b5f0d1d6a9b>

Read about GetHashCode() webpage:

[5][https://msdn.microsoft.com/en-us/library/system.object.gethashcode\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.object.gethashcode(v=vs.110).aspx)

Solution 3:

[6]<https://gist.github.com/jianminchen/8f6bd4631f0b5f0bdee7>

Solution 4.

use Dictionary class, sort the key string, then anagram strings will be the same.

[7]<https://gist.github.com/jianminchen/59e326cbd1d8910c01c7>

30 minutes to practice:

study the blog: Dictionary OrderByDescending

[8]<https://goo.gl/6ZbTPz>

baby step to learn C # Dictionary class API - Order by and distinct

[9]<https://gist.github.com/jianminchen/eff03bea08a95061deb4185af74fea18>

#### **Review a few of terms:**

1. generic type -
2. boxing and unboxing - Hashtable stores Object, need boxing and unboxing
3. thread safe - all members vs. only public static members
4. speed concern - Dictionary is faster than Hashtable

#### **Actionable items:**

1. Read all dictionary classes:

- ConcurrentDictionary - thread safe (can be safely accessed from several threads concurrently)
- HybridDictionary - optimized performance (for few items and also for many items)
- OrderedDictionary - values can be accessed via int index (by order in which items were added)
- SortedDictionary - items automatically sorted
- StringDictionary - strongly typed and optimized for strings

#### **Motivation talk:**

1. Julia likes to use Dictionary class instead of Hashtable, since Dictionary uses explicit type checking to match its declaration, and it is in compile time. Much better than run time Hashtable box/ unbox - type conversion.
2. Julia likes to use Dictionary class because it is easy to complete task like order by feature. She likes to find  $O(N)$  solution to sort, but in reality,  $O(n \log n)$  is fine, just call orderBy API.
3. Read article extension method:

#### **Extension Methods:**

[10]<https://msdn.microsoft.com/en-CA/library/bb383977.aspx>

1. <http://stackoverflow.com/questions/301371/why-is-dictionary-preferred-over-hashtable>
2. <http://www.codeproject.com/Tips/602537/Hashtable-vs-Dictionary>

3. <https://gist.github.com/jianminchen/ffcca0582b5f0d1d6a9b>
  4. <https://gist.github.com/jianminchen/ffcca0582b5f0d1d6a9b>
  5. [https://msdn.microsoft.com/en-us/library/system.object.gethashcode\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.object.gethashcode(v=vs.110).aspx)
  6. <https://gist.github.com/jianminchen/8f6bd4631f0b5f0bdee7>
  7. <https://gist.github.com/jianminchen/59e326cbd1d8910c01c7>
  8. <https://goo.gl/6ZbTPz>
  9. <https://gist.github.com/jianminchen/eff03bea08a95061deb4185af74fea18>
  10. <https://msdn.microsoft.com/en-CA/library/bb383977.aspx>
- 

## Learn to lose - my favorite verse to study (2016-06-11 11:13)

June 11, 2016

Learn to lose, I like the teaching from sprinter, most marketable athlete - Usain Bolt.

Like the video, remind me over hundreds of hours on tennis court, play sports in order to get rid of waist bubble, excess body fat, build some muscle and also learn some sports - team spirit, mental toughness, and a lot of fun.

[1]<https://www.youtube.com/watch?v=QL5mCN-TTBs>

Reading:

1. [2][https://en.wikipedia.org/wiki/Usain\\_Bolt](https://en.wikipedia.org/wiki/Usain_Bolt)

2. [3]<http://www.runnersworld.co.uk/interview/im-still-here-and-im-still-the-best-rw-talks-to-usain-bolt/13333.html> (10 minute reading, excellent to know how athlete keeps working hard)

copy some sentences Julia likes to read:

I no longer think about what the fans want, or who is running what times and who I need to be on the lookout for. I just think about me, and doing my job. Legacy is the main word in my life now and that means if I'm five meters ahead with 10 to go I won't be slowing down and beating my chest. I'll be pushing to get the best time I can, every time.

### Julia's comment :

Sprinter is much harder compared to software coding in 45 minutes range or system design. Julia, you can do it. Learn to lose.

1. <https://www.youtube.com/watch?v=QL5mCN-TTBs>

2. [https://en.wikipedia.org/wiki/Usain\\_Bolt](https://en.wikipedia.org/wiki/Usain_Bolt)

3. <http://www.runnersworld.co.uk/interview/im-still-here-and-im-still-the-best-rw-talks-to-usain-bolt/13333.html>

---

## Top 10 Sales techniques for Entrepreneurs (2016-06-11 11:56)

June 11, 2016

Learn to sell! Spend 1 - 2 hours to do some research on how to sell. Being a software programmer, how to sell the skills/ problem solving skills/ 45 minutes coding in a sprint.

[1]<https://www.youtube.com/watch?v=PwwgGOBw1oE>

1. Motivation Speaker: **Jordan Belfort** ,

[2][https://en.wikipedia.org/wiki/Jordan\\_Belfort](https://en.wikipedia.org/wiki/Jordan_Belfort)

2. Advice from **Tim Ferriss**

[3][https://en.wikipedia.org/wiki/Tim\\_Ferriss](https://en.wikipedia.org/wiki/Tim_Ferriss)

Practicing is only way to get better. Speaking ..., but that is the reason why written word is so powerful. Writing on a crispy paper and then can be reviewed. It is very difficult to do that for the speech. That is the reason I recommend to hold on skills on writing.

Take time to do study ads, printout etc. Then go back to review, what he falls for, this, or that. What is pushed you for the edge. And then, sell to people like you.

Book to read outside business books:

**Recommendation 1:**

**Sales** -> go to " **On Writing Well** "

On Writing Well by: William Zinsser

[4][https://books.google.ca/books/about/On\\_Writing\\_Well.html?id=USd6AAAAIAAJ&redir\\_esc=y&hl=en](https://books.google.ca/books/about/On_Writing_Well.html?id=USd6AAAAIAAJ&redir_esc=y&hl=en)

On Writing Well, which grew out of a course that William Zinsser taught at Yale, has been praised for its sound advice, its clarity, and for the warmth of its style. It is a book for anybody who wants to learn how to write or who needs to do some writing to get through the day, as almost everybody does. Whether you want to write about people or places, science and technology, business, sports, the arts, or about yourself in the increasingly popular memoir genre, On Writing Well offers you both fundamental principles as well as the insights of a distinguished practitioner. How to Write a Memoir tells you how to write the story of your life. Everyone has a story - whether you're a professional writer or just want to validate your personal and family reminiscences, William Zinsser explains how to do it, and do it well.

Recommendation 2:

**Bird by Bird** by: Anne Lamott

Good at writing, improving communication -

[5][https://books.google.ca/books/about/Bird\\_by\\_Bird.html?id=LISjPwAACAAJ&redir\\_esc=y](https://books.google.ca/books/about/Bird_by_Bird.html?id=LISjPwAACAAJ&redir_esc=y)

This volume presents the author's offbeat wisdom about how to write. She recounts her personal experiences to reveal her writing techniques and how she overcomes obstacles that interfere with the writing flow. The author offers concrete suggestions about character, plot, setting, and other topics of interest to writers. She also offers irreverent advice about how to navigate through the dark underbelly feelings of self-doubt, inadequacy, and jealousy that are



inevitable parts of any writer's experience. Her humorous advice is based on her own experience and honest self-analysis which provides writers the necessary perspective to keep writing through the difficult times that all writers encounter.

Recommendation 3:

Ogilvy on Advertising by: David Ogilvy

<https://books.google.ca/books?id=FIguPyk2w6YC&q=ogilvy+on+advertising+book&dq=ogilvy+on+advertising+book&hl=en&sa=X&ved=0ahUKEwiXyOqo2aDNAhUMzGMKHSKoChgQ6AEIJDA>

An advertising authority updates his analysis of the elements of successful advertising and assesses the advertising environment that has emerged during the past twenty years

### 3. **Gary Vaynerchuk**

[6][https://en.wikipedia.org/wiki/Gary\\_Vaynerchuk](https://en.wikipedia.org/wiki/Gary_Vaynerchuk)

### 4. **Brian Tracy** - Life long learning. Quality of thinking -

[7]<http://www.briantracy.com/>

[8]<http://www.briantracy.com/blog/time-management/3-underrated-tips-to-achieve-work-life-balance/>

[9]<http://www.briantracy.com/blog/category/sales-success/>

### 5. **Guy Kawasaki**

[10][https://en.wikipedia.org/wiki/Guy\\_Kawasaki](https://en.wikipedia.org/wiki/Guy_Kawasaki)

Follow him on twitter, and find this twitter and an article:

[11]<http://www.npr.org/sections/ed/2016/06/01/479335421/practice-makes-possible-what-we-learn-by-studying-amazing-kids>

### 6. **Eben Pagan**

[12]<https://www.facebook.com/Eben-Pagan-135028473246104/>

Package info to Ebooks to sell them.

Market them and sell them. Better to learn market and sales themselves.

Best sell method is to have a conversation with them, and try to fit in. Learn a new word today:

### **Consultative**

### **sales**

### 7.

### **Mohnish Pabrai**

[13][https://en.wikipedia.org/wiki/Mohnish\\_Pabrai](https://en.wikipedia.org/wiki/Mohnish_Pabrai)

### 8. **Grant Cardone**

[14][https://en.wikipedia.org/wiki/Grant\\_Cardone](https://en.wikipedia.org/wiki/Grant_Cardone)

Author of book: Sell To Survive

## 9. Peter Sage

[15][https://en.wikipedia.org/wiki/Grant\\_Cardone](https://en.wikipedia.org/wiki/Grant_Cardone)

## 10. Zig Ziglar

[16][https://en.wikipedia.org/wiki/Zig\\_Ziglar](https://en.wikipedia.org/wiki/Zig_Ziglar)

More to study:

**Keith Ferrazzi**

[17][https://en.wikipedia.org/wiki/Keith\\_Ferrazzi](https://en.wikipedia.org/wiki/Keith_Ferrazzi)

[18]<http://knowledge.wharton.upenn.edu/article/keith-ferrazzi-relationships-crucial-success/>

People relationship:

Invite people in, be generous to the people around me. Universal currency.

Athlete has more than 1 coaches.

Relationship coach -

Professional currency -

3rd layer - some one cares about

Read the script of interview - write down main points:

1. <https://www.youtube.com/watch?v=PwwgGOBw1oE>
2. [https://en.wikipedia.org/wiki/Jordan\\_Belfort](https://en.wikipedia.org/wiki/Jordan_Belfort)
3. [https://en.wikipedia.org/wiki/Tim\\_Ferriss](https://en.wikipedia.org/wiki/Tim_Ferriss)
4. [https://books.google.ca/books/about/On\\_Writing\\_Well.html?id=USd6AAAAIAAJ&redir\\_esc=y&hl=en](https://books.google.ca/books/about/On_Writing_Well.html?id=USd6AAAAIAAJ&redir_esc=y&hl=en)
5. [https://books.google.ca/books/about/Bird\\_by\\_Bird.html?id=LISjPwAACAAJ&redir\\_esc=y](https://books.google.ca/books/about/Bird_by_Bird.html?id=LISjPwAACAAJ&redir_esc=y)
6. [https://en.wikipedia.org/wiki/Gary\\_Vaynerchuk](https://en.wikipedia.org/wiki/Gary_Vaynerchuk)
7. <http://www.briantracy.com/>
8. <http://www.briantracy.com/blog/time-management/3-underrated-tips-to-achieve-work-life-balance/>
9. <http://www.briantracy.com/blog/category/sales-success/>
10. [https://en.wikipedia.org/wiki/Guy\\_Kawasaki](https://en.wikipedia.org/wiki/Guy_Kawasaki)
11. <http://www.npr.org/sections/ed/2016/06/01/479335421/practice-makes-possible-what-we-learn-by-studying-amazing-kids>
12. <https://www.facebook.com/Eben-Pagan-135028473246104/>
13. [https://en.wikipedia.org/wiki/Mohnish\\_Pabrai](https://en.wikipedia.org/wiki/Mohnish_Pabrai)
14. [https://en.wikipedia.org/wiki/Grant\\_Cardone](https://en.wikipedia.org/wiki/Grant_Cardone)
15. [https://en.wikipedia.org/wiki/Grant\\_Cardone](https://en.wikipedia.org/wiki/Grant_Cardone)
16. [https://en.wikipedia.org/wiki/Zig\\_Ziglar](https://en.wikipedia.org/wiki/Zig_Ziglar)
17. [https://en.wikipedia.org/wiki/Keith\\_Ferrazzi](https://en.wikipedia.org/wiki/Keith_Ferrazzi)
18. <http://knowledge.wharton.upenn.edu/article/keith-ferrazzi-relationships-crucial-success/>

## Algorithms to study (2016-06-11 23:07)

June 11, 2016

Prepare to write some code. Find algorithms to work on.

1. Leetcode 76: Minimum Window Substring

Study the code:

[1]<http://blog.csdn.net/sunnyyoona/article/details/43925035>

2. Leetcode 91: Decode the ways

[2][https://github.com/jianminchen/puzzles/blob/master/decode\\_ways.cc](https://github.com/jianminchen/puzzles/blob/master/decode_ways.cc)

2D. Leetcode 128: Longest Consecutive Sequence in Array

2E. Leetcode 139, 140, Word break

2F. Leetcode 150 Evaluate Reverse Polish Notation

3A. Leetcode 289: Game of Life

3B. Leetcode 298: Longest Consecutive Sequence in BT

4. LeetCode 351 - Android Unlock Patterns

Study the code:

[3]<http://massivealgorithms.blogspot.ca/2016/06/leetcode-351-android-unlock-patterns.html>

5. Read project Euler website:

[4]<https://projecteuler.net/archives>

6. Strobogrammatic number I, II, III (246, 247, 248)

Study code:

[5]<http://buttercola.blogspot.ca/2015/08/leetcode-strobogrammatic-number.html>

[6][https://tonycao.gitbooks.io/leetcode-locked/content/LeetCode %20Locked/c1.5.html](https://tonycao.gitbooks.io/leetcode-locked/content/LeetCode%20Locked/c1.5.html)

[7]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/246.strobogrammatic-number.java>

[8]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/247.strobogrammatic-number-ii.java>

[9]<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/248.strobogrammatic-number-iii.java>

7. One minute call 60 times at most

Leaky bucket algorithm

[10][https://en.wikipedia.org/wiki/Leaky\\_bucket](https://en.wikipedia.org/wiki/Leaky_bucket)

## 8. Stock Maximization

[11]<https://www.hackerrank.com/challenges/stockmax>

9.

[12]<http://www.careercup.com/question?id=5840928073842688>

11. Leetcode: shortest word distance I, II, III

[13]<https://segmentfault.com/a/1190000003906667>

study Leetcode solution:

[14]<https://github.com/MaskRay/LeetCode>

12. Leetcode 53: Maximum Subarray

[15][https://github.com/jianminchen/Leetcode\\_C-/blob/master/53MaximumSubArray1.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/53MaximumSubArray1.cs)

[16][https://github.com/jianminchen/Leetcode\\_C-/blob/master/53MaximumSubArray2.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/53MaximumSubArray2.cs)

[17]<http://juliachencoding.blogspot.ca/2015/07/dp-problem-kadanes-algorithm-maximum.html>

13. Leetcode 352 - Data Stream as Disjoint Intervals (Hard - 34 % passing rate)

[18]<http://www.cnblogs.com/grandyang/p/5548284.html>

14. Leetcode 174 Dungeon game

15. Leetcode 247 H-index

16. **Leetcode 269 Alien dictionary**

17. Leetcode 142 Linked List Cycle II

17B. Leetcode 286 Walls and Gates

18. Leetcode 329 Longest increasing path in matrix

19. Leetcode 345 Reverse Vowels of a string

20. Leetcode 354 Russian Doll Envelopes

Flatten Linked List

[19]<http://www.geeksforgeeks.org/flattening-a-linked-list/>

Blogs: Learn system design:

[20]<https://www.quora.com/How-should-I-prepare-for-my-Google-interview-if-I-have-1-month-left>

[21]<https://www.educative.io/collection/5642554087309312/5679846214598656>

1. <http://blog.csdn.net/sunnyyoona/article/details/43925035>
2. [https://github.com/jianminchen/puzzles/blob/master/decode\\_ways.cc](https://github.com/jianminchen/puzzles/blob/master/decode_ways.cc)
3. <http://massivealgorithms.blogspot.ca/2016/06/leetcode-351-android-unlock-patterns.html>
4. <https://projecteuler.net/archives>
5. <http://buttercola.blogspot.ca/2015/08/leetcode-strobogrammatic-number.html>
6. <https://tonycao.gitbooks.io/leetcode-locked/content/LeetCode%20Locked/c1.5.html>
7. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/246.strobogrammatic-number.java>
8. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/247.strobogrammatic-number-ii.java>
9. <https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/248.strobogrammatic-number-iii.java>
10. [https://en.wikipedia.org/wiki/Leaky\\_bucket](https://en.wikipedia.org/wiki/Leaky_bucket)
11. <https://www.hackerrank.com/challenges/stockmax>
12. <http://www.careercup.com/question?id=5840928073842688>
13. <https://segmentfault.com/a/1190000003906667>
14. <https://github.com/MaskRay/LeetCode>
15. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/53MaximumSubArray1.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/53MaximumSubArray1.cs)
16. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/53MaximumSubArray2.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/53MaximumSubArray2.cs)
17. <http://juliachencoding.blogspot.ca/2015/07/dp-problem-kadanes-algorithm-maximum.html>
18. <http://www.cnblogs.com/grandyang/p/5548284.html>
19. <http://www.geeksforgeeks.org/flattening-a-linked-list/>
20. <https://www.quora.com/How-should-I-prepare-for-my-Google-interview-if-I-have-1-month-left>
21. <https://www.educative.io/collection/5642554087309312/5679846214598656>
22. <https://zhuanlan.zhihu.com/p/19873823?refer=qinchao>
23. <http://poj.org/problem?id=3494>
24. [https://en.wikipedia.org/wiki/Flood\\_fill](https://en.wikipedia.org/wiki/Flood_fill)
25. <http://www.cnblogs.com/yrbbest/p/5023584.html>
26. <https://gist.github.com/jianminchen/07546625d828f63e762ba03b463fe8aa>
27. <https://gist.github.com/jianminchen/b984ccba8dabb0bb78160c6e5c6e8b4>

---

## HackerRank: Sherlock and anagram - warmup practice after 3 months (2016-06-12 13:31)

June 12, 2016

### First, software coding -> tennis sports -> gardening - a small talk

Julia likes blossom of flowers, she enjoyed but does not have time to do any gardening work. She spends a lot of hours on tennis court to keep her physical fit, cheer her up, play social games to build up very good team sports spirit through her workout.

She observes neighborhood gardeners watering the flower, grass every day in summer time, in the city of Vancouver, it takes a lot of hard work. She enjoys the neighborhood gardening.

So, to show some talent in software coding, she likes to learn from a patient gardener, watering, taking away weeds. In other words, read her own coding blog, learn to educate herself better through the short period of time.

Today, she likes to bring back her favorite practice 3 months ago, an algorithm - "Sherlock and anagram", and then review them, write more practice.

3 months is not long time, but those past 3 months, Julia found out that she is street-smart again. She starts to get more organized, more time-savvy, and know the importance to push herself to write her own words about experience. It does help.

One example, she stops to check in C # file using github, she just uses gist to quickly create a link. She did create almost 300 gists, each gist saves 5 minutes compared to check in files using github, she saved 1500 minutes, close to 25 hours time.

Another example, she tries to focus on reasoning and analysis, writing things to help her solve problem. One problem a time. So, she works on one problem, using more than 5 solutions - BFS, DFS, using Queue, using Stack, **[1]phone number problem** , therefore, she can apply the problem solving to all other similar problems.

Now, work on the coding:

1. HackerRank: Sherlock and anagram - 7 blogs with practices

**Practice on HackerRank is like to play tennis sports, you have to experience various hitting partners, good workout!**

No. 1.

[2]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-string-sherlock-and-anagrams.html>

No. 2

[3]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-ii.html>

No. 3

[4]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlocks-and-anagram-ii.html>

No. 4 **Study 6 solutions - Julia chose from over 200 solutions**

[5]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-iv.html>

**Julia, work on this code, and put test case in, write your own C # practice:**

**study the code, order by values, not by key, try it!**

[6]<https://gist.github.com/jianminchen/ffcca0582b5f0d1d6a9b>

study the blog: Dictionary OrderByDescending

[7]<https://goo.gl/6ZbTPz>

baby step to learn C # Dictionary class API - Order by and distinct

[8]<https://gist.github.com/jianminchen/eff03bea08a95061deb4185af74fea18>

No. 5

[9][http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-iv\\_27.html](http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-iv_27.html)

No. 6

[10]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagram-vii.html>

**Warm up 12 solutions - "Sherlock and anagram" one by one. Each one for a blog.** Focus on speed, correctness.

1. <http://juliachencoding.blogspot.ca/search/label/phone%20number>
  2. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-string-sherlock-and-anagrams.html>
  3. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-ii.html>
  4. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlocks-and-anagram-ii.html>
  5. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-iv.html>
  6. <https://gist.github.com/jianminchen/ffcca0582b5f0d1d6a9b>
  7. <https://goo.gl/6ZbTPz>
  8. <https://gist.github.com/jianminchen/eff03bea08a95061deb4185af74fea18>
  9. [http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-iv\\_27.html](http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagrams-iv_27.html)
  10. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-sherlock-and-anagram-vii.html>
- 

## **Array Class - C#, C++, JavaScript, Java (2016-06-12 20:20)**

June 12, 2016

Memorize all API of array definitely will help performance, help to communicate and fast coding. Just invest time to read, memorize, and practice. More reading leads great coding experience.

Ask questions about design, why they share the same, what is difference. So, like bible verse, you will come out the API just in second when you have a problem to solve.

A small research about good programmer vs good googler:

[1]<http://juliachencoding.blogspot.ca/2016/06/good-programmer-or-just-good-googler.html>

So, Julia starts to go over all API of Array class first:

**in C #: (once a week, spend 30 minutes to go over all examples, memorize them all!)**

[2][https://msdn.microsoft.com/en-us/library/system.array\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.array(v=vs.110).aspx)

It is an abstract class Array, implementing 6 interfaces:

IConeable,  
IList,  
ICollection,  
IEnumerable,



IStructuralComparable,  
IStructuralEquatable

Property:  
IsFixedSize  
IsReadOnly  
IsSynchronize  
Length  
LongLength  
Rank  
SyncRoot

C # array method:  
[3][https://msdn.microsoft.com/en-us/library/system.array\\_methods\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.array_methods(v=vs.110).aspx)

**40 methods: (June 16, 2016, Go over them one by one, mark favorite ones)**

AsReadOnly(T) (20 minutes June 19, 2016)

BinarySearch  
Clear  
Clone  
ConstrainedCopy  
ConvertAll  
Copy  
CopyTo  
CreateInstance  
Empty(T)

Exists(T)  
Find(T)  
FindAll(T)  
FindIndex  
FindLast(T)  
FindLastIndex  
ForEach(T)  
GetEnumerator  
GetLength  
GetLongLength

GetLowerBound  
GetUpperBound  
GetValue  
IndexOf  
Initialize  
LastIndexOf  
Resize(T)  
Reverse  
SetValue  
IList.Add

IList.Clear  
IList.Contains  
IList.IndexOf  
IList.Insert  
IList.Remove  
IList.RemoveAt  
IStructuralComparable.CompareTo  
IStructuralEquatable.Equals  
IStructuralEquatable.GetHashCode  
TrueForAll(T)

### **Have to work on Enumerable 50 methods first:**

System.Linq > Enumerable Class > Enumerable 50 Methods:

[4][https://msdn.microsoft.com/en-us/library/bb342261\(v=vs.100\).aspx](https://msdn.microsoft.com/en-us/library/bb342261(v=vs.100).aspx)

Aggregate  
All(TSource)  
Any  
AsEnumerable(TSource)  
Average  
Cast(TResult)  
Concat(TSource)  
Contains  
Count  
DefaultIfEmpty  
  
Distinct  
ElementAt(TSource)  
ElementAtOrDefault(TSource)  
Empty(TResult)  
Except  
First  
FirstOrDefault  
GroupBy  
GroupJoin  
Intersect  
  
Join  
Last  
LastOrDefault  
LongCount  
Max  
Min  
OfType(TResult)  
OrderBy  
OrderByDescending  
Range

Repeat(TResult)  
Reverse(TSource)  
Select  
SelectMany  
SequenceEqual  
Single  
SingleOrDefault  
Skip(TSource)  
SkipWhile  
Sum

Take(TSource)  
TakeWhile  
ThenBy  
ThenByDescending  
ToArray(TSource)  
ToDictionary  
ToList(TSource)  
ToLookup  
Union  
Where

### **JavaScript**

[5][http://www.w3schools.com/jsref/jsref\\_obj\\_array.asp](http://www.w3schools.com/jsref/jsref_obj_array.asp)

[6][https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Array/prototype](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/prototype) (June 14, 2016 - go over 2 hours)

Array Methods - 25 methods

concat  
copyWithin  
every  
fill  
filter  
findIndex  
forEach  
indexOf  
isArray  
join

lastIndexOf  
map  
pop  
push  
reduce  
reduceRight  
reverse  
shift  
slice

some

sort  
splice  
toString  
unshift  
valueOf

Study on June 13, 2016:

copyWithin - 3 arguments, target, start (required), end(required) (optional)

You should not use an array as associative arrays

[7]<http://andrewdupont.net/2006/05/18/javascript-associative-arrays-considered-harmful/>

[8][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Array/from](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/from)

Study:

compare to JavaScript Set object

Set

[9][https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Set](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Set)

Map

[10][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Array/map](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/map)

### JavaScript reference:

[11][https://msdn.microsoft.com/en-us/library/yek4tbz0\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/yek4tbz0(v=vs.94).aspx)

Array object:

[12][https://msdn.microsoft.com/en-us/library/k4h76zbx\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/k4h76zbx(v=vs.94).aspx)

JavaScript array study: **June 30, 2016**

PROPERTY:

3 properties:

constructor,  
length  
prototype

constructor property:

[https://msdn.microsoft.com/en-us/library/jj155291\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/jj155291(v=vs.94).aspx)

length:

array is sparse, so the array is not contiguous. The length is not necessarily the number of elements in the array.

[https://msdn.microsoft.com/en-us/library/d8ez24f2\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/d8ez24f2(v=vs.94).aspx)

Prototype:

[https://msdn.microsoft.com/en-us/library/jj155285\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/jj155285(v=vs.94).aspx)

JavaScript array 29 methods:

Spend 10 minutes a time to memorize all the function names, and then, try to guess each api's task, what are the arguments, how it is designed.

write down:

Array.from - copy array from, input argument is array.

isArray - Array.isArray(arr) input argument arr is array

of - ? wild guess -

concat - arr1.concat(arr2), concatenate the string

entries - entries - arguments: startIndex, endIndex, return subarray?

every - iterator - go through each node in the array to check some logic?

fill - fill - arr.fill(1), all the elements in the array are assigned to the same value

filter

findIndex

foreach

indexOf

join

keys

lastIndexOf

map

pop

push

reduce

reduceRight

reverse

shift

slice

some

sort

splice

toString

unshift

valueOf

values

challenges:

Arguments:

callback function -

callback function syntax

3 things Julia likes the JavaScript Array.fill function design:

- start, end arguments are options
- negative start, end arguments handling
- 

- end of June 30, 2016 study -

- End of JavaScript -

— Java —

**Java Array reference:**

[13]<https://docs.oracle.com/javase/7/docs/api/java/lang/reflect/Array.html>

Java Array

Method inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

Method detail:

Array.newInstance

getLength

get - Array.get(array, index)

getBoolean - Array.getBoolean(array, index)

getByte - Array.getByte(array, index)

getChar - Array.getChar(array, index)

getShort - Array.getShort(array, index)

getInt

getLong

getFloat

getDouble

set - Array.set(array, index, Object value)

setBoolean

setByte

setChar - Array (Object array, int index, char c)

setShort

setInt

setLong

setFloat

setDouble

## Java

### Arrays

<https://docs.oracle.com/javase/7/docs/api/java/util/Arrays.html>

asList

binarySearch

copyOf

copyOfRange

deepEquals

deepHashCode

deepToString

equals

fill

hashCode

sort

toString

Methods inherited from class java.lang.Object

clone, equals, finalize, hashCode, notify, notifyAll, toString, wait.

blogs to read:

asList

[14]<http://stackoverflow.com/questions/20538869/what-is-the-best-way-of-using-arrays-aslist-to-initialize-a-list>

copyOfRange:

[15][http://www.tutorialspoint.com/java/util/arrays\\_copyofrange\\_short.htm](http://www.tutorialspoint.com/java/util/arrays_copyofrange_short.htm)

[16]<http://stackoverflow.com/questions/11001720/get-only-part-of-an-array-in-java>

more than 2 ways:

1. copyOfRange

2. Arrays.asList(array).subList(index, array.Length)

[17]<http://stackoverflow.com/questions/19389609/array-vs-arraylist-in-performance>

[18]<http://stackoverflow.com/questions/716597/array-or-list-in-java-which-is-faster>

notes: List vs ArrayList List is interface, whereas ArrayList is a concrete class to create object.

[19]<https://docs.oracle.com/javase/tutorial/java/nutsandbolts/arrays.html>

1. <http://juliachencoding.blogspot.ca/2016/06/good-programmer-or-just-good-googler.html>
2. [https://msdn.microsoft.com/en-us/library/system.array\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.array(v=vs.110).aspx)
3. [https://msdn.microsoft.com/en-us/library/system.array\\_methods\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.array_methods(v=vs.110).aspx)
4. [https://msdn.microsoft.com/en-us/library/bb342261\(v=vs.100\).aspx](https://msdn.microsoft.com/en-us/library/bb342261(v=vs.100).aspx)
5. [http://www.w3schools.com/jsref/jsref\\_obj\\_array.asp](http://www.w3schools.com/jsref/jsref_obj_array.asp)
6. [https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Array/prototype](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/prototype)
7. <http://andrewdupont.net/2006/05/18/javascript-associative-arrays-considered-harmful/>
8. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Array/from](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/from)
9. [https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Set](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Set)
10. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Array/map](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/map)
11. [https://msdn.microsoft.com/en-us/library/yek4tbz0\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/yek4tbz0(v=vs.94).aspx)
12. [https://msdn.microsoft.com/en-us/library/k4h76zbx\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/k4h76zbx(v=vs.94).aspx)
13. <https://docs.oracle.com/javase/7/docs/api/java/lang/reflect/Array.html>
14. <http://stackoverflow.com/questions/20538869/what-is-the-best-way-of-using-arrays-as-list-to-initialize-a-list>
15. [http://www.tutorialspoint.com/java/util/arrays\\_copyofrange\\_short.htm](http://www.tutorialspoint.com/java/util/arrays_copyofrange_short.htm)
16. <http://stackoverflow.com/questions/11001720/get-only-part-of-an-array-in-java>
17. <http://stackoverflow.com/questions/19389609/array-vs-arraylist-in-performance>
18. <http://stackoverflow.com/questions/716597/array-or-list-in-java-which-is-faster>
19. <https://docs.oracle.com/javase/tutorial/java/nutsandbolts/arrays.html>

---

## Hashtable - C++, C#, Java, JavaScript (2016-06-12 20:23)

June 12, 2016

Start to read all Hashtable class API, read code and start to memorize all of them. Invest time on them, 30 minutes a time.

### C #:

Dictionary class

[1][https://msdn.microsoft.com/en-us/library/xfhwa508\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/xfhwa508(v=vs.110).aspx)

Interface implemented:

IDictionary,

ICollection,

IEnumerable,

IReadOnlyDictionary

ISerializable



IDeserializationCallback

**Properties:**

Comparer,  
Count,  
Item[TKey]  
Keys  
Values

**Methods: 15 methods**

Add(TKey, TValue)  
Clear()  
ContainsKey(TKey)  
ContainsValue(TValue)  
Equals(Object)

Finalize()  
GetEnumerator()  
GetHashCode()

**Extension Methods:**

[2]<https://msdn.microsoft.com/en-CA/library/bb383977.aspx>

C++:

Java:

JavaScript:

**Statistics:**

1. June 12, 2016 30 minutes

1. [https://msdn.microsoft.com/en-us/library/xfhwa508\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/xfhwa508(v=vs.110).aspx)

2. <https://msdn.microsoft.com/en-CA/library/bb383977.aspx>

---

**Queue Class - C++, C#, Java, JavaScript (2016-06-12 20:55)**

June 12, 2016

Start to spend time to go over all API for those 4 languages.

1. C++:

2. C #:

3. Java:

1. PriorityQueue

[1]<https://docs.oracle.com/javase/7/docs/api/java/util/PriorityQueue.html>

2. Queue

[2]<https://docs.oracle.com/javase/7/docs/api/java/util/Queue.html>

4. JavaScript

Will come back.

1. <https://docs.oracle.com/javase/7/docs/api/java/util/PriorityQueue.html>

2. <https://docs.oracle.com/javase/7/docs/api/java/util/Queue.html>

---

## Stack Class - C++, C#, Java, JavaScript (2016-06-12 20:55)

June 12, 2016

Start to spend time to go over all API for those 4 languages.

Will come back.

---

## Double Linked List - C++, C#, Java, JavaScript (2016-06-12 21:10)

June 12, 2016

Motivation behind memorization APIs as a programmer:

### Good programmer or just good Googler

20 minutes research about good programmer or just good Googler?

[1]<http://www.hanselman.com/blog/AmIReallyADeveloperOrJustAGoodGoogler.aspx>

[2]<http://getinvolved.hanselman.com/>

[3]<http://www.hanselman.com/blog/ImAPhonyAreYou.aspx>

[4]<http://codekata.pragprog.com/>

[5]<https://www.quora.com/Do-developers-memorize-all-tags-classes-and-functions>

[6]<http://app.pluralsight.com/courses/hanselman-speaking>

**Memorization brings up a lot of learning process**

1. Where to find documents?
2. How to design, understandable?
3. Start to work on simple examples.
4. To get prepared early. It makes difference.

Work on Double Linked List class/ interface - go over all APIs of 4 languages -> Memorize -> Compare, ask questions -> start to prepare to use them in the future projects.

#### **C #:**

LinkedList class

[7][https://msdn.microsoft.com/en-us/library/he2s3bh7\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/he2s3bh7(v=vs.110).aspx)

Properties:

Count,  
First,  
Last

25 Methods:

AddAfter(LinkedListNode<T>, T)  
AddAfter(LinkedListNode<T>, ListListNode<T>)

AddBefore(LinkedListNode<T>, T)  
AddBefore(LinkedListNode<T>, ListListNode<T>)

AddFirst(LinkedListNode<T>, T)  
AddFirst(LinkedListNode<T>, ListListNode<T>)

AddLast(LinkedListNode<T>, T)  
AddLast(LinkedListNode<T>, ListListNode<T>)

Clear()  
Contains(T)  
CopyTo(T[], int32)

Equals(Object)

Finalize()  
Find(T) - Find the first node that contains the specified value  
FindLast(T) - Find the last node that contains the specified value

GetEnumerator()  
GetHashCode()  
GetObjectData()  
GetType()

MemberwiseClone()  
OnDeserialization(Object)

Remove(T)  
Remove(LinkedListNode<T>)

RemoveFirst()  
RemoveLast()

LinkedListNode:  
[8][https://msdn.microsoft.com/en-us/library/ahf4c754\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ahf4c754(v=vs.110).aspx)

## Statistics:

June 12, 2016 10 minutes

More reading:  
[9]<http://matt-welsh.blogspot.ca/2014/01/getting-job-at-google-for-phd-students.html>

[10]<http://matt-welsh.blogspot.ca/2010/11/why-im-leaving-harvard.html>

[11]<http://matt-welsh.blogspot.ca/2010/05/secret-lives-of-professors.html>

1. <http://www.hanselman.com/blog/AmIReallyADeveloperOrJustAGoodGoogler.aspx>
2. <http://getinvolved.hanselman.com/>
3. <http://www.hanselman.com/blog/ImAPhonyAreYou.aspx>
4. <http://codekata.pragprog.com/>
5. <https://www.quora.com/Do-developers-memorize-all-tags-classes-and-functions>
6. <http://app.pluralsight.com/courses/hanselman-speaking>
7. [https://msdn.microsoft.com/en-us/library/he2s3bh7\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/he2s3bh7(v=vs.110).aspx)
8. [https://msdn.microsoft.com/en-us/library/ahf4c754\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ahf4c754(v=vs.110).aspx)
9. <http://matt-welsh.blogspot.ca/2014/01/getting-job-at-google-for-phd-students.html>
10. <http://matt-welsh.blogspot.ca/2010/11/why-im-leaving-harvard.html>
11. <http://matt-welsh.blogspot.ca/2010/05/secret-lives-of-professors.html>

---

## Good programmer or just good Googler - a small research (2016-06-14 20:23)

June 14, 2016

Julia knows that it makes difference if she knows APIs very well. It helps her to communicate and also problem solving on algorithm problems. So, she starts to try idea - use memorization to help coding. Memorize Array/ Double Linked List/ Stack/ Queue/ String/ Hashtable class in C #, C++, Java, JavaScript, and then, prepare more coding in the future. She likes to read more C++/ Java/ Java Script code.

Get organized. Get more prepared by more reading as a software programmer. Practice Leetcode algorithms/ practice on HackerRank, also, reading APIs, and then memorize APIs like bible verse memorization, is a good practice.

### **Good programmer or just good Googler**

20 minutes research about good programmer or just good Googler?

[1]<http://www.hanselman.com/blog/AmIReallyADeveloperOrJustAGoodGoogler.aspx>

[2]<http://getinvolved.hanselman.com/>

[3]<http://www.hanselman.com/blog/ImAPhonyAreYou.aspx>

[4]<http://codekata.pragprog.com/>

[5]<https://www.quora.com/Do-developers-memorize-all-tags-classes-and-functions>

[6]<http://app.pluralsight.com/courses/hanselman-speaking>

### **Memorization brings up a lot of learning process**

1. Where to find documents?
2. Learn the design, improve reading, learn something new, or the API document easy to understand?
3. Start to work on simple examples.
4. Know the difference to get prepared early.

Know how to make difference to get prepared early.

5. Preparation takes time to learn.

1. <http://www.hanselman.com/blog/AmIReallyADeveloperOrJustAGoodGoogler.aspx>

2. <http://getinvolved.hanselman.com/>

3. <http://www.hanselman.com/blog/ImAPhonyAreYou.aspx>

4. <http://codekata.pragprog.com/>

5. <https://www.quora.com/Do-developers-memorize-all-tags-classes-and-functions>

6. <http://app.pluralsight.com/courses/hanselman-speaking>

---

### **Industry work vs academic research - blogs reading time (2016-06-14 20:25)**

June 14, 2016

Come cross the article about an engineer working for Google, a computer professor made a career change.

Have some readings:

1. [1]<http://matt-welsh.blogspot.ca/2010/11/why-im-leaving-harvard.html>
2. [2]<http://matt-welsh.blogspot.ca/2010/05/secret-lives-of-professors.html>
3. [3]<http://matt-welsh.blogspot.ca/2014/01/getting-job-at-google-for-phd-students.html>
4. Find a good topic to read:  
[4]<http://matt-welsh.blogspot.ca/2013/02/grad-students-learn-how-to-give-talk.html>

**Julia's comment about how to give a talk:**

Know your topic very well. I still remember the advice I got many years ago, "Do not waste other people's time - even it is 5 minutes. If you do not understand the topic, then stop talking!", "Reading related articles more than 10 hours, not working on a concrete example, coding/ experiment/ third party, it does not count!"

For example, Julia worked on binary tree least common ancestor more than 10 hours in less than 1 week, she wrote code more than 5 times, two solutions; definitely, she can give a talk over 15 minutes about the algorithm any time without any preparation, present the problem, draw some diagram, write some code, and then go over it, line by line, each variable, each executable path, discuss all kinds of bugs if there is a mistake. She knows 4-5 possible mistakes, **because she made one by one, and then, spent more than half hour to fix one by one**. It will be a fun talk.

[5]<http://juliachencoding.blogspot.ca/2016/04/find-lowest-common-ancestor-of-two.html>

**5. Code Review**

[6]<http://matt-welsh.blogspot.ca/2012/02/my-love-affair-with-code-reviews.html>

**Julia's comment:**

**Julia's favorite books:** [7]The art of readable code, [8]clean code, and then, [9]C++ code guidelines.

**And also favorite object oriented principles:** S.O.L.I.D., [10]good design principle - simple vs difficult.

**Choose "Old school" or "new school", prepare a guideline for code review. To write readable code/ clean code, practice the code review based on guidelines.**

**Also, really work on coding skills, read a lot of code. For example, HackerRank, Two string, hundreds of solutions - compared the difference, find the best one. And be the best one in your coding practice.**

HackerRank: Two string - thinking in JavaScript over 10 ways

[11]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-two-string-thinking-in.html>

Thinking in C++ over 15 ways

[12]<http://juliachencoding.blogspot.ca/2016/03/hackerrank-two-string-thinking-in-c.html>

**6.**

[13]<http://matt-welsh.blogspot.ca/2010/06/working-for-google.html>

7.

[14]<http://matt-welsh.blogspot.ca/2016/06/death-by-peer-review.html>

Very good analysis - peer review - anonymous/ ad hoc vs principled leadership organization

Which is more controllable? Work on controllable things - NO Matter how hard it is, you work hard, you will make it one day.

1. <http://matt-welsh.blogspot.ca/2010/11/why-im-leaving-harvard.html>
  2. <http://matt-welsh.blogspot.ca/2010/05/secret-lives-of-professors.html>
  3. <http://matt-welsh.blogspot.ca/2014/01/getting-job-at-google-for-phd-students.html>
  4. <http://matt-welsh.blogspot.ca/2013/02/grad-students-learn-how-to-give-talk.html>
  5. <http://juliachencoding.blogspot.ca/2016/04/find-lowest-common-ancestor-of-two.html>
  6. <http://matt-welsh.blogspot.ca/2012/02/my-love-affair-with-code-reviews.html>
  7. <http://juliachencoding.blogspot.ca/2016/04/the-art-of-readable-code-book-review.html>
  8. <http://juliachencoding.blogspot.ca/2016/04/clean-code-book-reading.html>
  9. <http://juliachencoding.blogspot.ca/2015/12/cpp-core-guidelines.html>
  10. <http://juliachencoding.blogspot.ca/search/label/design%20principle>
  11. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-two-string-thinking-in.html>
  12. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-two-string-thinking-in-c.html>
  13. <http://matt-welsh.blogspot.ca/2010/06/working-for-google.html>
  14. <http://matt-welsh.blogspot.ca/2016/06/death-by-peer-review.html>
- 

## **The Art of Speaking: Scott Hanselman (2016-06-14 20:30)**

June 14, 2016

Julia works on the most important skills - how to prepare a tech talk? She needs to work on a few of things: talk concise and stop; think about careful and make a short statement; focus on the main point.

She spends 60 minutes to learn something through talk on Pluralsight.

The Art of Speaking: Scott Hanselman

### **What do you learn through the talk?**

Coffee Script is much more readable compared to JavaScript, and also easy to get started with a few videos - quick study.

The design of talk is interesting, Julia, learn 3 things:

1. The example with 11 lines of code, written in coffee script, comparison to Java Script is very easy to follow;

2. The speaker knows how to give a very smooth talk - target wide range of users through the video watching.
3. The Delivery 14m8s - somewhere in first 5 minutes - Julia likes the talk about a loop, using business sense, explain what is going on, how easy things can go wrong: makes a mix of vegetables and chocolate, and then, go through them one by one. etc.

To entertain the study, provide a video link about Microsoft azure network/ storage. Spent 10 minutes to watch the video:

[1][https://www.youtube.com/watch?v=ZNgvZE0MLeo&list=PLHqk7IOX-\\_BYHuz0ZJt4HO8RXkh5nUjHG&index=2](https://www.youtube.com/watch?v=ZNgvZE0MLeo&list=PLHqk7IOX-_BYHuz0ZJt4HO8RXkh5nUjHG&index=2)

1. [https://www.youtube.com/watch?v=ZNgvZE0MLeo&list=PLHqk7IOX-\\_BYHuz0ZJt4HO8RXkh5nUjHG&index=2](https://www.youtube.com/watch?v=ZNgvZE0MLeo&list=PLHqk7IOX-_BYHuz0ZJt4HO8RXkh5nUjHG&index=2)

---

## **Leetcode 269: Alien Dictionary (2016-06-15 22:42)**

**June 15, 2016**

### **Leetcode 269 Alien dictionary**

Choose to work on a graph problem - using Topological Sorting:

Study C++ solution:

<http://www.cnblogs.com/jcliBlogger/p/4758761.html>

Discussion about the problem description:

<https://leetcode.com/discuss/53997/the-description-is-wrong>

Study C++ solution:

[#q54024](https://leetcode.com/discuss/54024/straightforward-c-solution?show=54024)

Java solution to study:

<https://github.com/jianminchen/LeetCode-Java-Solutions/blob/master/269.alien-dictionary.java>

[1]<http://www.cnblogs.com/yrbbest/p/5023584.html>



Julia worked on C # code:

[2]<https://gist.github.com/jianminchen/07546625d828f63e762ba03b463fe8aa>

line 75, 76, after queue.peek() is called, need to call dequeue. (dead loop)

[3] <https://gist.github.com/jianminchen/a49496ea21cadcbdda7c1669216c05a5>

Add more comment, where to be careful, to avoid bugs:

[4]<https://gist.github.com/jianminchen/47f516b54686080c3a68bc8c3f1d04cb>

More comment, variable name refactor:

[5]<https://gist.github.com/jianminchen/85129ed50ce597f896b0f0c5a2fa5586>

Read blogs:

[6]<http://www.geeksforgeeks.org/topological-sorting-indegree-based-solution/>

Comparison between 2 versions:

variable name change: graph -> dependencyList, more meaningful. The graph has nodes, dependency list, inDegree array.

```
// Source code from blog:
// http://www.geeksforgeeks.org/topological-sorting-indegree-based-solution/
// Topological Sorting - Kahn's Algorithm

public static string topoOrder(string[] words) {
    if(words == null || words.Length == 0) {
        return "";
    }

    Dictionary<char, HashSet<char>> graph = new Dictionary<char, HashSet<char>>();
    int[] inDegree = new int[words.Length];

    graphSetup(words, graph, inDegree);

    return topologicalSort(words, graph, inDegree);
}

// Graph presentation:
// nodes, nodes's dependency list and inDegree array
// nodes - Function getNodes()

Dictionary<char, HashSet<char>> dependencyList = new Dictionary<char, HashSet<char>>();
int[] inDegree = new int[words.Length];

graphSetup(words, dependencyList, inDegree);

return topologicalSort(words, dependencyList, inDegree);
}
```

Change function name: getCharSet -> getNodes

```
// First time to use HashSet UnionWith api - good practice!

public static HashSet<char> getCharSet(string[] words) {
    HashSet<char> set = new HashSet<char>();

    foreach (string word in words) {
        set.UnionWith(word.ToCharArray());
    }

    return set;
}

public static HashSet<char> getNodes(string[] words) {
    HashSet<char> hashSet = new HashSet<char>();

    foreach (string s in words) {
        hashSet.UnionWith(s.ToCharArray());
    }

    return hashSet;
}
```

Love those comment, so helpful to get start to coding...



Left side first implementation:

```
for(int j=0; j < shortLength; j++)
```

```
{
```

```
...
```

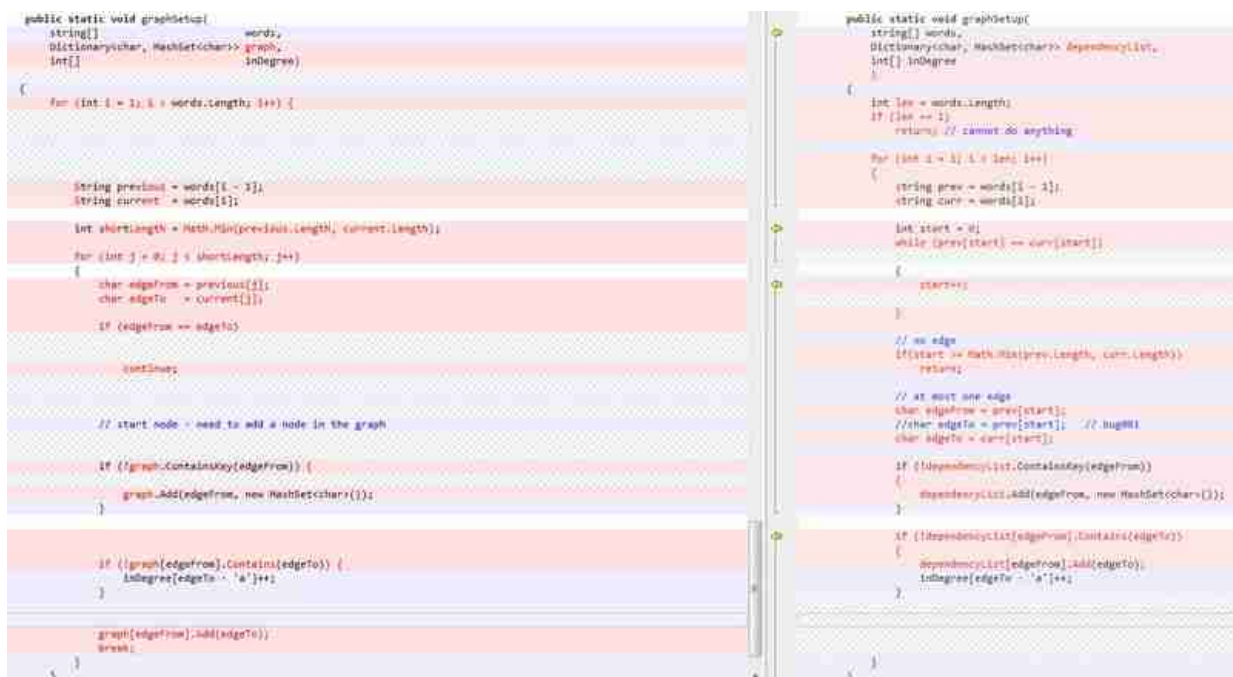
```
break;
```

```
}
```

confusing, not very easy to follow. line 154 - line 176, 22 lines of code. The big scope to handle.

Also, this for loop is nested loop, too many lines of code inside.

Replacement of while loop is short, only 3 lines of code.



graphSetup -> what we can tell here? ... later!

```
public static void graphSetup(
    string[] words,
    Dictionary<char, HashSet<char>> graph,
    int[] inDegree)
{
    for (int i = 1; i < words.Length; i++) {
        string previous = words[i - 1];
        string current = words[i];

        int shortestLength = Math.Min(previous.Length, current.Length);
        for (int j = 0; j < shortestLength; j++) {
            char edgeFrom = previous[j];
            char edgeTo = current[j];

            if (edgeFrom == edgeTo)
                continue;

            // start node - need to add a node in the graph
            if (!graph.ContainsKey(edgeFrom)) {
                graph.Add(edgeFrom, new HashSet<char>());
            }

            // Avoid bugs
            // Do not add same node twice in inDegree array
            // For example, art-nrf -> t-rf, 't' is inDegree from 'a', should not count twice.
            // filter out duplicated relationship
            // ["ta","ab","ca","ab"], then, a-b will show up twice
            // Try to describe what code is doing here:
            // If adjacency list does not contain edgeTo, then, it is the
            // first time visited, then, increment one to inDegree array for
            // the char
        }
    }
}

public static void graphSetup(
    string[] words,
    Dictionary<char, HashSet<char>> dependencyList,
    int[] inDegree)
{
    int len = words.Length;
    if (len == 1)
        return; // cannot do anything

    for (int i = 1; i < len; i++) {
        string prev = words[i - 1];
        string curr = words[i];

        int start = 0;
        while (prev[start] == curr[start])
            start++;

        // no edge
        if (start == Math.Min(prev.Length, curr.Length))
            return;

        // at next one edge
        char edgeFrom = prev[start];
        //char edgeTo = prev[start]; // bug!!!
        char edgeTo = curr[start];

        if (!dependencyList.ContainsKey(edgeFrom)) {
            dependencyList.Add(edgeFrom, new HashSet<char>());
        }

        if (!dependencyList[edgeFrom].Contains(edgeTo))
    }
}
```

### Question and answer:

1. Can you work on a simple example to explain the idea of your solution?

Here is the warm up for topological sorting using two strings {"wrt","wrf"}:

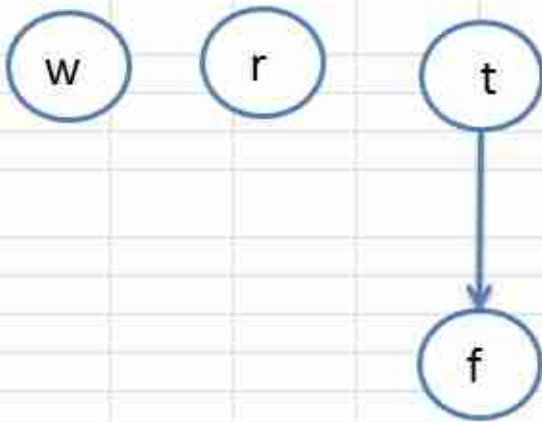
"wrt", "wrf"

Nodes in graph: 'w', 'r', 't', 'f'

Node 'f' -> dependency list - a HashSet {'t'}

inDegree['t'->'a'] = 0

inDegree['f'->'a'] = 1



Topological Sort - output: wrtf, or rwtf, trwf or twrf etc.

Review previous blog:

[7]<http://juliachencoding.blogspot.ca/2016/04/leetcode-208-course-schedule.html>

#### Warmup practice:

statistics: 1 bug, more than 60 minute to write. Totally new program

[8]<https://gist.github.com/jianminchen/58d80aa86a027af7a3e52277d15c3733>

a few changes to highlight:

1. line 24 - 26 add comment what to do about graph
2. line 49 - 65 motivation talk - help to design the function using the graph above {"wrt", "wrf"}, help to write code
3. line 72 - 74 special case words length is 1
4. line 81 - line 85 use one pointer to slide forward <- more flat code
5. line 87 add comment - no edge -> very good comment
6. line 91 - 94 first writing with a bug - prev, curr, but prev twice

Here is the comparison file:

[9][https://github.com/jianminchen/Leetcode\\_C/blob/master/Leetcode269FirstAndSecondPractice.pdf](https://github.com/jianminchen/Leetcode_C/blob/master/Leetcode269FirstAndSecondPractice.pdf)

#### Statistics:

time spent: 3hours +

I don't feel fear when I am on court. That's where I feel at home [10] #CreateYourMark [11] #RG16 [12]@adidastennis [13]@adidasFR [14]pic.twitter.com/ZVNcZvWZE0

— Kristina Mladenovic (@KikiMladenovic) [15]May 20, 2016

1. <http://www.cnblogs.com/yrbbest/p/5023584.html>
2. <https://gist.github.com/jianminchen/07546625d828f63e762ba03b463fe8aa>
3. <https://gist.github.com/jianminchen/a49496ea21cadcbdda7c1669216c05a5>
4. <https://gist.github.com/jianminchen/47f516b54686080c3a68bc8c3f1d04cb>
5. <https://gist.github.com/jianminchen/85129ed50ce597f896b0f0c5a2fa5586>
6. <http://www.geeksforgeeks.org/topological-sorting-indegree-based-solution/>
7. <http://juliachencoding.blogspot.ca/2016/04/leetcode-208-course-schedule.html>
8. <https://gist.github.com/jianminchen/58d80aa86a027af7a3e52277d15c3733>
9. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/Leetcode269FirstAndSecondPractice.pdf](https://github.com/jianminchen/Leetcode_C-/blob/master/Leetcode269FirstAndSecondPractice.pdf)
10. <https://twitter.com/hashtag/CreateYourMark?src=hash>
11. <https://twitter.com/hashtag/RG16?src=hash>
12. <https://twitter.com/adidastennis>
13. <https://twitter.com/adidasFR>
14. <https://t.co/ZVNcZvWZE0>
15. <https://twitter.com/KikiMladenovic/status/733569899964989441>

---

## LINQ - Language-Integrated Query (2016-06-15 22:50)

June 15, 2016

Spend 20 minutes to work on C # Dictionary class extension method: All, Any method

[1][https://msdn.microsoft.com/en-us/library/bb548541\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/bb548541(v=vs.110).aspx)

So, look into how many people are using LINQ:

equinox - C # .Net Application Developer,

Knowledge of, or a desire to work with:

- AngularJS, jquery, json
- Soap/XML/XSLT/XSD/WSDL
- LINQ.

Canfor:

C #, VB.net, LINQ, JavaScript , HTML, XML, SQL, VBA, CSS

So, study more about LINQ.

It is interesting journey, fast coding -> plan to memorize C # Dictionary API -> work on extension method All, Any -> start to learn LINQ -> C # .Net Application development

### Actionable Items:

Do some research on LINQ - good study material - get some coding experience.

Put LINQ, Lambda expression code practice here - 10 - 20 of them first:

1. Practice debug - watch - lambda expression

[2]<https://gist.github.com/jianminchen/79608e80e1915ecdb5df118a54001086>

2. review two string solution - use Any

[3]<https://gist.github.com/jianminchen/acedbb7cb86cf1c00131>

3. Review two string solution - use Any

[4]<https://gist.github.com/jianminchen/50fc6b5a13b7d62dfa1d>

4. Take two courses on pluralsight.com (June 17, 2016):

LINQ Architecture - 2 hours

1. [https://msdn.microsoft.com/en-us/library/bb548541\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/bb548541(v=vs.110).aspx)

2. <https://gist.github.com/jianminchen/79608e80e1915ecdb5df118a54001086>

3. <https://gist.github.com/jianminchen/acedbb7cb86cf1c00131>

4. <https://gist.github.com/jianminchen/50fc6b5a13b7d62dfa1d>

---

### Leetcode 269 - Alien Dictionary - Practice 2 more times (2016-06-16 21:27)

June 16, 2016

Try to find a small topic to start to do some research every day at least 20 minutes, and then build up more later on.

Today topic is about writing algorithm code - problem solving, **1st writing (study other's code and understand) vs 1st writing (with simple test case with a diagram, more focus, a small problem) :**

Here is the cycle Julia goes through for Leetcode 269 - Alien Dictionary:

Coding process ->

choose an algorithm to code ->

study 4 or 5 solution from over 10 solutions ->

cannot learn an algorithm just by reading, stop reading ->  
write C # code based on one of them (Java, or C++) ->  
add comment, make code readable, debugging ->  
code works ->  
wait 10 - 20 minutes ->

**draw a diagram to work on a simple test case**

->  
rewrite the code 2nd time->  
**big difference, a new story to write - it takes close to one hour ->**

**interesting experience. ->**

**3rd rewrite ->**

**document the difference**

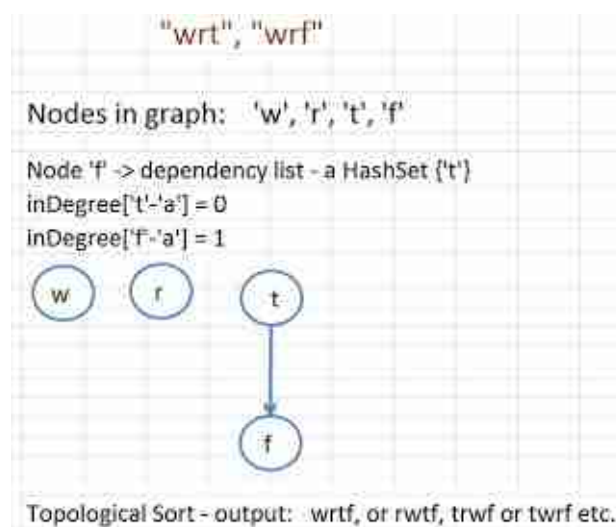
First blog of Leetcode 269 Alien Dictionary:

[1]<http://juliachencoding.blogspot.ca/2016/06/leetcode-269-alien-dictionary.html>

1st good writing in C #:

[2]<https://gist.github.com/jianminchen/85129ed50ce597f896b0f0c5a2fa5586>

Afterwards, work out a simple test case, and then, write down the graph with detail data structure and data, write code based on the picture. New ideas come out naturally to improve:



1st writing focusing on the above diagram: C # implementation

[3]<https://gist.github.com/jianminchen/58d80aa86a027af7a3e52277d15c373>

3rd writing using C # code:

[4]<https://gist.github.com/jianminchen/8d4c1f601bae0ca7ef27e470dfe1e636>

Highlight the differences 3rd time:

1. line 26, base case is updated: words.Length <= 1 instead of <1
2. Add comments what to find in the function alienOrder
  1. spell error topological -> topological
  2. add nodes variable as part of graph
3. getNodes function -  
look into string.ToList() -> List<char>  
study HashSet.UnionWith() input argument and its type IEnumerable<T>
4. rewrite graphSetup function tasks from line 62 - 79  
explain very clearly to find an edge if there is one;  
what to construct in the graph.  
add precondition for the function.
5. rewrite the function comment for topologicalSort
6. missing line 163 - 164, run time error - array access - out-of-index

### statistics:

Time spent: 3rd practice - more than 40 minutes

### Quick tip:

```
{"wrt", "wrf", "er", "ett", "rftt" }
```

first char, comparison (all five words) => w -> e -> r

third char comparison (first two words - "wrt", "wrf") => t->f

second char comparison (third, fourth words - "er", "ett") => r-> t

So, the order is "wertf"

Comparison:

alienOrder - two things noticed documented in comment.

```
public static string alienOrder(string[] words)
{
    if (words == null || words.Length == 0)
        return "";

    // Graph presentation:
    // nodes, node's dependency list and inDegree array
    // nodes - function getNodes()
    Dictionary<char, HashSet<char>> dependencyList = new Dictionary<char, HashSet<char>>();
    List<int> inDegree = new List<int>();

    graphSetup(words, dependencyList, inDegree);

    return topologicalSort(words, dependencyList, inDegree);
}

/*
 * Things found in 3rd writing:
 * 1. new spell error topologicalSort function name - topologicalSort not topologicalSort
 * 2. add nodes variable as part of graph - explicitly add the variable in the graph
 */
public static string alienOrder(string[] words)
{
    if (words == null || words.Length <= 1)
        return "";

    // Graph presentation - 3 variables at least
    // nodes, node's dependency list and inDegree array
    // nodes - function getNodes()
    HashSet<char> nodes = getNodes(words);
    Dictionary<char, HashSet<char>> dependencyList = new Dictionary<char, HashSet<char>>();

    List<int> inDegree = new List<int>();

    graphSetup(words, dependencyList, inDegree);

    return topologicalSort(words, dependencyList, inDegree, nodes);
}
```

3rd writing - take some time to look up HashSet.UnionWith argument type: IEnumerable<T>, good time to learn something here.



```

public static HashSet<char> getNodes(string[] words)
{
    HashSet<char> hashset = new HashSet<char>();
    foreach (string s in words)
    {
        hashset.UnionWith(s.ToList());
    }
    return hashset;
}

```

```

/*
 * 3rd writing:
 * look into string.ToList() -> string -> List<char>
 * try to use HashSet.UnionWith(IEnumerable<T>)
 *
 * because List<char> should be IEnumerable<T>, T is char here.
 */
public static HashSet<char> getNodes(string[] words)
{
    HashSet<char> hashset = new HashSet<char>();
    foreach (string s in words)
    {
        hashset.UnionWith(s.ToList()); // List<char>
    }
    return hashset;
}

```

Practice to write down what to do, in order to write good code, also need to explain what to do first. Good writing right side! Bravo!

```

/*
 * Motivation talk:
 * set up graph for ["wt", "wt"]
 * output:
 * 't' -> add 't' into dependency list's dictionary, also, update context {'t'}
 * indegree array, setup for indegree['t'] = 2
 */

/* two words, at most one edge
 * two words, no edge - special case discussion!
 * test case:
 * case 1: "a", "ac"
 * case 2:
 *
 * That is it!
 */
public static void graphSetup(
    string[] words,
    Dictionary<char, HashSet<char>> dependencyList,
    int[] indegree
)
{
    int len = words.Length;
    if (len == 1)
        return; // cannot do anything

    for (int i = 1; i < len; i++)
    {
        string prev = words[i - 1];
        string curr = words[i];
    }
}

```

```

/*
 * Explain the function graphSetup design using the most simple example:
 * set up graph for ["wt", "wt"]
 */

/*
 * here are the task to complete:
 * 1. Find the first different char - actually, it is an edge in the graph,
 *    here is t -> t
 * 2. Need to handle case - no edge
 * 3. At most one edge for two consecutive words
 * 4. Only handle the array of string by two neighboring nodes - transitive
 */

/* what to construct in the graph:
 * 1. Add 't' to dependencyList, and also add 't' as char, its neighbors {'t'}, 't' depends on 't'.
 * 2. work on indegree, increment indegree['t'] += 1, 't' has one more indegree from 't'.
 * 3. need to get the first different char from two strings.
 */

/* precondition:
 * words's length >= 2
 * base case >= 1 is handled in caller function
 */
public static void graphSetup(
    string[] words,
    Dictionary<char, HashSet<char>> dependencyList,
    int[] indegree
)
{
    int len = words.Length;
    for (int i = 1; i < len; i++)
    {
        string prev = words[i - 1];
        string curr = words[i];
    }
}

```

Find a bug, 2nd version, left side, no edge case: return; should be continue; line 89

[5]<https://gist.github.com/jianminchen/58d80aa86a027af7a3e52277d15c3733>

```

public static void graphSetup(
    string[] words,
    Dictionary<char, HashSet<char>> dependencyList,
    int[] indegree
)
{
    int len = words.Length;
    if (len == 1)
        return; // cannot do anything

    for (int i = 1; i < len; i++)
    {
        string prev = words[i - 1];
        string curr = words[i];

        int start = 0;
        while (prev[start] == curr[start])
        {
            start++;
        }

        // no edge
        if (start >= Math.Min(prev.Length, curr.Length))
            return;

        // at most one edge
        char edgeFrom = prev[start];
        //char edgeTo = prev[start]; // bug!!!
        char edgeTo = curr[start];

        if (!dependencyList.ContainsKey(edgeFrom))
        {
            dependencyList.Add(edgeFrom, new HashSet<char>());
        }

        if (!dependencyList[edgeFrom].Contains(edgeTo))
        {
            dependencyList[edgeFrom].Add(edgeTo);
            indegree[edgeTo] = 'a'++;
        }
    }
}

```

```

public static void graphSetup(
    string[] words,
    Dictionary<char, HashSet<char>> dependencyList,
    int[] indegree
)
{
    int len = words.Length;

    for (int i = 1; i < len; i++)
    {
        string prev = words[i - 1];
        string curr = words[i];

        int start = 0;
        while (prev[start] == curr[start])
        {
            start++;
        }

        // no edge
        if (start >= Math.Min(prev.Length, curr.Length))
            continue;

        // work on style - double checking - using correct variable
        char edgeFrom = prev[start];
        char edgeTo = curr[start];

        if (!dependencyList.ContainsKey(edgeFrom))
        {
            dependencyList.Add(edgeFrom, new HashSet<char>());
        }

        if (!dependencyList[edgeFrom].Contains(edgeTo))
        {
            dependencyList[edgeFrom].Add(edgeTo);
            indegree[edgeTo] = 'a'++;
        }

        // skip if edgeTo is in the hashset.
    }
}

```

Also, write down tasks before writing code, good habit to train to focus on tasks when writing.

```

* https://en.wikipedia.org/wiki/topological_sorting
* review the idea of topological sorting:
* 1. push all indegree nodes with 0 into the queue
* 2. work on queue
*   step 1: dequeue the node from the queue,
*   step 2: update indegree node's value effected - decrement one
*   step 3: if the dependency list's node indegree value down to zero,
*
*   add the node to the queue
*
* 3. construct the output string
*
* Is it just the normal queue process, can you do a bug free writing?
*/

public static String topologicalSort(
    String[] words,
    Dictionary<char, HashSet<char>> dependencyList,
    int[] indegree
) {
    HashSet<char> nodes = getNodes(words);

    Queue<char> queue = new Queue<char>();
    for (int i = 0; i < 26; i++)
    {
        char runner = (char) ('a' + i);
        if (!nodes.Contains(runner))
            continue; // skip it

        if (indegree[i] == 0)
            queue.Enqueue(runner);
    }

    // Two tasks:
    // 1. get nodes with indegree 0 - enqueue those nodes
    // 2. play with queue
    //   * node is dequeued, append to the output stringBuilder
    //   * adjust its dependency list - indegree value - decrement one
    //   * if the neighbor node is with 0 indegree value, push it to the queue
    //
    // Checking list about queue
    // 1. first queue is not empty - add nodes into queue first - with indegree 0 nodes
    // 2. when node is dequeued, the indegree is updated accordingly, then, more nodes will be
    //   added to queue when its indegree is 0
    //
    // 3. Every node in the queue is with indegree 0 - this is a fact.
    //
    public static String topologicalSort(
        String[] words,
        Dictionary<char, HashSet<char>> dependencyList,
        int[] indegree,
        HashSet<char> nodes
    ) {
        // HashSet<char> nodes = getNodes(words);

        Queue<char> queue = new Queue<char>();
        for (int i = 0; i < 26; i++)
        {
            char runner = (char) ('a' + i);
            if (!nodes.Contains(runner))
                continue;

            if (indegree[i] == 0)
                queue.Enqueue(runner);
        }
    }
}

```

Use node in neighbors to make code more readable.

```

public static String topologicalSort(
    String[] words,
    Dictionary<char, HashSet<char>> dependencyList,
    int[] indegree
) {
    HashSet<char> nodes = getNodes(words);

    Queue<char> queue = new Queue<char>();
    for (int i = 0; i < 26; i++)
    {
        char runner = (char) ('a' + i);
        if (!nodes.Contains(runner))
            continue; // skip it

        if (indegree[i] == 0)
            queue.Enqueue(runner);
    }

    StringBuilder sb = new StringBuilder();
    while (queue.Count > 0)
    {
        char runner = queue.Dequeue();

        sb.Append(runner);

        if (!dependencyList.ContainsKey(runner))
            continue;

        HashSet<char> neighbors = dependencyList[runner];
        foreach (char node in neighbors)
        {
            int index = node - 'a';
            indegree[index]--;

            if (indegree[index] == 0)
                queue.Enqueue(node);
        }
    }

    // edge case
    return sb.Length < nodes.Count ? "" : sb.ToString();
}

public static String topologicalSort(
    String[] words,
    Dictionary<char, HashSet<char>> dependencyList,
    int[] indegree,
    HashSet<char> nodes
) {
    // HashSet<char> nodes = getNodes(words);

    Queue<char> queue = new Queue<char>();
    for (int i = 0; i < 26; i++)
    {
        char runner = (char) ('a' + i);
        if (!nodes.Contains(runner))
            continue;

        if (indegree[i] == 0)
            queue.Enqueue(runner);
    }

    StringBuilder sb = new StringBuilder();
    while (queue.Count > 0)
    {
        char runner = queue.Dequeue();

        sb.Append(runner);

        if (!dependencyList.ContainsKey(runner)) // bug!!! - forget these 2 lines, runtime execution error
            continue;

        HashSet<char> neighbors = dependencyList[runner];
        foreach (char node in neighbors)
        {
            int index = node - 'a';
            indegree[index]--;

            if (indegree[index] == 0)
                queue.Enqueue(node);
        }
    }

    // edge case
    return sb.Length < nodes.Count ? "" : sb.ToString();
}

```

One algorithm a time. Take your time to master an algorithm.

Everyone asks me what's next. I'm here to stay, taking it one game at a time.[6] #CreateYourMark [7]pic.twitter.com/CzDq5CIGMA

— Angelique Kerber (@AngeliqueKerber) [8]May 21, 2016

1. <http://juliachencoding.blogspot.ca/2016/06/leetcode-269-alien-dictionary.html>
2. <https://gist.github.com/jianminchen/85129ed50ce597f896b0f0c5a2fa5586>
3. <https://gist.github.com/jianminchen/58d80aa86a027af7a3e52277d15c3733>
4. <https://gist.github.com/jianminchen/8d4c1f601bae0ca7ef27e470dfe1e636>
5. <https://gist.github.com/jianminchen/58d80aa86a027af7a3e52277d15c3733>
6. <https://twitter.com/hashtag/CreateYourMark?src=hash>
7. <https://t.co/CzDq5CIGMA>
8. <https://twitter.com/AngeliqueKerber/status/734019444868059136>

## **LINQ Architecture - Pluralsight.com (2016-06-17 23:28)**

June 17, 2016

Spend two hours to study the course LINQ Architecture by Scott Allen.

The lecturer webpage:

[1]<https://www.pluralsight.com/authors/scott-allen>

Plan to study a few more LINQ courses:

1. LINQ Fundamentals
2. LINQ architecture
3. LINQ data access

1. <https://www.pluralsight.com/authors/scott-allen>

---

## **LINQ Fundamentals - pluralsight.com (2016-06-18 11:43)**

June 18, 2016

Work on course - LINQ Fundamentals - pluralsight.com - 4 hours courses -

June 18, 2016 11:43am - 2:43pm

### **Manipulating Data**

object data (generics, algorithms)

relational data (ADO.net, SQL)

XML data (XmlDocument, XPath/ XSLT)

Language Integrated Query (LINQ) works for all 3 kinds of data -

### **Standard Query Operators:**

Defined in the System.Linq namespace

Work on any IEnumerable<T>

CLS compliant (generics required)

### **LINQ's Extensibility**

Operator extensibility

Provider extensibility

A LINQ provider is a gateway to query-able types:

PLINQ, LINQ to LDAP, LINQ to Flickr, LINQ to Amazon

## **LINQ to Objects**

Replace foreach loops and other iterative code with LINQ expressions:

## **Deferred Execution**

Query expression does not execute until we access the result

## **LINQ to XML**

Not just another XML API

XElement is the core class in the System.Xml.Linq namespace

## **Entity Framework**

Entity framework provides a rich layer of object data services

Object services - LINQ to Entities and Entity SQL, Change tracking and identity management,

Serialization and data binding, and Connection and transaction management

## **Digging into C # Features For LINQ**

Extension methods

Lambda Expressions

Expression Trees

Query Expressions

Type Inference

Anonymous Types

Partial Methods

blogs:

[1]<http://www.albahari.com/nutshell/10linqmyths.aspx>

1. <http://www.albahari.com/nutshell/10linqmyths.aspx>

---

## **.Net programming technologies (2016-06-18 16:17)**

June 18, 2016

Spend 20 minutes to work on a small research, what .NET technologies are popular in the Vancouver market. Learn advanced technology, reduce time to do development and maintenance.

indeed.ca

## **equinox:**

AngularJS, jquery, json

Soap/ XML/ XSLT/ XSD/ WSDL

LINQ

## Health Employers Association:

MVC, Ext.Net or ExtJS

MCSD, MCPD for web and desktop application, database and integration service.

---

## Rest Fundamentals - pluralsight.com (2016-06-19 12:55)

June 19, 2016

Plan to spend 3 hours to take this course:

Rest Fundamentals

Lecturer:

[1]<http://app.pluralsight.com/author/howard-dierking>

Plan to study 3 hours:

Five Essential Tools for Building REST API Is

by

[2]Elton Stoneman

Reading:

<https://msdn.microsoft.com/en-us/library/jj819168.aspx>

Go over Rest API example, learn the basics of Rest API ...

<http://www.asp.net/web-api/overview/older-versions/build-restful-apis-with-aspnet-web-api>

Rest API project - developing a REST web service using C # - A walkthrough

<http://www.codeproject.com/Articles/112470/Developing-a-REST-Web-Service-using-C-A-walkthrough>

[3] <http://www.codeproject.com/Articles/21174/Everything-About-REST-Web-Services-What-and-How-Pa>

[4] <http://www.codeproject.com/Articles/21258/Everything-about-REST-web-services-what-and-how>

<https://www.quora.com/How-do-I-start-learning-or-strengthen-my-knowledge-of-data-structures-and-algorithms>

[5]<http://stackoverflow.com/questions/671118/what-exactly-is-restful-programming> (20 minutes reading)

RSET is a lightweight alternative to mechanisms like RPC (Remote Procedure Calls) and Web Services (SOAP, WSDL, et al.).

1. <http://app.pluralsight.com/author/howard-dierking>
  2. <https://app.pluralsight.com/author/elton-stoneman>
  3. <http://www.codeproject.com/Articles/21174/Everything-About-REST-Web-Services-What-and-How-Pa>
  4. <http://www.codeproject.com/Articles/21258/Everything-about-REST-web-services-what-and-how>
  5. <http://stackoverflow.com/questions/671118/what-exactly-is-restful-programming>
- 

## Leetcode 329: Longest increasing path in matrix (2016-06-20 23:23)

June 20, 2016

problem statement:

Given an integer matrix, find the length of the longest increasing path. From each cell, you can either move to four directions: left, right, up or down. You may NOT move diagonally or move outside of the boundary (i.e. wrap-around is not allowed).

### 329. Longest Increasing Path in a matrix

#### Example 1:

```
nums = [
  [
    9
  ,9,4],
  [
    6
  ,6,8],
  [
    2
  ,
  1
  ,1]
]
```

Return 4

The longest increasing path is [1, 2, 6, 9].

Study the code written in Java:

[1]<http://blog.csdn.net/sbitswc/article/details/50707203>

1st practice using C #:

[2]<https://gist.github.com/jianminchen/b984ccba8dabbb0bb78160c6e5c6e8b4>

Question and answer:

1. How is the practice?

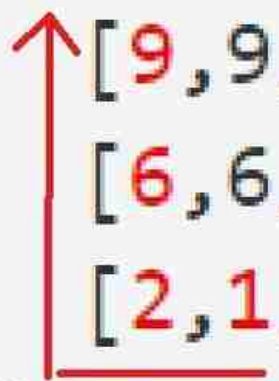
The study was very good. DFS - using recursive call, and then, tricky part is to use memorization to avoid duplicated calculation.

2. Talk more with an easy example, therefore, next time you will not forget the problem after a few months.

Answer: Will write something here.

```
nums = [  
  [9,9,4],  
  [6,6,8],  
  [2,1,1]  
]
```

```
nums = [  
  [9,9,4],  
  [6,6,8],  
  [2,1,1]  
]
```



1. Talk about node row = 2, col = 1, value is 1, denoted as start1, 3 neighbors, left (value 2), right (value 1), up (value 6)
2. Try to avoid loops - base case checking, always go to the node value  $\geq$  current value  $\geq$  start node '1'
3. Also, if neighbour node left (value 2) has maximum increasing path in matrix n, then, what we can know:  
value 1 < value 2, then, start1's longest path at least 1 + value2's longest increasing path. Need to check other 2 neighbors to see if it is larger value

left neighbor (value 2)'s longest increasing path in matrix 2  $\rightarrow$  6  $\rightarrow$  9, length is 3;

right neighbor (value 1)'s longest increasing path in matrix 1  $\rightarrow$  8, length is 2;

upper neighbor (value 6)'s longest increasing path in matrix (actually 2 of them, 6  $\rightarrow$  9 or 6  $\rightarrow$  8), length is 2.

4. DFS algorithm can be set up using recursive function; also, there are total  $3 \times 3 = 9$  nodes in the above matrix, each node's increasing path in matrix should not count more than once.

For example,  $\text{nums}[0,0] = 9$ , is maximum value of matrix, so the longest path = 1.

Let us put a sentence together. How about the following:

Be greedy, check your neighbors (at most 4), and find maximum one with longest increasing path in matrix, and then use the path

to build your own. The value is 1 + that neighbor's problem, where the recursive function is constructed.

5. The design of algorithm - **brute force solution** - how many paths in the matrix -  $n^4 = n \times n \times n \times n$ ; filter out non-increasing path, then find maximum value - not efficient

6. For each node in matrix, find its longest increasing path in matrix,  $n \times n$  nodes, each node, DFS algorithm is applied. at most  $n \times n$  node is checked about increasing order. <- try to reason the time complexity -> ...

Will be less than  $n^4$ .

- 7.

## **P prepare a check list for the design:**

1. Run time issue - index out of range - array - boundary check  
loops - always go to bigger value - no loop
2. time complexity -  $n \times n$  node, each one does DFS; each DFS, at most  $4n^2$  comparison of value.
3. space complexity - use extra array  $n \times n$  to store bool value - memorization
4. value is bigger/ smaller / 0, 0 - not possible, at least 1, miss count - recursive, should be easy.
5. Brute force solution and its issues - more time consuming etc., duplication calculation

## **Most important discussion:**

find the idea to store in extra array:

1. Extra array to store the length of "longest increasing path in matrix" for each node in the array - (working idea)
2. Extra array to mark the node is visited or not - (not enough for calculation!)

denote DFS function to calculate the longest increasing path in matrix, then starting from start1, the function will be left->left->up->up, in other words, from 1->2->6->9, and then,

up->right, 1->6->8

up-> up, 1->6->9,

the visiting order of neighbors is left, down, right, up.

[3]<https://gist.github.com/jianminchen/b984ccb8dabbb0bb78160c6e5c6e8b4>

anti-clockwise, left, down, right, up - line 116 - 120

So, the cache value of (2, 1) is 4, longest path is 4 (1->2->6->9); and 6 nodes in matrix are calculated and saved with cache value - 2 matrix(2,0), 6(1,0), 9(0,0), 6(1,1), 8(1,2), 9(0,1), the order of saving is

9(0,0), - 0 neighbor

6(1,0), - 1 neighbor

2 matrix(2,0) - 1neighbor, value 1

6(1,1)

8(1,2)

9(0,1)

6(1,1) - 2 neighbors, comparison 1 vs 1

1(2,1) - 2 neighbors, comparison 3 vs 2

Use stack to help to track the order:

start1 node, 2 neighbors

go to left neighbor first,

push to stack, 2, 6, 9

no down neighbor, skip right

then go to up neighbour,

...



## Statistics:

Time to work out order of calculation cache value in the above list takes more than 10 minutes.

Two motivations to work on reasoning and analysis:

1. Leetcode 329 is hard
2. Being able to write down bug free, executable code in 20 minutes.
3. Study more solutions from others, and then, practice it using C #. Study one using C++.

C++ solution:

[4]<http://gaocegege.com/Blog/algorithm/leetcode329>

[5]<http://codingmelon.com/2016/01/20/longest-increasing-path-in-a-matrix-leetcode-329/>

To be continued.

1. <http://blog.csdn.net/sbitswc/article/details/50707203>
  2. <https://gist.github.com/jianminchen/b984ccba8dabb0bb78160c6e5c6e8b4>
  3. <https://gist.github.com/jianminchen/b984ccba8dabb0bb78160c6e5c6e8b4>
  4. <http://gaocegege.com/Blog/algorithm/leetcode329>
  5. <http://codingmelon.com/2016/01/20/longest-increasing-path-in-a-matrix-leetcode-329/>
- 

## Strength knowledge of data structures and algorithms (2016-06-21 00:21)

June 21, 2016

20 minutes research on the topic - strength knowledge of data structures and algorithms.

[1]<https://www.quora.com/How-do-I-start-learning-or-strengthen-my-knowledge-of-data-structures-and-algorithms>

Write down some notes.

Top coder website to read:

[2]<https://www.topcoder.com/community/data-science/data-science-tutorials/>

Read the Quora question:

[3]<https://www.quora.com/What-are-the-algorithms-required-to-solve-all-problems-using-C++-in-any-competitive-coding-contest>

86 algorithms: (get to know the algorithms - memorize the name first)

[4]<https://www.quora.com/What-are-the-10-algorithms-one-must-know-in-order-to-solve-most-algorithm-problems/answer/Pratyush-Khare-1?srid=iGPI&share=1>

[5]<http://www.ideserve.co.in/>

[6]<http://www.ideserve.co.in/learn/lowest-common-ancestor-of-two-nodes-binary-search-tree>

[7][https://docs.google.com/document/d/1\\_dc3lfg7Gg1LxhiqMMmE9UbTsXpdRiYh4pKILYG2eA4/edit?pref=2&pli=1](https://docs.google.com/document/d/1_dc3lfg7Gg1LxhiqMMmE9UbTsXpdRiYh4pKILYG2eA4/edit?pref=2&pli=1)

Algorithm webpage: in Russian, code is good:

[8]<http://e-maxx.ru/algo/>

Julia, if you miss the good lecture, then just click the link:

[9]<http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-046j-introduction-to-algorithms-sma-5503-fall-2005/video-lectures/>

Plan to learn some technologies:

List can be very long, ...

Angular JS, Bootstrap, Entity framework, Rest API, AWS, Azure ...

1. <https://www.quora.com/How-do-I-start-learning-or-strengthen-my-knowledge-of-data-structures-and-algorithms>
  2. <https://www.topcoder.com/community/data-science/data-science-tutorials/>
  3. <https://www.quora.com/What-are-the-algorithms-required-to-solve-all-problems-using-C++-in-any-competitive-coding-contest>
  4. <https://www.quora.com/What-are-the-10-algorithms-one-must-know-in-order-to-solve-most-algorithm-problems/answer/Pratyush-Khare-1?srid=iGPI&share=1>
  5. <http://www.ideserve.co.in/>
  6. <http://www.ideserve.co.in/learn/lowest-common-ancestor-of-two-nodes-binary-search-tree>
  7. [https://docs.google.com/document/d/1\\_dc3Ifg7Gg1LxhiqMMmE9UbTsXpdRiYh4pKILYG2eA4/edit?pref=2&pli=1](https://docs.google.com/document/d/1_dc3Ifg7Gg1LxhiqMMmE9UbTsXpdRiYh4pKILYG2eA4/edit?pref=2&pli=1)
  8. <http://e-maxx.ru/algo/>
  9. <http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-046j-introduction-to-algorithms-sma-5503-fall-2005/video-lectures/>
- 

## ATP - Tennis Coaching - Sports coaching (2016-06-21 23:47)

June 21, 2016

Julia spent 2 hours to do a small research on ATP player Novak and how he works very well with his coach - Boris Becker. Champions need coach, how they select and work with their coaches?

Spent time to watch the interview:

Boris Becker on coaching Novak Djokovic

[1]<https://www.youtube.com/watch?v=1a5w-LBot1c>

I like how relax the interview is, and Boris expressed his idea so clearly, so calm. Julia likes to learn how mental tough or focus Boris is to answer the interview questions.

Justin Gimelstob Interview Boris Becker Part 2

[2]<https://www.youtube.com/watch?v=FV4GUd60pjl&spfreload=5>

Julia coaches herself to be a better programmer, she pushes herself to write down her thought, and then, re-view, criticize her own writing/ thoughts, and then try to find what she is looking for - **a coach makes difference** .

1. <https://www.youtube.com/watch?v=1a5w-LBot1c>
  2. <https://www.youtube.com/watch?v=FV4GUd60pjI&spfreload=5>
- 

## Leetcode 139: word break I - 3+ practices (2016-06-22 18:01)

June 22, 2016

Work on leetcode 139 - word break I

Problem statement:

Given a string *s* and a dictionary of words *dict*, determine if *s* can be segmented into a space-separated sequence of one or more dictionary words.

For example, given  
*s* = "leetcode",  
*dict* = ["leet", "code"].

Return true because "leetcode" can be segmented as "leet code".

Programming skills is like muscle, if you do not maintain it, it will go back to fat. When Julia tries to stretch DP muscle, she found out that she has nothing to stretch. She has to work on it, build up by hours, build up again. Leetcode 139: word break I (medium).

Julia read the code/ solution, but she could not figure out in detail, specially for those two loops - for DP algorithm.

[1]<http://bangbingsyb.blogspot.ca/2014/11/leetcode-word-break-i-ii.html>

1 hour, play with 2 test cases.

3 practices using C #; Set up a test case, run the code, and then, understand the algorithm.

Same code, different test cases

### 1st practice:

[2]<https://gist.github.com/jianminchen/f763ce5033ea4b152c7a56aaf98b7075>

### 2nd practice:

[3]<https://gist.github.com/jianminchen/a1bafb4271b8bbdb18b92d0b3dad913e>

### 3rd practice:

[4]<https://gist.github.com/jianminchen/bd9301c3ad38d801001d1d66a08256fe>

Let us walk through a test case:

**string test** = "abcd",  
**and dictionary string array**: {"a","bc","d" }

We know that, "a" can be constructed by word "a" in the dictionary.  
"ab" cannot be constructed by the dictionary.

Now, what we can do to work on substring "abc", assuming that "a", "ab" are processed already.

### How to approach the problem?

Use dynamic programming, reduce the work to minimum.

abc, so 'c' is the new comer, assuming that "a" and "ab" have been processed. By debugging the code, Julia put together the following comment:

```
// "abc" is constructable "a" + "bc"
// exhaustive search three times:
//   check dp       check dictionary
// 1. "ab"         "c"
// 2. "a"          "bc"  <- it is constructable, break the loop
// 3. ""           "abc"
```

so, **step 1**, "ab" cannot be constructed by the words in dictionary, so it does not matter if 'c' is in dictionary.

Use array bool[] dp to store the cached value;

Cannot conclude that "abc" can be constructed by words in dictionary, need to go through all possible new words:  
{"c", "bc", "abc" }

**step 2**, backward one more char, new word: "bc",

"a" is in cache - dp[1] = true,

"bc" is in the dictionary.

so, "abc" can be constructed by words in dictionary.

Stop here, break the loop.

At most there are 3 words ending at 'c':

"c",  
"bc",  
"abc",

All other substrings are the substring of "ab", assuming that DP is used, so no need to worry about.

### 4th C # practice:

[5]<https://gist.github.com/jianminchen/36de4c46292e79290af95098dfec3cb0>

### Question and Answer:

1. What is most time consuming part in coding?

Julia spent over 20 minutes to figure out what should be looped on - line 80, it is hard to figure out meaningful ways to loop. She tried a few of ideas, then, she chose to loop on the position of right string.

"abc",

i = 2, then, 3 things are tried:

1. "ab", "c"
2. "a", "bc"
3. "", "abc"

two strings are in each case, right string's starting position in the original string "abc"

line 80 for( int pos = i; pos >= 0; pos--)

i = 2, "c"

i = 1, "bc"

i = 0, "abc"

#### 4th C # practice:

[6]<https://gist.github.com/jianminchen/36de4c46292e79290af95098dfce3cb0>

Spend 10 minutes to write the program again.

#### 5th C # practice:

[7]<https://gist.github.com/jianminchen/9766aea70c9015fabbea98fb71b56d6d>

highlight of changes:

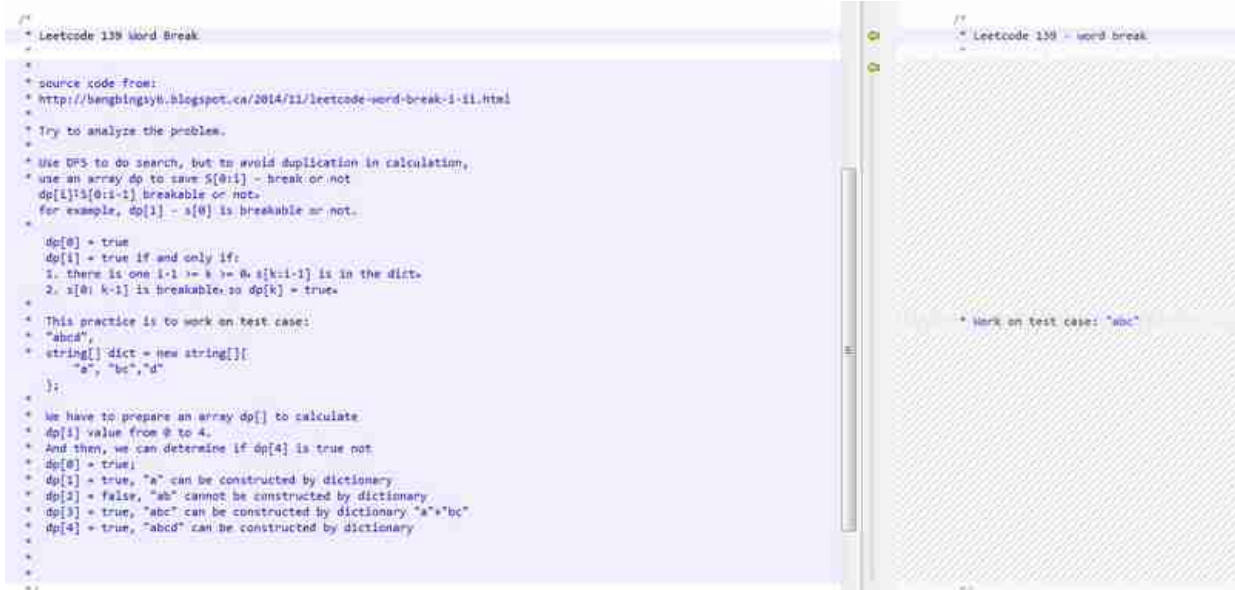
1. line 41 - variable name is changed from " left " to " **existingWord** ", existingWord is already processed, which can be looked up through cache array - true/ false.

2. line 42, **newWord** , each time the new char is processed, there are ith words ending at position i.

Need to check if it is in word dictionary.

Comparison between 4th practice and 5th practice:

comment:



code:

4th practice:

1. Do not know only work on substring 0-i, length i+1 substring, bottom up approach as DP - dynamic programming.
2. Confuse with string left, right, why left string needs to check cache, and then right string checks word dictionary.

```

public static bool wordBreak(
    string s,
    HashSet<string> wordDict)
{
    if (s == null || s.Length == 0)
        return false;

    int len = s.Length;

    bool[] cache = new bool[len + 1];
    cache[0] = true; // "" empty string

    for (int i = 1; i < len + 1; i++)
    {
        // "abc", "abac"
        // "abc"
        // right string start position point index
        for (int pos = 1; pos <= i; pos++)
        {
            string left = s.Substring(0, pos);
            string right = s.Substring(pos, i - left.Length); // if right == "" then left should be i
            string word = s.Substring(pos, i - pos + 1);

            if (word.Length > 0 && wordDict.Contains(word))
            {
                cache[pos] = true;
                break;
            }
        }
    }

    return cache[i];
}

```

Read blog to review dynamic programming:  
[\[8\]https://en.wikipedia.org/wiki/Dynamic\\_programming](https://en.wikipedia.org/wiki/Dynamic_programming)

Review a few concepts:

**Principle of Optimality**

**optimal substructure**

larger problem - sub-problems

**Bellman equation** - optimization literature names relationship

optimal substructure and overlapping sub-problems

Divide and Conquer vs dynamic programming

Top-down approach vs Bottom-up approach

Word break I uses bottom up solution -

**6th practice:**

[\[9\]https://gist.github.com/jianminchen/29758bad5d56d54e87c2a6f52d057835](https://gist.github.com/jianminchen/29758bad5d56d54e87c2a6f52d057835)

comparison line by line:

6th practice 5th practice

```

public static bool wordBreak(
    string s,
    HashSet<string> wordDict)
{
    if (s == null || s.Length == 0)
        return false;

    int len = s.Length;

    bool[] cache = new bool[len + 1];
    cache[0] = true;

    // work on dynamic programming - bottom-up approach
    // not top-down approach
    // first loop is to handle bottom up, from first char to the
    // string in the cache used as a string
    for (int i = 1; i < len + 1; i++)
    {
        // second loop, loop on the start position of new word
        // start point starts from 0 decreasing to 1 by 1 to 0
        // For example, "abc", when working on "ab", "c".
        // new word "abc", "ab", "a"
        // use variable new startpos instead of pos
        for (int startpos = 1; startpos <= i; startpos++)
        {
            // new word, first word is part of subproblem, processed string
            int startposIndex = i; // string to work on: index 0 - i
            string processedStr = s.Substring(0, startposIndex);
            int processedLen = processedStr.Length;
            string wordend = s.Substring(startpos, i - startpos + 1);

            if (wordend.Length > 0 && wordDict.Contains(wordend))
            {
                cache[startposIndex] = true;
                break;
            }
        }
    }

    return cache[i];
}

```

**7th practice:**

second loop on DP algorithm, new word loop starts from start position from 0 to i, in ascending order, line 43.  
[10]<https://gist.github.com/jianminchen/2e56665d934f9b9cbc2f94adc8e567bb>

Statistics:

Time spent:

2 hour +

Train insane or remain the same - focus on training!

Train insane or remain the same '[11] #progress [12] #nopainnogain [13] #mondaymotivation  
Séance de 1/2squat aujourd'hui? [14]pic.twitter.com/K8tGiOXixC

— Kristina Mladenovic (@KikiMladenovic) [15]June 20, 2016

1. <http://bangbingsyb.blogspot.ca/2014/11/leetcode-word-break-i-ii.html>
2. <https://gist.github.com/jianminchen/f763ce5033ea4b152c7a56aaf98b7075>
3. <https://gist.github.com/jianminchen/a1bafb4271b8bbdb18b92d0b3dad913e>
4. <https://gist.github.com/jianminchen/bd9301c3ad38d801001d1d66a08256fe>
5. <https://gist.github.com/jianminchen/36de4c46292e79290af95098dfee3cb0>
6. <https://gist.github.com/jianminchen/36de4c46292e79290af95098dfee3cb0>
7. <https://gist.github.com/jianminchen/9766aea70c9015fabbea98fb71b56d6d>
8. [https://en.wikipedia.org/wiki/Dynamic\\_programming](https://en.wikipedia.org/wiki/Dynamic_programming)
9. <https://gist.github.com/jianminchen/29758bad5d56d54e87c2a6f52d057835>
10. <https://gist.github.com/jianminchen/2e56665d934f9b9cbc2f94adc8e567bb>
11. <https://twitter.com/hashtag/progress?src=hash>
12. <https://twitter.com/hashtag/nopainnogain?src=hash>
13. <https://twitter.com/hashtag/mondaymotivation?src=hash>
14. <https://t.co/K8tGiOXixC>
15. <https://twitter.com/KikiMladenovic/status/744887194016452609>

---

## HackerRank: world code sprint #4 - minimum distance (2016-06-26 00:56)

June 26, 2016

problem statement:

[1]<https://www.hackerrank.com/contests/june-world-codesprint/challenges/minimum-distances>

C # practice:

[2]<https://gist.github.com/jianminchen/f2732794c68a5f2602d17e0052ebbd41>

1. <https://www.hackerrank.com/contests/june-world-codesprint/challenges/minimum-distances>
2. <https://gist.github.com/jianminchen/f2732794c68a5f2602d17e0052ebbd41>

---

## HackerRank: Equal stacks (2016-06-26 00:59)

June 26, 2016

Problem statement:

[1]<https://www.hackerrank.com/contests/june-world-codesprint/challenges/equal-stacks>

C # practice:

[2]<https://gist.github.com/jianminchen/28a6120097026961f9671710fd9a911d>

1. <https://www.hackerrank.com/contests/june-world-codesprint/challenges/equal-stacks>

2. <https://gist.github.com/jianminchen/28a6120097026961f9671710fd9a911d>

---

## HackerRank: AorB - intense workout - 3 hours+ (2016-06-26 01:03)

June 26, 2016

HackerRank: AorB

problem statement:

[1]<https://www.hackerrank.com/contests/june-world-codesprint/challenges/a-or-b>

Quick summary of 3 practices:

1st practice: failed all test cases except first one.

C # practice: only pass 1 test case, failed all other cases, score 0/ 50

[2]<https://gist.github.com/jianminchen/ebbb58587568127a3f2d9254d650fa20>

2nd practice: fix the bug - major issue, but failed most test cases still.

Fix the above issue - comparison issue, but still failed most of test cases:

[3]<https://gist.github.com/jianminchen/15c0470a5bb86fb06bc30a4b83465313>

3rd practice: pass all test cases.

Work on the test case, fix the bug. C # practice passes all test cases:

[4]<https://gist.github.com/jianminchen/cf823376a23a36304da82ca975f77fa1>

So, starting from very beginning:

**1st practice:** failed all test cases except first one.

C # practice: only pass 1 test case, failed all other cases, score 0/ 50

[5]<https://gist.github.com/jianminchen/ebbb58587568127a3f2d9254d650fa20>

Spent over hours, go over again and again the problem statement, and then play with HackerRank unknown test case, try to figure out what is wrong.

So, work on problems one by one, it takes hours up to 8 hours. Great workout!



Here are a list of problems to work on, very good workout:

1. Remove leading 0 in HexDecimal string, for example, "08" should be "8"

[6]<https://gist.github.com/jianminchen/fb14be371486b0fa3659cc2542a41bc1>

2. Get a char from a binary number with 4 bits: 1111, return F; 1000, return 8

[7]<https://gist.github.com/jianminchen/fbea867f130b7d84e44085eab3671883>

3. Get an integer value from HexDecimal char - for example, 'F', return 15; '9' return 9

[8]<https://gist.github.com/jianminchen/2fbf865e2bdb49681bb8c7e9899d5908>

4. Work out a small problem - AorB - try to define a function, get count of bits to change from 1 to 0, in A, based on AorB value.

[9]<http://juliachencoding.blogspot.ca/2016/06/aorb-hackerrank-workout-example.html>

5. ( **6/27/2016 added** )

Write a function to do A or B calculation, A and B is bigger than 0, and smaller than  $16^{(50000)}$

Test case:

Hexadecimal: 1000 | 11000 = 11000

[10]<http://juliachencoding.blogspot.ca/2016/06/hexadecimal-or-calculation.html>

June 27, 2016

**Question and Answer:**

**1. How is the practice?**

The design has issues, here is the detail:

function getChangeBits \_From0To1, getFirstNumberFirst1To0 \_LetSecondOneTake1, getBitNumber \_From1To0  
\_strVersion all have same fatal error:

**Review the design:  $A \mid B = C$**

**D8 A**

**A01 C**

**For example, A string "D8", AorB string "A01",  
then, D should compare to 0, and 8 should compare to 1**

**So, the solution is to reverse the string.**

**8D A's reverse**

**10A C's reverse**

But in the the code from line 397 - 399, 'D' is compared to 'A'. The design cannot handle difference lenght correctly.

That is the reason of failing most of test cases.

1. line 385 function `getChangeBits_From0To1` has a bug:

inside the function,

line 395: the Hexadecimal char is not matching between from and AorB sum.

For example, A string "D8", AorB string "A01",

then, D should compare to 0, and 8 should compare to 1

But in the the code from line 397 - 399, 'D' is compared to 'A'. The design cannot handle difference length correctly.

That is the reason of failing most of test cases.

2. Same problem applies to function `getFirstNumberFirst1To0_LetSecondOneTake1`

3. And also, line 268 - line 282 can be shortened, remove if/else from line 279 - 291. Just update variable: `afterAnd_From` value

4. Same problem applies to function `getBitNumber_From1To0_strVersion`

**2nd practice:** fix the bug - major issue, but failed most test cases still.

Fix the above issue - comparison issue, but still failed most of test cases:

[11]<https://gist.github.com/jianminchen/15c0470a5bb86fb06bc30a4b83465313>

Need to make this test case work, through test case provided by downloads:

input:

3  
25  
B631EB5AE  
601C227E1  
707AC8792  
12  
CAF7028FD  
59B5AC1CE  
CAF1B7B7F  
3  
81B9BB94E  
8AB3CA95E  
8BBBFB95E

output:

C8582  
707A00790  
8AF10287D  
570

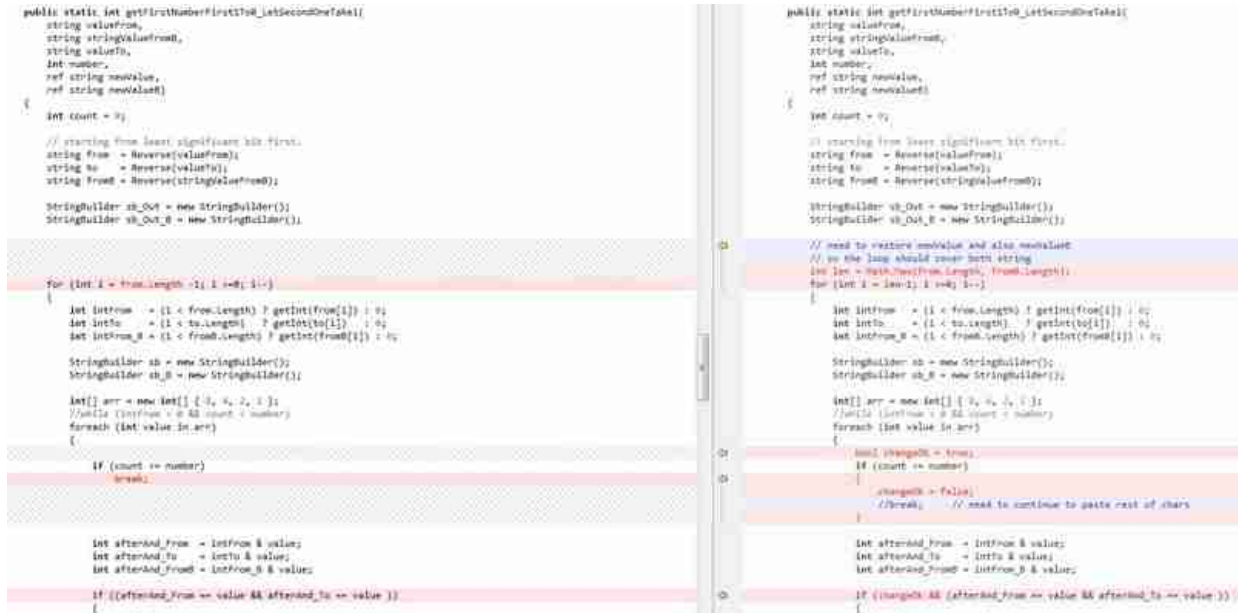
48B1B534E  
B9BB94E  
8BB3CA95E

**3rd practice** : pass all test cases.

Work on the test case, fix the bug. C # practice passes all test cases:

[12]<https://gist.github.com/jianminchen/cf823376a23a36304da82ca975f77fa1>

Bug fix comparison - what to change:



2. How long is the practice and bug fix? What are major issues?

Time spent is much more than expected.

Major issue is about the design of function, too long to handle without a bug. Need to think about breaking into small function.

**function getFirstNumberFirst1To0\_LetSecondOneTake1 -**

**line 249 to 330, almost 100 lines. Need to redesign**

**the function, into multiple functions.**

Warmup the algorithm, write another one in practice:

**things to work on:**

1. function name should be changed: (A|B = c AorB problem)

function name:

getFirstNumberFirst1To0\_LetSecondOneTake1

new function name:

getGreedy\_A1To0\_BStayUpdateTo1 -> get\_A\_SmallestPossible

greedy algorithm to make A as small as possible

2. Use array to reduce the total of variables, and make the code less lines

3 variables: A, B, C 3 string

line 260 - line 262, string from, to, fromB,

Declare a string array with length [3],

0 - A

1 - B

2 - C

3. line 272 - line 274, 3 variables,

intFrom, intTo, intFrom \_B,

use array to reduce variable to one

4. Line 279, array variable - arr - move definition out of for loop (line 270 - 330)

5. Declare an array to store chars, and add a new function called toChar(int)

line 298, line 317, line 318

```
public static char tochar(int val, int powerof2) {
```

```
    return (char)(value/powerof2 + '0');
```

```
}
```

6. getCharFromBinaryNumber function name is too long, maybe, define a class BinaryNumber, and then add getChar api

1. <https://www.hackerrank.com/contests/june-world-codesprint/challenges/aorb>

2. <https://gist.github.com/jianminchen/ebbb58587568127a3f2d9254d650fa20>

3. <https://gist.github.com/jianminchen/15c0470a5bb86fb06bc30a4b83465313>

4. <https://gist.github.com/jianminchen/cf823376a23a36304da82ca975f77fa1>

5. <https://gist.github.com/jianminchen/ebbb58587568127a3f2d9254d650fa20>

6. <https://gist.github.com/jianminchen/fb14be371486b0fa3659cc2542a41bc1>

7. <https://gist.github.com/jianminchen/fbea867f130b7d84e44085eab3671883>

8. <https://gist.github.com/jianminchen/2fbf865e2bdb49681bb8c7e9899d5908>

9. <http://juliachencoding.blogspot.ca/2016/06/aorb-hackerrank-workout-example.html>

10. <http://juliachencoding.blogspot.ca/2016/06/hexadecimal-or-calculation.html>

11. <https://gist.github.com/jianminchen/15c0470a5bb86fb06bc30a4b83465313>

12. <https://gist.github.com/jianminchen/cf823376a23a36304da82ca975f77fa1>

---

## Remove leading 0 in Hexadecimal string (2016-06-26 10:21)

June 26, 2016

1. Remove leading 0 in HexDecimal string, for example, "08" should be "8"

[1]<https://gist.github.com/jianminchen/fb14be371486b0fa3659cc2542a41bc1>

1. <https://gist.github.com/jianminchen/fb14be371486b0fa3659cc2542a41bc1>

## Convert a binary number from 0 - 16 to a char (HexaDecimal binary string to a char) (2016-06-26 10:23)

June 26, 2016

Get a char from a binary number with 4 bits: 1111, return F; 1000, return 8

[1]<https://gist.github.com/jianminchen/fbea867f130b7d84e44085eab3671883>

1. <https://gist.github.com/jianminchen/fbea867f130b7d84e44085eab3671883>

---

## Convert a HexaDecimal char to an integer value (2016-06-26 10:26)

June 26, 2016

Get an integer value from HexaDecimal char - for example, 'F', return 15; '9' return 9

[1]<https://gist.github.com/jianminchen/2fbf865e2bdb49681bb8c7e9899d5908>

1. <https://gist.github.com/jianminchen/2fbf865e2bdb49681bb8c7e9899d5908>

---

## Reverse a string (2016-06-26 10:32)

June 26, 2016

Reverse a string in C #:

[1]<https://gist.github.com/jianminchen/ea5f706117ef8316aef66e057e0c86ee>

Read blog:

[2]<http://stackoverflow.com/questions/228038/best-way-to-reverse-a-string>

C # implementation

[3]<https://gist.github.com/jianminchen/9ad0fff7c0e1d3f7294ef3dcc24b933d>

Reverse, Array reverse, XOR, benchmark testing etc.

[4]<https://gist.github.com/jianminchen/b8d94943ff002fc617c9b46ef472b8f5>

1. <https://gist.github.com/jianminchen/ea5f706117ef8316aef66e057e0c86ee>

2. <http://stackoverflow.com/questions/228038/best-way-to-reverse-a-string>

3. <https://gist.github.com/jianminchen/9ad0fff7c0e1d3f7294ef3dcc24b933d>

4. <https://gist.github.com/jianminchen/b8d94943ff002fc617c9b46ef472b8f5>

---

## AorB - HackerRank workout example (2016-06-26 12:45)

June 26, 2016

A small problem related to AorB problem, and get some workout on bit manipulation. Still work on designing a well-defined problem - something small enough, meaningful enough.

[1]<https://www.hackerrank.com/contests/june-world-codesprint/challenges/a-or-b>

The coding is time consuming, but it is good workout.

Try to put together a meaningful, small function:

HexaDecimal Char update

For example, A = 1000

A | B = C,

C = 0111

Then, 4th bit 1 in A should be set to 1.

number is 1, A' = 0000; **assuming that B will provide first 3 bits of 1.**

Another example, A = 1100,

A | B = C,

C = 0111,

then, 4th bit 1 in A should be set to 0

function return is 1, A' = 0100; assuming that B will take care first 3 bits of 1.

1. count how many bits to update (only bit with value 1)
2. return A's new value A'
3. Work on bits from left to right, lowest significant bit first
4. Assume the value > 0, and value < 10^50000

Work on HexaDecimal string,

For example, D8, work on '8' first, 4 bits 1000, from right to left;

and then, work on 'D', 1101 - 13

edge case:

A= "8D",

C = "E",

C is short in length, one char only.

**Practice 1:** with a bug

[2]<https://gist.github.com/jianminchen/f4d8761ca6126052e3df792212861436>

Question and Answer:

The design has issues, here is the detail:

function

**updateHexaDecimal\_Remove1s (line 55 - 75)** has

same fatal error:

**Review the design: A | B = C**

D8 A  
A01 C

For example, A string "D8", AorB string "A01",  
then, D should compare to 0, and 8 should compare to 1

So, the solution is to reverse the string.

8D A's reverse  
10A C's reverse

But in the the code from line 61 -65, 'D' is compared to 'A'. The design cannot handle difference length correctly.

**Practice 2:** with bug fix

[3]<https://gist.github.com/jianminchen/c8501eabdb774710c100a1aa06049d8e>

Read blogs:

From high school:

[4]<http://tarawa.github.io/>

From Tsinghua University:

[5]<http://maskray.me/>

[6]<https://threads-iiith.quora.com/>

[7][http://heliang.me/blog/?page\\_id=488](http://heliang.me/blog/?page_id=488)

[8][http://heliang.me/blog/?tag= %E9 %9D %A2 %E8 %AF %95](http://heliang.me/blog/?tag=%E9%9D%A2%E8%AF%95)

[9]<https://threads-iiith.quora.com/>

[10]<https://www.hackerrank.com/roba>

[11]<https://www.linkedin.com/in/changhai-zhou-823ba265>

[12][https://github.com/maximizedchang/projecteuler/tree/master/Java %20Files](https://github.com/maximizedchang/projecteuler/tree/master/Java%20Files)

[13]<http://yufeizhao.com/olympiad.html>

1. <https://www.hackerrank.com/contests/june-world-codesprint/challenges/aorb>
  2. <https://gist.github.com/jianminchen/f4d8761ca6126052e3df792212861436>
  3. <https://gist.github.com/jianminchen/c8501eabdb774710c100a1aa06049d8e>
  4. <http://tarawa.github.io/>
  5. <http://maskray.me/>
  6. <https://threads-iiith.quora.com/>
  7. [http://heliang.me/blog/?page\\_id=488](http://heliang.me/blog/?page_id=488)
  8. <http://heliang.me/blog/?tag=%E9%9D%A2%E8%AF%95>
  9. <https://threads-iiith.quora.com/>
  10. <https://www.hackerrank.com/roba>
  11. <https://www.linkedin.com/in/changhai-zhou-823ba265>
  12. <https://github.com/maximizedchang/projecteuler/tree/master/Java%20Files>
  13. <http://yufeizhao.com/olympiad.html>
- 

## Leetcode solution to study (2016-06-26 13:31)

June 26, 2016

Spend some time to study Leetcode solution from the following blog:

[1]<http://maskray.me/blog/2014-06-29-leetcode-solutions>

Leetcode 56: Merge Intervals

[2]<http://xiaoyaoworm.com/blog/2016/06/27/%E6%96%B0leetcode56-merge-intervals/>

Leetcode 75: sort colors

[3]<http://fisherlei.blogspot.ca/2013/01/leetcode-sort-colors.html>

Leetcode 130 surrounded region

[4]<http://www.cnblogs.com/higerzhang/p/4149040.html>

Learn system design:

[5]<https://www.facebook.com/careers/life/preparing-for-your-software-engineering-interview-at-facebook>

[6]<https://codelab.interviewbit.com/> (facebook codelab)

Plan to work on: 1 hour daily starting from July 1, 2016

[7]<http://www.hiredintech.com/system-design>

A small research - hiring best:

[8]<https://www.toptal.com/freelance/in-search-of-the-elite-few-finding-and-hiring-the-best-developers-in-the-industry>

1. <http://maskray.me/blog/2014-06-29-leetcode-solutions>

2. <http://xiaoyaoworm.com/blog/2016/06/27/%E6%96%B0leetcode56-merge-intervals/>



3. <http://fisherlei.blogspot.ca/2013/01/leetcode-sort-colors.html>
  4. <http://www.cnblogs.com/higerzhang/p/4149040.html>
  5. <https://www.facebook.com/careers/life/preparing-for-your-software-engineering-interview-at-facebook>
  6. <https://codelab.interviewbit.com/>
  7. <http://www.hiredintech.com/system-design>
  8. <https://www.toptal.com/freelance/in-search-of-the-elite-few-finding-and-hiring-the-best-developers-in-the-industry>
- 

## Hexadecimal number bit manipulation practice (2016-06-26 22:31)

June 26, 2016

Need to work on problem solving, [1]AorB, how to design a few simple APIs first?

Write a few function to check bit by bit from right to left, or left to right.

[2]<https://gist.github.com/jianminchen/103859e5e8287c9fd865c59363ad500b>

$A \mid B = C$ , to make A as small as possible, with maximum number of bits to update.

For example,

1

1001000 A

0 0000000 B

-----  
11001000 C

If there are 2 more bits allowed to be update, then to make A smallest as possible, update A's leftmost bit to 0, and update C's leftmost bit to 1.

01001000 A'

10000000 B'

-----  
11001000 C

Review blogs related to bit manipulations:

power of 2:

[3]<http://juliachencoding.blogspot.ca/2016/05/algorithm-check-if-number-is-power-of-2.html>

Blogs to read:

1. Angular 2 minutes review:

[4]<http://henriquat.re/intro/angular/angularjsForDotNetDevelopers.html>

2. So many courses on pluralsight, how to choose courses to learn day by day.

[5]<http://app.pluralsight.com/author/joe-eames>

### 3. Reading material:

[6]<http://web.stanford.edu/class/cs9/>

[7]<http://web.stanford.edu/class/cs9/lectures/07/Numbers.pdf>

[8]<https://courses.csail.mit.edu/iap/interview/index.php>

1. <https://www.hackerrank.com/contests/june-world-codesprint/challenges/aorb>

2. <https://gist.github.com/jianminchen/103859e5e8287c9fd865c59363ad500b>

3. <http://juliachencoding.blogspot.ca/2016/05/algorithm-check-if-number-is-power-of-2.html>

4. <http://henriquat.re/intro/angular/angularjsForDotNetDevelopers.html>

5. <http://app.pluralsight.com/author/joe-eames>

6. <http://web.stanford.edu/class/cs9/>

7. <http://web.stanford.edu/class/cs9/lectures/07/Numbers.pdf>

8. <https://courses.csail.mit.edu/iap/interview/index.php>

---

## A comparison of Microsoft web technology - pluralsight (2016-06-27 20:54)

June 27, 2016

One hour lecture:

A comparison of Microsoft Web Technologies.

Start to work on short course first, work on foundation, and then, move to Angular JS, MVC later.

Course:

A Comparison of Microsoft Web Technologies

[1]<https://www.pluralsight.com/authors/michael-palermo>

1. <https://www.pluralsight.com/authors/michael-palermo>

---

## Hexadecimal OR calculation (2016-06-27 21:32)

June 26, 2017

To work on HackerRank algorithm [1]AoB, Julia found out that it took her more than 10 5+ hours to work on/ debugging/ bug fixing. She needs to work on design carefully, work on small functions first.

### Strategies talk:

In order to save time and help herself think clearly/ simple, at the very beginning, she should define this small function, play with the code, and make it bug-free first. So, choose an easy battle first, something she can finish in

20 minutes and well-defined, easy to set up test cases and verify, and then, put together a complicate function to complete the task. If there is doubt or unsolvable issues, go back to the easy function, work on it more.

### Problem statement:

Write a function to do A or B calculation, A and B is bigger than 0, and smaller than  $16^{(50000)}$

### Analysis:

The size of number is too big, 64 bit integer is only  $< 2^{64}$ , so only work on one char at a time - one Hexadecimal char (0-F) a time. Only convert char and integer between Hexadecimal char(0-F) and integer (0-15).

Test case:

Hexadecimal: 1000 | 11000 = 11000

A | B

For example:

A = "111A"

B = "1B"

so, A | B = "111B"

C # practice:

[2]<https://gist.github.com/jianminchen/5442cb579b13ffc203ce89b3e54e8a36>

1. <https://www.hackerrank.com/contests/june-world-codesprint/challenges/aorb>

2. <https://gist.github.com/jianminchen/5442cb579b13ffc203ce89b3e54e8a36>

---

## A small research on HackerRank world sprint #4 (2016-06-27 22:37)

June 27, 2016

On Sunday June 26, Julia spent 3 hours to do some research, evaluate where she is in the competition, and what kind of target she can set, how to reach the target next time. Actually she went through **current leaderboard**, and study who are best players, top-performance programmers.

### Statistics:

**Time spent on HackeRank world sprint #4: 4+ hours on June 25, 2016**

Score: 40, around 2276 rank in over 5000 people.

### Target:

She needs to score another 50 - 100 in order to be competitive, where she found out that some Google, Facebook employee score' range (above 100, some above 140).

Top 16 - full score (420), there are a lot of world champions over there. So, the problems have some pattern, people can get trained to be good at them.

Here is the link of HackerRank world sprint #4.

[1]<https://www.hackerrank.com/contests/june-world-codesprint/challenges>

She completed first two easy algorithms in first 2+ hours. Get 40 points.

**Gamble the luck is not a good strategy. Set low target is also not a good strategy. 4 hours can be spent on more meaningful learning experience.**

### **Why?**

Julia spent most of time to work on one problem, AorB, try to score another 50 points. She enjoyed coding and found one problem after another, but it is time-consuming, not efficient approach.

In first hour, she needs to write down every small function she should solve first, and then, design well-defined small function to be helpers, understand possible test cases. Simple problems are easy to work on, save time.

Some problems were discovered after she worked on the problem almost 4+ hours. Julia, you have to learn to be a good thinker first, force yourself think hard on first 30 minutes. Write down things to consider in design of function.

### **1. What is the problem she found out after 3+ hours?**

The solution has problem, Hexadecimal should be reversed first, then, first char is least significant bit, much easy to iterate, compare among A, B, C. Mistakes like A, B, C to compare different bit, not align properly.

### **2. What is the problem she found out after 4+ hours?**

**how to make A as small possible?**

**Provided with extra allowed bits, Give up 1 from A (1->0), by the order of most significant bit first, and then, set B's bit to 1 if need.**

### **3. What is the problem she found after 8+ hours?**

**A|B = C, so C's length as string should be longest one. C's length = Math.Max(A.Length, B.Length). This helps the design and also testing.**

From 12:00pm - 12:00am, she spent a lot of hours to play with code, until midnight, she gave up AorB, scored 0 with many hours passion/ learning experience/ lesson learned.

### Lesson learned:

#### Passion does not work, good workout; but not practical.

Break the problem into 5 - 6 well defined small functions;

First 30 minutes, be a good thinkers. Find all possible problems, assertions, read problem statement again and again.  
( *make sense? :-)* )

Write a few small functions to warmup, fully test with test cases. 2 - 3 hours.

For example, work on this small well-defined function first,

[2]<http://juliachencoding.blogspot.ca/2016/06/hexadecimal-or-calculation.html>

[3]<https://gist.github.com/jianminchen/5442cb579b13ffc203ce89b3e54e8a36>

And then, spend 1 hour to put together for solution AorB.

The best performer in the world (172minutes/7 algorithms), average less than 30 minutes/ algorithm.

### Good about the practice:

#### Practice code on problem solving:

1. reverse a string in C # -> ToCharArray() -> IEnumerable first
  2. check least significant bit using And 1 (bit manipulation)
  3. check most significant bit using And 1 - define value array {8,4,2,1 }
- 3B: leftshift >>1 in C # practice
4. Hexadecimal char <-> binary format string <-> integer value expression
  5. int, long, BigInteger(?) cannot handle big number  $10^{5000}$ , force me to work on string manipulation
  6. Look into C # const keyword, how to separate variables - using existing variable, set new value, less code.
  7. StringBuilder class needs System.Text package
  8. break statement - be careful when the function is not well-defined small function, causing bugs.
  9.  $A|B = C$ , what we can tell the length relationship among A, B, C.

### Actionable items:

Choose 10 people's code to read.

Learn to be a good thinker - work on first 30 minutes, thinking, reading, define well-defined small functions.

Blog to read:

[4]<https://www.linkedin.com/pulse/why-best-designers-problem-solvers-katy-tsai?trk=hp-feed-article-title-like>

I've tasted success and I'm hungry for more. I'll prove I've got what it takes to win. [5] #CreateYourMark  
[6][pic.twitter.com/jansdjv9pG](https://pic.twitter.com/jansdjv9pG)

— Lucas Pouille (@la\_pouille) [7]May 20, 2016

1. <https://www.hackerrank.com/contests/june-world-codesprint/challenges>

2. <http://juliachencoding.blogspot.ca/2016/06/hexadecimal-or-calculation.html>

3. <https://gist.github.com/jianminchen/5442cb579b13ffc203ce89b3e54e8a36>

4. <https://www.linkedin.com/pulse/why-best-designers-problem-solvers-katy-tsai?trk=hp-feed-article-title-like>

5. <https://twitter.com/hashtag/CreateYourMark?src=hash>

6. <https://t.co/jansdjv9pG>

7. [https://twitter.com/la\\_pouille/status/733783371705421824](https://twitter.com/la_pouille/status/733783371705421824)

## HackerRank: World codesprint #4 - Roads in HackerLand (2016-06-29 22:07)

June 29, 2016

Problem statement:

[1]<https://www.hackerrank.com/contests/june-world-codesprint/challenges/johnland>

John lives in HackerLand, a country with cities and

bidirectional roads. Each of the roads has a distinct length, and each length is a power of two (i.e., raised to some exponent). It's possible for John to reach any city from any other city.

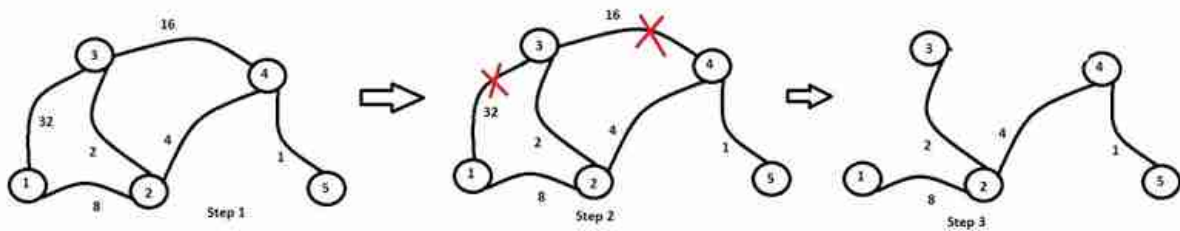
Given a map of HackerLand, can you help John determine the sum of the minimum distances between each pair of cities? Print your answer in [2]binary representation.

Max Score: 60

Difficulty: Moderate

Submissions: 1319

The problem is a graph problem, so here is her analysis:



Spent over 1 hour to try to figure out the graph problem, what is my best bet to solve the problem.

Find simple problems to work on first:

**Algorithm 1. Any two directed connected nodes, find shortest distance :**

distance (n1, n3) = 10, go through node 1 -> node 2 -> node 3, instead of directly connected edge - 32

**2. Any two nodes in the graph, find the shortest distance :**

based on graph in the step 3, find a route from one node to another node, using BFS or DFS, for example, node 1 to node 4, node 1 -> node 2 -> node 4.

min distance (node1, node4)

Spend a lot of time here to try to design the algorithm:

1. using DFS/recursive/memorization solution:

evaluate the solution, problems in the design cannot be solved:

problem and its subproblems are overlapping.

min distance (node2, node4)

Not working

2. using BFS/queue, the design concerns:

1. Avoid loops

2. Stop early -

3. using DFS/stack, the design concerns:

Also, each edge's length is power of 2, different length.

1. Step 1-> Step 2:

Try to work on this graph, for each directed connected graph, use BFS algorithm, try to find a shorter route. For example,

edge(1, 3) = 32,

edge(1, 2) = 8,

edge(2, 3) = 2,

edge(1,3) > edge(1, 2) + edge(2,3), since  $32 > 8 + 2$ ,

So, the edge(1, 3) can be marked removable, this direct route will never be traveled.

Work out the above case:

start from node 1, add node 1 to the queue, and then, go through the queue, remove node from queue, add all neighbors to the queue, if it is not visited previously, and if it is the node 3, then ~~only allow twice~~, as long as the sum is less than edge(1,3) = 32, continue; Make a single linked list,  $n2.prev = n1$ ,  $n3.prev = n2$ ,

2. To work on graph in step 3,

For each node in the graph, for example,  $n1$ , using DFS or BFS to find a route to node  $j$ ,  $j < i$ .

So, with those removed edges, only one route will be found through any two nodes,  $i < j$ .

To be continued.

I've tasted success and I'm hungry for more. I'll prove I've got what it takes to win. [3] #CreateYourMark  
[4]pic.twitter.com/jansdjv9pG — Lucas Pouille (@la\_pouille) [5]May 20, 2016

1. <https://www.hackerrank.com/contests/june-world-codesprint/challenges/johnland>

2. [https://en.wikipedia.org/wiki/Binary\\_number#Representation](https://en.wikipedia.org/wiki/Binary_number#Representation)

3. <https://twitter.com/hashtag/CreateYourMark?src=hash>

4. <https://t.co/jansdjv9pG>

5. [https://twitter.com/la\\_pouille/status/733783371705421824](https://twitter.com/la_pouille/status/733783371705421824)

## JavaScript Array - 2+ hour study (2016-06-30 19:35)

June 30, 2016

Work on JavaScript Array - Design a drill to understand JavaScript Array, and learn some API design.

### Target:

Memorize all 29 API, be able to write down API function syntax, understand why, how for each API.

Each practice takes 20 minutes:

10 minutes to remember Array properties.

10 minutes to remember Array 29 APIs.

Prepare the study content first and document the experience:

### 1. PROPERTY:

3 properties:

constructor,

length

prototype

constructor property:

[1][https://msdn.microsoft.com/en-us/library/jj155291\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/jj155291(v=vs.94).aspx)

length:

array is sparse, so the array is not contiguous. The length is not necessarily the number of elements in the array.

[2][https://msdn.microsoft.com/en-us/library/d8ez24f2\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/d8ez24f2(v=vs.94).aspx)

Prototype:

[3][https://msdn.microsoft.com/en-us/library/jj155285\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/jj155285(v=vs.94).aspx)

### 2. JavaScript array 29 methods:



Spend 10 minutes a time to remember all the function names,

Spend 10 minutes to guess each API's functionality, and details about arguments, how it is designed.

Spend 10 minutes reading/ practice/ ask questions.

Repeat the process.

[4][https://msdn.microsoft.com/en-us/library/k4h76zbx\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/k4h76zbx(v=vs.94).aspx)

**write down my guess: (June 30, 2016 - too honest here! Will improve JavaScript knowledge, learn design!)**

Array.from - copy array from, input argument is array.

isArray - Array.isArray(arr) input argument arr is array

of - ? wild guess -

concat - arr1.concat(arr2), concatenate the string

entries - entries - arguments: startIndex, endIndex, return subarray?

Discussion:

**Array.from study :**

two concepts:

1. array-like object, an iterable object.

Array.from

syntax: Array.from(arrayLike[,mapfn [,thisArg]]);

Array-like object:

[5]<http://stackoverflow.com/questions/11886578/creating-array-like-objects-in-javascript>

Iterable object:

[6][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterables\\_and\\_Generators](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterables_and_Generators)

JavaScript String, Array, TypedArray, Map and Set are all built-in iterables.

Read JavaScript Map

[7][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Map](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Map)

Detail see the blog:

[8]<http://juliachencoding.blogspot.ca/2016/06/javascript-arrayof-function-study.html>

every - iterator - go through each node in the array to check some logic?

fill - fill - arr.fill(1), all the elements in the array are assigned to the same value

filter

findIndex

foreach

indexOf

join

keys

lastIndexOf

map

pop

push

reduce

reduceRight

reverse

shift

slice

586

some

sort

splice

toString

unshift

valueOf

values

challenges:

Arguments:

callback function -

callback function syntax

3 things Julia likes the JavaScript Array.fill function design:

- start, end arguments are options
- negative start, end arguments handling

**Questions:**

**1. Why JavaScript Array API is designed so differently? need to learn more about functional programming?**

**Reference:**

Previous blog:

Array

[9]<http://juliachencoding.blogspot.ca/2016/06/array-class-c-c-javascript-java.html>

Array-like object:

[10]<http://stackoverflow.com/questions/11886578/creating-array-like-object-in-javascript>

Iterable object:

[11][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterators\\_and\\_Generators](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterators_and_Generators)

1. [https://msdn.microsoft.com/en-us/library/jj155291\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/jj155291(v=vs.94).aspx)
  2. [https://msdn.microsoft.com/en-us/library/d8ez24f2\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/d8ez24f2(v=vs.94).aspx)
  3. [https://msdn.microsoft.com/en-us/library/jj155285\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/jj155285(v=vs.94).aspx)
  4. [https://msdn.microsoft.com/en-us/library/k4h76zbx\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/k4h76zbx(v=vs.94).aspx)
  5. <http://stackoverflow.com/questions/11886578/creating-array-like-objects-in-javascript>
  6. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterators\\_and\\_Generators](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterators_and_Generators)
  7. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Map](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Map)
  8. <http://juliachencoding.blogspot.ca/2016/06/javascript-array-of-function-study.html>
  9. <http://juliachencoding.blogspot.ca/2016/06/array-class-c-c-javascript-java.html>
  10. <http://stackoverflow.com/questions/11886578/creating-array-like-objects-in-javascript>
  11. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterators\\_and\\_Generators](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterators_and_Generators)
- 

## JavaScript Array.from function - 60 minutes study (2016-06-30 22:13)

June 30, 2016

### Array.from study :

[1][https://msdn.microsoft.com/en-us/library/k4h76zbx\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/k4h76zbx(v=vs.94).aspx)

two concepts:

1. array-like object, an iterable object.

Array.from

syntax: `Array.from(arrayLike[,mapfn [,thisArg]])`;

Array-like object:

[2]<http://stackoverflow.com/questions/11886578/creating-array-like-objects-in-javascript>

Iterable object:

[3] [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterators\\_and\\_Generators](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterators_and_Generators)

JavaScript String, Array, TypedArray, Map and Set are all built-in iterables.

Read JavaScript Map

[4] [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Map](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Map)

1. [https://msdn.microsoft.com/en-us/library/k4h76zbx\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/k4h76zbx(v=vs.94).aspx)
2. <http://stackoverflow.com/questions/11886578/creating-array-like-objects-in-javascript>

3. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterators\\_and\\_Generators](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterators_and_Generators)
  4. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Map](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Map)
- 

## JavaScript Array.of Function study (2016-06-30 22:23)

June 30, 2016

Start to work on JavaScript 29 function one by one. Focus on one function at a time.

[1][https://msdn.microsoft.com/en-us/library/k4h76zbx\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/k4h76zbx(v=vs.94).aspx)

return an array from passed in arguments.

Syntax:

```
Array.of(element0[, element1][, ...][,elementN]);
```

Read blogs:

[2][https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Array/of](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/of)

[3]<https://gist.github.com/rwaldron/1074126>

5 functions - indexOf(), filter(), forEach(), map(), reduce() (1 hour study June 30, 2016 10:00pm - 11:00pm)

Learn technique called **demethodizing**.

[4]<http://colintoh.com/blog/5-array-methods-that-you-should-use-today>

JavaScript Array's reduce Method:

[5][https://www.youtube.com/watch?v=J21rsVw\\_NBE](https://www.youtube.com/watch?v=J21rsVw_NBE)

accumulator

Spend 1 hour to watch the video:

Code genius - Using JavaScript to Teach JavaScript by John Resig

[6][https://www.youtube.com/watch?v=H4sSldXv\\_S4](https://www.youtube.com/watch?v=H4sSldXv_S4)

### Actionable Items:

Use JsFiddle to edit JavaScript code snippets.

1. [https://msdn.microsoft.com/en-us/library/k4h76zbx\(v=vs.94\).aspx](https://msdn.microsoft.com/en-us/library/k4h76zbx(v=vs.94).aspx)

2. [https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Array/of](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/of)

3. <https://gist.github.com/rwaldron/1074126>
  4. <http://colintoh.com/blog/5-array-methods-that-you-should-use-today>
  5. [https://www.youtube.com/watch?v=J21rsVw\\_NBE](https://www.youtube.com/watch?v=J21rsVw_NBE)
  6. [https://www.youtube.com/watch?v=H4sSldXv\\_S4](https://www.youtube.com/watch?v=H4sSldXv_S4)
- 

## 2.7 July

### Office politics, hard conversation, company culture study (2016-07-01 10:51)

July 1, 2016

Spend 20 minutes to do some research about office politics. Study the article and write down some notes. Julia did some research before - office politics.

It is better to focus on thing easy to control, work on communication instead. She takes baby steps, learns to improve communication. But, it is worthy of time to study the article.

How facebook tries to prevent office politics?

[1]<https://hbr.org/2016/06/how-facebook-tries-to-prevent-office-politics>

Office politics is very good topic, Julia learns not to waste time on it. Try to move it positive directions. Do not take it personal, think about how to win together, do not let emotion take control. Learn to play very good team sports through tennis helps a lot. Julia just keeps calm, and active and move forward with creative ideas/ solutions.

Here is the notes from the article.

Five tactics:

**1. Look for empire builders, self-servers, and whiners in the hiring process - and don't hire them .**

Hire best and smartest people they can find.

Priorities are company, team, and self - in that order.

**2. Take the incentive out of "climbing the ladder".**

At Facebook, moving into management is not a promotion. It's a lateral move, a parallel track.

Managers are people with strong people skills.

Individual contributors (ICs) should be your best minds for informing team goals and direction.

Use "hackamonth", where ICs can take a month to help another team on a specific project. It helps to keep talent at the company, keep mold from growing on our team.

3.

**Be open and transparent, and create opportunities for voices to be heard.**

- make escalation "legal"
- frequent question-and-answer sessions with leadership.
- Engagement surveys

4. Make everyone accountable, so personal bias can't creep into decision making.

5. Train your leaders to effectively manage politics out of conversations.

... we train employees on how to have hard conversation. We tell them when they see something they don't like, they should start by telling the other person what they saw, how they felt, and what the result of that action was.

... we tell them not to assume you understand the why - start by trying to understand the other person's perspective. Doing so helps avoid the kinds of resentment that can lead to political behavior.

More readings:

Facebook core values:

1. Be bold 2. Focus on impact 3. Move fast 4. Be Open 5. Build Social Value

[2]<http://recruiterbox.com/blog/3-ways-facebook-recruits-hires-the-best-candidates/>

Hire best, not just for next 6 months.

2. Break All rules - for each new manager

[3]<http://www.businessinsider.com/facebook-recommends-this-book-to-new-managers-2016-2>

[4]<http://www.gallup.com/businessjournal/529/The-Four-Keys-to-Great-Management.aspx>

3.

[5]<http://www.businessinsider.com/facebook-hr-chief-explains-its-performance-reviews-2016-2>

4.

[6]<http://www.businessinsider.com/facebook-best-managers-exhibit-these-7-behaviors-2016-1>

5. Hard Conversation culture

[7]<http://fortune.com/2015/12/09/facebook-company-culture/>

6. Chinese blogs about career, facebook:

[8]<https://zhuanlan.zhihu.com/p/20879014?refer=qinchao>

[9]<https://zhuanlan.zhihu.com/p/20865236?refer=qinchao>

7. Data driven, A/B testing, great article:

[10]<https://zhuanlan.zhihu.com/p/20865236?refer=qinchao>

[11]<https://zhuanlan.zhihu.com/p/20226008>

1. <https://hbr.org/2016/06/how-facebook-tries-to-prevent-office-politics>
  2. <http://recruiterbox.com/blog/3-ways-facebook-recruits-hires-the-best-candidates/>
  3. <http://www.businessinsider.com/facebook-recommends-this-book-to-new-managers-2016-2>
  4. <http://www.gallup.com/businessjournal/529/The-Four-Keys-to-Great-Management.aspx>
  5. <http://www.businessinsider.com/facebook-hr-chief-explains-its-performance-reviews-2016-2>
  6. <http://www.businessinsider.com/facebook-best-managers-exhibit-these-7-behaviors-2016-1>
  7. <http://fortune.com/2015/12/09/facebook-company-culture/>
  8. <https://zhuanlan.zhihu.com/p/20879014?refer=qinchao>
  9. <https://zhuanlan.zhihu.com/p/20865236?refer=qinchao>
  10. <https://zhuanlan.zhihu.com/p/20865236?refer=qinchao>
  11. <https://zhuanlan.zhihu.com/p/20226008>
- 

## **Teach people how to be a manager from a programmer background (2016-07-01 13:24)**

July 1, 2016

Teach people how to be a manager as a programmer

[1]<http://www.businessinsider.com/8-habits-of-highly-effective-google-managers-2011-3>

1. Be a good coach (provide specific, constructive feedback, balance +, - feedbacks)

Have regular one-on-ones, tailored to the employee's strengths.

2. Empower your team and don't micro-manage

Make "sketch" assignments

3. Express interest in employee's success and well-being

4. Be productive and results-oriented

5. Be a good communicator and listen to your team

6. Help your employees with career development

7. Have a clear vision and strategy for the team

8. Have key technical skills, so you can help advise the team.



## 7. Better bosses

"people analytics" teams - Eight habits of Highly Effective Google Managers

"cognitive biases" - "halo/horns" effect

Have some time for them and to be consistent.

1. <http://www.businessinsider.com/8-habits-of-highly-effective-google-managers-2011-3>

---

## Nine principles of innovation (2016-07-01 13:26)

July 1, 2016

Study 9 core principles of innovation -

[1]<http://www.fastcompany.com/3021956/how-to-be-a-success-at-everything/g>  
innovation

oogles-nine-principles-of-

### 1. Innovation comes from anywhere

Moral obligation to extend help, change search result, for example: "How to commit suicide" - National Suicide Prevention Hotline

2.

### Focus on the user

Worry about the money later, when you focus on the user, all else will follow.

Predictive analysis so search suggestions come up after the user types a few keystrokes. This instant search feature saves the user a few microseconds with each entry. Google sales reps were concerned that this shortened the time customers would review ads, but the company went ahead and believed that it was worth the risk.

"Create a great user experience and the revenue will take care of itself."

### 3. Aim to be ten times better

If you come into work thinking that you will improve things by ten percent, you will only see incremental change. Think 10 times improvement instead.

### 4. Bet on technical strengths

a self-driving car - all building blocks - Google maps, Google Earth, and Street View cars.

### 5. Ship and iterate

Ship your products often and early, and do not wait for perfection.

Google Chrome - launched in 2008, every six weeks Google pushed out an improved version.

6. Give employees 20 percent time  
"They will delight you with their creative thinking"  
Google street view car -> Google tricycle

7. Default to open processes  
Make your processes open to all users. Tap into the collective energy of the user base to obtain great ideas.

Android platform-> open source  
voice search app-> children sent clever videos that rivaled the campaigns of the big ad agencies.

8. Fail well  
There should be no stigma attached to failure.

9.  
Have a mission that matters  
Chinese version:  
[2]<https://buzzorange.com/techorange/2013/12/09/google-reveals-its-9-principles-of-innovation/>

1. <http://www.fastcompany.com/3021956/how-to-be-a-success-at-everything/googles-nine-principles-of-innovation>
  2. <https://buzzorange.com/techorange/2013/12/09/google-reveals-its-9-principles-of-innovation/>
- 

## JavaScript Array developer.Mozilla.org - 3+ hours study (2016-07-02 11:02)

July 2, 2016

First thing in the morning, read JavaScript Array webpage, and then, start to work on the target:

1. Memorize all the content on the page. (Estimated time to understand every term on the page: 10+ hours)

[1][https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Array/prototype](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/prototype)

2. List all concepts to learn
3. Questions about the content

Training on JavaScript - start from memorizing JavaScript Array APIs

Time spent:  
3 hours (July 2, 2016)

[2][https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Array/prototype](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/prototype)

Design drills to memorize the JavaScript function names:

## 1. Properties

Array.prototype

Array.length

Method (33 methods)

Array.from()

Array.isArray()

Array.of()

Array.prototype.concat()

Array.prototype.copyWithin()

Array.prototype.entries()

Array.prototype.every()

Array.prototype.fill()

Array.prototype.filter()

Array.prototype.find()

Array.prototype.findIndex()

Array.prototype.forEach()

Array.prototype.includes()

Array.prototype.indexOf()

Array.prototype.join()

Array.prototype.keys()

Array.prototype.lastIndexOf()

Array.prototype.map()

Array.prototype.pop()

Array.prototype.push()

Array.prototype.reduce()

Array.prototype.reduceRight()

Array.prototype.reverse()

Array.prototype.shift()

Array.prototype.slice()

Array.prototype.some()

Array.prototype.sort()

Array.prototype.splice()

Array.prototype.toLocaleString()

Array.prototype.toSource()

Array.prototype.toString()

Array.prototype.unshift()

Array.prototype.values()

Inheritance:

Function -> properties -> (5, please remember: c, d, l, n, p)

Function.caller

Function.displayName

Function.length

Function.name

Function.prototype

Function -> Methods -> (6, please remember: a, b, c, i, t, t)

Function.prototype.apply()

Function.prototype.bind()

Function.prototype.call()

Function.prototype.isGenerator()

Function.prototype.toSource()

Function.prototype.toString()

Object

Methods

Object.prototype.\_\_defineGetter\_\_(\_\_())

...

## 2. Study notes (July 2, 2016):

a global object,

list-like objects

Access(index into) an Array item

Loop over an array - forEach

**push** - Add to the end of an Array

**pop** - Remove from the end of an Array

**shift** - Remove from the front of an Array

**unshift** - Add to the front of an array

Adds one or more elements to the front of an array and returns the new length of the array.

**push** - find the index of an item in the Array

**pop** - Remove an item by Index Position

**splice** - Remove an item by Index Position

**slice** - Copy an Array

julia's comment: slice or dice, slice is just to get part of array, can be the whole array.

function syntax, startIndex, endIndex, arr.slice([begin[, end]])

**from** - Array.from(), creates a new Array instance from an array-like or iterable object.

**isArray** - Array.isArray(), returns true if a variable is an array, if not false.

**of** - `Array.of()`, creates a new Array instance within a variable number of arguments, regardless of number or type of the arguments.

**Mutator methods** - modify the array

**copyWithin** - Copies a sequence of array elements with the array

**fill** - Fills all the elements of an array from a start index to an end index with a static value.

**splice** - Adds and/or removes elements from an array.

**join** - joins all elements of an array into a string.

#### Iteration methods:

`forEach()`

`entries()`

`every()`

`some()`

`filter()`

`find()`

`findIndex()`

`keys()`

`map()`

`reduce()`

`reduceRight()`

`values()`

Question: Syntax - what kind of definition? [ <- optional argument?

traversal and **mutation** operations

Arrays can be stored at non-contiguous locations in the array, JavaScript arrays are not guaranteed to be dense.

Questions: When to use typed arrays?

JavaScript arrays are not guaranteed to be dense, so typed arrays are guaranteed to be dense. (? Look into it later)

**Questions: an associated array - what is "associated" meaning?**

**zero-indexed** -

using

**bracket notation**

- `renderer['ed'].setTexture(model, 'character.png')` - works properly

with **dot notation** - cannot be referenced with dot notation, must be accessed using bracket notation.

`Array.length` talk -

A JavaScript array's length property and numerical properties are connected. Several of the built-in array methods (e.g.,

join, slice, indexOf

, etc.) take into account value of an array's length property when they are called. Other methods (e.g., push, splice, etc.) also result in updates to an array's length property.

**Questions: built-in array methods ? Array.prototype.join, those functions are defined for all iterable object.**

**Good reading with one example, two lines of JavaScript code:**

**Creating an array using the result of a match**

**RegExp.exec, String.match, and String.replace (? study later)**

**Memorize this example - one regular express a time**

```
// Match one d followed by one or more b's followed by one d
// Remember matched b's and the following d
// Ignore case
```

```
var myRe = /d(b+)(d)/i;
var myArray = myRe.exec('cdbBdbsbz')
```

**Question: review regular expression in JavaScript?**

**Array instances**

**All array instances inherit from Array.prototype.**

Some people play not to lose. I play to win each and every second I'm on court. [3] #SpeedTakes aggression. [4]pic.twitter.com/ae5xlriogs

— Simona Halep (@Simona\_Halep) [5]July 2, 2016

1. [https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Array/prototype](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/prototype)
  2. [https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Array/prototype](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/prototype)
  3. <https://twitter.com/hashtag/SpeedTakes?src=hash>
  4. <https://t.co/ae5xlriogs>
  5. [https://twitter.com/Simona\\_Halep/status/749299161510600704](https://twitter.com/Simona_Halep/status/749299161510600704)
- 

**C# questions to review (2016-07-03 09:52)**

July 3, 2016

Just come cross the blogs about C # questions, like to review those questions and see if I have time to catch up something. Write down what I learn most.

[1]<http://www.csharpstar.com/csharp-interview-questions-part-2/>

[2]<http://www.csharpstar.com/csharp-interview-questions-part-3/>

1. <http://www.csharpstar.com/csharp-interview-questions-part-2/>

2. <http://www.csharpstar.com/csharp-interview-questions-part-3/>

---

## **C#, C++, JavaScript, Java String, StringBuilder class study (2016-07-03 16:06)**

July 3, 2016

Spend time to work on String class API on C #, C++, JavaScript, Java language. Always keep busy, study APIs first, go over all APIs.

### **1. C # programming language:**

First, work on C # StringBuilder class:

[1][https://msdn.microsoft.com/en-us/library/system.text.stringbuilder\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.text.stringbuilder(v=vs.110).aspx)

### **2. C++ string class study (30 minutes a time)**

Review and go through each one (30 minutes study .....)

[2]<http://www.cplusplus.com/reference/string/string/>

#### **member type (12):**

value \_type char

traits \_type char \_traits<char>

allocator \_type allocator<char>

reference char &

const \_reference const char &

pointer char\*

iterator a random access iterator to char (convertible to const \_iterator)

const \_iterator a random access iterator to const char

reverse \_iterator reverse \_iterator<iterator>

const \_reverse \_iterator reverse \_iterator<const \_iterator>

different \_type ptrdiff \_t

size \_type size \_t

questions: traits \_type ?

#### **Iterators (8):**

begin

end  
(C++ 11)  
rbegin  
rend  
cbegin  
cend  
crbegin  
crend

### **Capacity (9):**

size  
length  
max\_size  
resize  
capacity  
reserve  
clear  
empty  
shrink\_to\_fit (C++ 11)

### **Element access (3)**

operator[]  
at  
back - access last character  
front - access first character

### **Modifiers (9):**

operator+=  
append  
push\_back  
assign  
insert  
erase  
replace  
swap  
pop\_back

### **String operations (12):**

c\_str  
data  
get\_allocation  
copy  
find  
rfind  
find\_first\_of  
find\_last\_of  
find\_first\_not\_of



find\_last\_not\_of  
substr  
compare

#### **Member constants**

npos - maximum value for size\_t

#### **Non-member function overloads (6)**

operator+  
relational operators  
swap  
operator>>  
operator<<  
getline

### **3. JavaScript String reference** (30 minutes a time to study)

[3][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String)

#### **Properties:**

String.prototype  
String.length

#### **Methods:**

Exclude deprecated methods, try to remember all the following methods: 34 methods

String.fromCharCode()  
String.fromCodePoint()

String.prototype.anchor()  
String.prototype.charAt()  
String.prototype.codePointAt()  
String.prototype.concat()  
String.prototype.endsWith()

String.prototype.includes()  
String.prototype.indexOf()  
String.prototype.lastIndexOf()  
String.prototype.link()  
String.prototype.localeCompare()

String.prototype.match()  
String.prototype.normalize()  
String.prototype.repeat()  
String.prototype.replace()

String.prototype.search()

String.prototype.slice()

String.prototype.split()

String.prototype.startsWith()

String.prototype.normalize()

String.prototype.substr()

String.prototype.substring()

String.prototype.toLocaleLowerCase()

String.prototype.toLocaleUpperCase()

String.prototype.toLowerCase()

String.prototype.toString()

String.prototype.toUpperCase()

String.prototype.trim()

String.prototype.trimLeft()

String.prototype.trimRight()

String.prototype.valueOf()

String.prototype[@@iterator]()

String.raw()

## July 7, 2016 - Reading time: more than 30 minutes

[4][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/substring](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/substring)

substring

data descriptor/ accessor descriptor

In order to understand string - JavaScript, go through a lot of topics:

10 minutes reading of define property

[5][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Object/defineProperty](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/defineProperty)

string literal

global object directly

Template literals

back-tick ( ` )

Expression interpolation

syntactic sugar making substitutions like this more readable

tagged template literal - more advanced form of template literals

Another 10 - 15 minutes reading of template literals

[6][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Template\\_literals](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Template_literals)

July 12, 2016

Java String class

[7][https://docs.oracle.com/javase/7/docs/api/java/lang/String.html#length\(\)](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html#length())

1. [https://msdn.microsoft.com/en-us/library/system.text.stringbuilder\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.text.stringbuilder(v=vs.110).aspx)
  2. <http://www.cplusplus.com/reference/string/string/>
  3. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String)
  4. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/substring](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/substring)
  5. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Object/defineProperty](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/defineProperty)
  6. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Template\\_literals](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Template_literals)
  7. [https://docs.oracle.com/javase/7/docs/api/java/lang/String.html#length\(\)](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html#length())
- 

## **HackerRank: AorB - code submission C# study (2016-07-03 16:37)**

July 3, 2016

Study submission code of AorB, find out things to work on in C # code.

First study of code:

1. [1]<https://gist.github.com/jianminchen/505d4b6a20d10aaa7a859443fa5fcaf2>
2. [2]<https://gist.github.com/jianminchen/1b44d66ce8ac106b38495ea72dc79314>
3.  
[3]<https://gist.github.com/jianminchen/4f7f354675d9d762d1493de0213a0342>
4.  
[4]<https://gist.github.com/jianminchen/2f346cb9230967708b9db0154563cc61>
5.  
[5]<https://gist.github.com/jianminchen/1a8344f1fd0729dfc9e3e3e9be98d303>
6.  
[6]<https://gist.github.com/jianminchen/bcf7ade62dc35a85993b102c40584abe>

### **Statistics:**

Only 11 submissions score 50/50.

1. <https://gist.github.com/jianminchen/505d4b6a20d10aaa7a859443fa5fcaf2>
  2. <https://gist.github.com/jianminchen/1b44d66ce8ac106b38495ea72dc79314>
  3. <https://gist.github.com/jianminchen/4f7f354675d9d762d1493de0213a0342>
  4. <https://gist.github.com/jianminchen/2f346cb9230967708b9db0154563cc61>
  5. <https://gist.github.com/jianminchen/1a8344f1fd0729dfc9e3e3e9be98d303>
  6. <https://gist.github.com/jianminchen/bcf7ade62dc35a85993b102c40584abe>
- 

## HackerRank: AorB - JavaScript code study (2016-07-03 18:51)

July 3, 2016

Problem statement on HackerRank:

[1]<https://www.hackerrank.com/contests/june-world-codesprint/challenges/a-orb>

Work on JavaScript code on HackerRank AorB submission - code study:

1.

[2]<https://gist.github.com/jianminchen/b5ce95c07e3dd027e3753b8c81f00c74>

2.

[3]<https://gist.github.com/jianminchen/2f6b252009868ebcf4d0449aafa01b30>

### Statistics:

Only 2 submissions score 50/50. So, study those two submissions.

### Notes for code 1:

The code is so beautiful!

### Great ideas to highlight (Go over it 30 minutes):

1. line 13 - line 30

**var dec** - a map, '0' maps to 0, ..., 'F' to 15, char to integer;  
variable name is called "dec" - very short!

2. line 32 - line 49

**var hexBin** - a map, hexadecimal char maps to binary format,  
'0' - '0000', ..., 'F' - 'FFFF'  
variable name is called "hexBin" - very good name.  
hexadecimalToBinaryFormat is too long, remove "To", capital case 'B', so hexBin.

3. line 51 - 68

**var hex** - a map, binary number to hexadecimal char  
'0000' - '0', ..., '1111' - 'F'

4. line 70 - line 75

function getBinaryArray

### Actionable item:

1. Learn the code, write a C # version using exactly ideas!

Spend 30 minutes to do the workout!

2. Go over excellent code 30 minutes, save time to write code; Julia tries to figure out best design.  
10 hours hard work -> 30 minutes work! (Learn from best student in CMU? :-)

1. <https://www.hackerrank.com/contests/june-world-codesprint/challenges/aorb>
  2. <https://gist.github.com/jianminchen/b5ce95c07e3dd027e3753b8c81f00c74>
  3. <https://gist.github.com/jianminchen/2f6b252009868ebcf4d0449aafa01b30>
- 

### **AorB - HackerRank - C++ code study (2016-07-03 19:01)**

July 3, 2016

Study C++ code submission on HackerRank - AorB:

1.  
[1]<https://gist.github.com/jianminchen/4a608ffd202433f9be264c392f1bd75a>
- 2  
[2]<https://gist.github.com/jianminchen/d8879283c292454e2638444328633f09>
3.  
[3]<https://gist.github.com/jianminchen/d65f78caec4c8953949639b70dbdf192>
4.  
[4]<https://gist.github.com/jianminchen/9e72d35b50dc570a948e6a308d1c2e55>

#### **Notes to study code #1:**

1. Read string class in C++:  
<http://www.cplusplus.com/reference/string/string/string/>  
std::string::string  
fill(6) string(size\_t n, char c)  
fill constructor

Fills the string with n consecutive copies of character c.

2. std::string::c\_str  
[http://www.cplusplus.com/reference/string/string/c\\_str/](http://www.cplusplus.com/reference/string/string/c_str/)

3. hexmap[128]

ASCII table from 0 -128  
<http://www.asciitable.com/>

#### 4. reverse in C++ - function template

<http://www.cplusplus.com/reference/algorithm/reverse/>

#### 5. write string to stdout

<http://www.cplusplus.com/reference/cstdio/puts/>

[http://stackoverflow.com/questions/25311011/how-does-include-bits-std c-h-work-in-c](http://stackoverflow.com/questions/25311011/how-does-include-bits-std-c-h-work-in-c)

#### statistics:

1. over 300 submission scores 50/50 - Choose 50 submission to study

blog reading:

[5]<https://zhuanlan.zhihu.com/qinchao>

1. <https://gist.github.com/jianminchen/4a608ffd202433f9be264c392f1bd75a>
  2. <https://gist.github.com/jianminchen/d8879283c292454e2638444328633f09>
  3. <https://gist.github.com/jianminchen/d65f78caec4c8953949639b70dbdf192>
  4. <https://gist.github.com/jianminchen/9e72d35b50dc570a948e6a308d1c2e55>
  5. <https://zhuanlan.zhihu.com/qinchao>
- 

#### Facebook code lab - daily coding practice (2016-07-05 21:35)

July 5, 2016

Spend 30 minutes a time, choose to work on facebook code lab.

Julia's new favorite place to get training daily:

[1]<https://codelab.interviewbit.com/>

Try to write code without compiler error, executable code.

Favorite practice:

1. GCD - great common denominator

[2]<https://gist.github.com/jianminchen/631086210db7c41fcdda677c3aaf726c>

Things to work on: No run-time exception

line 47 - 48

if

(b ==

606

0

)

// bug001 - forget to check if(b==0) first writing, not reaching facebook coding requirement!

return a;

**// bug001 - forget to check if(b==0) first writing!**

2. is same tree

[3]<https://gist.github.com/jianminchen/2c55d67226b86d38f4158160801b82ff>

3. Number 1 bits

[4]<https://gist.github.com/jianminchen/d2f6022094fdb41fb7e7cae8b30c725>

things to work on: No compiler's help

**line 37: a = a >>**

**1**

;

**// bug001 - writing, forget**

**"a =" in first writing, need compiler's help**

Blog reading:

[5]<https://zhuanlan.zhihu.com/p/20941886?refer=qinchao>

DDD - Debugger Driven Development

PDD - Print Driven Development

BDD - Bug-Driven Development

RDD - Rat-Race-Game-Driven Development

DDD - Deadline-Driven Development

1. <https://codelab.interviewbit.com/>

2. <https://gist.github.com/jianminchen/631086210db7c41fcdda677c3aaf726c>

3. <https://gist.github.com/jianminchen/2c55d67226b86d38f4158160801b82ff>

4. <https://gist.github.com/jianminchen/d2f6022094fdb41fb7e7cae8b30c725>

5. <https://zhuanlan.zhihu.com/p/20941886?refer=qinchao>

## Leetcode 130: surrounded region (2016-07-06 21:41)

July 6, 2016

Work on Leetcode 130: surrounded region

Leetcode 130 surrounded region

Study the solution discussions - very detail discussion, comparison DFS/timeout/stack issues, and then BFS solution, ideas to approach the problem.

1. code study 1: C #

[1]<http://www.cnblogs.com/higerzhang/p/4149040.html>

Julia's C # practice:

Will post here very soon!

2. Code study 2: C++

C++ code: read C++ code

[2]<http://neuzxy.github.io/2016/03/14/leetcode-130.html>

Things learned in the code:

1. The description of ideas to solve the problem:

start from all four edges outside most, from each node 'O', work on BFS, mark '+' for those node with 'O', and then, set those not '+', but 'O' - whereas those are not accessible, surrounded by 'X' - set as 'X'.

2. learn typedef pair<int, int> state \_t;

C++ solution

[3]<https://ask.julyedu.com/question/164>

3. Read every blog from the author: The code is with good quality and some excellent sharing:

[4]<http://xiaoyaoworm.com/blog/>

To be continued!

1. <http://www.cnblogs.com/higerzhang/p/4149040.html>

2. <http://neuzxy.github.io/2016/03/14/leetcode-130.html>

3. <https://ask.julyedu.com/question/164>

4. <http://xiaoyaoworm.com/blog/>

---

## Tennis Legends - hingis tactics (2016-07-06 22:31)

July 6, 2016

Learn from my favorite tennis legends - hingis:

[1]<http://baseline.tennis.com/article/59052/learning-legends-hingis-tactics>



Julia works on her tennis skills so hard, she turns the boring weight control project into fascinating sports hobby - chasing tennis tournament, stars, watching hundreds hours tennis matches, and then practice on tennis courts hundreds of hours.

Julia learns from tennis sport:

Hard working,  
Very patient  
Work on drills,  
Focus on training.

She is transferring her sports skills and trains herself on algorithm, problem solving, system design.

Inspired by Wimbledon? Former No.1 Martina Hingis shares her fitness secrets  
[2]<https://t.co/HpSD9UnuAG> [3][pic.twitter.com/UikZ7mlHiB](https://pic.twitter.com/UikZ7mlHiB)  
— Red Magazine (@RedMagDaily) [4]June 27, 2016

Blog reading:

[5]<https://www.linkedin.com/pulse/how-i-learned-importance-self-service-analyticsand-found-elissa-fink?trk=prof-post>

[6]<http://qz.com/525496/done-what-a-recruiter-sees-on-your-resume-at-first-glance/>

[7]<https://hbr.org/2015/12/how-to-spot-a-bad-boss-during-an-interview>

[8]<https://www.linkedin.com/pulse/3-tips-getting-started-new-boss-jeffrey-kelly?trk=prof-post>

1. <http://baseline.tennis.com/article/59052/learning-legends-hingis-tactics>
2. <https://t.co/HpSD9UnuAG>
3. <https://t.co/UikZ7mlHiB>
4. <https://twitter.com/RedMagDaily/status/747468541583101952>
5. <https://www.linkedin.com/pulse/how-i-learned-importance-self-service-analyticsand-found-elissa-fink?trk=prof-post>
6. <http://qz.com/525496/done-what-a-recruiter-sees-on-your-resume-at-first-glance/>
7. <https://hbr.org/2015/12/how-to-spot-a-bad-boss-during-an-interview>
8. <https://www.linkedin.com/pulse/3-tips-getting-started-new-boss-jeffrey-kelly?trk=prof-post>

---

## Encouraging story from tennis player - World No. 50 Elena Vesnina - Wimbledon 2016 (2016-07-06 23:02)

July 6, 2016

Enjoy the Wimbledon and also learn the sports - how hard people work, no matter age, seeded or unseeded.

## Here is the thought process of the player - Elena Vesnina:

It is a far cry from her struggles in Melbourne, or even last year's Wimbledon campaign, which ended in the first round. So what changed?

"Self-belief," said the 29-year-old. "Positive thinking. I was not putting any pressure on myself going on the court – I know I'm in good shape, I'm playing good."

"I was not thinking about my draw. I didn't see who I was playing next round. I was trying to enjoy myself on the court first of all, because I'm here at Wimbledon – it's one of the best tennis tournaments in history, with the best players in the world. I have to enjoy myself here."

"You really appreciate more what you have now. You really enjoy what you're doing. I love playing tennis. It's the best thing what can happen with me. I'm really enjoying my time on the court, and off the court as well."

"I had really difficult beginning of the year, end of the year: I dropped out of the top 100, I was playing all tournaments starting from qualification. I had a lot of matches under my belt. It was not easy, to be honest, because I was in top 30, then I was like 120 or something."

"I'm really happy that it didn't break me up. I think the difficult times – every single player has to go through it because it makes you better. It makes you stronger."

### Julia's comment:

**" The difficult times – every single player has to go through it because it makes you better. It makes you stronger "**

Memorize the sentence!

Ranking: 50

Previous Slam semis: 0[1]@EVesnina001 emerges as [2] #Wimbledon surprise package:  
[3]<https://t.co/V0oyxB3XqK> [4][pic.twitter.com/8sHI8RLmVg](https://t.co/V0oyxB3XqK)

— Wimbledon (@Wimbledon) [5]July 5, 2016

[6][http://www.wimbledon.com/en\\_GB/news/articles/2016-07-05/unseeded\\_vesnina\\_reaches\\_first\\_grand\\_slam\\_semifinal.html](http://www.wimbledon.com/en_GB/news/articles/2016-07-05/unseeded_vesnina_reaches_first_grand_slam_semifinal.html)

"Images she'll remember for the rest of her life"

World No.50 Elena Vesnina celebrates her breakthrough moment [7]<https://t.co/XFnGTeyusO>

— Wimbledon (@Wimbledon) [8]July 5, 2016

Reading blog:

[9]<http://parentingmentallytoughtennis.com/?p=185>

1. <https://twitter.com/EVesnina001>

2. <https://twitter.com/hashtag/Wimbledon?src=hash>

3. <https://t.co/V0oyxB3XqK>
  4. <https://t.co/8sHl8RLmVg>
  5. <https://twitter.com/Wimbledon/status/750433083623501824>
  6. [http://www.wimbledon.com/en\\_GB/news/articles/2016-07-05/unseeded\\_vesnina\\_reaches\\_first\\_grand\\_slam\\_semifinal.html](http://www.wimbledon.com/en_GB/news/articles/2016-07-05/unseeded_vesnina_reaches_first_grand_slam_semifinal.html)
  7. <https://t.co/XFnGTeyus0>
  8. <https://twitter.com/Wimbledon/status/750352576860790784>
  9. <http://parentingmentallytoughtennis.com/?p=185>
- 

## String primitive - JavaScript study (2016-07-07 20:17)

July 7, 2016

### Motivation talk:

Given an array, for example, ['a','b','c','e'], insert 'd' before 'e' in the array. Thinking in JavaScript.

1. **Do not know JavaScript splice API**, write a loop;

2. **Know JavaScript splice API**,

```
var arr = ['a','b','c','e'];
```

```
arr.splice(3,0,'d');
```

3. **Know JavaScript splice API, but careless counting error, 3 goes to 2**

```
var arr = ['a','b','c','e'];
```

```
arr.splice(2,0,'d');
```

Assuming that the array is longer than length 3, and also need to remove some elements before insertion. Counting error is a reasonable mistake.

### Consequence:

Take extra 5 - 10 minutes to find the issue.

### 4. Ideal solution:

Use indexOf to find 'e' position, and then, call splice next.

```
var arr = ['a','b','c','e'];
```

```
arr.splice(arr.indexOf('e'),0,'d');
```

So, the programmer can be lazy, no counting, just ask indexOf API's help, and then, ask splice API's help. Get smart, invest time to get familiar with all APIs, then, it is the game time.

Start to work on JavaScript string reference:

### Target:

1. Memorize all APIs;

2. Write down learning process - notes, blogs to read, questions to ask myself, measurement of progress etc.

**JavaScript String reference** (30 minutes a time to study -Aug.2, 2016, )

[1][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String)

### Properties:

string.prototype

`string.length`

**Methods:**

Exclude deprecated methods, try to remember all the following methods: 34 methods

`String.fromCharCode()`

`String.fromCodePoint()`

`String.prototype.anchor()`

`String.prototype.charAt()`

`String.prototype.codePointAt()`

`String.prototype.concat()`

`String.prototype.endsWith()`

`String.prototype.includes()`

`String.prototype.indexOf()`

`String.prototype.lastIndexOf()`

`String.prototype.link()`

`String.prototype.localeCompare()`

`String.prototype.match()`

`String.prototype.normalize()`

`String.prototype.repeat()`

`String.prototype.replace()`

`String.prototype.search()`

`String.prototype.slice()`

`String.prototype.split()`

`String.prototype.startsWith()`

`String.prototype.normalize()`

`String.prototype.substr()`

`String.prototype.substring()`

`String.prototype.toLocaleLowerCase()`

`String.prototype.toLocaleUpperCase()`

`String.prototype.toLowerCase()`

`String.prototype.toString()`

`String.prototype.toUpperCase()`

`String.prototype.trim()`

`String.prototype.trimLeft()`

`String.prototype.trimRight()`

`String.prototype.valueOf()`

String.prototype[ @@iterator]()

String.raw()

**July 7, 2016 - More than 60 minutes reading - get into the detail - "wild wild west", JavaScript coding is such a challenging one.**

[2][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/substring](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/substring)

substring

data descriptor/ accessor descriptor

In order to understand string - JavaScript, go through a lot of topics:

1. defineProperty

2. template literals

...

10 minutes reading of define property

[3][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Object/defineProperty](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/defineProperty)

string literal

global object directly

Template literals

**back-tick ( ' ' )**

Expression interpolation

syntactic sugar making substitutions like this more readable

tagged template literal - more advanced form of template literals

10 - 15 minutes reading of template literals

[4][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Template\\_literals](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Template_literals)

### August 2, 2016 30 minutes+

// August 2, 2016 - read JavaScript 0 Standard built-in object > string - 30 minutes reading

1. [5][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/undefined](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/undefined)

2. [6][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/codePointAt](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/codePointAt)

read polyfill - read the definition of the function is fun:

charCodeAt()

defineProperty

Number()

TypeError()

3. [7][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/concat](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/concat)

4. [8][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/endsWith](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/endsWith)

5. [9][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/includes](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/includes)

6. [10][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/lastIndexOf](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/lastIndexOf)

syntax: str.lastIndexOf(searchValue[, fromIndex])

optional argument: fromIndex

It can be any integer. Default value is +Infinity. If fromIndex >= str.length, the whole string is searched. If fromIndex < 0, the behavior will be the same as if it would be 0.

7. read 10 minutes

[11][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/match](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/match)

Julia's practice - match function - html file

[12]<https://gist.github.com/jianminchen/528872aa8853b22151325990506d60fc>

8. substring function

[13][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/substring](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/substring)

Julia's practice:

[14]<https://gist.github.com/jianminchen/941a13f0283e8063cf8207715e19e4ac>

9. split function

split function -

[15][https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/split](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/split)

Julia's practice - html

[16]<https://gist.github.com/jianminchen/45806800342101f3b4af7ac7c8ca8efa>

1. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String)
  2. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/substring](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/substring)
  3. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Object/defineProperty](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/defineProperty)
  4. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Template\\_literals](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Template_literals)
  5. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/undefined](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/undefined)
  6. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/codePointAt](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/codePointAt)
  7. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/concat](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/concat)
  8. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/endsWith](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/endsWith)
  9. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/includes](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/includes)
  10. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/lastIndexOf](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/lastIndexOf)
  11. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/match](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/match)
  12. <https://gist.github.com/jianminchen/528872aa8853b22151325990506d60fc>
  13. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/substring](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/substring)
  14. <https://gist.github.com/jianminchen/941a13f0283e8063cf8207715e19e4ac>
  15. [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/String/split](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/split)
  16. <https://gist.github.com/jianminchen/45806800342101f3b4af7ac7c8ca8efa>
- 

## pluralsight - Building Highly Scalable Web Applications in Azure (2016-07-07 22:44)

July 7, 2016

2 hours courses on pluralsight.com. The course is very well prepared, great teaching of system design, a lot of discussions are new to Julia.

Lecturer's website:

[1]<http://www.edinkapic.com/2015/09/learn-building-highly-scalable-applications-in-azure.html>

### Topics included:

storage locking removal

caching

asynchronous calls or non-relational data storage

[2][https://www.youtube.com/watch?time\\_continue=79&v=muiGJ2gK9XI](https://www.youtube.com/watch?time_continue=79&v=muiGJ2gK9XI)

1. <http://www.edinkapic.com/2015/09/learn-building-highly-scalable-applications-in-azure.html>

2. [https://www.youtube.com/watch?time\\_continue=79&v=muiGJ2gK9XI](https://www.youtube.com/watch?time_continue=79&v=muiGJ2gK9XI)

---

## A small talk on Learning Style Change - coding, system design (2016-07-09 00:43)

July 9, 2016

Try to select a topic to work on, 20 minutes, and see how far Julia can go on the topic. Later, she can add more on her research.



Take easy, choose easy topic to work on, choose learning style change as a topic.

Start to work on new style of learning:

1. Take courses on **pluralsight.com (starting from Dec. 2015)**
2. Read **Leetcode solutions** from **last 2-3 years blogs** - written by a younger generation, compare different languages - Java, C++, JavaScript, C ( **starting from Feb. 2015** )
3. Work on **HackerRank** , and go over **hundreds of** submissions, try to find best solutions and analyze the solution. ( **starting from Feb. 2016** )
4. Every day write some code/ read some code/ study APIs/ Try to memorize something about APIs ( **starting from Feb. 2015** )
5. Try different ideas to explain the algorithm problem solving, understand learning process/ mental process to solve unseen problems. ( **starting from May. 2016** )
6. Work on algorithm problems everyday, discuss it on blogs, and read other people's code; keep to find best/ top talent through blogs reading (code in C/C++/Java/JavaScript). Appreciate people sharing through blogs, and then, Julia also likes to join the community to write blogs.

Will be back to work on more on this topic.

---

### **A small research about frugality (2016-07-09 17:55)**

July 9, 2016

Frugality - Try to save time, save money, and save the resources.

How to build the habit to think in frugality?

How to measure the progress of good thinking in frugality?

What to do to prevent impulse shopping or other things?

How to save time, focus on important things first?

...

A list can be long, first of all, memorize the verse:

Proverbs 28:6

Better to be a poor person who has integrity

than to be rich and double-dealing.

Lessons she learned:

1. Spent a lot of time to debug/maintain the code in last 6 years, if she learns to focus on writing bug-free code in the first writing, that is a lot of savings on time. It is not frugal to write code, without a few of rewriting followed, some refinements, following some code standards, following certain design principles closely;

That time goes to study of algorithm, problem solving, code standards, etc.

2. Spend a lot of time reading/ studying, without practice (write code/ write blogs/ help others at work), or review later on, the knowledge is easy to come/ easy to go. Not focusing on the most important thing - data structure, algorithm, math, problem solving, try to overcome the limitation of cognitive ability etc.

3. Should work on time management / planning / decision making. Good planning saves a lot of time later.

---

### **Business intelligent study (2016-07-09 22:46)**

July 9, 2016

Julia spent over years to work on Microsoft Excel, powerpivot table and vertipaq engine to do data analysis. Now, she likes to do some study on Tableau:

She could not recall the engine name - vertipaq, so she googled and then found this article -

1. [1]<http://searchsqlserver.techtarget.com/feature/Importing-SQL-Server-data-into-PowerPivot-for-Excel>

### **Tableau - Spend a few hours to read and then learn something about Tableau products:**

#### **Read blogs:**

2. [2]<http://www.36dsj.com/archives/42081>

Take some notes here from the blog:

Business intelligence (BI)

a list of features: "A B C D E S"

1. A = Automation

To collect, analyze data, it takes time to work on

2. B = Big Data

Hadoop based

Tableau supports Hadoop, Spark.

3. C = Complex Analysis - Include Machine Learning, predictive modeling

4. D = Data Integration/ Connection

5. E = Easy-to-use.

6. W = Workflow Embedded

3.

[3]<http://www.36dsj.com/archives/46378>

4. Learn how to read a graph using Tableau:

Sankey Diagram:

Others call DecisionTree/ Flow Diagram/ Alluvial Diagram

[4]<http://www.36dsj.com/archives/22337>

[5]<http://www.36dsj.com/archives/16647>

5. Podcast - Tableau - Vancouver

[6][http://www.vancouvertechpodcast.ca/episodes/41291-episode-33-tableau-s ofware](http://www.vancouvertechpodcast.ca/episodes/41291-episode-33-tableau-s%20software)

Last 20 minutes, Jesse Calderon talked about everything about Tableau - how team works together, how new project is organized etc.

6. A lot of great insights - how to think about venture capital/ product/ problem solving?

[7]<https://www.youtube.com/watch?v=d8o20TP7270>

7. Tableau - Visual Analysis

[8]<https://www.youtube.com/watch?v=hoAbjerDak4>

Podcast: Microsoft Canada project manager talk:

[9][http://www.vancouvertechpodcast.ca/episodes/40343-episode-31-microsoft #0:27:53](http://www.vancouvertechpodcast.ca/episodes/40343-episode-31-microsoft%20#0:27:53)

1. <http://searchsqlserver.techtarget.com/feature/Importing-SQL-Server-data-into-PowerPivot-for-Excel>

2. <http://www.36dsj.com/archives/42081>

3. <http://www.36dsj.com/archives/46378>

4. <http://www.36dsj.com/archives/22337>

5. <http://www.36dsj.com/archives/16647>

6. <http://www.vancouvertechpodcast.ca/episodes/41291-episode-33-tableau-software>

7. <https://www.youtube.com/watch?v=d8o20TP7270>

8. <https://www.youtube.com/watch?v=hoAbjerDak4>

9. <http://www.vancouvertechpodcast.ca/episodes/40343-episode-31-microsoft#0:27:53>

---

## Leetcode 56: Merge Intervals (2016-07-10 00:24)

July 10, 2016

## Leetcode 56: Merge Intervals

study the code:

[1]<http://xiaoyaoworm.com/blog/2016/06/27/%E6%96%B0leetcode56-merge-intervals/>

Write down C # code - 20 minutes workout later.

Have some drawing to analyze the problem.

blog reading:

[2]<http://www.canadianbusiness.com/leadership/office-space/microsoft-canada-vancouver/>

[3]<http://www.canadianbusiness.com/leadership/office-space/jordan-banks-facebook-canada/image/2/>

1. <http://xiaoyaoworm.com/blog/2016/06/27/%E6%96%B0leetcode56-merge-intervals/>

2. <http://www.canadianbusiness.com/leadership/office-space/microsoft-canada-vancouver/>

3. <http://www.canadianbusiness.com/leadership/office-space/jordan-banks-facebook-canada/image/2/>

---

## CSS learning / forgetting - a better cycle (2016-07-10 15:58)

July 10, 2016

A few days ago, Julia could not recall CSS ~~text-decoration~~ text-decoration rule. She likes to find underline a word using a CSS property.

So, she tried to **fail better** next time when she forgets CSS - how she can do that?

Answer: Try the following

1. Use her memory - push herself to memorize a **[1]CSS cheat sheet in a 2 weeks/ 30 minutes a day** , and then, she will read more about CSS properties as a daily routine.

2. More cheat sheet about CSS and Regular Express:

[2]<https://codingsec.net/2016/04/complete-cheatsheet-csscascading-style-sheet/>

3. [3]<https://www.cheatography.com/davechild/cheat-sheets/css2/>

4. Read document 30 minutes a time:

[4]<https://developer.mozilla.org/en-US/docs/Web/CSS>

[5][https://developer.mozilla.org/en/docs/Web/CSS/Attribute\\_selectors](https://developer.mozilla.org/en/docs/Web/CSS/Attribute_selectors)

CSS Selectors

Basic Selectors

Type selectors

Class selectors

ID selectors

Universal selectors

Attribute selectors (July 14, 2016 30 minutes - go over , |, ^, \$, \*, i)

, |, ^, \$, \*, i

one of which, value or value-, prefix, suffix, contains, case insensitive,

Combinators

Adjacent sibling selectors

General sibling selectors

Child selectors

Descendant selectors

Pseudo-classes (36)

:active

:checked

:default

:disabled

:empty

:enabled

:first

:first-child

:first-of-type

:focus

:hover

:indeterminate

:in-range

:invalid

:lang

:last-child

:last-of-type

:left

:link

:not()

:nth-child

:nth-last-child

:nth-last-of-type

:nth-of-type

:only-child

:only-of-type

:optional

:out-of-range

:read-only  
:read-write

:required  
:right  
:root  
:target  
:valid

:valid  
Julia had hard time to memorize 36 pseudo classes, get some reading:  
Pseudo classes:

[6]<https://www.smashingmagazine.com/2016/05/an-ultimate-guide-to-css-pseudo-classes-and-pseudo-elements/>

[7]<https://css-tricks.com/pseudo-class-selectors/>

5. CSS courses on pluralsight.com

CSS position

Introduction website layout

[8]<http://app.pluralsight.com/author/susan-simkins>

CSS in-depth 6 hour course

(July25, 30mins/ Specificity 10m)

[9]<http://app.pluralsight.com/author/estelle-weyl>

#### - Past Experience -

In 2013,

Her favorite book: [10]Head first Html with CSS & Xhtml

Her favorite CSS learning starting from 2013: She found out that she could not read CSS code without knowing CSS selectors.

In 2013, Julia tried to memorize all selectors - she worked on the drills to memorize all of them, write on papers etc. Show the paper writing to celebrate over 36 months - dated on May 9, 2013 - CSS learning handscript .  
Post a picture here.

Her favorite CSS blog about CSS selector.

[11]<http://code.tutsplus.com/tutorials/the-30-css-selectors-you-must-memorize-net-16048>

#### - Motivation talk -

Share a favorite verse from Julia's favorite professional ATP player:

[12]<http://heavy.com/sports/2015/06/stan-wawrinka-tattoo-say-mean-daughter-wife-french-open/>

[13]<https://www.theguardian.com/sport/2014/jun/09/stan-wawrinka-queens-rafael-nadal-novak-djokovic>

Stan Wawrinka is one of the most intense, hard-working players on the ATP Tour; dedicating himself to every point, giving everything in the quest for victory, no matter the cost.

And Wawrinka's ink, unsurprisingly, reflects his tennis weltanschauung. On the inside of his left arm are the words of poet Samuel Beckett: Ever tried. Ever failed. No matter. Try Again. Fail again. Fail better.

SF!! [14] stanwawrinka@[15] #RG16 [16] #RolandGarros [17]pic.twitter.com/vrVeNcK78y  
— Erika Tanaka (@ErikaTanaka\_) [18]June 1, 2016

1. <https://www.smashingmagazine.com/wp-content/uploads/images/css3-cheat-sheet/css3-cheat-sheet.pdf>
2. <https://codingsec.net/2016/04/complete-cheatsheet-csscascading-style-sheet/>
3. <https://www.cheatography.com/davechild/cheat-sheets/css2/>
4. <https://developer.mozilla.org/en-US/docs/Web/CSS>
5. [https://developer.mozilla.org/en/docs/Web/CSS/Attribute\\_selectors](https://developer.mozilla.org/en/docs/Web/CSS/Attribute_selectors)
6. <https://www.smashingmagazine.com/2016/05/an-ultimate-guide-to-css-pseudo-classes-and-pseudo-elements/>
7. <https://css-tricks.com/pseudo-class-selectors/>
8. <http://app.pluralsight.com/author/susan-simkins>
9. <http://app.pluralsight.com/author/estelle-weyl>
10. <http://www.headfirstlabs.com/books/hfhtml/>
11. <http://code.tutsplus.com/tutorials/the-30-css-selectors-you-must-memorize--net-16048>
12. <http://heavy.com/sports/2015/06/stan-wawrinka-tattoo-say-mean-daughter-wife-french-open/>
13. <https://www.theguardian.com/sport/2014/jun/09/stan-wawrinka-queens-rafael-nadal-novak-djokovic>
14. <https://twitter.com/stanwawrinka>
15. <https://twitter.com/hashtag/RG16?src=hash>
16. <https://twitter.com/hashtag/RolandGarros?src=hash>
17. <https://t.co/vrVeNcK78y>
18. [https://twitter.com/ErikaTanaka\\_/status/738065559401697281](https://twitter.com/ErikaTanaka_/status/738065559401697281)

---

## Data Analysis Fundamentals with Tableau - pluralsight.com (2016-07-10 21:09)

July 10, 2016

Plan to spend 4 hours to study the course on pluralsight.com - "Data Analysis Fundamentals with Tableau".

[1]<https://www.pluralsight.com/authors/ben-sullins>

Watch first 2 hours while going through twitter account of instructor - data geek - get ideas how / what data geek is so good at presentation through social media.

People working on BI are in general much better to do presentation.

Show Me The Data!

Demo: Mapping

1. <https://www.pluralsight.com/authors/ben-sullins>

---

## Leetcode 142: Linked List Cycle II - two examples to show the idea (2016-07-14 22:04)

July 14, 2016

Work on facebook code lab - Linked List Cycle, same question as Leetcode 142.

Problem statement:

Given a linked list, return the node where the cycle begins. If there is no cycle, return

null .

Note: Do not modify the linked list.

Follow up :

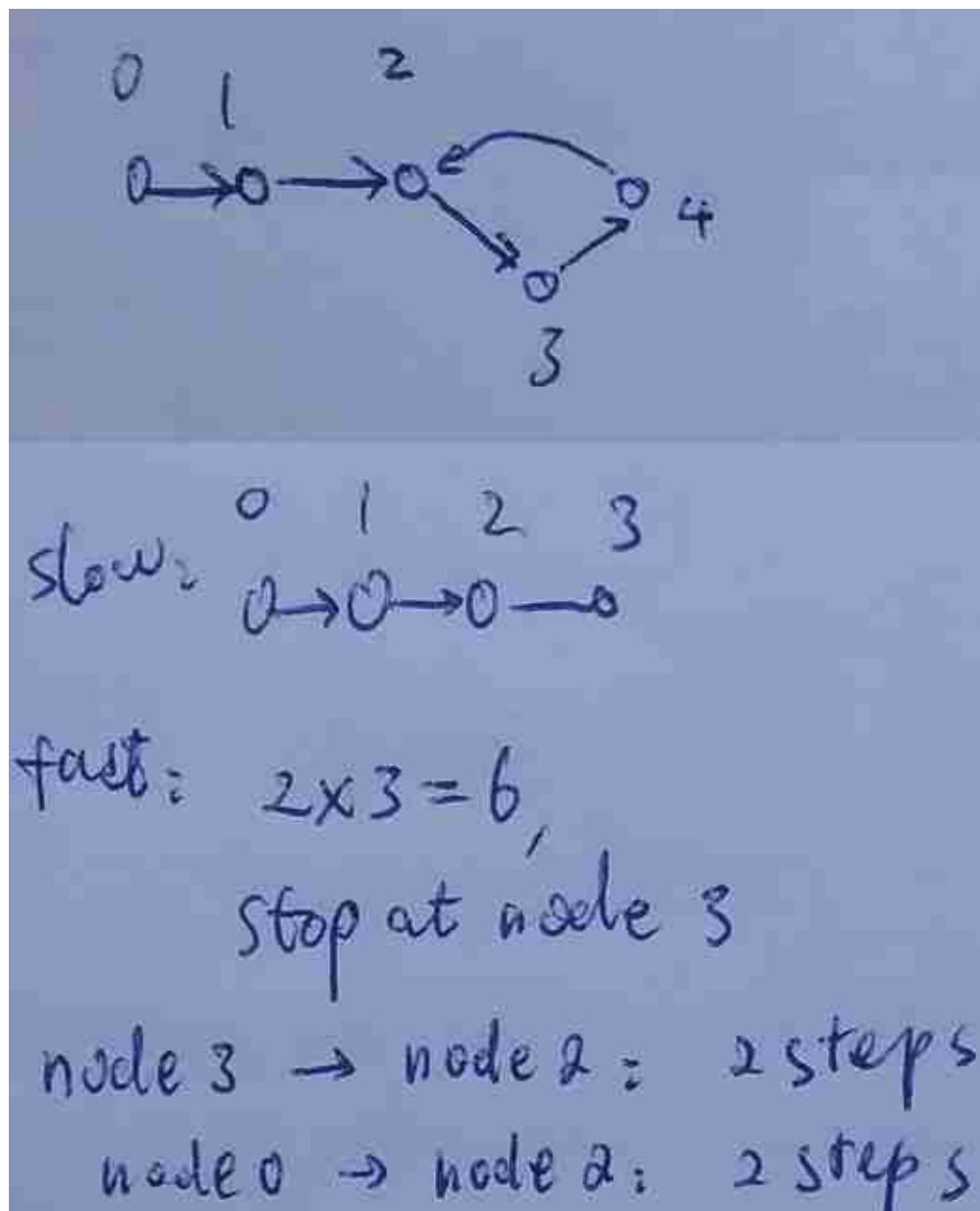
Can you solve it without using extra space?

Work on two simple examples, and then, figure out the solution first:

**Example 1:**

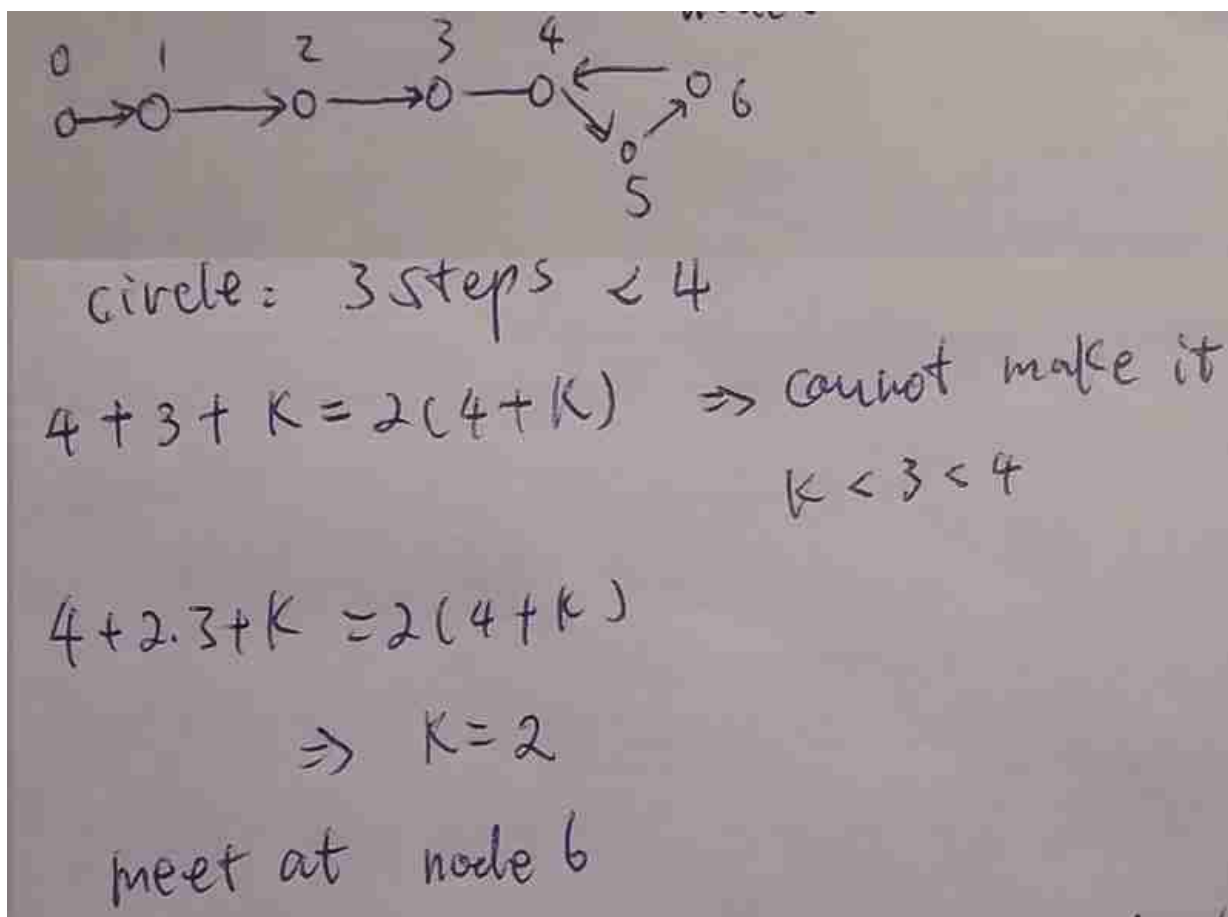
624





[1]

Example 2:



First, find the node two pointers meet - one slow runner, one step a time; one fast runner, two steps a time. If there is a cycle, two nodes will meet.

And then, start from meeting place, one node starts from node 0, slow node continues from meeting place, then, two nodes will meet at the beginning of cycle.

#### Facts about the practice :

1. Julia worked on this problem before. But she has to figure out the solution again.

2. Spent over 20+ minutes and still confused about the solution.

two distance values:

meeting place -> cycle starting node

cycle starting node -> meeting place

Use the first one, not second one. But, Julia was trying to figure out the second one.

Denote d1 as starting node to cycle starting node

Denote d2 as cycle length

Discuss two cases:

$d1 > d2$ ,

$d1 \leq d2$ ,

and when the two pointers meet, the fast node may go through cycle more than once.

The formula is not deterministic, so it is better to go through the simple example, and then, make a guess.

3. Google the blog about the solution:

[2][https://siddontang.gitbooks.io/leetcode-solution/content/linked\\_list/linked\\_list\\_cycle.html](https://siddontang.gitbooks.io/leetcode-solution/content/linked_list/linked_list_cycle.html)

July 15, 2016

Read more blogs:

The analysis has some issues:

[3]<http://www.cnblogs.com/hiddenfox/p/3408931.html>

[4]<http://www.cnblogs.com/wuyuegb2312/p/3183214.html>

[5]<http://yucoding.blogspot.ca/2013/12/leetcode-question-linked-list-cycle-ii.html>

1. [https://1.bp.blogspot.com/-yjmUVldDxSY/V4h5MHa-2BI/AAAAAAAAAGN4/hQZ90q\\_ZXo8gyVHTC6X1xSSJrivPJIXywCLcB/s1600/LinkedList\\_CaseII.jpg](https://1.bp.blogspot.com/-yjmUVldDxSY/V4h5MHa-2BI/AAAAAAAAAGN4/hQZ90q_ZXo8gyVHTC6X1xSSJrivPJIXywCLcB/s1600/LinkedList_CaseII.jpg)

2. [https://siddontang.gitbooks.io/leetcode-solution/content/linked\\_list/linked\\_list\\_cycle.html](https://siddontang.gitbooks.io/leetcode-solution/content/linked_list/linked_list_cycle.html)

3. <http://www.cnblogs.com/hiddenfox/p/3408931.html>

4. <http://www.cnblogs.com/wuyuegb2312/p/3183214.html>

5. <http://yucoding.blogspot.ca/2013/12/leetcode-question-linked-list-cycle-ii.html>

---

## Leetcode 142: Linked List Cycle II - first practice (2016-07-14 23:17)

July 14, 2016

Continue after first blog on this problem:

[1]<http://juliachencoding.blogspot.ca/2016/07/leetcode-142-linked-list-cycle-ii.html>

Java solution written in facebook code lab:

[2]<https://gist.github.com/jianminchen/cc0f97525e3eb68db23f4a07ec0b7e90>

The code failed to pass some test cases.

**Highlight problems on the first writing:**

```

16 public ListNode detectCycle(ListNode a) {
17     if(a == null)
18         return null;
19
20     ListNode normalRunner = a;
21     ListNode doubleRunner = a;
22
23     while(normalRunner != null
24         && doubleRunner != null
25         && normalRunner != doubleRunner)
26     {
27         normalRunner = normalRunner.next;
28         if(doubleRunner.next == null)
29             return null;
30         else
31             doubleRunner = doubleRunner.next.next;
32     }
33
34     //if(normalRunner == null) // bug001
35     if(normalRunner == null || doubleRunner == null)
36         return null;
37
38     ListNode thirdRunner = a;
39
40     while(thirdRunner != null
41         && normalRunner != null
42         && thirdRunner != normalRunner)
43     {
44         thirdRunner = thirdRunner.next;
45         normalRunner = normalRunner.next;
46     }

```

1. line 20,  
normalRunner - one step a time, the variable name is too long; slow will be better.

2. line 21,  
doubleRunner - the variable name is not so accurate; fast will be better.

**problem 1** : base case checking:

line 17: if(a == null)

return null

if the list is null or only has 1 node, then no cycle.

**problem 2** : doubleRunner and normalRunner is in the same list, and doubleRunner is ahead of normalRunner if there is no cycle; and if there is a cycle, two runners will run forever. In both cases, only need to check doubleRunner != null, if it is true, then normalRunner != null will be true as well.

In short, doubleRunner != null => normalRunner != null

**The while checking has duplicate checking!**

**problem 3 :**

line 24: doubleRunner is not null <= **it is duplicated with base case checking (line 17)!**

While loop line 24, doubleRunner != null is duplicate with base case checking.

While loop design:

Let us check fast node can move 2 steps, next two nodes are not null;

if slow and fast node meets, then break the loop. <- **make the logic checking positive, not negative (line 25)**

**problem 4:**

while loop is not designed very well - easy to make bugs, hidden bugs.

line 25: negative checking - make the code more confusing

**problem 5:**

line 34: bug001 - if clause checking problem

line 35: doubleRunner != null => normalRunner != null

so if statement is not optimal

**problem 6:**

line 35: there is a hidden bug here - still cannot figure out.

**problem 7:**

line 38: extra variable - not necessary, reuse the old one, still in the scope.

line 40: not necessary

line 41: not necessary

**Actionable item:**

1. Document the mistakes made in the practice, learn from the experience.
2. Review code and figure out issues without any help.

Could not figure out bugs in the first practice

- > Failed facebook code lab test cases
- > Studied optimal solution
- > Ran optimal solution in facebook code lab
- > passed all test cases
- > then, started to find problems in the first practice
- > wrote down problems one by one

Train insane or remain the same - focus on training!

Train insane or remain the same '[3] #progress [4] #nopainnogain [5] #mondaymotivation  
Séance de 1/2squat aujourd'hui? get vous? [6]pic.twitter.com/K8tGiOXixC

— Kristina Mladenovic (@KikiMladenovic) [7]June 20, 2016

1. <http://juliachencoding.blogspot.ca/2016/07/leetcode-142-linked-list-cycle-ii.html>
  2. <https://gist.github.com/jianminchen/cc0f97525e3eb68db23f4a07ec0b7e90>
  3. <https://twitter.com/hashtag/progress?src=hash>
  4. <https://twitter.com/hashtag/nopainnogain?src=hash>
  5. <https://twitter.com/hashtag/mondaymotivation?src=hash>
  6. <https://t.co/K8tGiOXixC>
  7. <https://twitter.com/KikiMladenovic/status/744887194016452609>
- 

## Leetcode 142: Linked List Cycle II - second practice (2016-07-14 23:57)

July 14, 2016

Second practice:

[1]<https://gist.github.com/jianminchen/65796a60e7320784231bd1883af0d6f8>

```

18     public ListNode detectCycle(ListNode a) {
19         if(a == null || a.next == null) {
20             return null;
21         }
22
23         ListNode fast = a;
24         ListNode slow = a;
25
26         while(fast.next != null && fast.next.next != null) {
27             fast = fast.next.next;
28             slow = slow.next;
29             if(fast == slow) {
30                 slow = a;
31                 while(slow != fast) {
32                     fast = fast.next;
33                     slow = slow.next;
34                 }
35
36                 return slow;
37             }
38         }
39
40         return null;
41     }
42 }

```

Transcript:

Assuming that you have the knowledge how to find the start node in the cycle. It took 20+ minutes to figure out a solution using simple examples, two simple examples are discussed here:

[2]<http://juliachencoding.blogspot.ca/2016/07/leetcode-142-linked-list-cycle-ii.html>

So, base case coding, line 19 - 20, if the node is null or linked list has only one node, no cycle, return null;

Two runners start from start node in the linked list, one moves one step at a time, the fast one moves two steps at a time.

Since the fast node explores the linked list in front of the slow one, only check fast node can make 2 steps move or not.

The while loop - iteration is to determine if the fast can move forward, if it is true, then slow one can move as well.

Line 26, while loop, check fast node next two nodes are not empty nodes.

Line 27, 28: go to next iteration step,

Line 29: check if the slow meets the fast. if it is, find the start node in the cycle:

Line 29 - 37. The idea is to set slow runner to the start node of linked list, and fast/ slow runners both move one step a time until they meet.

One concern is the nested while loop from line 29 - 37, the code can be moved out from the while loop, see the practice III:

[3]<http://juliachencoding.blogspot.ca/2016/07/leetcode-142-linked-list-cycle-ii-third.html>

1. <https://gist.github.com/jianminchen/65796a60e7320784231bd1883af0d6f8>
  2. <http://juliachencoding.blogspot.ca/2016/07/leetcode-142-linked-list-cycle-ii.html>
  3. <http://juliachencoding.blogspot.ca/2016/07/leetcode-142-linked-list-cycle-ii-third.html>
- 

### **Leetcode 142: Linked List Cycle II - third practice (2016-07-14 23:58)**

July 14, 2016

Third practice:

[1]<https://gist.github.com/jianminchen/3b7194dd0f2e3672542046bda3d95d66>

1. <https://gist.github.com/jianminchen/3b7194dd0f2e3672542046bda3d95d66>
- 

### **Sorted Linked List duplicates removal - Facebook code lab (2016-07-16 10:17)**

July 16, 2016

Julia spent over 1 hour to work on this question. So, her first two practices using facebook code lab:

1. Use recursive function - stack overflow issue

[1]<https://gist.github.com/jianminchen/b31347aed2fe140384215156004790a5>

2. Use iteration solution, but run time performance needs to improve (July15, 2016)/ unreachable code, showing different error messages on July 16, 2016

[2]<https://gist.github.com/jianminchen/0318844f35723d6e28d9a90092b2a85c>

Then, Julia spent 20 minutes to think if she should use binary search to expedite the search not distinct node - extra array to help, but then, she checked the code lab.

Recall her previous blog on the linked list:

[3]<http://juliachencoding.blogspot.ca/2016/05/hackerrank-delete-duplicate-value-nodes.html>

(Pass HackerRank test, but fail facebook code lab test? double check)

[4]<https://gist.github.com/jianminchen/9013539e71764a018f3745c514da9d91>



Her favorite solution: (Pass facebook code lab test)

[5]<https://gist.github.com/jianminchen/a13738b4ab32dec8601f29777172209>

Julia likes to talk about the issues of her practice on July 16, 2016:

1. <https://gist.github.com/jianminchen/b31347aed2fe140384215156004790a5>
  2. <https://gist.github.com/jianminchen/0318844f35723d6e28d9a90092b2a85c>
  3. <http://juliachencoding.blogspot.ca/2016/05/hackerrank-delete-duplicate-value-nodes.html>
  4. <https://gist.github.com/jianminchen/9013539e71764a018f3745c514da9d91>
  5. <https://gist.github.com/jianminchen/a13738b4ab32dec8601f29777172209>
- 

## Coding practice talk (2016-07-16 10:32)

July 16, 2016

Spend some time to work on a small research topic this weekend: " **coding practice** ", " **How to conduct rigorous training** ".

Read the article about facebook coding requirement, since Julia spent over 3 hours on facebook code lab and then start to find her weakness to work on.

[1]<https://www.facebook.com/careers/life/preparing-for-your-android-engineering-interview-at-facebook>

- Write a working solution and iterate.

It's better to have a non-optimal but working solution than random fragments of an optimal but unfinished solution. ( **Julia's comment: 10/10** )

- Listen for hints. If your interviewer gives you hints to improve your code, please run with them.
- Prep questions for us in advance. You'll most likely have some time at the end for questions for your interviewer. Some people find it easier to come up with a few questions in advance rather than think of them on the spot.
- Don't worry about memorizing tables of runtimes or API calls. It's always good to know how to figure out approximate runtimes on the fly but the code you write is more important.
- If your solution is getting ugly, step back. Most coding interview questions are designed to have reasonably elegant solutions. If you have festoons of if-else blocks and special cases everywhere, you might be taking the wrong approach. Look for patterns and try to generalize.

( **Julia's comment: 10/10** )

(**Julia's thought: practice more, and also review each practice, write down the issues.** )

How to Prepare:

- Do as many coding questions as you can. Visit [2]Glassdoor, [3]Careercup, [4]Project Euler, or [5]Facebook Code Lab or another site that hosts questions. The idea isn't to see every question, but to become familiar with the pattern of interpreting a question, formulating a solution, and writing an efficient, bug-free program without a compiler.
- Practice on a whiteboard or with pencil and paper.  
Practice under time pressure: coding speed is important. **The more rigorous your training** , the easier you'll find the interviews.
- Go over data structures, algorithms and complexity: Be able to discuss the big-O complexity of your approaches. Don't forget to brush up on your data structures like lists, arrays, hash tables, hash maps, stacks, queues, graphs, trees, heaps. Also sorts, searches, and traversals (BFS, DFS). Also review recursion and iterative approaches.
  - Our typical coding questions aren't phrased as "implement x"; they're "solve this problem." You can pick from a number of approaches. (No one is going to ask you "implement Knuth-Morris-Pratt" or "construct a 2-3-4 tree.")
  - Your reasoning is important. Engineering is all tradeoffs so be able to discuss those.
- Additional reading resources: [6]Cracking the Coding Interview, [7]Introduction to Algorithms,[8]Algorithms in C.

### Learn from the tennis professional player - Milos Raonic -

[9]<http://www.milosraonicofficial.com/about/>

[10][https://www.youtube.com/watch?v=A3Q4IIIL\\_fk](https://www.youtube.com/watch?v=A3Q4IIIL_fk)

Fresh from his first grass court final, [11]@milosraonic is thriving under coach McEnroe.[12] #Wimbledon[13]<https://t.co/AWVQyfUugl>

— Wimbledon (@Wimbledon) [14]June 22, 2016

1. <https://www.facebook.com/careers/life/preparing-for-your-android-engineering-interview-at-facebook>
2. [https://www.facebook.com/l.php?u=https%3A%2F%2Fwww.glassdoor.com%2Ffacebook&h=\\_AQEEjcBk&s=1](https://www.facebook.com/l.php?u=https%3A%2F%2Fwww.glassdoor.com%2Ffacebook&h=_AQEEjcBk&s=1)
3. <http://l.facebook.com/l.php?u=http%3A%2F%2Fwww.careercup.com%2Fpage&h=7AQFDsrGN&s=1>
4. <http://l.facebook.com/l.php?u=http%3A%2F%2Fprojecteuler.net%2Findex.php&h=RAQHTpPIa&s=1>
5. <https://codelab.interviewbit.com/>
6. <http://l.facebook.com/l.php?u=http%3A%2F%2Fwww.amazon.com%2Fdp%2F0984782850%2F&h=CAQE100Dq&s=1>
7. <http://www.amazon.com/dp/0262033844>
8. <http://l.facebook.com/l.php?u=http%3A%2F%2Fwww.amazon.com%2FAlgorithms-Parts-1-5-Bundle-Fundamentals%2Fdp%2F0201756080&h=uAQEOsfWz&s=1>
9. <http://www.milosraonicofficial.com/about/>
10. [https://www.youtube.com/watch?v=A3Q4IIIL\\_fk](https://www.youtube.com/watch?v=A3Q4IIIL_fk)
11. <https://twitter.com/milosraonic>
12. <https://twitter.com/hashtag/Wimbledon?src=hash>
13. <https://t.co/AWVQyfUugI>
14. <https://twitter.com/Wimbledon/status/745697224886083584>

## Bulbs - facebook code lab - five practices (2016-07-16 11:53)

July 16, 2016

Work on facebook code lab - bulbs

N light bulbs are connected by a wire. Each bulb has a switch associated with it, however due to faulty wiring, a switch also changes the state of all the bulbs to the right of current bulb. Given an initial state of all bulbs, find the minimum number of switches you have to press to turn on all the bulbs. You can press the same switch multiple times.

Note : 0 represents the bulb is off and 1 represents the bulb is on.

Example:

Input : [0 1 0 1]

Return : 4

Explanation :

press switch 0 : [1 0 1 0]

press switch 1 : [1 1 0 1]

press switch 2 : [1 1 1 0]

press switch 3 : [1 1 1 1]

### 1. First practice:

Recursive solution - stack overflow issue:

[1]<https://gist.github.com/jianminchen/ae6ab2d83a090b79e161cb4c870d6aa8>

### 2. Second practice: Partial correct - work on time complexity

[2]<https://gist.github.com/jianminchen/fb463d7bfe70022f26fc6e0854f94228>

Here is the error message:

- [3]

Time Complexity

Almost there. Your solution is not well optimized for runtime. Focus your effort in solving the problem in shorter time frame.

- Partially Correct Answer. Make your solution more efficient

### 3. Third practice:

[4]<https://gist.github.com/jianminchen/7093c5dcd52dd72f1ecd06e1d969e648>

Same message, almost there!

### 4. Fourth practice: Correct answer

[5]<https://gist.github.com/jianminchen/7e934805e2bfa9fe4ce79f3991a4b570>

Highlights of change:

1. No change on input argument List<Integer>, no List.remove call, no function call of getOpposite

5. Standard answer provided by author - C++

[6]<https://gist.github.com/jianminchen/7857bba5138d6c72067eddf40f0f5d5b>

Actually, just use one variable - state, and only two states: 0 or 1.

### Actionable Items:

1. Go over Java List Interface, and list all APIs here. Try to memorize all APIs.

2. Go over Java Integer class, and list all APIs here.

### Entertainment notes:

Julia's favorite practice, first, she forced herself to write Java code in the facebook code lab, no C # support; she started to read more about Java List API, and got some coding experience on basic List coding.

And then, she started to challenge herself to figure out step by step to elegant solution. One step a time.

The practices were quite entertaining, Julia likes the big surprise at the end.

1. <https://gist.github.com/jianminchen/ae6ab2d83a090b79e161cb4c870d6aa8>
  2. <https://gist.github.com/jianminchen/fb463d7bfe70022f26fc6e0854f94228>
  3. [https://codelab.interviewbit.com/problems/bulbs/#view-code-error-time\\_complexity](https://codelab.interviewbit.com/problems/bulbs/#view-code-error-time_complexity)
  4. <https://gist.github.com/jianminchen/7093c5dcd52dd72f1ecd06e1d969e648>
  5. <https://gist.github.com/jianminchen/7e934805e2bfa9fe4ce79f3991a4b570>
  6. <https://gist.github.com/jianminchen/7857bba5138d6c72067eddf40f0f5d5b>
- 

## Flatten - facebook code lab - 2 practices (2016-07-16 13:58)

July 16, 2016

Given a binary tree, flatten it to a linked list in-place.

Example :

Given

```
1
 / \
2 5
 / \ \
3 4 6
```

The flattened tree should look like:

```
1
 \
 2
 \
 3
 \
 4
 \
 5
 \
 6
```

Note that the left child of all nodes should be NULL.

First practice: Run time error

[1]<https://gist.github.com/jianminchen/5cad1bd01525e58da40bff23ab541fc7>

Second practice: Pass all test cases!

[2]<https://gist.github.com/jianminchen/8388016355c3e3a393c71b17d508e79b>

```
1
 / \
2  3
 /
4
```

The above tree is serialized by level order traversal, 1 2 3 -1 -1 4 -1 -1 -1

So, that is the reason in the second practice, line ..., line 12, line 34, line 35, discussion of -1 value.

3. Study the iterative solution - editorial solution in C++:

[3]<https://gist.github.com/jianminchen/082118552020666ec5c794d1383b6ae8>

4. Study the Java solution -

[4]<https://gist.github.com/jianminchen/d9f18909417476f7102b5573173336ca>

1. <https://gist.github.com/jianminchen/5cad1bd01525e58da40bff23ab541fc7>
2. <https://gist.github.com/jianminchen/8388016355c3e3a393c71b17d508e79b>
3. <https://gist.github.com/jianminchen/082118552020666ec5c794d1383b6ae8>
4. <https://gist.github.com/jianminchen/d9f18909417476f7102b5573173336ca>

---

## Reverse unsigned 32 bit integer - facebook code lab - 5+ practice (2016-07-16 16:47)

July 16, 2016

Problem statement:

REVBITS

Reverse bits of an 32 bit unsigned integer

Example 1:

x = 0,

```
00000000000000000000000000000000
=> 00000000000000000000000000000000
```

return 0

Example 2:

x = 3,

```
00000000000000000000000000000011
=> 11000000000000000000000000000000
```

return 3221225472

Practice:

Java Practice:

**1st practice:**

code is working, but Java code is using Math.pow(2, index).

[1]<https://gist.github.com/jianminchen/395ccf0277250e5d3d68498e0d2238f1>

**2nd practice: not working, error on 32 bit integer, left most significant bit 1 means negative number.**

[2]<https://gist.github.com/jianminchen/7bfdcd0c1643bf01c2bf6ba7acea6f70>

Sorry, wrong answer. Your program's output doesn't match the expected output. You can try testing your code with **custom input** and try putting **debug statements** in your code.

Your submission failed for the following input:

A : 3

Your function returned the following :

-1073741824

The expected returned value :

3221225472

638

line 11:

```
ret
```

```
+=
```

```
1
```

```
<<
```

```
power;
```

1 is int with 32 bit.

[3]<http://stackoverflow.com/questions/1032982/is-a-java-int-always-32-bit> s

should be: ret

```
+=
```

```
((
```

```
long
```

```
)
```

```
1
```

```
)
```

```
<<
```

```
power;
```

**3rd practice: fail to fix the issue.**

[4]<https://gist.github.com/jianminchen/2b753978ffe983600a031a0fcda7b3a6>

**4th practice :** Use bit manipulation - left shift to get  $2^n$  value - code is working

[5]<https://gist.github.com/jianminchen/6a5c1ab71b01414f008b22386000ee59>

**5th practice:**

First, study the solution provided by lab, great idea:

Reversing bits could be done by swapping the  $n/2$  least significant bits with its most significant bits.

The trick is to implement a function called `swapBits(i, j)`, which swaps the 'i'th bit with the 'j'th bit.

If you still remember how XOR operation works:

here

$0 \wedge 0 == 0,$

$1 \wedge 1 == 0,$

$0 \wedge 1 == 1,$  and

$1 \wedge 0 == 1.$

We only need to perform the swap when the 'i'th bit and the 'j'th bit are different.

To test if two bits are different, we could use the XOR operation. Then, we need to toggle both 'i'th and 'j'th bits.

We could apply the XOR operation again.

By XOR-ing the 'i'th and 'j'th bit with 1, both bits are toggled.

Bonus approach (The divide and conquer approach):

Remember how merge sort works? Let us use an example of  $n == 8$  (one byte) to see how this works:

Remember how merge sort works? Let us use an example of  $n == 8$  (one byte) to see how this works:

01101001

/ \

0110 1001

/ \ / \

01 10 10 01

/\ /\ /\ /\

0 1 1 0 1 0 0 1

The first step is to swap all odd and even bits. After that swap consecutive pairs of bits, and so on ...

Therefore, only a total of  $\log(n)$  operations are necessary.

Example:

For the first step, you would do:

```
x = ((x & 0x55555555) << 1) | ((x & 0xAAAAAAAA) >> 1);
```

**Julia's comment:** *5 minutes workout - how to swap odd bit and even bits?*

$5 = 101$

How to swap odd bit and even bits?

In other words, odd bit becomes even bit; even bit goes to odd bit

First, odd bit shifts to left; even bit shifts to right.

32 bits: 0101 0101 0101 0101 0101 0101 0101 0101

all odd bit number in x:

**$x \& 0x55555555$**

left shift 1 bit:

$(x \& 0x55555555) << 1$



So, even bits shift to right;

A in bits presentation: 1010

all even bit number in x:

`x & 0xAAAAAAAA`

After the swap, the new value is

`x = ((x & 0x55555555) << 1) | ((x & 0xAAAAAAAA) >> 1);`

**- end of Julia's comment -**

study the code in C++:

[6]<https://gist.github.com/jianminchen/3526dd68e72ae97563cdd61580052016>

Fresh from his first grass court final, [7][@milosraonic](#) is thriving under coach McEnroe.[8] #Wimbledon[9]<https://t.co/AWVQyfUugI> — Wimbledon (@Wimbledon) [10]June 22, 2016

1. <https://gist.github.com/jianminchen/395ccf0277250e5d3d68498e0d2238f1>
2. <https://gist.github.com/jianminchen/7bfdcd0c1643bf01c2bf6ba7acea6f70>
3. <http://stackoverflow.com/questions/1032982/is-a-java-int-always-32-bits>
4. <https://gist.github.com/jianminchen/2b753978ffe983600a031a0fcda7b3a6>
5. <https://gist.github.com/jianminchen/6a5c1ab71b01414f008b22386000ee59>
6. <https://gist.github.com/jianminchen/3526dd68e72ae97563cdd61580052016>
7. <https://twitter.com/milosraonic>
8. <https://twitter.com/hashtag/Wimbledon?src=hash>
9. <https://t.co/AWVQyfUugI>
10. <https://twitter.com/Wimbledon/status/745697224886083584>

---

## Swap odd bit and even bit of unsigned 32 bit integer (2016-07-16 17:26)

July 16, 2016

Work on bit manipulation is so much fun. Julia likes to get workout on bit manipulation, and work on the code one by one.

This blog has everything, but it is hard to know what is going on.

[1]<http://juliachencoding.blogspot.ca/2016/07/reverse-unsigned-32-bit-integer.html>

For the first step, you would do:

```
x = ((x & 0x55555555) << 1) | ((x & 0xAAAAAAAA) >> 1);
```

**Julia's comment:**

***5 minutes workout - how to swap odd bit and even bits?***

5 = 101

How to swap odd bit and even bits?

In other words, odd bit becomes even bit; even bit goes to odd bit

First, odd bit shifts to left; even bit shifts to right.

32 bits: 0101 0101 0101 0101 0101 0101 0101 0101

all odd bit number in x:

**x & 0x55555555**

left shift 1 bit:

$(x \& 0x55555555) \ll 1$

So, even bits shift to right;

A in bits presentation: 1010

all even bit number in x:

**x & 0xAAAAAAAA**

After the swap, the new value is

```
x = ((x & 0x55555555) << 1) | ((x & 0xAAAAAAAA) >> 1);
```

**- end of Julia's comment -**

1. <http://juliachencoding.blogspot.ca/2016/07/reverse-unsigned-32-bit-integer.html>

## Reverse unsigned 32 bit integer - facebook code lab - No.1 Java code (2016-07-16 17:26)

July 16, 2016

Work on bit manipulation is so much fun. Julia likes to get workout on bit manipulation, and work on the code one by one.

This blog has everything, but it is hard to know what is going on.

[1]<http://juliachencoding.blogspot.ca/2016/07/reverse-unsigned-32-bit-integer.html>

Java code to study:

[2]<https://gist.github.com/jianminchen/5a866f2abe5f6afcf6a17ef9bbd4c05d>

1. <http://juliachencoding.blogspot.ca/2016/07/reverse-unsigned-32-bit-integer.html>

2. <https://gist.github.com/jianminchen/5a866f2abe5f6afcf6a17ef9bbd4c05d>

---

## Reverse unsigned 32 bit integer - facebook code lab - 5th practice - C++, swap bits (2016-07-16 17:32)

July 16, 2016

First, study the solution provided by lab, great idea:

Reversing bits could be done by swapping the  $n/2$  least significant bits with its most significant bits.

The trick is to implement a function called `swapBits(i, j)`, which swaps the 'i'th bit with the 'j'th bit.

If you still remember how XOR operation works:

here

```
0 ^ 0 == 0,
1 ^ 1 == 0,
0 ^ 1 == 1, and
1 ^ 0 == 1.
```

We only need to perform the swap when the 'i'th bit and the 'j'th bit are different.

To test if two bits are different, we could use the XOR operation. Then, we need to toggle both 'i'th and 'j'th bits.

We could apply the XOR operation again.

By XOR-ing the 'i'th and 'j'th bit with 1, both bits are toggled.

Bonus approach (The divide and conquer approach):

Remember how merge sort works? Let us use an example of  $n == 8$  (one byte) to see how this works:

Remember how merge sort works? Let us use an example of  $n == 8$  (one byte) to see how this works:

01101001

/ \

0110 1001

/ \ / \

01 10 10 01

/\ /\ /\ /\

0 1 1 0 1 0 0 1

The first step is to swap all odd and even bits. After that swap consecutive pairs of bits, and so on ...

Therefore, only a total of  $\log(n)$  operations are necessary.

study the code in C++:

[1]<https://gist.github.com/jianminchen/3526dd68e72ae97563cdd61580052016>

1. <https://gist.github.com/jianminchen/3526dd68e72ae97563cdd61580052016>

---

**Reverse unsigned 32 bit integer - facebook code lab - how to express Math.pow using bit shift**  
(2016-07-16 17:47)

**July 16, 2016**

Julia enjoyed the workout using facebook code lab. She actually learned a few things about Type conversion in bit manipulation:

int - java - 32 bit

Java does not have unsigned int, use long.

But it causes problems!

She struggles and then she learns lessons.

This blog has everything about more than 5 practices, but it is hard to know what is going on.

[1]<http://juliachencoding.blogspot.ca/2016/07/reverse-unsigned-32-bit-integer.html>

So, dedicate one blog on this issue:

**2nd practice: not working, error on 32 bit integer, left most significant bit 1 means negative number.**

[2]<https://gist.github.com/jianminchen/7bfdcd0c1643bf01c2bf6ba7acea6f70>

Sorry, wrong answer. Your program's output doesn't match the expected output. You can try testing your code with **custom input** and try putting **debug statements** in your code.

Your submission failed for the following input:

A : 3

Your function returned the following :

-1073741824

The expected returned value :

3221225472

line 11:

ret

+=

1

<<

power;

1 is int with 32 bit.

[3]<http://stackoverflow.com/questions/1032982/is-a-java-int-always-32-bit> s

should be: ret

+=

((

long

)

1

)

<<

power;

**3rd practice: fail to fix the issue.**

[4]<https://gist.github.com/jianminchen/2b753978ffe983600a031a0fcda7b3a6>

**4th practice** : Use bit manipulation - left shift to get  $2^n$  value - code is working

[5]<https://gist.github.com/jianminchen/6a5c1ab71b01414f008b22386000ee59>

Everyone faces challenges on court and I'm no different. Get out on court and overcome yours to [6]  
#CreateYourMark [7][pic.twitter.com/RSNn7AqFch](https://pic.twitter.com/RSNn7AqFch)

— Ana Ivanovic (@Analvanovic) [8]May 21, 2016

1. <http://juliachencoding.blogspot.ca/2016/07/reverse-unsigned-32-bit-integer.html>
  2. <https://gist.github.com/jianminchen/7bfdcd0c1643bf01c2bf6ba7acea6f70>
  3. <http://stackoverflow.com/questions/1032982/is-a-java-int-always-32-bits>
  4. <https://gist.github.com/jianminchen/2b753978ffe983600a031a0fcda7b3a6>
  5. <https://gist.github.com/jianminchen/6a5c1ab71b01414f008b22386000ee59>
  6. <https://twitter.com/hashtag/CreateYourMark?src=hash>
  7. <https://t.co/RSNn7AqFch>
  8. <https://twitter.com/AnaIvanovic/status/734079097756913664>
- 

## Remove duplicates from Sorted Array (2016-07-16 22:20)

July 16, 2016

Problem statement:

Remove duplicates from Sorted Array

Given a sorted array, remove the duplicates in place such that each element appears only once and return the new length.

Note that even though we want you to return the new length, make sure to change the original array as well in place

Do not allocate extra space for another array, you must do this in place with constant memory.

Example:

Given input array A = [1,1,2],

Your function should return length = 2, and A is now [1,2].

First practice:

[1]<https://gist.github.com/jianminchen/f7156b033b783b3a01c2ad2cb3d1b313>

1. <https://gist.github.com/jianminchen/f7156b033b783b3a01c2ad2cb3d1b313>
- 

Marry Starry (2016-07-25 04:49:29)

This comment has been removed by a blog administrator.

## DiffK - facebook code lab - practices (2016-07-16 23:16)

July 16, 2016

Problem statement:

Given an array 'A' of sorted integers and another non negative integer k, find if there exists 2 indices i and j such that  $A[i] - A[j] = k$ ,  $i \neq j$ .

Example:

Input :

A : [ 1 3 5]

k : 4

Output : YES

as  $5 - 1 = 4$

Return 0 / 1 ( 0 for false, 1 for true ) for this problem

Try doing this in less than linear space complexity.

**First practice:** The solution is not optimized for the time. Almost there.

[1]<https://gist.github.com/jianminchen/decb1042a4e8e66c50be2c80c70f063a>

**Second practice:** The solution works

[2]<https://gist.github.com/jianminchen/f24bc4f8561f29ef5a98d8bd3460d8dd>

Study the solution provided by code lab:

[3]<https://gist.github.com/jianminchen/e4906acce9c763626453adb182ee3b03>

1. <https://gist.github.com/jianminchen/decb1042a4e8e66c50be2c80c70f063a>

2. <https://gist.github.com/jianminchen/f24bc4f8561f29ef5a98d8bd3460d8dd>

3. <https://gist.github.com/jianminchen/e4906acce9c763626453adb182ee3b03>

---

**Big Data Analytics with Tableau - pluralsight.com (2016-07-17 12:24)**

July 17, 2016

Plan to spend a few hours to study the course on pluralsight.com

## Majority - facebook code lab - optimal solution (2016-07-18 00:08)

July 18, 2016

Problem statement:

Given an array of size  $n$ , find the majority element. The majority element is the element that appears more than  $\text{floor}(n/2)$  times.

You may assume that the array is non-empty and the majority element always exist in the array.

Example :

Input : [2, 1, 2]

Return : 2 which occurs 2 times which is greater than  $3/2$ .

**Think about brute force solution first ,**

sort the array first, and then, medium of the array is the majority element. And the time complexity is  $O(n \log n)$ .

If the size of the array is even, then, medium is two nodes in the center. And if the size of the array is odd, the medium is the middle one.

Spent 10 -15 minutes to find the optimal solution, beat the time complexity  $O(n \log n)$ . The array does not need to be sorted, try to think about the partition of the array, pivot value - how to find the one?

No progress in over 15 minutes. Then, read the hint from the facebook code lab.

**Optimal solution idea:**

Just try to go through the array once, and set the majority element, and keep the count of majority element.

Really like the **hint** from facebook code lab:

If two distinct elements are removed from the array, the majority element is still the same one in the array.

1st practice:

[1]<https://gist.github.com/jianminchen/c6f01498a6de4ca2df3d813b375a3d6d>

Study the code provided by facebook code lab:

[2]<https://gist.github.com/jianminchen/7406702476e8fe7b58934fccb9a4e674>

The code is much simple and short.

Julia wrote ugly if statement for 4 cases, but the editorial solution only has one if statement. How does that happen?

**The reasoning is more stronger— one step further, if the count is 0 for majority element, then just set last one as majority element and count = 1. No candidate, and then, set the last one as the candidate.** There is no worry about it. But make the code more simple, not too much if statement.

1. <https://gist.github.com/jianminchen/c6f01498a6de4ca2df3d813b375a3d6d>

2. <https://gist.github.com/jianminchen/7406702476e8fe7b58934fccb9a4e674>



---

**A small research how top tennis player progresses to be top 7 from 152 ranking (2016-07-18 00:31)**

**July 18, 2016**

**Julia likes to spend some time - more than 1 hour to do some research every day. She chose to study the Wimbledon runnup - Milos Raonic. She was too busy to play tennis to do some study about top players.**

**Spend time to get familiar with a tennis player - Milos Raonic -**

Watch the tennis player - Milos Raonic's interview

**How to get top 37 from over 152 ranking in just one month?**

Children's foundation

1. [1]<https://www.youtube.com/watch?v=ZIk6k-RJk8E>

Philanthropy

[ [2]edit ]

In 2011, while recovering from a hip injury sustained at Wimbledon, Raonic decided to become involved with philanthropic work, focusing on helping disadvantaged children.<sup>[3][38]</sup> The following year, in 2012, Raonic launched the Milos Raonic Foundation,<sup>[4][39]</sup>notesize[5][40][6][41] which aims to "support children from disadvantaged backgrounds in order to remove economic, physical and other barriers that might prevent them from becoming healthy, productive members of society. ... In the initial stages of its work, the foundation will focus, in particular, on children with physical disabilities."<sup>[7][42]</sup>

Is Time Running Out?

2. [8]<https://www.youtube.com/watch?v=4leeDF5Qla4>

John McEnroe talked about his coaching - how to help Raonic?

3. [9]<https://www.youtube.com/watch?v=9Kd2cuhp5LI>

4. [10]<https://www.youtube.com/watch?v=qYt98fHNksE>

Hardwork pays off, but the timing may not be you expect.

Raonic's expectations sky high - **Julia likes the seeded tennis professional players, really smart and also good at talking/ sharing.**

5. [11]<https://www.youtube.com/watch?v=fTAR38gcL6U>

### Julia's most favorite:

How to handle the success?

Sudden success? It is difficult to manage the success as managing failure.

How does Raonic copy with the sudden success?

6. [12]<https://www.youtube.com/watch?v=WLzADfghPDA>

7. [13]<https://www.youtube.com/watch?v=FwVgrhWq6oA>

Took risk to turn down tennis scholarship for college, and chose to be a professional tennis player. There is a life after tennis, and tennis professional can be very short career.

Blogs:

Leetcode 47 Permutation II

Leetcode 75 Sort colors

[14]<http://www.jianshu.com/p/82fe72072226>

Leetcode 286 Walls and Gates

1. <https://www.youtube.com/watch?v=ZIk6k-RJk8E>

2. [https://en.wikipedia.org/w/index.php?title=Milos\\_Raonic&action=edit&section=2&editintro=Template:BLP\\_editintro](https://en.wikipedia.org/w/index.php?title=Milos_Raonic&action=edit&section=2&editintro=Template:BLP_editintro)

3. [https://en.wikipedia.org/wiki/Milos\\_Raonic#cite\\_note-39](https://en.wikipedia.org/wiki/Milos_Raonic#cite_note-39)

4. [https://en.wikipedia.org/wiki/Milos\\_Raonic#cite\\_note-:49-40](https://en.wikipedia.org/wiki/Milos_Raonic#cite_note-:49-40)

5. [https://en.wikipedia.org/wiki/Milos\\_Raonic#cite\\_note-:50-41](https://en.wikipedia.org/wiki/Milos_Raonic#cite_note-:50-41)

6. [https://en.wikipedia.org/wiki/Milos\\_Raonic#cite\\_note-:51-42](https://en.wikipedia.org/wiki/Milos_Raonic#cite_note-:51-42)

7. [https://en.wikipedia.org/wiki/Milos\\_Raonic#cite\\_note-:3-43](https://en.wikipedia.org/wiki/Milos_Raonic#cite_note-:3-43)

8. <https://www.youtube.com/watch?v=4IeeDF5Qla4>

9. <https://www.youtube.com/watch?v=9Kd2cuhp5LI>

10. <https://www.youtube.com/watch?v=qYt98fHNksE>

11. <https://www.youtube.com/watch?v=fTAR38gcL6U>

12. <https://www.youtube.com/watch?v=WLzADfghPDA>

13. <https://www.youtube.com/watch?v=FwVgrhWq6oA>

14. <http://www.jianshu.com/p/82fe72072226>

---

### Leetcode 173: Binary Search Tree Iterator - facebook code lab (2016-07-18 22:15)

July 18, 2016

Problem statement:

Implement an iterator over a binary search tree (BST). Your iterator will be initialized with the root node of a BST. The first call to next() will return the smallest number in BST. Calling next() again will return the next smallest number in the BST, and so on.

Note: next() and hasNext() should run in average  $O(1)$  time and uses  $O(h)$  memory, where  $h$  is the height of the tree.

Try to optimize the additional space complexity apart from the amortized time complexity.

The time estimated is 41 minutes in facebook code lab.

Plan to work on this problem in short future.

---

### Leetcode 23: Merge K Sorted Lists (2016-07-19 22:13)

July 19, 2016

Problem statement:

Merge  $k$  sorted linked lists and return it as one sorted list. Analyze and describe its complexity.

Julia's first two practices in 2015:

1. Naive solution:

[1][https://github.com/jianminchen/Leetcode\\_C-/blob/master/MergeKSortedLists\\_A\\_No23.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/MergeKSortedLists_A_No23.cs)

2. Implementation using merge sort: August 12, 2015

[2][https://github.com/jianminchen/Leetcode\\_C-/blob/master/MergeKSortedLists\\_B\\_No23.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/MergeKSortedLists_B_No23.cs)

Come back to work on this problem in 2016, July 19:

Study those blogs:

1. Most favorite one:

[3]<http://bangbingsyb.blogspot.ca/2014/11/leetcode-merge-k-sorted-lists.html>

Specially, the third discussion using divide and conquer, the conclusion that the merge solution is same time efficiency using heap solution  $Nk\log(k)$ ,  $k$  is the number of arrays, and  $N$  is the length of the array, assuming that each array has same length.

1. [4]<http://www.cnblogs.com/TenosDolt/p/3673188.html>

2. [5]<http://bangbingsyb.blogspot.ca/2014/11/leetcode-merge-k-sorted-lists.html>

3. [6][http://codeganker.blogspot.ca/2014/08/leetcode\\_26.html](http://codeganker.blogspot.ca/2014/08/leetcode_26.html)

4. [7]<http://www.cnblogs.com/springfor/p/3869217.html>

5. [8][https://github.com/shawnfan/LintCode/blob/master/Java/Merge %20k %20Sorted %20Arrays.java](https://github.com/shawnfan/LintCode/blob/master/Java/Merge%20k%20Sorted%20Arrays.java)

7. Excellent article - **Julia, share your personal story as well .**

[9][http://codeganker.blogspot.ca/2014/08/leetcode\\_26.html](http://codeganker.blogspot.ca/2014/08/leetcode_26.html)

Read the article:

[10]<http://www.cs.princeton.edu/courses/archive/spr07/cos226/lectures/04MergeQuick.pdf>

**Julia's question and answer:**

1. **Ideas to improve practice**

?

Answer:

Ideas to construct the practice:

1. Need to work on blog 7 - 3 solutions using C #, apply to "merge k sorted array".

Learn bottom-up merge sort, and know the time complexity as well.

C # solution - merge k sorted array - using bottom-up implementation merge sort

[11]<https://gist.github.com/jianminchen/d5fab2036647d380f20c908e91a81132>

2. Write down the proof of "divide and conquer" time complexity comparison using heap,  $O(nk \log k)$

Show some simple example to help understand the time complexity.

3. Warm up the code practice a few times, aiming to write 20 minutes for a solution

4. Need to figure out master theorem quickly.

2. Can you work on a small example to illustrate some analysis?

Copy the image from Princeton's lecture note:

← → ↻ algs4.cs.princeton.edu/lectures/22/Mergesort.pdf 🔍 ⚙️ 📄

## Divide-and-conquer recurrence: proof by picture

---

**Proposition.** If  $D(N)$  satisfies  $D(N) = 2 D(N/2) + N$  for  $N > 1$ , with  $D(1) = 0$ , then  $D(N) = N \lg N$ .

**Pf 1.** [assuming  $N$  is a power of 2]

$D(N)$   
 $D(N/2)$   $D(N/2)$   
 $D(N/4)$   $D(N/4)$   $D(N/4)$   $D(N/4)$   
 $D(N/8)$   $D(N/8)$   $D(N/8)$   $D(N/8)$   $D(N/8)$   $D(N/8)$   $D(N/8)$   $D(N/8)$   
 $\vdots$

$N = N$   
 $2(N/2) = N$   
 $4(N/4) = N$   
 $8(N/8) = N$   
 $\vdots$   


---

 $T(N) = N \lg N$

3. Can you write some C # code this time to make your own mark?

**Actionable items:**

**Read the article, and write down questions - prepare a further reading list. (30 minutes reading )**

[12][https://en.wikipedia.org/wiki/Merge\\_sort](https://en.wikipedia.org/wiki/Merge_sort)

**Notes:**

**keywords: (remember 14 keywords)**

average and worst-case performance

Bitonic Mergesort

Bottom-up implementation

comparison-based sorting algorithm

external merge sort - disk/ tape drives

**external merge sort**

master theorem

merge sort - not inplace - must be allocated for the sorted output to be stored in

**parallel mergesort**

polyphase merge sort

natural merge sort - similar to a bottom up merge sort

**stable sort,**  
**TimSort,**  
tiled merge sort algorithm  
Top-down implementation

Variants:

1. reducing the space complexity
2. cost of copying

locality of reference

memory hierarchies  
cache-aware version of merge sort algorithm  
tiled merge sort algorithm

Parallel merge sort

Merge sort parallelizes well due to use of the divide-and-conquer method.

-

Comparison with other sort algorithms

heapsort vs merge sort

$O(1)$  auxiliary space instead of merge sort's  $O(n)$ .

efficient quicksort implementations generally outperform mergesort for sorting RAM-based arrays - ?

merge sort is a stable sort and is more efficient at handling slow-to-access sequential media.

Merge sort is often the best choice for sorting a linked list.

the slow random-access performance of a linked list makes some other algorithms (such as quicksort) perform poorly, and others (such as heapsort) completely impossible.

Java, `Array.sort()` methods use merge sort or a tuned quicksort depending on the datatypes and for implementation efficiency switch to insertion sort when fewer than seven array elements are being sorted.

Python uses Timsort, another tuned hybrid of merge sort and insertion sort.

-

**More reading:**

30 minutes to review:

[13]<http://algs4.cs.princeton.edu/lectures/22Mergesort.pdf>

5 minutes reading:

[14]<http://infolab.stanford.edu/~ullman/fcscnotes/notes9.pdf>

20 minutes reading - plan to read - parallel merge sort

[15][http://stanford.edu/rezab/dao/notes/Lecture04/cme323\\_lec4.pdf](http://stanford.edu/rezab/dao/notes/Lecture04/cme323_lec4.pdf)

Read 10 minutes a time - get lost and enjoy reading ...

[16]<http://web.archive.org/web/20150120063131/https://android.googlesource.com/platform/libcore/+jb-mr2-release/luni/src/main/java/java/util/TimSort.java>

Will come back very soon.

1. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/MergeKSortedLists\\_A\\_No23.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/MergeKSortedLists_A_No23.cs)
  2. [https://github.com/jianminchen/Leetcode\\_C-/blob/master/MergeKSortedLists\\_B\\_No23.cs](https://github.com/jianminchen/Leetcode_C-/blob/master/MergeKSortedLists_B_No23.cs)
  3. <http://bangbingsyb.blogspot.ca/2014/11/leetcode-merge-k-sorted-lists.html>
  4. <http://www.cnblogs.com/TenosDoIt/p/3673188.html>
  5. <http://bangbingsyb.blogspot.ca/2014/11/leetcode-merge-k-sorted-lists.html>
  6. [http://codeganker.blogspot.ca/2014/08/leetcode\\_26.html](http://codeganker.blogspot.ca/2014/08/leetcode_26.html)
  7. <http://www.cnblogs.com/springfor/p/3869217.html>
  8. <https://github.com/shawnfan/LintCode/blob/master/Java/Merge%20k%20Sorted%20Arrays.java>
  9. [http://codeganker.blogspot.ca/2014/08/leetcode\\_26.html](http://codeganker.blogspot.ca/2014/08/leetcode_26.html)
  10. <http://www.cs.princeton.edu/courses/archive/spr07/cos226/lectures/04MergeQuick.pdf>
  11. <https://gist.github.com/jianminchen/d5fab2036647d380f20c908e91a81132>
  12. [https://en.wikipedia.org/wiki/Merge\\_sort](https://en.wikipedia.org/wiki/Merge_sort)
  13. <http://algs4.cs.princeton.edu/lectures/22Mergesort.pdf>
  14. <http://infolab.stanford.edu/~ullman/fcscnotes/notes9.pdf>
  15. [http://stanford.edu/~rezab/dao/notes/Lecture04/cme323\\_lec4.pdf](http://stanford.edu/~rezab/dao/notes/Lecture04/cme323_lec4.pdf)
  16. <http://web.archive.org/web/20150120063131/https://android.googlesource.com/platform/libcore/+jb-mr2-release/luni/src/main/java/java/util/TimSort.java>
- 

## HackerRank: World codesprint #4 - Roads in HackerLand - II - code study (2016-07-20 22:27)

July 20, 2016

First blog on this algorithm:

HackerRank: World codesprint #4 - Roads in HackerLand

[1]<http://juliachencoding.blogspot.ca/2016/06/hackerrank-world-codesprint-4-roads-in.html>

Come back to work on this problem. Study editorial solution provided by HackerRank, and then, review union find, minimum spanning tree algorithm with code first:

### 3 steps :

step 1: union find algorithm

step 2: minimum spanning tree

step 3: roads in hackerLand

Blogs to read:

1.

[2][https://en.wikipedia.org/wiki/Minimum\\_spanning\\_tree](https://en.wikipedia.org/wiki/Minimum_spanning_tree)

2.

[3]<http://www.ics.uci.edu/~eppstein/161/960206.html>

Work on union find algorithm first: ( **step 1** )

July 21, 2016

1. Union find algorithm - detect cycle

[4]<http://www.geeksforgeeks.org/union-find/>

2. Union by rank and path compression

[5]<http://www.geeksforgeeks.org/union-find-algorithm-set-2-union-by-rank/>

Then, work on minimum spanning tree algorithm: ( step II )

3.

[6]<http://www.geeksforgeeks.org/greedy-algorithms-set-2-kruskals-minimum-spanning-tree-mst/>

4.

[7]<http://www.geeksforgeeks.org/greedy-algorithms-set-5-prims-minimum-spanning-tree-mst-2/>

And then, study 10 code submission scoring 60/60 in Java, C++, C#. Practice one by one. ( **step III** )

A better way to learn an algorithm - minimum spanning tree - work on a concrete example, make it fun learning experience.

1. Java implementation:

[8]<https://gist.github.com/jianminchen/20775a7ac1eeb83fa141e92633ea7d78>

2. C++ 14

[9]<https://gist.github.com/jianminchen/28b5c3bf191a27250b67ce4e7656166b>

3. C #

[10]<https://gist.github.com/jianminchen/7a21dfe2a62f1bcf4bc305480ef2f51e>

4. C #

[11]<https://gist.github.com/jianminchen/8a15be7b83770d22647f34371fb48a97>

5. C++

[12]<https://gist.github.com/jianminchen/5a3e680b86d2c5ffc0f090f29f074089>

6. C++

[13]<https://gist.github.com/jianminchen/dcac67099aa7ddcb4497565f0732fe5e656>

656



7. C++

[14]<https://gist.github.com/jianminchen/10983fdcbb15fef37bd73a60fe87430>

8.

9.

10.

1. <http://juliachencoding.blogspot.ca/2016/06/hackerrank-world-codesprint-4-roads-in.html>
2. [https://en.wikipedia.org/wiki/Minimum\\_spanning\\_tree](https://en.wikipedia.org/wiki/Minimum_spanning_tree)
3. <http://www.ics.uci.edu/~eppstein/161/960206.html>
4. <http://www.geeksforgeeks.org/union-find/>
5. <http://www.geeksforgeeks.org/union-find-algorithm-set-2-union-by-rank/>
6. <http://www.geeksforgeeks.org/greedy-algorithms-set-2-kruskals-minimum-spanning-tree-mst/>
7. <http://www.geeksforgeeks.org/greedy-algorithms-set-5-prims-minimum-spanning-tree-mst-2/>
8. <https://gist.github.com/jianminchen/20775a7ac1eeb83fa141e92633ea7d78>
9. <https://gist.github.com/jianminchen/28b5c3bf191a27250b67ce4e7656166b>
10. <https://gist.github.com/jianminchen/7a21dfe2a62f1bcf4bc305480ef2f51e>
11. <https://gist.github.com/jianminchen/8a15be7b83770d22647f34371fb48a97>
12. <https://gist.github.com/jianminchen/5a3e680b86d2c5ffc0f090f29f074089>
13. <https://gist.github.com/jianminchen/dcac67099aa7ddcb4497565f0732fe5e>
14. <https://gist.github.com/jianminchen/10983fdcbb15fef37bd73a60fe87430>

---

## Introduction to website layout - pluralsight.com (2016-07-20 23:22)

July 20, 2016

Relax and try to find 3 hours to work on course provided by pluralsight.com

[1]<http://app.pluralsight.com/author/susan-simkins>

1. <http://app.pluralsight.com/author/susan-simkins>

---

## CSS in-depth - pluralsight.com (2016-07-20 23:24)

July 20, 2016

Plan to find 6 hours to work on this course: 30 minutes a time, one by one.

CSS in-depth 6 hour course

[1]<http://app.pluralsight.com/author/estelle-weyl>

1. Go over the slides on the website: (plan to take 1 hour first)

[2]<http://lanyrd.com/profile/estellew/slides/>

2. Julia's favorite - ***great input for Julia to build great mobile website!***

[3]<http://estelle.github.io/mobilecss/#slide4>

3. Get to know a new tool: YSlow

[4]<http://yslow.org/>

Read 34 rules of web performance best practices and rules:

Yahoo!'s Exceptional Performance team has identified 34 rules that affect web page performance. YSlow's web page analysis is based on the 23 of these 34 rules that are testable. Click each performance rule below to see the details.

1. [5]Minimize HTTP Requests
2. [6]Use a Content Delivery Network
3. [7]Avoid empty src or href
4. [8]Add an Expires or a Cache-Control Header
5. [9]Gzip Components
6. [10]Put StyleSheets at the Top
7. [11]Put Scripts at the Bottom
8. [12]Avoid CSS Expressions
9. [13]Make JavaScript and CSS External
10. [14]Reduce DNS Lookups
11. [15]Minify JavaScript and CSS
12. [16]Avoid Redirects
13. [17]Remove Duplicate Scripts
14. [18]Configure ETags
15. [19]Make AJAX Cacheable
16. [20]Use GET for AJAX Requests
17. [21]Reduce the Number of DOM Elements
18. [22]No 404s
19. [23]Reduce Cookie Size

20. [24]Use Cookie-Free Domains for Components
21. [25]Avoid Filters
22. [26]Do Not Scale Images in HTML
23. [27]Make favicon.ico Small and Cacheable

#### 4. Julia loves the CSS Filter Effects

[28]<http://html5-demos.appspot.com/static/css/filters/index.html>

( **memorize 10 keywords: b b c d g h i o s s** )

blur  
brightness  
contrast  
drop-shadow  
grayscale  
hue-rotate  
invert  
opacity  
saturate  
sepia

5.

[29]<http://static.lukew.com/TouchGestureGuide.pdf>

Core gestures:

Tap,  
double tap,  
Drag,  
Flick,  
Pinch,  
Spread,  
Press,  
Press and tap,  
Press and drag,  
Rotate

#### 6. Google page speed tools

- Analyze your site performance -

[30]<https://developers.google.com/speed/pagespeed/?csw=1>

6B. Page Speed - Go over each main point for 5 minute study:

1. leverage browser caching

Leveraging Browsing caching for images, CSS and JS

<https://www.siteground.com/kb/leverage-browser-caching/>

<http://stackoverflow.com/questions/23923039/wordpress-leverage-browser-caching>

2. Enable compression
3. Defer parsing of JavaScript
4. Minimize request size
5. Specify a cache validator
6. Optimize images
7. Minify JavaScript
8. Minify HTML
9. specify image dimensions
10. Specify a character set
11. Specify a Vary: Accept Encoding Header
12. Avoid long-running scripts
13. Avoid CSS @import
14. Avoid bad requests
15. Enable Keep-Alive
16. Make landing page redirects cacheable
17. Minify CSS
18. Minimize redirects
19. Optimize the order of styles and scripts
20. Put CSS in the document head
21. Remove query strings from static resources

7. Sprites - reduce requests -

8. ImageAlpha

Reduce image file size - lossy compression

9. <https://pngmini.com/>

10. Responsive design - RDS

<http://estelle.github.io/rwdpanacea/#slide12>

11. Debugging tools for mobile website:

1. Charles proxy

<https://www.charlesproxy.com/>

2. fiddler

3. browser stack

[www.browserstack.com](http://www.browserstack.com)

4. device anywhere

[www.keynoteddeviceanywhere.com](http://www.keynoteddeviceanywhere.com)

Julia's comment:

1. Great teaching, first CSS expert Julia knows - love the course - CSS in-depth - pluralsight.com
2. web performance best practices and rules - how many Julia breaks?

1. <http://app.pluralsight.com/author/estelle-weyl>
2. <http://lanyrd.com/profile/estellewv/slides/>
3. <http://estelle.github.io/mobilecss/#slide4>
4. <http://yslow.org/>
5. [http://developer.yahoo.com/performance/rules.html#num\\_http](http://developer.yahoo.com/performance/rules.html#num_http)
6. <http://developer.yahoo.com/performance/rules.html#cdn>
7. <http://developer.yahoo.com/performance/rules.html#emptysrc>
8. <http://developer.yahoo.com/performance/rules.html#expires>
9. <http://developer.yahoo.com/performance/rules.html#gzip>
10. [http://developer.yahoo.com/performance/rules.html#css\\_top](http://developer.yahoo.com/performance/rules.html#css_top)
11. [http://developer.yahoo.com/performance/rules.html#js\\_bottom](http://developer.yahoo.com/performance/rules.html#js_bottom)
12. [http://developer.yahoo.com/performance/rules.html#css\\_expressions](http://developer.yahoo.com/performance/rules.html#css_expressions)
13. <http://developer.yahoo.com/performance/rules.html#external>
14. [http://developer.yahoo.com/performance/rules.html#dns\\_lookups](http://developer.yahoo.com/performance/rules.html#dns_lookups)
15. <http://developer.yahoo.com/performance/rules.html#minify>
16. <http://developer.yahoo.com/performance/rules.html#redirects>
17. [http://developer.yahoo.com/performance/rules.html#js\\_dupes](http://developer.yahoo.com/performance/rules.html#js_dupes)
18. <http://developer.yahoo.com/performance/rules.html#etags>
19. <http://developer.yahoo.com/performance/rules.html#cacheajax>
20. [http://developer.yahoo.com/performance/rules.html#ajax\\_get](http://developer.yahoo.com/performance/rules.html#ajax_get)
21. [http://developer.yahoo.com/performance/rules.html#min\\_dom](http://developer.yahoo.com/performance/rules.html#min_dom)
22. <http://developer.yahoo.com/performance/rules.html#no404>
23. [http://developer.yahoo.com/performance/rules.html#cookie\\_size](http://developer.yahoo.com/performance/rules.html#cookie_size)
24. [http://developer.yahoo.com/performance/rules.html#cookie\\_free](http://developer.yahoo.com/performance/rules.html#cookie_free)
25. [http://developer.yahoo.com/performance/rules.html#no\\_filters](http://developer.yahoo.com/performance/rules.html#no_filters)
26. [http://developer.yahoo.com/performance/rules.html#no\\_scale](http://developer.yahoo.com/performance/rules.html#no_scale)
27. <http://developer.yahoo.com/performance/rules.html#favicon>
28. <http://html5-demos.appspot.com/static/css/filters/index.html>
29. <http://static.lukew.com/TouchGestureGuide.pdf>
30. <https://developers.google.com/speed/pagespeed/?csw=1>

---

## Typograph for web - pluralsight.com (2016-07-20 23:29)

July 20, 2016

Plan to find 3 hours to work on this course - 30 minutes a time, one by one,

Typograph for web

[1]<http://app.pluralsight.com/author/estelle-weyl>

1. <http://app.pluralsight.com/author/estelle-weyl>

---

## **Creating Responsive Landing Pages in Photoshop and CSS - pluralsight.com (2016-07-20 23:36)**

July 20, 2016

Plan to spend 2 hours to work on this course:

Creating Responsive Landing Pages in Photoshop and CSS

[1]<http://app.pluralsight.com/author/gary-simon>

1. <http://app.pluralsight.com/author/gary-simon>

---

## **The Art of A/B Testing for Web Design - pluralsight.com (2016-07-20 23:38)**

July 20, 2016

Plan to spend 2 hours to work on this course:

The Art of A/B Testing for Web Design

[1]<http://app.pluralsight.com/author/gary-simon>

1. <http://app.pluralsight.com/author/gary-simon>

---

## Merge N sorted Array - merge sort from bottom-up implementation (2016-07-21 23:51)

July 21, 2016

Merge N sorted arrays - C # practice:

[1]<https://gist.github.com/jianminchen/d5fab2036647d380f20c908e91a81132>

Time complexity analysis:

N sorted array, each array length is k, so the time complexity:

$$2k * N/2 + 4k * N/4 + \dots + 2^{\log N} k * N / (2^{\log N}) = kN \log N,$$

### Today's research topic :

Coding is more of science, or just muscle memory, go for the intuition.

### Review:

1. bottom-up implementation merge sort
2. Master Theorem -  $T(N) = 2T(N/2) + O(N)$
3. Time complexity - come out  $kN \log N$  analysis

Read the article, and understand better about merge sort:

[2][https://en.wikipedia.org/wiki/Merge\\_sort](https://en.wikipedia.org/wiki/Merge_sort)

### Keywords:

Bitonic Mergesort

external merge sort - disk/ tape drives

external merge sort

merge sort

- not inplace - must be allocated for the sorted output to be stored in

parallel mergesort

polyphase merge sort

natural merge sort - similar to a bottom up merge sort

stable sort,

TimSort,

tiled merge sort algorithm

Bottom-up implementation

Top-down implementation

comparison-based sorting algorithm

master theorem

average and worst-case performance

**Lecture notes study: (work on lecture notes, write down some interesting topic, do some research!)**

[3]<http://algs4.cs.princeton.edu/lectures/22Mergesort.pdf>

**Actionable item:**

Add study time 10 minutes for computer science, review theorems, lecture notes when writing a new blog.

1. <https://gist.github.com/jianminchen/d5fab2036647d380f20c908e91a81132>

2. [https://en.wikipedia.org/wiki/Merge\\_sort](https://en.wikipedia.org/wiki/Merge_sort)

3. <http://algs4.cs.princeton.edu/lectures/22Mergesort.pdf>

---

**AWS Developer Fundamentals - pluralsight.com (2016-07-22 20:05)**

July 22, 2016

Spend a few hours on AWS developer fundamentals.

Lecturer website:

[1]<http://app.pluralsight.com/author/richard-seroter>

[2]<https://seroter.wordpress.com/>

Take some notes here:

1. <http://app.pluralsight.com/author/richard-seroter>

2. <https://seroter.wordpress.com/>

---

**GALS show - favorite time to spend 30 minutes (2016-07-23 13:46)**

July 23, 2016

Spend 1 hour to watch twice, write down a few notes:

[1]<https://channel9.msdn.com/Shows/GALS/Interview-with-Kathleen-Hogan>

Growth mind set - fixed mind set vs growth mind set, showing more

Julia likes to read article:

[2]<http://www.bizjournals.com/seattle/print-edition/2015/11/20/kathleen-hogan.html>

[3]<https://www.linkedin.com/pulse/empower-your-employees-leverage-own-data-kathleen-hogan?trk=prof-post>



[4]<https://www.linkedin.com/pulse/how-millennials-changing-workforce-three-data-trends-im-hogan?trk=prof-post>

Enjoy Vancouver city video 2 minutes, and browse through Microsoft office building in Vancouver through the video.

[5]<http://blogs.microsoft.com/jobs/new-microsoft-offices-boast-ultramodern-design-stunning-views/?WT.tsrc=MSSharing>

1. <https://channel9.msdn.com/Shows/GALs/Interview-with-Kathleen-Hogan>
2. <http://www.bizjournals.com/seattle/print-edition/2015/11/20/kathleen-hogan.html>
3. <https://www.linkedin.com/pulse/empower-your-employees-leverage-own-data-kathleen-hogan?trk=prof-post>
4. <https://www.linkedin.com/pulse/how-millennials-changing-workforce-three-data-trends-im-hogan?trk=prof-post>
5. <http://blogs.microsoft.com/jobs/new-microsoft-offices-boast-ultramodern-design-stunning-views/?WT.tsrc=MSSharing>

---

## Union-Find Algorithm - an undirected graph algorithm (2016-07-23 15:08)

July 23, 2016

Julia just misses the fun time to play with [1]HackerRank world codesprint #4 in June 2016, in order to prepare next HackerRank code sprint on July 24, 2016, she has to review undirected graph algorithm - Union-Find algorithm. She likes the challenge, but she has to discipline herself always work on the simplest algorithm first. Aim small target first, baby step; avoid dealing with complicate function in the design at the beginning.

Write her own C # code:

[2]<http://www.geeksforgeeks.org/union-find/>

August 3, 2016

1. Go over Union-find algorithm lecture notes:

[3]<https://www.cs.princeton.edu/rs/AlgsDS07/01UnionFind.pdf>

2.

[4][http://blog.csdn.net/dm\\_vincent/article/details/7655764](http://blog.csdn.net/dm_vincent/article/details/7655764)

[5][http://blog.csdn.net/dm\\_vincent/article/details/7769159](http://blog.csdn.net/dm_vincent/article/details/7769159)

1. <http://juliachencoding.blogspot.ca/2016/06/a-small-research-on-hackerrank-world.html>
2. <http://www.geeksforgeeks.org/union-find/>

3. <https://www.cs.princeton.edu/~rs/AlgsDS07/01UnionFind.pdf>
  4. [http://blog.csdn.net/dm\\_vincent/article/details/7655764](http://blog.csdn.net/dm_vincent/article/details/7655764)
  5. [http://blog.csdn.net/dm\\_vincent/article/details/7769159](http://blog.csdn.net/dm_vincent/article/details/7769159)
- 

## **HackerRank - project euler (2016-07-23 15:17)**

July 23, 2016

Favorite place to write code in June 2016 is facebook code lab. Try this one, project euler, start with 1 or 2 hours first.

[1]<https://www.hackerrank.com/contests/projecteuler/challenges>

blog about euler practice:

[2]<http://cenalulu.github.io/python/euler-project-experience/>

1. <https://www.hackerrank.com/contests/projecteuler/challenges>
  2. <http://cenalulu.github.io/python/euler-project-experience/>
- 

## **Short Palindrome - HackerRank - world codesprint #5 (2016-07-25 00:03)**

July 25, 2016

Spent 3+ hours to score 13 out of 40 points. But, Julia enjoyed the time to work on the problem.

A few issues, wrong answer, time out, run time error.

More than 3 submissions, failed to solve time out issue.

1. [1]<https://gist.github.com/jianminchen/c5b6d09decd5f599392c0209e65b82> 36

2. add memorization using Dictionary to save time, failed to solve issues

[2]<https://gist.github.com/jianminchen/cb46292340f86cdcf35a5b913e1588b1>

3. add one more memorization, failed to solve issues.

[3]<https://gist.github.com/jianminchen/894c9b723aa847f1a261658147e1ecaf>

4. Review code and then add some comment, and analyse design issues:

[4]<https://gist.github.com/jianminchen/bf0f554dd32ea8f4055f86a2c569e414>

Review the design, try to find the flaws - a few ideas to improve timeout, no run-time error issue.

review code and then add comment for function countingPalindromes:

```
/*
* use two loops
*
* get 0110 pairs
* get 1001 paris
*
* Think about DP solution - dynamic programming - not working, too complicate
*
* brute force solution:
* 0 - start char, end char, and then, count how many possibilities
* Go through  $O(n*n)$  case, for any two 0 in pseudo string, check in-between
*
* For example:
* 001010110
*
* The above case, there are 5 '0',
* position of '0' is 0, 1, 3, 5, 8,
* So, any two '0' combinations are  $5 * 4 / 2 * 1 = 10$ 
*
* A: 0, 1, in between, there is no chars, n/a
* B: 0, 3, 2 chars in between - "01", count how many '1' in the substring, only 1.
* C: 0, 5, 4 chars in between - "0101", two one inside, so how many combination
* of 11, only 1 choice.
* D: 0, 8, 7 chars in-between - "0101011", four one's in the substring, how many combination -  $4 * 3 / 2 = 6$  choices
*/
Add comment for function getPseudoString(...)

/*
* using 0 and 1 to construct the string
* 1001
* 0110
*
* There are 26 chars in alphabetic string, any two combination is  $26 * 25 / 2 * 1 > 250$ 
* Instead, using 0 and 1 to stand for two distinct characters.
```

\* for example, abab

\* 0101

\* So, we can conclude one thing:

\* ababa

\* 01010

\* or

\* 10101

\* So, use memorization, 01010 and 10101 should be same key

\* See if this can solve time out issues - July 25, 2016

\*

\* reverse of string should be same key

\*/ It took Julia more than 3 hours to gain 13/40 point, but learning experience was so great, and she enjoyed the coding experience.

Starting from brute force solution, she made some progress to gain 13 points; go through optimal solution - DP, should be next step.

### **Statistics:**

450 score 40/40

1600 score above 12/40

Total submission: 2180

### **Facts about HackerRank:**

aiming brute force, 30 % score.

### **Dynamic solution:**

### **detail:**

[5]<https://www.hackerrank.com/contests/world-codesprint-5/challenges/short-palindrome/editorial>

**The idea of DP from the above website:** string length is n

pattern to search

xyyx

xy ends position at i - iterate from 1 to n-1,

denote l[i]

yx starts at position i - iteration from i to n-1

denote  $r2[i]$

yx starts at position  $\geq i$

denote  $r[i] = r2[i] + r2[i+1] + \dots + r2[n-1]$

So, to get a solution:

Iterate  $i$  from 1 to  $n-1$

sum of  $l[i] * r[i]$

And one more thing,

$r[i] = r2[i] + r[i+1]$

**The idea Julia tried on July 24, 2016 -**

suppose that  $T[i]$  is the short palindrome like "xyyx" at end of string  $i$ , how to construct  $T[i+1]$ ?

I've tasted success and I'm hungry for more. I'll prove I've got what it takes to win. [6] #CreateYourMark  
[7][pic.twitter.com/jansdjv9pG](https://pic.twitter.com/jansdjv9pG)

— Lucas Pouille (@la\_pouille) [8] May 20, 2016

1. <https://gist.github.com/jianminchen/c5b6d09decd5f599392c0209e65b8236>
2. <https://gist.github.com/jianminchen/cb46292340f86cdcf35a5b913e1588b1>
3. <https://gist.github.com/jianminchen/894c9b723aa847f1a261658147e1ecaf>
4. <https://gist.github.com/jianminchen/bf0f554dd32ea8f4055f86a2c569e414>
5. <https://www.hackerrank.com/contests/world-codesprint-5/challenges/short-palindrome/editorial>
6. <https://twitter.com/hashtag/CreateYourMark?src=hash>
7. <https://t.co/jansdjv9pG>
8. [https://twitter.com/la\\_pouille/status/733783371705421824](https://twitter.com/la_pouille/status/733783371705421824)

---

## Camel Case - HackerRank - world codesprint #5 (2016-07-25 00:03)

July 25, 2016

C # practice: Easy question, score 25/25

[1]<https://gist.github.com/jianminchen/75b984851f03cc369d1f59f41b6415f2>

1. <https://gist.github.com/jianminchen/75b984851f03cc369d1f59f41b6415f2>
-

## String Construction - HackerRank - world codesprint #5 (2016-07-25 00:05)

July 25, 2016

String Construction: Easy question, score 25/25

[1]<https://gist.github.com/jianminchen/9c61868800a9835347ffc1047960ea08>

1. <https://gist.github.com/jianminchen/9c61868800a9835347ffc1047960ea08>

---

## HackerRank - world codesprint #5 Longest increasing subsequence arrays (2016-07-25 00:16)

July 25, 2016

Problem website:

[1]<https://www.hackerrank.com/contests/world-codesprint-5/challenges/long est-increasing-subsequence-arrays>

Analysis from editorial:

Because we must use numbers from  
to fill each array and we must be able to build an  
-element LIS for each array, we know each number from  
to  
must appear in the array in increasing order.  
We can select  
positions where  
to  
will be placed such that  
is placed in the first selected position,  
,  
is placed in the second position,  
, and so on. To avoid overcounting, we impose that there is no  
before  
, no  
between the position of  
and  
, and so on. Note that after  
670

, we are free to place any value  
we want.

For each chosen arrangement, how many ways are there to fill the remaining gaps? There are  
unfilled cells and there are

values we can use to fill each position (except the segment from

, which can accommodate until

). If we let

, we can place

.

If we loop over the values of

from

to

, then:

Note that after we place the values in the last segment of length

, we're left with an array of length

in which the last element must be

. That's why we can only choose

integers.

This has a complexity of

, provided you precalculate properly.

C # code to study:

[2]<https://gist.github.com/jianminchen/c7feb15436e8eaa6ceac5b9253f78b54>

1. <https://www.hackerrank.com/contests/world-codesprint-5/challenges/longest-increasing-subsequence-arrays>

2. <https://gist.github.com/jianminchen/c7feb15436e8eaa6ceac5b9253f78b54>

---

## Balanced forest - HackerRank - world codesprint #5 (2016-07-25 00:23)

July 25, 2016

Spent more than 30 minutes to read the problem statement:

[1]<https://www.hackerrank.com/contests/world-codesprint-5/challenges/balanced-forest>

Try to come out the idea to solve the problem.

Will come back later.

1. <https://www.hackerrank.com/contests/world-codesprint-5/challenges/balanced-forest>

---

## Build a palindrome - HackerRank world codesprint #5 (2016-07-25 00:28)

July 25, 2016

Read the problem statement more than 30 minutes:

[1]<https://www.hackerrank.com/contests/world-codesprint-5/challenges/challenging-palindromes>

Try to come out the idea to solve the problem first. (Advanced problem) - Learning starts from reading the analysis. Prepare for advanced level challenges from HackerRank, one by one.

Read the editorial notes:

[2]<https://www.hackerrank.com/contests/world-codesprint-5/challenges/challenging-palindromes/editorial>

Here are a list terms to review:

1. Consider that  $\geq (L+1)/2$  characters are present in the first string.
2. string hashing/ a palindromic tree
3. Iterate on each index

$i$

of the given string  $a$  and consider the longest palindrome starting from

$i$

as a part of your solution to find the maximum length possible,  $L$ , for string  $s$ .

4. Suffix array
5. LCP - largest common prefix array

Required Knowledge:

Suffix Array, Palindromic Tree, String Hashing, Implementation

one selected code to study - Just use it as an example to motivate herself to work hard one by one on small topic, and then, one day in the future, build complicated stuff like this challenge.

C # code to study:

[3]<https://gist.github.com/jianminchen/8dbcd3ddafec645498b2a60b7edc1ce3>

Java code to study:

[4]<https://gist.github.com/jianminchen/5817849853dc5643677f74e4ea88c06c>

C++ code to study:

[5]<https://gist.github.com/jianminchen/f1a54449fc4d5ab216a84c8e59148745>

Previous blog about suffix array:

[6]<http://juliachencoding.blogspot.ca/2016/04/hackerrank-string-calculate-function-iii.html>

1. <https://www.hackerrank.com/contests/world-codesprint-5/challenges/challenging-palindromes>
2. <https://www.hackerrank.com/contests/world-codesprint-5/challenges/challenging-palindromes/editorial>
3. <https://gist.github.com/jianminchen/8dbcd3ddafec645498b2a60b7edc1ce3>
4. <https://gist.github.com/jianminchen/5817849853dc5643677f74e4ea88c06c>
5. <https://gist.github.com/jianminchen/f1a54449fc4d5ab216a84c8e59148745>
6. <http://juliachencoding.blogspot.ca/2016/04/hackerrank-string-calculate-function-iii.html>



## Longest common prefix - LCP - facebook code lab (2016-07-25 21:42)

July 25, 2016

Write a function to find the longest common prefix string amongst an array of strings.

Longest common prefix for a pair of strings S1 and S2 is the longest string S which is the prefix of both S1 and S2.

As an example, longest common prefix of "abcdefgh" and "abcefg" is "abc".

Given the array of strings, you need to find the longest S which is the prefix of ALL the strings in the array.

Example:

Given the array as:

```
[  
  "abcdefgh",  
  "aefghijk",  
  "abcefg"  
]
```

The answer would be "a".

Plan to work on the problem

Blog reading:

1. Talk about CMU computer science - network security course in 2015 - good to know the college - computing teaching

<http://article.heron.me/2015/05/cmu-nctu-final.html>

course project:

[https://github.com/heronyang/youtube\\_fake\\_view/blob/master/doc/paper.pdf](https://github.com/heronyang/youtube_fake_view/blob/master/doc/paper.pdf)

2. Four algorithm questions to review

<http://codechen.blogspot.ca/search/label/interview>

Inspired by the above blog, write code for **binary tree path sum - two with same value checking** :

[1]<https://gist.github.com/jianminchen/b5aa95fb574fcb6f4135f88a4b47d5f8>

checklist of code style and design issues:

[2]<https://gist.github.com/jianminchen/b5aa95fb574fcb6f4135f88a4b47d5f8>

2.1. if statement is minimum - avoid using if statement, tips: let it fall through base case.

2.2. code style:

[3]Readable code

[4]Clean code

2.3. use LINQ to code select clause like SQL statement

2.4. Avoid early return when the duplicate path sum is added.

2.5. Let main function to take care of discussion of two path sum with same value

### 3. Find k most frequent numbers in the array

Problem:

```
// nums = [5, 3, 1, 1, 1, 3, 73, 1]
```

```
// k = 1
```

```
// return [1]
```

```
// k = 2
```

```
// return [1, 3]
```

```
// k = 3
```

```
// return [1, 3, 5]
```

First, go through the array once, and keep the count for every distinct value:

We can use LINK to call order by and then get Top k values.

But, in the analysis of the top k values in the array -

[5]<http://www.geeksforgeeks.org/k-largestor-smallest-elements-in-an-array/>

1. sorting the array  $O(n \log n)$
2. use selection sort  $O(nk)$
3. use sorting
4. use min heap
5. use max heap
6. use temporary array
7. use order statistics

7A. randomized selection algorithm - a deterministic algorithm that runs in  $O(n)$  in the worst case

<http://www.cse.ust.hk/dekai/271/notes/L05/L05.pdf>

1. the idea is to divide n items into  $n/5$  sets (denoting m sets), each contains 5 items.  $O(n)$
2. Find the median of each of the m sets.  $O(n)$
3. Take those m medians and put them in another array. Use Dselection() to recursively calculate the median of these medians. Call this x.  $T(n/5)$
4. ...

7B. Use QuickSort partition algorithm to partition around the kth largest number  $O(n)$ .

7C. Sort the k-1 elements (elements greater than the kth largest element)  $O(k \log k)$ . This step is needed only if sorted output is required.

1. <https://gist.github.com/jianminchen/b5aa95fb574fcb6f4135f88a4b47d5f8>
  2. <https://gist.github.com/jianminchen/b5aa95fb574fcb6f4135f88a4b47d5f8>
  3. <http://juliachencoding.blogspot.ca/2016/04/the-art-of-readable-code-book-review.html>
  4. <http://juliachencoding.blogspot.ca/2016/04/clean-code-book-reading.html>
  5. <http://www.geeksforgeeks.org/k-largestor-smallest-elements-in-an-array/>
- 

## longest palindrome - facebook code lab (2016-07-25 21:42)

July 25, 2016

Problem statement:

Given a string  $S$ , find the longest palindromic substring in  $S$ .

Substring of string  $S$ :

$S[i...j]$  where  $0 \leq i \leq j < \text{len}(S)$

Palindrome string:

A string which reads the same backwards. More formally,  $S$  is palindrome if  $\text{reverse}(S) = S$ .

In case of conflict, return the substring which occurs first (with the least starting index).

Example :

Input : "aaaabaaa"

Output : "aaabaaa"

**Plan to work on java coding in short future.**

**Hint:**

**Brute force:**

How many substrings? - start position and end position,  $O(N^2)$  variables, and each substring needs one palindrome check, so time complexity is  $O(N^3)$ .

**Facebook code lab editorial hint:**

A simpler approach,  $O(N^2)$

$N^2$

) time and  $O(1)$  space:

In fact, we could solve it in  $O(N^2)$

$N^2$

) time without any extra space.

We observe that a palindrome mirrors around its center. Therefore, a palindrome can be expanded from its center, and there are only  $2N-1$  such centers.

You might be asking why there are  $2N-1$  but not  $N$  centers?

The reason is that the center of a palindrome can be in between two letters.

Such palindromes have even number of letters (such as "abba") and their center are between the two 'b's.

Since expanding a palindrome around its center could take  $O(N)$  time, the overall complexity is  $O(N^2)$ .

### 1. first practice: (in Java)

[1]<https://gist.github.com/jianminchen/08126ba3d168498695e619f7fded444a>

highlights:

1. String - compile error: string, Java String class, not string. Different from C #
2. String.substring(int beginIndex, endIndex), endIndex is exclusive <- understand exclusive meaning here: line 66, line 90
3. Java String [] operator - compiler error, using charAt() function
4. line 33 - 36 bug removal - use extra boolean update, since line 35 - update maxLength, so line 36 update boolean should not be used. Saved status
5. Two for loops can be merged into one loop
6. Design concern, use start pos and end pos, substring has  $O(n^2)$ , but if only consider the center position of palindrome, only  $O(n)$  case.

2. 2nd practice - merge two for loops

[2]<https://gist.github.com/jianminchen/ee7e6512cd8fbffb1895e51ebaa4487c>

### Blogs to read:

[3][https://segmentfault.com/a/1190000006059081?utm\\_source=weekly&utm\\_medium=email&utm\\_campaign=email\\_weekly](https://segmentfault.com/a/1190000006059081?utm_source=weekly&utm_medium=email&utm_campaign=email_weekly)

[4]<http://blog.iderzheng.com/coding-is-not-everything/>

[5]<http://code.iderzheng.com/leetcode/>

[6]<http://blog.iderzheng.com/i-like-interview/>

Excellent analysis - brute force  $O(n^4)$  ->  $O(n^3)$

[7]<http://blog.iderzheng.com/longest-common-substring-problem-optimization/>

1. <https://gist.github.com/jianminchen/08126ba3d168498695e619f7fded444a>

2. <https://gist.github.com/jianminchen/ee7e6512cd8fbffb1895e51ebaa4487c>

3. [https://segmentfault.com/a/1190000006059081?utm\\_source=weekly&utm\\_medium=email&utm\\_campaign=email\\_weekly](https://segmentfault.com/a/1190000006059081?utm_source=weekly&utm_medium=email&utm_campaign=email_weekly)

4. <http://blog.iderzheng.com/coding-is-not-everything/>

5. <http://code.iderzheng.com/leetcode/>

6. <http://blog.iderzheng.com/i-like-interview/>

7. <http://blog.iderzheng.com/longest-common-substring-problem-optimization/>

---

### substring - facebook code lab - Leetcode 30 (2016-07-25 21:43)

July 25, 2016

You are given a string, S, and a list of words, L, that are all of the same length.

Find all starting indices of substring(s) in S that is a concatenation of each word in L exactly once and without any intervening characters.

Example :

S: "barfoothefoobarman"

L: ["foo", "bar"]

You should return the indices: [0,9].

(order does not matter).

Plan to work on the problem

Get Java, C++ code:

1. C++ solution:

study C++ code:

[1]<https://gist.github.com/jianminchen/1eb3d40f6173f07c7db90d39c1c71edf>

from blog:

[2]<https://yujia.io/blog/2015/11/17/LeetCode-30-Substring-with-Concatenation-of-All-Words/>

2. sliding windows method - introduction - great idea - ...

study Java code:

[3]<https://gist.github.com/jianminchen/c1f4d4f0b53cd197e3cfaa968a76f62e>

from the blog:

[4]<http://yuanhsh.iteye.com/blog/2187543>

Review sliding window blog:

[5]<http://juliachencoding.blogspot.ca/2015/07/leetcode-longest-substring-without.html>

1. <https://gist.github.com/jianminchen/1eb3d40f6173f07c7db90d39c1c71edf>

2. <https://yujia.io/blog/2015/11/17/LeetCode-30-Substring-with-Concatenation-of-All-Words/>

3. <https://gist.github.com/jianminchen/c1f4d4f0b53cd197e3cfaa968a76f62e>

4. <http://yuanhsh.iteye.com/blog/2187543>

5. <http://juliachencoding.blogspot.ca/2015/07/leetcode-longest-substring-without.html>

## Pascal's triangle - facebook code lab (2016-07-25 21:44)

July 25, 2016

Given numRows, generate the first numRows of Pascal's triangle.

Pascal's triangle : To generate A[C] in row R, sum up A'[C] and A'[C-1] from previous row R - 1.

Example:

Given numRows = 5,

Return

```
[
  [1],
  [1,1],
  [1,2,1],
  [1,3,3,1],
  [1,4,6,4,1]
]
```

Plan to work on the problem.

Blogs to read:

1. [1]<http://cenalulu.github.io/linux/all-about-cpu-cache/>
2. [2]<http://cenalulu.github.io/mysql/how-i-become-a-facebook-dba/>
3. [3]<http://cenalulu.github.io/python/online-programming-test/>
4. [4]<http://cenalulu.github.io/python/euler-project-experience/>

1. <http://cenalulu.github.io/linux/all-about-cpu-cache/>
  2. <http://cenalulu.github.io/mysql/how-i-become-a-facebook-dba/>
  3. <http://cenalulu.github.io/python/online-programming-test/>
  4. <http://cenalulu.github.io/python/euler-project-experience/>
- 

## Factorial - facebook code lab (2016-07-25 21:45)

July 25, 2016

Given an integer n, return the number of trailing zeroes in n!.

Note: Your solution should be in logarithmic time complexity.

Example :

678

```
n = 5
n! = 120
Number of trailing zeros = 1
So, return 1
```

Plan to work on the problem.

Blogs to read:

1. Find maximum length snake sequence:

[1]<http://www.geeksforgeeks.org/find-maximum-length-snake-sequence/>

2. Root to leaf path sum equal to a given number

[2]<http://www.geeksforgeeks.org/root-to-leaf-path-sum-equal-to-a-given-number/>

Blog to read:

1. How to construct a blog in 10 minutes? Using github and jekyll?

[3]<http://cenalulu.github.io/jekyll/how-to-build-a-blog-using-jekyll-markdown/>

2. Buy a domain, buy a Alibaba cloud virtual computer, Nginx + WordPress - set up youownsite.com

[4]<http://storypku.com/2014/05/%E5%88%9B%E7%AB%99%E8%AE%B0/>

```
1. http://www.geeksforgeeks.org/find-maximum-length-snake-sequence/
2. http://www.geeksforgeeks.org/root-to-leaf-path-sum-equal-to-a-given-number/
3. http://cenalulu.github.io/jekyll/how-to-build-a-blog-using-jekyll-markdown/
4. http://storypku.com/2014/05/%E5%88%9B%E7%AB%99%E8%AE%B0/
```

---

## Lintcode 79: longest common substring (2016-07-26 21:59)

July 26 2016

Work on lintcode: longest common substring.  
problem statement:

Given two strings, find the longest common substring.

Return the length of it.

### Note

The characters in substring should occur continuously in original string. This is different with subsequence.

Study the blog:

[1]<http://blog.iderzheng.com/longest-common-substring-problem-optimization/>

1. Brute force solution -> from  $O(n^4)$  to  $O(n^3)$ , great analysis in the above blog:

Good thinking adds value to your coding practice:

Brute force - get smart on brute force solution:

For example, a string `s1 = "abcdefg"`,

One way to think about brute force:

The length of string is 7.

How many substring in `s1`, substring - start position,

it can be 0 to 6, denoted as `i`, and then,

end position from `i` to `n-1`.

the variation formula,  $\text{Sum} = (n-1) + (n-2) + \dots + 1 = (n-1)n/2$ ,

So, the total is  $O(n^2)$ ;

Try to reduce brute force variation from  $O(n^2)$  to  $O(n)$ , instead of letting start position and end position both varies, just work on start position only.

Second way to think about using brute force:

the start position of substring is from 0 to `n-1`, so considering the start position, start a new search.

So, the total of search is  $O(n)$ .

Longest common substring - start from start-position, and then, compare both of chars are equal, if yes, continue, record length and compare to maximum length, else then break the search.

1. C++ code:

[2]<https://gist.github.com/jianminchen/cdfebbc432600467f300c9e914352574>

From blog:

[3]<http://blog.iderzheng.com/longest-common-substring-problem-optimization/>

The time complexity is  $O(n^3)$ . There is duplicated calculation.

Will write C # practice very soon.

2. Use Dynamic programming, in the above blog, the analysis is very helpful.

Work on dynamic programming, improve time complexity from  $O(n^4)$  or  $O(n^3)$  to  $O(n^2)$ , using memorization, space  $O(n^2)$ , bottom up approach.

The idea is to find the formula of DP - dynamic programming.

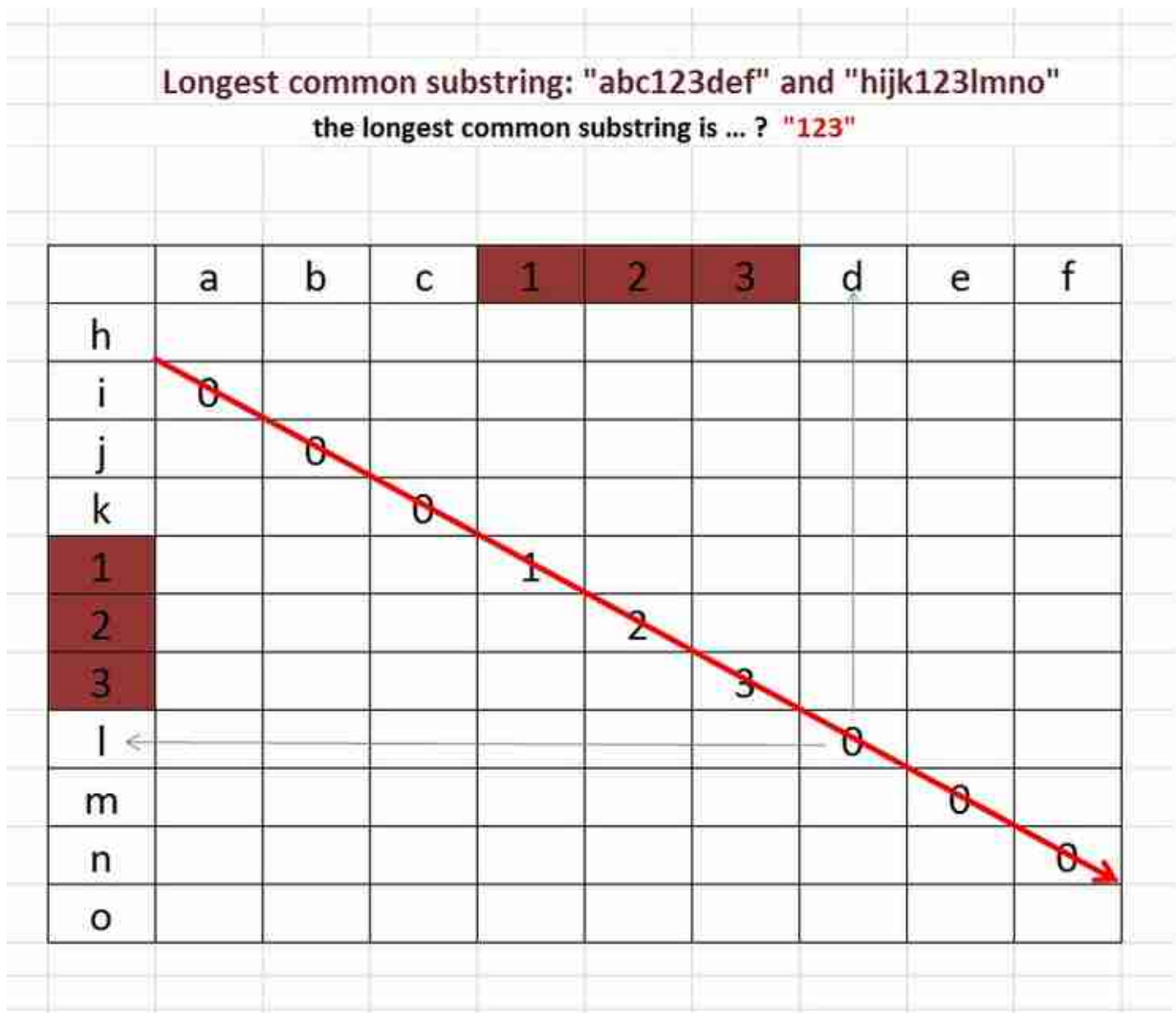
table `T(i, j)` - common substrings, using end position as a variable.



one is in s1, ending at position s1[i];  
 one is in s2, ending at position s2[j].

We know that if  $T(i, j) > 0$ , then,  $s1[i] = s2[j]$ ;  
 if  $(s1[i+1] == s2[j+1])$ , then,  $T[i+1, j+1] = T[i, j] + 1$ ,  
 otherwise,  $T[i+1, j+1] = 0$ .

Will think about to put together a graph here to explain the idea as well.



2. C++ code:

[4]<https://gist.github.com/jianminchen/315120970b40d829d6a17ef59242be8f>

From blog: C++ code

DP solution, time complexity  $O(nm)$ , space  $O(nm)$

[5]<http://blog.iderzheng.com/longest-common-substring-problem-optimization/>

3. further improvement: C++ code

DP solution, time complexity  $O(nm)$ , space  $O(nm) \rightarrow O(n+m) \rightarrow O(1)$

because the recurrence formula tells us that the current position only relies on diagonal position - left-up corner  $(i-1, j-1)$

[6]<https://gist.github.com/jianminchen/0061dcf562bd0bdb091301241c38730f>

From blog:

[7]<http://blog.iderzheng.com/longest-common-substring-problem-optimization/>

Julia's practice:

1. brute force solution C #

A. first writing, static analysis catching 1 bug, left 2 bugs for debugging. (not so good!)

[8]<https://gist.github.com/jianminchen/66e8b6637e68cd0a94e6c0ab99509b35>

B. fix all bugs - final presentation: C #

[9]<https://gist.github.com/jianminchen/b75fb58837ffc49abdb6b04aa6218f02>

2. Dynamic programming solution using C #:

[10]<https://gist.github.com/jianminchen/dc76e5baa3cb5d570609356a23b2afe4>

highlights of code writing and execution:

1. static analysis - find bugs

change made: line 56 - 61, if the longest common substring is with length 1 and also start from row 0 or col 0.  
line 67 - 72

2. test case failed on line `longestCommonSubstring("abc1def", "1ghijk l")` - should be "1",

change made: move `end1` variable to line 49, and set variable from line 56 - 61, line 67 - line 72

2nd version: add comments

[11]<https://gist.github.com/jianminchen/4f7dbad13f441beb14e1e2cadb60687e>

3. **DP solution with space reduction:  $O(nm)$  ->  $O(n+m)$  ->  $O(1)$**

practice later.

Design issues:

4 variables - memo, longest, end1, searchFirstRowCol

1. memorization using two dimension array - memo

2. variable int longest - get maximum length

3. variable int end1 - string s1 - end position - s1's substring end position

4. variable bool searchFirstRowCol - check first row and first col to update maximum length

DP problems:

[12]<http://ihuafan.com/%E7%AE%97%E6%B3%95/lintcode-dynamic-programming-summary>

**Coding practice is like sports - I don't feel fear when I am on court. That's where I feel at home.**

I don't feel fear when I am on court. That's where I feel at home [13] #CreateYourMark [14] #RG16  
[15]@adidastennis [16]@adidasFR [17][pic.twitter.com/ZVNcZvWZEO](https://pic.twitter.com/ZVNcZvWZEO)

— Kristina Mladenovic (@KikiMladenovic) [18]May 20, 2016

1. <http://blog.iderzheng.com/longest-common-substring-problem-optimization/>
  2. <https://gist.github.com/jianminchen/cdfebbc432600467f300c9e914352574>
  3. <http://blog.iderzheng.com/longest-common-substring-problem-optimization/>
  4. <https://gist.github.com/jianminchen/315120970b40d829d6a17ef59242be8f>
  5. <http://blog.iderzheng.com/longest-common-substring-problem-optimization/>
  6. <https://gist.github.com/jianminchen/0061dcf562bd0bdb091301241c38730f>
  7. <http://blog.iderzheng.com/longest-common-substring-problem-optimization/>
  8. <https://gist.github.com/jianminchen/66e8b6637e68cd0a94e6c0ab99509b35>
  9. <https://gist.github.com/jianminchen/b75fb58837ffc49abdb6b04aa6218f02>
  10. <https://gist.github.com/jianminchen/dc76e5baa3cb5d570609356a23b2afe4>
  11. <https://gist.github.com/jianminchen/4f7dbad13f441beb14e1e2cadb60687e>
  12. <http://ihuaфан.com/%E7%AE%97%E6%B3%95/lintcode-dynamic-programming-summary>
  13. <https://twitter.com/hashtag/CreateYourMark?src=hash>
  14. <https://twitter.com/hashtag/RG16?src=hash>
  15. <https://twitter.com/adidastennis>
  16. <https://twitter.com/adidasFR>
  17. <https://t.co/ZVncZvWZEO>
  18. <https://twitter.com/KikiMladenovic/status/733569899964989441>
- 

## Binary Tree Path Sum - two with same value checking (2016-07-28 20:06)

July 28, 2016

Problem:

Write a function that given a tree, returns true if at least 2 paths down the tree have the same sum. A path is a series of nodes from the root to the leaf.

```
// ex1:
// 2
// / \
// 4 5
// /
// 1
// returns true // 2+4+1 = 2+5
// ex2:
// 3
// /
// 4
// / \
// 1 1
// returns true // 3+4+1 = 3+4+1
// ex3:
// 1
// / \
// 3 4
// returns false // 1+3 != 1+4
```

Write code for **binary tree path sum - two with same value checking** :

Julia's first writing using C # language:

[1]<https://gist.github.com/jianminchen/b5aa95fb574fcb6f4135f88a4b47d5f8>

checklist of code style and design issues:

[2]<https://gist.github.com/jianminchen/b5aa95fb574fcb6f4135f88a4b47d5f8>

2.1. if statement is minimum - avoid using if statement.

**Tips: let it fall through base case.**

2.2. code style:

[3]Readable code

[4]Clean code

2.3. use LINQ to code select clause like SQL statement

2.4. Avoid early return when the duplicate path sum is added. (line 147 - line 154)

2.5. Let main function to take care of discussion of two path sum with same value (line 123 - 128)

2.6 It is part of the design, one path sum is added to the dictionary twice; so if there are two path sum with same value, the value of dictionary  $\geq 4$  instead  $\geq 2$ .

### Questions and answers:

1. Which blogs helps you to shape the idea to solve the problem quickly?

When Julia worked on lowest common ancestor, she used this blog to write a version as well.

[5]<http://www.geeksforgeeks.org/lowest-common-ancestor-binary-tree-set-1/>

One of Julia's practice:

[6]<http://juliachencoding.blogspot.ca/2016/04/leetcode-236-lowest-common-ancestor-in.html>

2. What is most important lessons learned through the practice?

Julia tried to avoid if statement when she designed the solution. Let base case take care of if statement only. Make the code simple as possible, therefore, design of the recursive function is void. line 143 - 162, function **pathSumTracking** .

1. <https://gist.github.com/jianminchen/b5aa95fb574fcb6f4135f88a4b47d5f8>

2. <https://gist.github.com/jianminchen/b5aa95fb574fcb6f4135f88a4b47d5f8>

3. <http://juliachencoding.blogspot.ca/2016/04/the-art-of-readable-code-book-review.html>

4. <http://juliachencoding.blogspot.ca/2016/04/clean-code-book-reading.html>
  5. <http://www.geeksforgeeks.org/lowest-common-ancestor-binary-tree-set-1/>
  6. <http://juliachencoding.blogspot.ca/2016/04/leetcode-236-lowest-common-ancestor-in.html>
- 

## Leetcode 347: Find k most frequent numbers in the array - C# solution (2016-07-28 23:00)

July 28, 2016

### Find k most frequent numbers in the array

Problem:

```
// nums = [5, 3, 1, 1, 1, 3, 73, 1]
```

```
// k = 1
```

```
// return [1]
```

```
// k = 2
```

```
// return [1, 3]
```

```
// k = 3
```

```
// return [1, 3, 5]
```

C # solution using Dictionary<int,int> and language Integrate Query (LINQ):

First, go through the array once, and keep the count for every distinct value:

We can use

### Language Integrated Query

(LINQ) to call order by and then get Top k values.

Write a solution using C # first, post code here.

1. First practice, using LINQ, OrderByDescending, Take

[1]<https://gist.github.com/jianminchen/1941a6b82525feefe15a2c477e0a0a0b>

LINQ reference:

[2]<http://stackoverflow.com/questions/5344805/linq-orderby-descending-query>

[3]<http://stackoverflow.com/questions/192203/whats-the-linq-to-sql-equivalent-to-top-or-limit-offset>

### Actionable Item:

Write one solution using bucket sort to get k most frequent numbers in the array.

1. <https://gist.github.com/jianminchen/1941a6b82525feefe15a2c477e0a0a0b>

2. <http://stackoverflow.com/questions/5344805/linq-orderby-descending-query>

3. <http://stackoverflow.com/questions/192203/whats-the-linq-to-sql-equivalent-to-top-or-limit-offset>

## Leetcode 347 - Find k most frequent numbers in the array - C++/ Java Solutions (2016-07-29 19:30)

July 28, 2016

### Find k most frequent numbers in the array

Problem:

```
// nums = [5, 3, 1, 1, 1, 3, 73, 1]
```

```
// k = 1
```

```
// return [1]
```

```
// k = 2
```

```
// return [1, 3]
```

```
// k = 3
```

```
// return [1, 3, 5]
```

### Code study: nothing can compare to code reading

**C++** code to study:

use unordered\_map and priority\_queue

1. [1]<https://gist.github.com/jianminchen/cd9f536708f6ec42ae229d853c88136> 1

2. use bucket sort -

[2]<https://gist.github.com/jianminchen/c3d49c11c090b91c94ae7b05bdc11786>

[3]<https://gist.github.com/jianminchen/88f3e2d441c62ecb7d9d706d1b9bda37>

3. use Map + std::sort

[4] <https://gist.github.com/jianminchen/3e5bba61847c5c84d95e1ca7806c81be>

**Java code** to study:

1. use a min-heap - Java PriorityQueue class - underneath heap sort

[5]<https://gist.github.com/jianminchen/76b2b3196a4e58312e1ba1d0c288d941>

2. use heap:

686

[6]<https://gist.github.com/jianminchen/2482dbacb59e464d94c4a4d16cbc6d3b>

3. use bucket sort:

[7]<https://gist.github.com/jianminchen/0626a0cb673526d4b7b58698004b87b8>

4. use Collections.sort - underneath merge sort,

[8]<https://gist.github.com/jianminchen/3d8fab01965efd19a0f72b2109501c8d>

5. use quicksort partition

[9]<https://gist.github.com/jianminchen/99d2dea800b28e24456827946409d32d>

### Conclusion:

Based on the analysis, the time complexity should be better than  $O(n \log n)$ , if using heap sort or merge sort, when  $k$  is big enough close to  $n$ , then,  $O(n \log k)$  is close to  $O(n \log n)$ .

However, the bucket sort is always  $O(n)$ , so the time complexity is  $O(n)$ , no matter  $k$ 's size.

1. <https://gist.github.com/jianminchen/cd9f536708f6ec42ae229d853c881361>
  2. <https://gist.github.com/jianminchen/c3d49c11c090b91c94ae7b05bdc11786>
  3. <https://gist.github.com/jianminchen/88f3e2d441c62ecb7d9d706d1b9bda37>
  4. <https://gist.github.com/jianminchen/3e5bba61847c5c84d95e1ca7806c81be>
  5. <https://gist.github.com/jianminchen/76b2b3196a4e58312e1ba1d0c288d941>
  6. <https://gist.github.com/jianminchen/2482dbacb59e464d94c4a4d16cbc6d3b>
  7. <https://gist.github.com/jianminchen/0626a0cb673526d4b7b58698004b87b8>
  8. <https://gist.github.com/jianminchen/3d8fab01965efd19a0f72b2109501c8d>
  9. <https://gist.github.com/jianminchen/99d2dea800b28e24456827946409d32d>
- 

## Introduction to LINQ Queries (C#) - Language Integrate Query (LINQ) study series (I) (2016-07-29 20:43)

July 28, 2016

LINQ:

[1]<https://msdn.microsoft.com/en-us/library/bb397906.aspx>

### Study more on LINQ:

1. LINQ - introduction

[2]<https://msdn.microsoft.com/en-us/library/bb397906.aspx>

Languages for different data source:

SQL - relational database

XQuery - XML

LINQ -

a consistent model

basic coding patterns

For example: ? patterns -

Five most popular data source: (5 sources: A, N, S, X, any)

ADO.NET Datasets

.NET collections

SQL databases

XML documents

any other format

## 2. More reading about LINQ:

Google search keywords:

1. basic code patterns used for LINQ

2. .NET collections

[3][https://msdn.microsoft.com/en-us/library/bb399365\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/bb399365(v=vs.110).aspx)

3. Generics in .NET framework

[4][https://msdn.microsoft.com/en-us/library/ms172192\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ms172192(v=vs.110).aspx)

## 4. LINQ - blogs to read:

[5]<http://igoro.com/archive/7-tricks-to-simplify-your-programs-with-linq/>

[6]<http://igoro.com/archive/extended-linq-additional-operators-for-linq-to-objects/>

5.

[7]<http://www.asp.net/mvc/overview/older-versions-1/models-data/creating-model-classes-with-linq-to-sql-cs>

1. <https://msdn.microsoft.com/en-us/library/bb397906.aspx>

2. <https://msdn.microsoft.com/en-us/library/bb397906.aspx>

3. [https://msdn.microsoft.com/en-us/library/bb399365\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/bb399365(v=vs.110).aspx)

4. [https://msdn.microsoft.com/en-us/library/ms172192\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ms172192(v=vs.110).aspx)

5. <http://igoro.com/archive/7-tricks-to-simplify-your-programs-with-linq/>

6. <http://igoro.com/archive/extended-linq-additional-operators-for-linq-to-objects/>

7. <http://www.asp.net/mvc/overview/older-versions-1/models-data/creating-model-classes-with-linq-to-sql-cs>

---

## 3+ tips - how to read technical articles quickly (2016-07-29 20:45)

July 29, 2016

Small research - how to save time and more efficient to read technical articles?

Read technical articles - how to save time and more efficient?

Any order in the following:

1. Write down some notes:

keywords

things to remember



things not so clear

things to help memorize

2. Write down the ideas in the article -

short

,

concise

,

keywords only

And also,

sort by alphabetical order

( **rate 10 out of 10** )

Separate Facts/ Arguments/ Rules/ Tips

(rate 9 out of 10)

short chars to iterate (ex: S.O.L.I.D. Principle, easy to remember)

(rate 8 out of 10)

Search more on google...

3. Get organized, maybe, better/ clear/ more pleasant ways

---

**.NET collections - Language Integrate Query - LINQ study series (II) (2016-07-29 20:47)**

**July 29, 2016**

**More reading about LINQ:**

Google search keywords:

1. basic code patterns used for LINQ

2. .NET collections

[1][https://msdn.microsoft.com/en-us/library/bb399365\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/bb399365(v=vs.110).aspx)

**Notes from the above article:**

LINQ queries

in-memory objects

object type

IEnumerable

IEnumerable<T>

object type implements IEnumerable or IEnumerable<T>

LINQ common pattern for accessing data vs standard foreach loops

more concise and readable

LINQ 3 capabilities:

filtering

ordering

grouping capabilities

LINQ

improve performance

common variations of data collections:

hash tables,

queues,

stacks,

bags ?

dictionaries

lists

3 Interfaces:

ICollection

ICollection

IDictionary

generic counterparts

ICollection -> IList

ICollection -> IDictionary

3 examples based on IList

Array

ArrayList

List<T>

4 examples based on ICollection - 1 value

Queue,

ConcurrentQueue<T>

Stack

ConcurrentStack<T>

LinkedList<T>

5 examples based on IDictionary - 2 values, both a key and a value

Hashtable

SortedList

Dictionary<TKey,TValue>

SortedList<TKey, TValue>

ConcurrentDictionary<TKey, TValue>

Special one - a list of values with keys embedded within the values and, therefore, it behaves like a list and like a dictionary

KeyedCollection<TKey,TItem>

class vs generic classes ?

Generic collections

strong typing

**Ready for this - memorize the claim: Julia likes it - reading is quickest way to become an expert!**

Generic collections are the best solution to strong typing

- Julia likes strong typing, find problems in compile time, fix bugs in static analysis, no debugging/test cases' help
- Design the function to less error prone - as simple as possible
- do one task only

Facts? Look into google:

some languages does not support generics - which one, not C #

behaviors:

how they are sorted

how searches are performed

how comparisons are made

1. [https://msdn.microsoft.com/en-us/library/bb399365\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/bb399365(v=vs.110).aspx)

---

## **Generics in .NET framework - Language Integrate Query (LINQ) study series (III) (2016-07-29 20:49)**

**July 29, 2016**

**Generics - article reading:**

[1][https://msdn.microsoft.com/en-us/library/ms172192\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ms172192(v=vs.110).aspx)

Notes:

Advantages:

code reusability and type safety

Better performance - no need to box the value types

type-safe callbacks

lightweight dynamic methods vs entire assemblies

compiler will help to do type safety - enforce at compile time

The need for type casting and the possibility of run-time errors are reduced.

Generics streamline dynamically generated code.

placeholder (type parameters) for one or more of the type that they store or use.

1. [https://msdn.microsoft.com/en-us/library/ms172192\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ms172192(v=vs.110).aspx)

---

## **Simplify your programs with LINQ - Language Integrate Query (LINQ) study series IV (2016-07-29 20:51)**

**July 29, 2016**

### **LINQ - blogs to read:**

[1]<http://igoro.com/archive/7-tricks-to-simplify-your-programs-with-linq/>

[2]<http://igoro.com/archive/extended-linq-additional-operators-for-linq-to-objects/>

Notes:

1. Initialize an array

```
int[] a = Enumerable.Repeat(-1,10).ToArray();
```

```
int[] b = Enumerable.Range(0,10).ToArray();
```

```
int[] c = Enumerable.Range(0,10).Select(i=>100+10*i).ToArray();
```

Enumerable.Repeat, Range,

no for loops necessary, using LINQ

2. Iterate over multiple arrays in a single loop

```
foreach(var x in array1)
```

```
DoSomething(x);
```

```
foreach(var x in array2)
```

```
DoSomething(x);
```

LINQ

```
foreach(var x in array1.Concat(array2))
```

```
DoSomething(x);
```

LINQ operates at the enumerator level (?), it will not allocate a new array to hold elements of array1 and array2. So, space-efficient.

3. Generate a random sequence

```
Random rand = new Random();
```

```
var randomSeq = Enumerable.Repeat(0,N).Select(i=>rand.Next());
```

lazy nature of LINQ, the sequence is not pre-computed and stored in an array, but instead random numbers are generated on-demand, as you

iterate over randomSeq.

#### 4. Generate a string

Generate a string with the repeating pattern "ABCABCABC..." of length N.

```
string str = new string(Enumerable.Range(0,N).Select(i => (char)('A' + i %3)).ToArray());  
string values = string.Join(string.Empty, Enumerable.Repeat(pattern, N).ToArray());
```

#### 5. Convert sequences or collections

```
IEnumerable<string> strEnumerable = ...;  
IEnumerable<object> objEnumerable = strEnumerable.Cast<object>();  
List<string> strList = ...;  
List<object> objList = new List<object>(strList.Cast<object>());  
var objList = strList.Cast<object>().ToList();
```

#### 6. Convert a value to a sequence of length 1

```
IEnumerable<int> seq = Enumerable.Repeat(myValue, 1);
```

#### 7. Iterate over all subsets of a sequence

For small input, three problems:

subset sum

boolean satisfiability

knapsack problem

NP-complete problems -

solve easily by iterating over all subsets of some sequence:

1. <http://igoro.com/archive/7-tricks-to-simplify-your-programs-with-linq/>
2. <http://igoro.com/archive/extended-linq-additional-operators-for-linq-to-objects/>

---

## Summary of study for Leetcode 347 - K most frequent element in the array (2016-07-29 20:54)

July 29, 2016

### Summary of study - selection algorithm and Language Integrated Query (LINQ)

1. [1] <http://www.geeksforgeeks.org/k-largestor-smallest-elements-in-an-array/>
  2. [2] <http://www.ardendertat.com/2011/10/27/programming-interview-questions-10-kth-largest-element-in-array/>
  3. [3] [https://en.wikipedia.org/wiki/Selection\\_algorithm](https://en.wikipedia.org/wiki/Selection_algorithm)
  4. [4] [https://msdn.microsoft.com/en-us/library/ms172192\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ms172192(v=vs.110).aspx)
- LINQ:
1. [5] <https://msdn.microsoft.com/en-us/library/bb397906.aspx>
  2. [6] [https://msdn.microsoft.com/en-us/library/bb399365\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/bb399365(v=vs.110).aspx)
  3. [7] <http://igoro.com/archive/7-tricks-to-simplify-your-programs-with-linq/>

1. <http://www.geeksforgeeks.org/k-largestor-smallest-elements-in-an-array/>
  2. <http://www.ardendertat.com/2011/10/27/programming-interview-questions-10-kth-largest-element-in-array/>
  3. [https://en.wikipedia.org/wiki/Selection\\_algorithm](https://en.wikipedia.org/wiki/Selection_algorithm)
  4. [https://msdn.microsoft.com/en-us/library/ms172192\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ms172192(v=vs.110).aspx)
  5. <https://msdn.microsoft.com/en-us/library/bb397906.aspx>
  6. [https://msdn.microsoft.com/en-us/library/bb399365\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/bb399365(v=vs.110).aspx)
  7. <http://igoro.com/archive/7-tricks-to-simplify-your-programs-with-linq/>
- 

## **K largest elements in the array - various ideas (2016-07-29 20:57)**

July 29, 2016

Top k values in the array - review the following article:

[1] <http://www.geeksforgeeks.org/k-largestor-smallest-elements-in-an-array/>

1. sorting the array  $O(n \log n)$
2. use selection sort  $O(nk)$
3. use sorting
4. use min heap
5. use max heap
6. use temporary array
7. use order statistics

7A. randomized selection algorithm - a deterministic algorithm that runs in  $O(n)$  in the worst case

[2] <http://www.cse.ust.hk/~dekai/271/notes/L05/L05.pdf>

1. the idea is to divide  $n$  items into  $n/5$  sets (denoting  $m$  sets), each contains 5 items.  $O(n)$
2. Find the median of each of the  $m$  sets.  $O(n)$
3. Take those  $m$  medians and put them in another array. Use `Dselection()` to recursively calculate the median of these medians. Call this  $x$ .  $T(n/5)$

4. ...

7B. Use QuickSort partition algorithm to partition around the kth largest number  $O(n)$ .

7C. Sort the k-1 elements (elements greater than the kth largest element)  $O(k \log k)$ . This step is needed only if sorted output is required.

1. <http://www.geeksforgeeks.org/k-largestor-smallest-elements-in-an-array/>

2. <http://www.cse.ust.hk/~dekai/271/notes/L05/L05.pdf>

---

## Binary search advanced (2016-07-31 23:27)

July 31, 2016

Study blogs:

[1]<http://blog.iderzheng.com/binary-search/>

[2]<http://blog.iderzheng.com/binary-search-advanced/>

Go over the blog, and write down great ideas one by one.

Will come back later.

1. <http://blog.iderzheng.com/binary-search/>

2. <http://blog.iderzheng.com/binary-search-advanced/>

---

## 2.8 August

### AWS study - pluralsight (2016-08-01 11:43)

August 1 2016

Canadian statutory holiday - civic day, choose some study material from pluralsight.com - learning AWS.

Lecturer's website:

[1]<http://app.pluralsight.com/author/richard-seroter>

Choose one or two in the following:

1. Amazon Web Service Databases in Depth

2. Architecting Highly Available Systems on AWS

3.

<https://app.pluralsight.com/library/courses/aws-auditing-environments-security-best-practices/table-of-contents>

Blog reading:

[2]<http://jane4532.blogspot.ca/>

I know what it takes to win. Forget everyone else and put the work in. [3][@jianminchenMP](#), you can do it. [4]<https://t.co/ZRlVZXlFeG> — jianmin chen (@jianminchen) [5]May 29, 2016

1. <http://app.pluralsight.com/author/richard-seroter>

2. <http://jane4532.blogspot.ca/>

3. <https://twitter.com/jianminchenMP>

4. <https://t.co/ZRlVZXlFeG>

5. <https://twitter.com/jianminchen/status/736795384991780865>

---

## Productivity Tips for the Busy Tech Professional - pluralsight.com (2016-08-01 11:48)

August 1, 2016

Lecture website:

[1]<http://app.pluralsight.com/author/richard-seroter>

One hour talk - great ideas!

Most favorite tips:

1.

Decompose big problems - never see big problem, always small problems

(10 out of 10)

2. Learn to say no, and leave buffer for important things to pop up.

3. Establishing a frame of mind - talk about his experience to write books etc.



I know what it takes to win. Forget everyone else and put the work in. [2]@jianminchenMP, you can do it. [3]<https://t.co/ZRlVZXlFeG> — jianmin chen (@jianminchen) [4]May 29, 2016

1. <http://app.pluralsight.com/author/richard-seroter>
2. <https://twitter.com/jianminchenMP>
3. <https://t.co/ZRlVZXlFeG>
4. <https://twitter.com/jianminchen/status/736795384991780865>

---

Richard Seroter (2016-08-02 22:43:17)  
I'm glad you found the course useful!

Jianmin Chen (2016-08-03 22:20:30)  
Hi, Richard, so glad to read your comment. The course is so great, I will watch one more time, and also take more notes.  
Thanks again. jianmin

## **HackerRank - Prepare to get experience on advanced level algorithms on HackerRank (2016-08-01 12:33)**

August 1, 2016

Choose a small topic to work on, when to choose to work on advanced algorithm and what to learn through the practice.

Pragmatic ideas:

1. How is the algorithm developed by editors? **Best algorithm lecture material to study** .
2. Study some code for classical problems through submissions.

Julia spent over 100 hours to work on HackerRank, solved over 50+ algorithms problems, and then, she is getting better to understand the problem statement on HackerRank. But she only chose to work on easy, medium difficult questions.

**Last time - 3 hours - really struggling - world code sprint #5 with advanced, difficult questions.**

### **Some facts on 3 hours activities:**

1. Tried to guess, break down small problems.
2. Wrote down some notes
3. Tried to guess what kind of problem it is - DP, DFS, graph, etc.

Julia likes to come back to review, and if she can write a blog on her 3 hours experience:

1. Spent 30+ minutes to read the problem statement
- [1]<http://juliachencoding.blogspot.ca/2016/07/build-forest-hackerrank-work.html>

2. Spent 30+ minutes to read the problem statement

[2]<http://juliachencoding.blogspot.ca/2016/07/build-palindrome-hackerrank-world.html>

### Actionable Items:

1. Work on suffix array first, learn basics first:

[3]<http://www.geeksforgeeks.org/suffix-array-set-1-introduction/>

2. Read the article, get all questions related to suffix array in the contest:

(plan to spend 2 hours to study)

[4]<http://www.stanford.edu/class/cs97si/suffix-array.pdf>

3. Review previous suffix array blog and C # implementation of suffix array:

[5][http://juliachencoding.blogspot.ca/search/label/suffix %20array %20C %23](http://juliachencoding.blogspot.ca/search/label/suffix%20array%20C%23)

1. <http://juliachencoding.blogspot.ca/2016/07/build-forest-hackerrank-world.html>

2. <http://juliachencoding.blogspot.ca/2016/07/build-palindrome-hackerrank-world.html>

3. <http://www.geeksforgeeks.org/suffix-array-set-1-introduction/>

4. <http://www.stanford.edu/class/cs97si/suffix-array.pdf>

5. <http://juliachencoding.blogspot.ca/search/label/suffix%20array%20C%23>

---

## Leetcode 124: Binary tree maximum path sum - a quick review (2016-08-04 21:22)

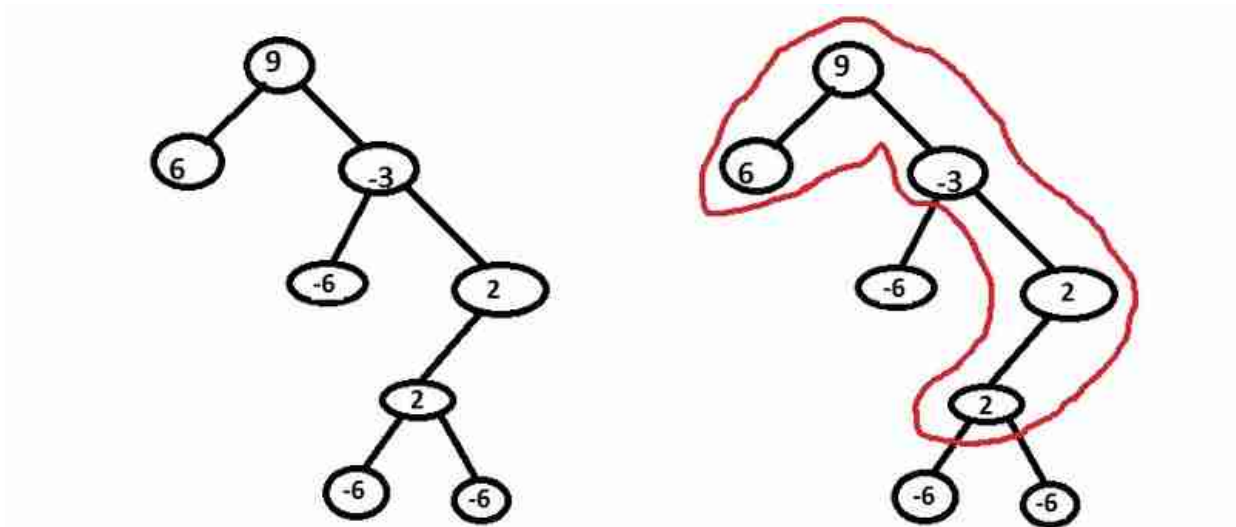
**August 4, 2016**

Come back to review the previous work, but it is still not easy to tell the ideas to solve the problem.

Let us talk about an example first:

In the practice on May 31, 2016, line 60 - 79, test case 3:

[1]<https://gist.github.com/jianminchen/578656e1079e8c58b08dd19f5b027e68>



The maximum path sum in the above tree is highlighted using red color, node 6->9->-3->2->2. Any two nodes in the tree can form a unique path, and the choice of path is  $N^2$ ,  $N$  is the total nodes in the tree.

Walk through the above example in the diagram and have some more discussion:

First talk about " **maximum value cross root** " - variable: `maxValueCrossRoot`,

each node is the root of its subtree, so the maximum one will be maximum one from the following list:

1. n1: root node (9): need to calculate `maximumEndByRoot` on right child (-3) first.
2. n2: left child (6):
3. n3: right child (-3):
4. n4: right->right child(-6): -6
5. n5: right->right->right (2): 4
6. n6: right->right->right->left(2) : 2
7. n7: right->right->right->left->left(-6): -6
8. n8: right->right->right->left->right(-6): -6

It is comparison by values.

**Tip** : 1.The idea to get the maximum value is to pass a reference int to any recursive function. Any subtree will have

one value, and compare with the global variable's value.

2. Use preorder traversal to travel tree once.

And " **maximum value end by root** " - variable: `maximumEndByRoot`

the above node `n8`: -6

`n7`: -6

`n6`: 2

`n5`: 4

`n3`: +1

`n2`: 6

So, the root node `maxValueCrossRoot` =  $9 + 6 + 1 = 16$

`maximumByEnd` = 15.

**Tip** : 1. use recursive function with return value - `maxValueEndByRoot`, the formula is easy to recall.

2. use preorder traversal to travel tree once.

The above case is coincident, the `maxValueCrossRoot` is ended at the root node `n1` (value 9), which may be anywhere (`ni`, `i` is one value from 1 to 8) in the tree.

### **Blogs about leetcode 124:**

Previous blogs about Leetcode 124:

1. blog 1:

[2]<http://juliachencoding.blogspot.ca/2015/07/leetcode-maximum-binary-tree-path-sum.html>

2. blog 2:

[3]<http://juliachencoding.blogspot.ca/2015/07/itint5-tree-maximum-path-sum.html>

3.

Practice on May 31, 2016, first version:

[4]<https://gist.github.com/jianminchen/432bdb1af38a1ec6f4a0741420891ebf>

Second version: add more comment:

[5]<https://gist.github.com/jianminchen/578656e1079e8c58b08dd19f5b027e68>

4. blog 4:

[6]<http://juliachencoding.blogspot.ca/2016/05/leetcode-124-binary-tree-maximum-path.html>

I like the encouraging verse. [7]<https://t.co/7wUhmyPoDx> — jianmin chen (@jianminchen) [8]June 5, 2016

1. <https://gist.github.com/jianminchen/578656e1079e8c58b08dd19f5b027e68>
2. <http://juliachencoding.blogspot.ca/2015/07/leetcode-maximum-binary-tree-path-sum.html>
3. <http://juliachencoding.blogspot.ca/2015/07/itint5-tree-maximum-path-sum.html>
4. <https://gist.github.com/jianminchen/432bdb1af38a1ec6f4a0741420891ebf>
5. <https://gist.github.com/jianminchen/578656e1079e8c58b08dd19f5b027e68>
6. <http://juliachencoding.blogspot.ca/2016/05/leetcode-124-binary-tree-maximum-path.html>
7. <https://t.co/7wUhmyPoDx>
8. <https://twitter.com/jianminchen/status/739254350355169281>

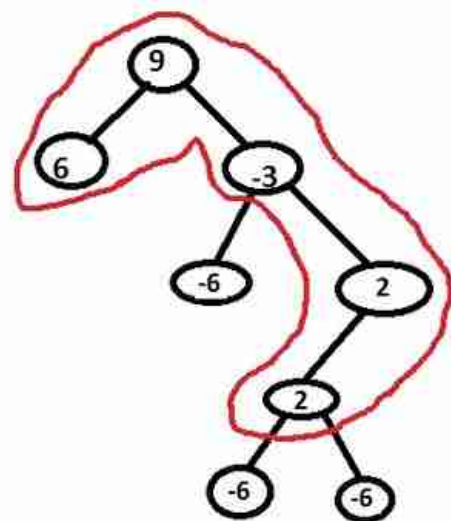
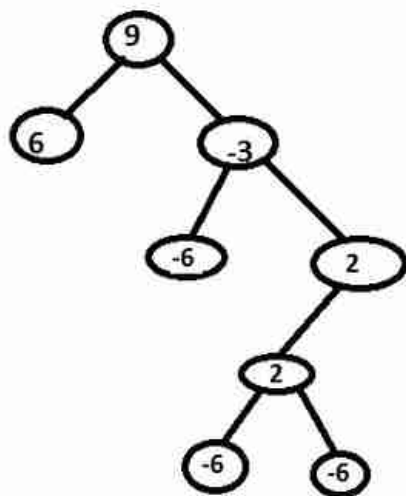
---

## Leetcode 124: Binary Tree Maximum Path Sum - Single Responsibility Principle (SRP) (2016-08-04 21:27)

August 4, 2016

If you do not have idea how to solve the Leetcode 124, please read the blog first, warm up with ideas to solve the problem:

[1]<http://juliachencoding.blogspot.ca/2016/08/leetcode-124-binary-tree-maximum-path.html>



The maximum path sum in the above tree is highlighted using red color, node 6->9->-3->2->2. Any two nodes in the tree can form a unique path, and the choice of path is  $N^2$ ,  $N$  is the total nodes in the tree.

**Creative way to solve the algorithm problem:**

**Review [2]S.O.L.I.D. principles, one of principle - Single Responsibility Principle.**

**Use SRP to write the function, one task a time.**

**1. Work on a simple problem first:**

Maximum value end by root in a binary tree - in other words, maximum value from the root node to any node in binary tree

[3]<https://gist.github.com/jjianminchen/8f3ec942e90bcdca5a1569d1a70e92df>

Goal: be able to write the function in 10 minutes, verify code with static analysis.

**1. Step 1 :** write a simple recursive function - preorder traversal.

A: Pay attention to negative value node.

(if both left and right child node's value are negative value, maximum value path ending at root node is root node's value itself; value  $\geq$  root node's value) -> come out formula: line 98, maximum value of 3 values.

B: Avoid if statement, just get minimum value through 3 values, make it one line statement - no if/else discussion of left/right child value  $> 0$ .

Only 5 lines of code. Short and concise.

```
89     private static int getMaximumEndByRoot(TreeNode root )
90     {
91         if (root == null)
92             return 0;
93
94         int left = getMaximumEndByRoot(root.left);
95         int right = getMaximumEndByRoot(root.right);
96
97         // 3 values, get maximum one.
98         return Math.Max(root.val, Math.Max(root.val + left, root.val + right));
99     }
```

**2. Add one more task in the above function -**

Usually the function should be designed to work on one task only. Need to add a second task to the function.

Based on the simple problem - maximum value end by the root node, add one more task to the function:

maxValueCrossRoot calculation. (bottom up solution)

Try to calculate the maximum path sum cross the root node in the tree.

[4]<https://gist.github.com/jianminchen/5eab22189f0fd7a58aa4fbc56b725dd8>

Add 2 more lines of code: line 104, 105, add one more input argument - ref int maxCrossRoot, 3 places update

Goal: complete the code change in 10 minutes.

2. **Step 2:** add 2 lines code (line 104, line 105), 3 changes - add one more argument (line 96, line 101, line 102):

```
96     private static int getMaximumEndByRoot(TreeNode root, ref int maxCrossRoot)
97     {
98         if (root == null)
99             return 0;
100
101         int left = getMaximumEndByRoot(root.left, ref maxCrossRoot);
102         int right = getMaximumEndByRoot(root.right, ref maxCrossRoot);
103
104         int value = root.val + ((left > 0) ? left : 0) + ((right > 0) ? right : 0);
105         maxCrossRoot = (value > maxCrossRoot) ? value : maxCrossRoot;
106
107         // 3 values, get maximum one.
108         return Math.Max(root.val, Math.Max(root.val + left, root.val + right));
109     }
```

So, overall, less than 20 minutes code writing. Follow the above 2 steps - write a function to complete the first task, and then, add second task to the function.

Questions and Answers:

1. How to make this algorithm an easy one?

A few people complain that the algorithm is too tough to work on through their blogs. Julia also spent hours to work on it in 2015, and then, Feb. 2016.

In August 2016, Julia spent hours to review the algorithm, wrote 2 blogs.

For easy to write, try SRP techniques, work on maximum path end by root first, then piggyback the max path cross the root. It will help to ease the stress.

**I learned to stay and work hard every day to get the chance to be the best.** - Karolina Pliskova

Julia, can you repeat the sentence word by word?

I like the encouraging verse. [5]<https://t.co/7wUhmyPoDx>

— jianmin chen (@jianminchen) [6]June 5, 2016

1. <http://juliachencoding.blogspot.ca/2016/08/leetcode-124-binary-tree-maximum-path.html>
  2. <http://juliachencoding.blogspot.ca/2015/12/oo-principle-solid-open-close-principle.html>
  3. <https://gist.github.com/jianminchen/8f3ec942e90bcdca5a1569d1a70e92df>
  4. <https://gist.github.com/jianminchen/5eab22189f0fd7a58aa4fbc56b725dd8>
  5. <https://t.co/7wUhmyPoDx>
  6. <https://twitter.com/jianminchen/status/739254350355169281>
- 

## ITINT5: tree maximum path sum (II) (2016-08-04 21:54)

August 4, 2016

Blog 1: July 6, 2015

[1]<http://juliachencoding.blogspot.ca/2015/07/itint5-tree-maximum-path-sum.html>

First writing in C # (July 6, 2015)

[2]<https://gist.github.com/jianminchen/754d29e47c491cfc271f764fb5dd8a61>

### Review comments (August 4, 2016, after 13 months):

1. first, the input argument res - variable name - not accurate - res should be maxValuCrossRoot
2. function name: maxTreePathSumRe is confusing, will be better called "maxTreePathSumEndByRoot"
3. line 121, 122 can be merged into one line statement - easy to read
4. add some design spec for the function - maxTreePathSumRe

Blog 2: August 4, 2016

C # code practice: 2nd writing

[3]<https://gist.github.com/jianminchen/c9be400e7bee71734ee7c454635846cf>

review comments:

1. Line 138 - 150, function ArrayMaximum

No need, call **Array.Max()** ;



3rd writing:

[4]<https://gist.github.com/jianminchen/3b2c8e0e84e52cbca2a7ec435b29a2e4>

highlight of changes:

1. line 124, use `Array.Max()`, remove the function: `ArrayMaximum(int[])`

Use Language Integrated Query (LINQ) - `Array.Max()`, detail see the blog:

[5]<http://juliachencoding.blogspot.ca/2016/06/array-class-c-c-javascript-java.html>

LINQ - Enumerable Methods Reference:

[6][https://msdn.microsoft.com/en-us/library/bb342261\(v=vs.100\).aspx](https://msdn.microsoft.com/en-us/library/bb342261(v=vs.100).aspx)

### **Blogs to read:**

Choose topic: extended merge sort

Algorithm: count inversions

count inversions - extended merge sort

1. [7]<http://jane4532.blogspot.ca/2013/06/zz-google-onsite-interview.html>

2. [8]<http://www.geeksforgeeks.org/counting-inversions/>

3. [9]<http://www.cs.umd.edu/class/fall2009/cmsc451/lectures/Lec08-inversions.pdf>

4. [10]<https://www.cp.eng.chula.ac.th/~piak/teaching/algo/algo2008/count-inv.htm>

Practice makes possible. [11]<https://t.co/Y9JyWKwLKY>

— jianmin chen (@jianminchen) [12]June 11, 2016

1. <http://juliachencoding.blogspot.ca/2015/07/itint5-tree-maximum-path-sum.html>

2. <https://gist.github.com/jianminchen/754d29e47c491cfc271f764fb5dd8a61>

3. <https://gist.github.com/jianminchen/c9be400e7bee71734ee7c454635846cf>

4. <https://gist.github.com/jianminchen/3b2c8e0e84e52cbca2a7ec435b29a2e4>

5. <http://juliachencoding.blogspot.ca/2016/06/array-class-c-c-javascript-java.html>

6. [https://msdn.microsoft.com/en-us/library/bb342261\(v=vs.100\).aspx](https://msdn.microsoft.com/en-us/library/bb342261(v=vs.100).aspx)

7. <http://jane4532.blogspot.ca/2013/06/zz-google-onsite-interview.html>

8. <http://www.geeksforgeeks.org/counting-inversions/>

9. <http://www.cs.umd.edu/class/fall2009/cmsc451/lectures/Lec08-inversions.pdf>

10. <https://www.cp.eng.chula.ac.th/~piak/teaching/algo/algo2008/count-inv.htm>

11. <https://t.co/Y9JyWKwLKY>

12. <https://twitter.com/jianminchen/status/741730885129510912>

---

## **Count inversions - Extended merge sort - 3 Lecture Notes Study (2016-08-05 21:10)**

August 5, 2016

Choose topic: extended merge sort

## Algorithm: count inversions

count inversions - extended merge sort

1. [1]<http://jane4532.blogspot.ca/2013/06/zz-google-onsite-interview.html>
2. [2]<http://www.geeksforgeeks.org/counting-inversions/>
3. [3][http://www.cs.umd.edu/class/fall2009/cmsc451/lectures/Lec08-inversions .pdf](http://www.cs.umd.edu/class/fall2009/cmsc451/lectures/Lec08-inversions.pdf)
4. [4]<https://www.cp.eng.chula.ac.th/piak/teaching/algo/algo2008/count-inv.htm>
5. [5]<https://www.cs.princeton.edu/wayne/kleinberg-tardos/pdf/05DivideAndConquerI.pdf>
6. [6][http://www.cs.colostate.edu/cs320/Slides/05 \\_inv.pdf](http://www.cs.colostate.edu/cs320/Slides/05_inv.pdf)

problem statement:

Inversion Count for an array indicates – how far (or close) the array is from being sorted. If array is already sorted then inversion count is 0. If array is sorted in reverse order that inversion count is the maximum.

Formally speaking, two elements  $a[i]$  and  $a[j]$  form an inversion if  $a[i] > a[j]$  and  $i < j$

Example:

The sequence 2, 4, 1, 3, 5 has three inversions (2, 1), (4, 1), (4, 3).

Lecture Notes -

### 1. First lecture study:

[7][http://www.cs.colostate.edu/cs320/Slides/05 \\_inv.pdf](http://www.cs.colostate.edu/cs320/Slides/05_inv.pdf)

#### 1. How many inversions at most in the array $n$ ?

$n(n-1)/2$ , special case, like  $\{n, n-1, \dots, 1\}$ , any two nodes in the array is one inversion pair.

or: **What is the maximum number of inversions for a list of length  $n$ ?**

$n(n-1)/2$

2. If each inversion is counted once, then the time of the algorithm is  $O(n^2)$ ,  $n$  is the number of elements in the array. Not optimal, we should not count each inversion.

3. Ideas to solve the algorithm:

Bubble sort?

Selection sort?

Insertion sort?

These are  $O(n^2)$

**Bubble and insertion sort count each individual inversion. To do better we must not count each individual inversion.**

So, better algorithm is to beat  $O(n^2)$ , using merge sort,  $n \log n$  - divide and conquer - sort and count inversion in the same time.

**In merge sort we do not swap all elements that are out of order with each other, we make larger distance "swaps".**

Questions: Sorting and counting inversion - merge part how to count the inversions.

**Keywords in the lecture notes (7):**

Collaborative filtering

inversions

Meta-search tools

Rank analysis

Recurrence Analysis -  $T(n) = 2 T(n/2) + cn$

similarity/ dissimilarity / in the middle

the number of out of place rankings

**Actionable Items:**

1. Merging part with diagram: <- Julia, can you draw a diagram as well

2. Count Inversions: Algorithm pseudo code - write down here:

# Counting Inversions: Algorithm

**Sort-and-Count(L)**

```
if list L has one element
    return 0 and the list L
divide the list into two halves A and B
( $r_A$ , A)  $\leftarrow$  Sort-and-Count(A)
( $r_B$ , B)  $\leftarrow$  Sort-and-Count(B)
( $r$ , R)  $\leftarrow$  Merge-and-Count(A, B)
return  $r = r_A + r_B + r$  and the sorted list R
```

**Merge-and-Count(L,R)**

```
count = 0
while L and R not empty:
    append smallest of  $L_i$  and  $R_j$  to result
    if  $R_j$  smallest
        add number of elements remaining in L to count
if one list empty
    append the other one to result
return count, result
```

## 2. 2nd Lecture Notes Study:

[8]<https://www.cs.princeton.edu/wayne/kleinberg-tardos/pdf/05DivideAndConquerI.pdf>

Julia, write down favorite notes one sentence a time, on page 16, 17

---

Counting inversions: how to combine two subproblems?

Q. How to count inversions (a,b) with a  $\in$  A and b  $\in$  B?

A. Easy if A and B are sorted!

Warmup algorithm.

Sort A and B.

For each element b  $\in$  B,

- binary search in A to find how elements in A are greater than b.

list A list B

7

10

708

18  
3  
14  
17  
23  
2  
11  
16  
  
sort A sort B  
  
3  
7  
10  
14  
18  
2  
11  
16  
17  
23  
  
binary search to count inversions (a, b) with a  
A and b  
B  
3  
7  
10  
14  
18  
2  
11  
16  
17  
23

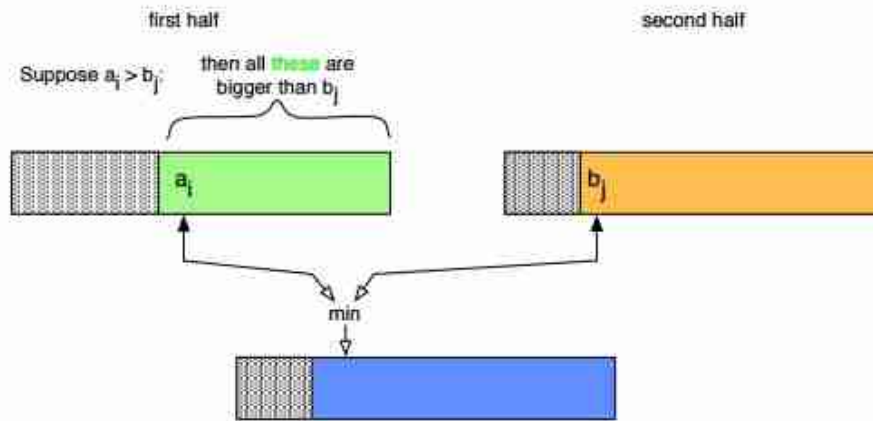
## 3. 3rd Lecture Notes Study: (Inversions Count)

[9][http://www.cs.umd.edu/class/fall2009/cmsc451/lectures/Lec08-inversions .pdf](http://www.cs.umd.edu/class/fall2009/cmsc451/lectures/Lec08-inversions.pdf)

## What if each of the half lists were sorted?

Suppose each of the half lists were sorted.

If we find a pair  $a_i > b_j$ , then we can infer many other inversions:



Each of the green items is an inversion with  $b_j$ .

Play to win; stop **Recognize** when you are using negative self- talks and replace it with **positive** ; when in doubt, remember: Play to win.

Great tips for sports coaching, [10]<https://t.co/6m29a8l1z0>

— jianmin chen (@jianminchen) [11]July 22, 2016

### Memorize 8 tips to help you to perform to your highest potential in Tennis (? code practice, etc.):

1. Let go of what others think
2. Perform for yourself, not to impress or to "not disappoint" others
3. Accept that you will make mistakes, and let them go
4. Focus on what you can control
5. Recognize when you are using negative self-talk and replace it with positive
6. Rather than performing perfectly, perform to see improvement
7. Be objective about your performance, not subjective
8. Focus on the Journey, not the Destination

## Play not to lose or Play to win - **Julia plays to win!**

Some people play not to lose. I play to win each and every second I'm on court. [12] #SpeedTakes aggression. [13] [pic.twitter.com/ae5xlriogs](https://pic.twitter.com/ae5xlriogs)

— Simona Halep (@Simona\_Halep) [14] July 2, 2016

1. <http://jane4532.blogspot.ca/2013/06/zz-google-onsite-interview.html>
  2. <http://www.geeksforgeeks.org/counting-inversions/>
  3. <http://www.cs.umd.edu/class/fall2009/cmsc451/lectures/Lec08-inversions.pdf>
  4. <https://www.cp.eng.chula.ac.th/~piak/teaching/algo/algo2008/count-inv.htm>
  5. <https://www.cs.princeton.edu/~wayne/kleinberg-tardos/pdf/05DivideAndConquerI.pdf>
  6. [http://www.cs.colostate.edu/~cs320/Slides/05\\_inv.pdf](http://www.cs.colostate.edu/~cs320/Slides/05_inv.pdf)
  7. [http://www.cs.colostate.edu/~cs320/Slides/05\\_inv.pdf](http://www.cs.colostate.edu/~cs320/Slides/05_inv.pdf)
  8. <https://www.cs.princeton.edu/~wayne/kleinberg-tardos/pdf/05DivideAndConquerI.pdf>
  9. <http://www.cs.umd.edu/class/fall2009/cmsc451/lectures/Lec08-inversions.pdf>
  10. <https://t.co/6m29a81lz0>
  11. <https://twitter.com/jianminchen/status/756286480528527360>
  12. <https://twitter.com/hashtag/SpeedTakes?src=hash>
  13. <https://t.co/ae5xlriogs>
  14. [https://twitter.com/Simona\\_Halep/status/749299161510600704](https://twitter.com/Simona_Halep/status/749299161510600704)
- 

## A small research - tennis coaching vs algorithm lecturing (2016-08-07 14:30)

August 7, 2016

Julia likes to pick up a small topic to do some research and practice her research muscle. Since she was amazed about USATP master professional Rick Macci's teaching video, and amazed how good the presentation is. A lot of technologies are applied on tennis coaching and tennis matches. For example, on the tennis court, on the grand slam matches, there are hundreds of camera on the court, hundreds of technologies on speed measuring, and all other things.

Here is the link:

[1] <https://www.youtube.com/watch?v=5MHugAF2DiQ>

1. Lecture on Roger Federer's tennis forehand stroke, and show 3D technologies how to do analysis
  2. Before vs After, two videos are comparing.
  3. **Statistics:** How many views? over half a million views. -
  4. **Producer :** USPTA
- how good USPTA can utilize the technologies and help the teaching.

### Lecture notes:

#### Forehand - 7 steps: (P, J, E, T, F, FS, F)

Preparation

Joint angles

Elbow Extension  
Tap the dog  
The Flip  
Forward Swing  
The Finish

Goal: Racket Speed, keep the ball in the court

And teaching lectures she spent time to work on this weekend:

Algorithm: count inversion  
extended merge sort:

Julia also likes the lecture content about count inversion - merge sort, the examples and discussions.  
So, Julia likes to study those lecture notes, take time to enjoy reading.

1. [2][http://www.cs.umd.edu/class/fall2009/cmsc451/lectures/Lec08-inversions .pdf](http://www.cs.umd.edu/class/fall2009/cmsc451/lectures/Lec08-inversions.pdf)
2. [3]<https://www.cs.princeton.edu/~wayne/kleinberg-tardos/pdf/05DivideAndConquerI.pdf>
3. [4][http://www.cs.colostate.edu/~cs320/Slides/05 \\_inv.pdf](http://www.cs.colostate.edu/~cs320/Slides/05_inv.pdf)

Stanford University:

- [5][https://www.coursera.org/learn/algorithm-design-analysis/lecture/GFmmJ /o-n-log-n-algorithm-for-counting-inversions-i](https://www.coursera.org/learn/algorithm-design-analysis/lecture/GFmmJ/o-n-log-n-algorithm-for-counting-inversions-i)
- [6][https://www.coursera.org/learn/algorithm-design-analysis/lecture/IUiUk /o-n-log-n-algorithm-for-counting-inversions-ii](https://www.coursera.org/learn/algorithm-design-analysis/lecture/IUiUk/o-n-log-n-algorithm-for-counting-inversions-ii)

Julia learns tennis from over 20 top coaches in the world and also actively practice what she learns through lessons:

Here is one of them she learned from the coach -

- [7]<https://www.youtube.com/watch?v=n7ApnK3BGcU>

Julia started to work on the double alley drill - learning is fun!



IFRAME: [8]<https://www.youtube.com/embed/s0yvvVN799w>

one more:



IFRAME: [9]<https://www.youtube.com/embed/XhHKR-9Ze1g>

1. <https://www.youtube.com/watch?v=5MHugAF2DiQ>
2. <http://www.cs.umd.edu/class/fall2009/cmsc451/lectures/Lec08-inversions.pdf>
3. <https://www.cs.princeton.edu/~wayne/kleinberg-tardos/pdf/05DivideAndConquerI.pdf>
4. [http://www.cs.colostate.edu/~cs320/Slides/05\\_inv.pdf](http://www.cs.colostate.edu/~cs320/Slides/05_inv.pdf)
5. <https://www.coursera.org/learn/algorithm-design-analysis/lecture/GFmmJ/o-n-log-n-algorithm-for-counting-in>



versions-i

6. <https://www.coursera.org/learn/algorithm-design-analysis/lecture/IUiUk/o-n-log-n-algorithm-for-counting-in-versions-ii>

7. <https://www.youtube.com/watch?v=n7ApnK3BGcU>

8. <https://www.youtube.com/embed/s0yvVvVN799w>

9. <https://www.youtube.com/embed/XhHKR-9Ze1g>

---

## **C# Extension Methods - pluralsight.com (2016-08-08 21:20)**

August 8, 2016

Plan to study the course "C # Extension Methods". 3 hours course.

Lecturer's website:

[1]<http://app.pluralsight.com/author/elton-stoneman>

August 8, 1 hour, Introduction Extension Methods

1. <http://app.pluralsight.com/author/elton-stoneman>

---

## **Become a Full-stack .NET Developer - Architecture and Testing - pluralsight.com (2016-08-08 22:08)**

August 8, 2016

Plan to study the course:

Become a Full-stack .NET Developer - Architecture and Testing

Lecturer's website:

[1]<http://app.pluralsight.com/author/mosh-hamedani>

Notes:

1. Remaining use cases
2. Modularize JavaScript Code
3. Refactoring towards the clean architecture
4. Programming against interfaces
5. Test Controllers
6. Test Repositories

## 7. Adding Integration Tests

Modularizing JavaScript Code - 40m 56s

Introduction

Extracting JavaScript Code

Revealing Module Pattern

Refactoring using Revealing Module Pattern

Cleaning the init Method

Applying the DRY Principle

Better Separation of Concerns

Module Dependencies

Physical Separation

Optimization

Exercise

Code Review

Summary

Programming Against Interfaces

Introduction

Dependency Inversion Principles

Extracting Interfaces

Dependency Injection

Adding Ninject

Does Entity Framework Really implement the Repository Pattern?

Package Dependencies

Restructure the Application

Complexity Fallacy

Productivity Fallacy

Exercise

Refactoring Data Annotations

Complexity Fallacy

Productivity Fallacy

Exercise

Refactoring Data Annotations

Summary

Testing Controllers

Introduction

Extracting Queries

Repository pattern

When to use the repository pattern

extracting repositories

extracting queries with eager loading

exercise

code review

clean architecture  
decoupling from entity framework  
unit of work pattern  
implementing the unit of work  
consolidating dependencies  
summary

Programming Against interfaces  
introduction  
Dependency inversion principle  
extracting interfaces

Repository pattern is already in entity framework - no need to use? it depends.

Design to solve problems - repository pattern

complexity of design - be pragmatic

medium/ large projects

problems :

complex queries/ fat controllers/ fat services

SQL Injection etc. website common attacks -

SQL injection, XSS, CSRF, preventing CSRF attacks -

<http://www.veracode.com/security/xss>

Blog reading:

August 12, 2016

[2]<http://www.peterprovost.org/blog/2012/06/19/adding-ninject-to-web-api>

[3]<http://stories.visualstudio.com/bing-continuous-delivery/>

1. <http://app.pluralsight.com/author/mosh-hamedani>
2. <http://www.peterprovost.org/blog/2012/06/19/adding-ninject-to-web-api>
3. <http://stories.visualstudio.com/bing-continuous-delivery/>

## **.NET distributed system architecture - pluralsight.com (2016-08-08 22:44)**

August 8, 2016

Plan to work on the course - .NET distributed system architecture - 5 hour 30 minutes

Lecturer's website:

[1]<http://app.pluralsight.com/author/scott-seely>

1. <http://app.pluralsight.com/author/scott-seely>

---

## **C# Design Strategies - pluralsight.com (2016-08-08 22:55)**

August 8, 2016

Plan to work on this course on pluralsight.com - C # design strategies - 3 hours.

[1]<http://app.pluralsight.com/author/jon-skeet>

1. <http://app.pluralsight.com/author/jon-skeet>

---

## **Mastering C# 4.0 - pluralsight.com (2016-08-08 22:58)**

August 8, 2016

Plan to work on this course - Mastering C # 4.0 - pluralsight.com, 11 hours.

[1]<http://app.pluralsight.com/author/jon-skeet>

1. <http://app.pluralsight.com/author/jon-skeet>

---

## **Designing Fluent APIs in C# - pluralsight.com (2016-08-08 23:29)**

August 8, 2016

Plan to work on course - Designing Fluent APIs in C #

[1]<http://app.pluralsight.com/author/floyd-may>

video to watch:

[2]<https://www.youtube.com/watch?v=uKtMwmWv6Q0>

1. <http://app.pluralsight.com/author/floyd-may>
  2. <https://www.youtube.com/watch?v=uKtMwmWv6Q0>
- 

## JavaScript - a programmer random thoughts (2016-08-11 23:51)

August 11, 2016

Julia likes to have a small research every day, later she can add some thoughts for the topic. Today the topic is "how to balance coding writing vs reading API documents

".

?

In order to write good JavaScript code, read a lot of JavaScript code first. She did write JavaScript/ CSS/ Html code 8 hours nonstop at work, she likes to take a break while she was at home enjoying the evening.

Julia reviewed the blog about JavaScript while watching interviews.

[1]HackerRank: Two string - thinking in JavaScript over 10 ways

I

interviews she enjoyed - Vogue 73 questions series:

1. Serena Williams

Serena offers 1 hour \$100 master class - video class - She is really good at teaching, she shows in the interview how to make a hidden drop shot.

[2][https://www.youtube.com/watch?v=Gpu\\_st4ynLc](https://www.youtube.com/watch?v=Gpu_st4ynLc)

2. Iggy Azalea - 7 3 questions

[3]<https://www.youtube.com/watch?v=AMh5f8xRLRE>

3. Song writer, singer Swift Taylor

[4]<https://www.youtube.com/watch?v=XnbCSboujF4>

### Random thoughts:

1. How to stay calm, answer questions very well?

2. Airbnb JavaScript Style Guide() - English Version

[5]<https://github.com/airbnb/javascript>

Reading while going through JavaScript style guide from Airbnb:

1. JavaScript spread syntax... - 10 minutes

[6][https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Spread\\_operator](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Spread_operator)

### 3. Read JavaScript Array.From - 20 minutes

[https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Array/from](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/from)

### 2. **Celebrate the day** by learning CSS, working on clipping image etc.:

A. clip techniques - outside - relative, inside, absolute  
how to define it and practice it!

[7]<http://www.xul.fr/en/css/clip.php>

B. Text over the image:

[8]<https://css-tricks.com/text-blocks-over-image/>

C. Bootstrap 3 Thumbnail Slider / Carousel

[9][#](http://www.bootply.com/XeZvDD059P)

1. <http://juliachencoding.blogspot.ca/2016/03/hackerrank-two-string-thinking-in.html>
2. [https://www.youtube.com/watch?v=Gpu\\_st4ynLc](https://www.youtube.com/watch?v=Gpu_st4ynLc)
3. <https://www.youtube.com/watch?v=AMh5f8xRLRE>
4. <https://www.youtube.com/watch?v=XnbCSboujF4>
5. <https://github.com/airbnb/javascript>
6. [https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Spread\\_operator](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Spread_operator)
7. <http://www.xul.fr/en/css/clip.php>
8. <https://css-tricks.com/text-blocks-over-image/>
9. <http://www.bootply.com/XeZvDD059P>

---

smith clark (2016-08-18 02:31:37)

This comment has been removed by a blog administrator.

## **Clip: cropping an image - CSS learning and sharing on JSFiddle (2016-08-12 21:39)**

August 12, 2016

Clip is the CSS feature to clip an image using scripts. Julia spent more than one hour to work on a few lines of CSS code, and then, she chose to use a specific image to help.

If you have an image with 500px x 500px, you may like to get one small image with size 100px x 100px without using Microsoft paint to cut and save as a new file. The solution is to use CSS clip.

Basics to find out the solution:

CSS:

1. position relative/ absolute
2. rect
3. clip
4. an example to explain how to put together
5. play with Firefox firebug, and then figure out CSS settings.

Study the article:

[1]<http://www.xul.fr/en/css/clip.php>

Spent more than 1 hour to figure out how to set up .clipzone width and height. There are a few ways to set up CSS, but she likes to write down her tips to solve the problem.

After more than one hour confusion, she decided to customize an image with some grids and text, with size of 500px x 500px, see the following:

<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>
<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>
<b>11</b>	<b>12</b>	<b>13</b>	<b>14</b>	<b>15</b>

With a few mistakes, she figured out how to configure them quickly. Let us talk about a problem, and then, solve the problem using CSS clip property based on the article (ref. 2).

Problem statement:

**How to clip the image 100px x 100px in the center of the above image, with text 13?**

**Here is the image:**



The element is to be cropped in an outer container, and the clip zone

- **Julia's tip 1** : make the zone the whole image:

```
.clipzone
{
position: relative ;
width:
500px
;
height: 500px ;
overflow:
hidden
;
}
```

```
.clipped
{
position: absolute;
}
```

```
<div class=
"clipzone"
> <img class=
"clipped"
id=
"image1"
src=
""
/> <
/div>
```

This container is integrated in the flow of the contents of the page with the relative position and allows the content to have the absolute position. The property overflow: hidden avoids exceeding the limit of the container.

so add the clip css property and value into class clipped, and add this rule:

```
.clipped
```



```
{
position: absolute ;
clip: rect(200px, 200px, 300px, 300px) ; /* top left right bottom */
}
```

And then, the image using clip CSS property will be:



### Julia's tip 2:

Add one more rule:

margin-left: -200px;

With the above rule, the clipped image will be positioned at 0 position; in other words, to get any image, for example,

with text: 11, set margin-left: 0px;

with text: 12, set margin-left: -100px;

with text: 13, set margin-left: -200px;

with text: 14, set margin-left: -300px;

with text: 15, set margin-left: -400px;

so the image is always positioned the same place.

References:

1. [2][http://www.w3schools.com/cssref/pr\\_pos\\_clip.asp](http://www.w3schools.com/cssref/pr_pos_clip.asp)

2. [3]<http://www.xul.fr/en/css/clip.php>

### Actionable Items:

1. clip an image using CSS

[4]<https://jsfiddle.net/jianminchen/ksdcgomx/>

2. original image vs clipped image

[5]<https://jsfiddle.net/jianminchen/hxgakk5n/>

1. clipped image:

IFRAME: [6]<https://jsfiddle.net/jianminchen/ksdcgomx/embedded/html,css,result/>

2. original image comparison to the clipped image

IFRAME: [7]<https://jsfiddle.net/jianminchen/hxgakk5n/embedded/html,css,result/>

### Blog reading:

1. Google keyword search: JavaScript Separate of concern

how to structure JavaScript better?

Study the code - example to follow

[8]<http://stephen-young.me.uk/2013/01/05/maintainable-js-with-modules.html>

Watch and learn: Microsoft agile development - Bing



IFRAME: [9]<https://www.youtube.com/embed/8RqzZaeDvkM>

Microsoft Development Services for the DevOps Era



IFRAME: [10]<https://www.youtube.com/embed/Gks7Ngzt5Yw>

1. <http://www.xul.fr/en/css/clip.php>
  2. [http://www.w3schools.com/cssref/pr\\_pos\\_clip.asp](http://www.w3schools.com/cssref/pr_pos_clip.asp)
  3. <http://www.xul.fr/en/css/clip.php>
  4. <https://jsfiddle.net/jianminchen/ksdcgomx/>
  5. <https://jsfiddle.net/jianminchen/hxgakk5n/>
  6. <file:///jsfiddle.net/jianminchen/ksdcgomx/embedded/html,css,result/>
  7. <file:///jsfiddle.net/jianminchen/hxgakk5n/embedded/html,css,result/>
  8. <http://stephen-young.me.uk/2013/01/05/maintainable-js-with-modules.html>
  9. <https://www.youtube.com/embed/8RqzZaeDvkM>
  10. <https://www.youtube.com/embed/Gks7Ngzt5Yw>
- 

## Microsoft Development Service - watch and learn (2016-08-14 11:32)

August 14, 2016

Great time to watch and learn. First thing in the Sunday morning is to relax and learn.

Blogs to read:

[1]<http://stories.visualstudio.com/bing-continuous-delivery/>

Videos:

1. Full Keynote: The Future of Microsoft Tools and Services for the New Role of Developers

[2]<https://www.youtube.com/watch?v=Gks7Ngzt5Yw>

2. Keynote: Visual Studio 2015 - Any App, Any Developer

[3]<https://www.youtube.com/watch?v=Nj1luMpj7ml>

Blogs to read:

1. Leetcode 380: ( **code study: Leetcode blogs** )

[4][http://www.guoting.org/leetcode/leetcode-380-insert-delete-getrandom-o 1/](http://www.guoting.org/leetcode/leetcode-380-insert-delete-getrandom-o-1/)

## 2. Leetcode 380 - Insert Delete GetRandom O(1)

[5]<https://leetcode.com/problems/insert-delete-getrandom-o1/>

## 3. Read the article

[6]<http://www.programcreek.com/2014/08/leetcode-insert-delete-getrandom-o1-java/>

## 4. Read Uber map article:

[7]<http://goo.gl/nmRv18>

Google search keyword:

red-black tree - C++ implements the map

2.

[8]<https://goo.gl/fWq132>

1. <http://stories.visualstudio.com/bing-continuous-delivery/>
  2. <https://www.youtube.com/watch?v=Gks7Ngzt5Yw>
  3. <https://www.youtube.com/watch?v=Nj1luMpj7mI>
  4. <http://www.guoting.org/leetcode/leetcode-380-insert-delete-getrandom-o1/>
  5. <https://leetcode.com/problems/insert-delete-getrandom-o1/>
  6. <http://www.programcreek.com/2014/08/leetcode-insert-delete-getrandom-o1-java/>
  7. <http://goo.gl/nmRv18>
  8. <https://goo.gl/fWq132>
- 

## Leetcode 380 - Insert, delete, getRandom() O(1) time - Practice 1 (2016-08-16 21:45)

August 8, 2016

Problem statement:

Design a data structure that supports all following operations in *average* O(1) time.

1. insert(val): Inserts an item val to the set if not already present.
2. remove(val): Removes an item val from the set if present.
3. getRandom: Returns a random element from current set of elements. Each element must have the same probability of being returned.

Go over the website Java code, write a C # version:

[1]<http://www.programcreek.com/2014/08/leetcode-insert-delete-getrandom-o1-java/>

Java code from the above webpage:

[2]<https://gist.github.com/jianminchen/b6ab09a8a047db42da4ad663036d621a>

C # code written by Julia

[3]<https://gist.github.com/jianminchen/9ebfbe7462c17689344578be31c804c0>

Highlights of practice:

1. Spent 20+ minutes to write the function `remove(int val)`
  2. Put test cases into main function, test the code to understand the problem.
  3. Did not understand the design, why two hashmaps are used.
  4. Concern about function `remove` - line 133 - 165
- The function is not readable - one function does so many things - break Single Responsibility Principle.

1. <http://www.programcreek.com/2014/08/leetcode-insert-delete-getrandom-o1-java/>
  2. <https://gist.github.com/jianminchen/b6ab09a8a047db42da4ad663036d621a>
  3. <https://gist.github.com/jianminchen/9ebfbe7462c17689344578be31c804c0>
- 

## Leetcode 380 - Insert, delete, getRandom() O(1) time - Practice 2 (2016-08-16 22:00)

August 16, 2016

First practice on Leetcode 380:

[1]<http://juliachencoding.blogspot.ca/2016/08/leetcode-380-insert-delete-getrandom-o1.html>

Add one more function - design API - Julia likes to design API, after she tried to memorize all APIs of JavaScript array.

Second practice - one function:

[2]<https://gist.github.com/jianminchen/84a0945eb49d3ae7af36b4ca5f0f0c41>

Design API - move an entry from the dictionary from one location  
to another location

assume that dictionary contains the entry of from

assume that to position is available in the dictionary

public bool move(Dictionary<Int32, Int32> dict, int from, int to)

### Blog:

1. CSS style -

[3]<http://codeguide.bootcss.com/>

2. [4][http://www.ruanyifeng.com/blog/2012/10/javascript\\_module.html](http://www.ruanyifeng.com/blog/2012/10/javascript_module.html)

1. <http://juliachencoding.blogspot.ca/2016/08/leetcode-380-insert-delete-getrandom-o1.html>

2. <https://gist.github.com/jianminchen/84a0945eb49d3ae7af36b4ca5f0f0c41>

3. <http://codeguide.bootcss.com/>

4. [http://www.ruanyifeng.com/blog/2012/10/javascript\\_module.html](http://www.ruanyifeng.com/blog/2012/10/javascript_module.html)

---

### Leetcode 380 - Insert, delete, getRandom() O(1) time - Practice 3 (2016-08-16 22:10)

August 16, 2016

After first 2 practices, Julia read the blog:

[1]<http://www.guoting.org/leetcode/leetcode-380-insert-delete-getrandom-o1/>

3rd practice using C #:

[2]<https://gist.github.com/jianminchen/9a8aef6220d285aac3f22abac984e0b3>

1. <http://www.guoting.org/leetcode/leetcode-380-insert-delete-getrandom-o1/>

2. <https://gist.github.com/jianminchen/9a8aef6220d285aac3f22abac984e0b3>

---

### Leetcode 380 - Insert, delete, getRandom() O(1) time - Practice 4 (2016-08-16 22:12)

August 16, 2016

The first 3 practices:

[1]<http://juliachencoding.blogspot.ca/2016/08/leetcode-380-insert-delete-getrandom-o1.html>

[2][http://juliachencoding.blogspot.ca/2016/08/leetcode-380-insert-delete-getrandom-o1\\_16.html](http://juliachencoding.blogspot.ca/2016/08/leetcode-380-insert-delete-getrandom-o1_16.html)

[3][http://juliachencoding.blogspot.ca/2016/08/leetcode-380-insert-delete-getrandom-o1\\_89.html](http://juliachencoding.blogspot.ca/2016/08/leetcode-380-insert-delete-getrandom-o1_89.html)

Read the blog:

[4]<http://www.guoting.org/leetcode/leetcode-380-insert-delete-getrandom-o1/>

Java code from the above blog:

[5]<https://gist.github.com/jianminchen/9561828feda21bac20bc9ba4da0d13c8>

4th practice using C #:

[6]<https://gist.github.com/jianminchen/7741fabf57413ccbb1080f6f691bb532>

### Highlights of 4th practice:

#### 1. Thinking problem solving using an array :

Insertion of array is to append the number at the end of array,  $O(1)$ ;

Delete a number from the array, for example, at index position  $i$  of array with length  $n$ , then, all elements from  $i+1$  to  $n-1$  should be shifted to left one position - time complexity is  $O(n-i) = O(n)$

`getRandom()` - time complexity is  $O(1)$ , just get a random number  $r$  from 0 to  $n$ , and then, return `arr[r]`.

*Array value can be any integer number, and in the range of `Int32.min` - `Int32.max`, so given an integer value, it may not be in the array; if it is, to find the index of array to hold the value, we need to look up a hashmap with value/ index of array pair first.*

So, in order to make it  $O(1)$  for the deletion, we can use extra space to speed up time to  $O(1)$ . The idea is to move last number in the array to the position of deleted element.

Extra space - a hashmap to store each number in the array as key, and the value is the position in the array.

#### 4. Introduce new design in the practice?

***arrayDeletionO1*** - line 150 API - **array deletion** , ***arrayDeletionO1 (Dictionary<Int32, Int32> arr, Dictionary<Int32, Int32> map, int val)***

- implement the array deletion in time complexity  $O(1)$

***move*** - line 173 API - **array move** - move an entry from the dictionary from one location to another location.  
***move(Dictionary<Int32, Int32> dict, int from, int to)***

***mapUpdateForArrayDeletion*** - line 129 - hashmap update -

***mapUpdateForArrayDeletion***

***(Dictionary<Int32, Int32> arr, Dictionary<Int32, Int32> map,***

***int***

***val***

***)***

**Cons and pros** for modified deletion for the array-like data structure:

1. Cons: If the array is sorted, then the order is lost. No longer in the order.
2. Study Array API for C, C++, C #, JavaScript, none of them has API like this. Why?

Array - JavaScript

[7]<http://juliachencoding.blogspot.ca/2016/07/javascript-array-mozilla-60-minutes.html>

C # array

[8]<http://juliachencoding.blogspot.ca/2016/06/array-class-c-c-javascript-java.html>

3. For the deletion, update can be done directly instead of first deletion, and then move of the last one to the deleted item's position. line 182, line 183 can be merged to one line - update.

4th practice using C #:

[9]<https://gist.github.com/jianminchen/7741fabf57413ccbb1080f6f691bb532>

4. JavaScript array is sparse, the array is not contiguous. For Leetcode 380, Julia thought about array deletion design, if the array is leaving as sparse after deletion, then, getRandom() can not be O(1). If the sparse slot is selected by random algorithm, need to select another one.

Related articles:

1. Uber map article:

[10]<http://goo.gl/nmRv18>

Google search keyword:

red-black tree - C++ implements the map

[11]<http://stackoverflow.com/questions/5288320/why-is-stdmap-implemented-as-a-red-black-tree>

2. Uber algorithm question:

[12]<https://goo.gl/fWq132>

1. <http://juliachencoding.blogspot.ca/2016/08/leetcode-380-insert-delete-getrandom-o1.html>
2. [http://juliachencoding.blogspot.ca/2016/08/leetcode-380-insert-delete-getrandom-o1\\_16.html](http://juliachencoding.blogspot.ca/2016/08/leetcode-380-insert-delete-getrandom-o1_16.html)
3. [http://juliachencoding.blogspot.ca/2016/08/leetcode-380-insert-delete-getrandom-o1\\_89.html](http://juliachencoding.blogspot.ca/2016/08/leetcode-380-insert-delete-getrandom-o1_89.html)
4. <http://www.guoting.org/leetcode/leetcode-380-insert-delete-getrandom-o1/>
5. <https://gist.github.com/jianminchen/9561828feda21bac20bc9ba4da0d13c8>
6. <https://gist.github.com/jianminchen/7741fabf57413ccbb1080f6f691bb532>
7. <http://juliachencoding.blogspot.ca/2016/07/javascript-array-mozilla-60-minutes.html>
8. <http://juliachencoding.blogspot.ca/2016/06/array-class-c-c-javascript-java.html>
9. <https://gist.github.com/jianminchen/7741fabf57413ccbb1080f6f691bb532>
10. <http://goo.gl/nmRv18>
11. <http://stackoverflow.com/questions/5288320/why-is-stdmap-implemented-as-a-red-black-tree>
12. <https://goo.gl/fWq132>

---

**JavaScript Style Guide Study - Airbnb (2016-08-17 21:05)**

August 17, 2016

Plan to spend 3+ hours to go over the JavaScript Style Guide.

1. Airbnb JavaScript Style Guide() - English Version

[1]<https://github.com/airbnb/javascript>

2. Airbnb JavaScript Style Guide() - Chinese version

[2]<https://github.com/yuche/javascript#table-of-contents>

36 Topics

Types

References

Objects

Arrays

Destructuring

Strings

Functions

Arrow Functions

Classes & Constructors

Modules

Iterators and Generators (NO. 11)

Properties

Variables

728



Hoisting

comparison Operators & Equality

Blocks

Comments

Whitespace

commas

Semicolons

Type Casting & Coercion (No. 21)

Naming Conventions

Accessors

Events

jQuery

ECMAScript 5 Compatibility

ECMAScript 6 Styles

Testing

Performance

Resources

In the Wild

Translation

The JavaScript Style Guide Guide

Chat With Us about JavaScript

Contributors

License

Reading while going through JavaScript style guide from Airbnb:

1. JavaScript **spread** syntax... - 10 minutes

[3][https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Spread\\_operator](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Spread_operator)

2. Read JavaScript **Array.From** - 20 minutes

[4][https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Array/from](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/from)

3. **Destructuring** assignment 10 minutes

[5][https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Destructuring\\_assignment](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Destructuring_assignment)

4. **Template literals** - 10 minutes reading

[6][https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Template\\_literals](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Template_literals)

new terms: back-tick(`), neither double nor single quotes

5. **Default parameters** -

[7][https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Template\\_literals](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Template_literals)

#### Editorial notes:

1. Write down all new terms in the guide. Understand the terms.
2. Try to memorize all the guide. Design some drills to help.
3. Invest time first to read the good style code, and then, start to write more JavaScript code.
4. Focus on the document, understand why? What to avoid? Good or bad or Excellent idea?

## Favorite blogs of day: web front technologies

1. [8]<https://github.com/qiu-deqing/FE-learning>

1. <https://github.com/airbnb/javascript>
  2. <https://github.com/yuche/javascript#table-of-contents>
  3. [https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Spread\\_operator](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Spread_operator)
  4. [https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Array/from](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/from)
  5. [https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Destructuring\\_assignment](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Destructuring_assignment)
  6. [https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Template\\_literals](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Template_literals)
  7. [https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Template\\_literals](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Template_literals)
  8. <https://github.com/qiu-deqing/FE-learning>
- 

## Become a Full-stack .NET Developer - pluralsight.com (2016-08-17 23:10)

August 17, 2016

Start to work on this 3 hours course on pluralsight.com.

Start from beginner level, enjoy the course. Take some notes.

Related course:

[1]<http://juliachencoding.blogspot.ca/2016/08/become-full-stack-net-developer.html>

## Leetcode workout:

1. **Leetcode 278** - binary search algorithm and discussion

1. [2]<http://blog.csdn.net/ebowtang/article/details/50735869>

A. Good discussion on the deadlock prevention - make sure that +1 or -1 somewhere

B. discussion of

Good discussion of tips:

1. base case
2. overflow issue

[3]<http://www.cnblogs.com/airwindow/p/4791234.html>

## Leetcode 125

1. study the code:

[4]<http://blog.csdn.net/DERRANTCM/article/details/47651267>

Leetcode 125 - write code based on this version

short function name - isalnum - very good name

[5]<http://blog.csdn.net/nomasp/article/details/50623165>

5 practices - interest journey !

**1st** practice: failed static analysis - miss a bug - easy to spot -, not++

[6]<https://gist.github.com/jianminchen/ed12d3004dbe85815f41fef047d48823>

there is a bug in first writing, line 62, should be right-; not right++.  
static analysis did not catch the bug - be careful next time.

**2nd** practice:

[7]<https://gist.github.com/jianminchen/1e1724c79805d2f40195f08135ec4d02>

highlights of practice 2:

1. Fail to pass the online judge
2. line 24 - 27, logic and reasoning has flaws

both are ok

both are failed

A is failed

B is failed

Those are 4 cases - in the specific order as well.

**3rd** practice:

[8]<https://gist.github.com/jianminchen/8f9ae3839499718790180aa58965d3de>

highlights of 3rd practice:

1. Fix the bug first, add these line 59 -67 two "else if", one "else", pass online judge
2. But code "else if" from line 59 -67 can be shortened, avoidable.

**4th** practice:

[9]<https://gist.github.com/jianminchen/c6a4318a06aa271de834e1e9c1afa7e4>

break else if statement, make it more flat - line 60 - 65, only two if, avoid " else if" in 3rd practice.

**5th** practice:

[10]<https://gist.github.com/jianminchen/8f9ae3839499718790180aa58965d3de>

**6th** practice:

**follow the flow of real processing - skip left/right if need, comparison failed, or continue to compare**

[11]<https://gist.github.com/jianminchen/71421f776948086fd653e441725b3732>

3. Hackthon - programming etc.

[12]<http://blog.csdn.net/DERRANTCM/article/category/6247554>

[13]<http://blog.csdn.net/derrantcm/article/details/51512284>

#### 4. Review palindrom blog

1. <http://juliachencoding.blogspot.ca/2016/08/become-full-stack-net-developer.html>
  2. <http://blog.csdn.net/ebowtang/article/details/50735869>
  3. <http://www.cnblogs.com/airwindow/p/4791234.html>
  4. <http://blog.csdn.net/DERRANTCM/article/details/47651267>
  5. <http://blog.csdn.net/nomasp/article/details/50623165>
  6. <https://gist.github.com/jianminchen/ed12d3004dbe85815f41fef047d48823>
  7. <https://gist.github.com/jianminchen/1e1724c79805d2f40195f08135ec4d02>
  8. <https://gist.github.com/jianminchen/8f9ae3839499718790180aa58965d3de>
  9. <https://gist.github.com/jianminchen/c6a4318a06aa271de834e1e9c1afa7e4>
  10. <https://gist.github.com/jianminchen/8f9ae3839499718790180aa58965d3de>
  11. <https://gist.github.com/jianminchen/71421f776948086fd653e441725b3732>
  12. <http://blog.csdn.net/DERRANTCM/article/category/6247554>
  13. <http://blog.csdn.net/derrantcm/article/details/51512284>
- 

### Leetcode 278 - binary search algorithm and discussion (2016-08-19 19:24)

August 19, 2016

#### Leetcode 278 - binary search algorithm and discussion

1. [1]<http://blog.csdn.net/ebowtang/article/details/50735869>

A. Good discussion on the deadlock prevention - make sure that +1 or -1 somewhere

B. discussion of ?

Good discussion of tips:

1. base case

2. overflow issue

- [2]<http://www.cnblogs.com/airwindow/p/4791234.html>

1. <http://blog.csdn.net/ebowtang/article/details/50735869>
  2. <http://www.cnblogs.com/airwindow/p/4791234.html>
-

## Leetcode 125 - valid palindrome - 5+ practice (2016-08-19 19:25)

August 19, 2016

### problem statement:

Given a string, determine if it is a palindrome, considering only alphanumeric characters and ignoring cases.

For example,

"A man, a plan, a canal: Panama" is a palindrome.

"race a car" is *not* a palindrome.

### Summary of practice:

After 2+ hours workout on coding, with 8th practice, Julia sets pragmatic goal - easy to control:

1. No nested if statement
2. No else statement
3. return early in the function
4. work smart to work around the logic to fit in rule 1, 2, 3

### Arguments of goal:

1. even if the code is written with a bug, easy to find, quick to find
2. at least more readable, aesthetic

better

.

### 8th practice code:

[1]<https://gist.github.com/jianminchen/54200adc1b68928b46fef5a087721f57>

### A table to summarize the workout:

ID Problem Practice Summary Statistics 1 a bug 1st practice increment one, should decrement 2 a bug 2nd practice run time error, logic with a mistake 3 pass Leetcode online judge 3rd practice Fix bugs in 1st and 2nd version. But code is too much with if/else if - 2 times, else if - 3 times, else - 1 time 4 pass Leetcode online judge 4th practice Work on 3rd practice, remove else if statement if - 4 times, else - 1 time 5 5th practice Remove extra checking in while statement (line 44) if - 4 times, else - 1 time 6 6th practice Break the rule, nested while loop, but it works better, code is more clean while - 2, if - 1 time 7 7th practice skip left char, or skip right char or fail because unmatched pair of chars, else statement for rest if - 1 time, else if - 2 times, else - 1 time 8 8th practice Set pragmatic goal, no nested if statement, no else, based on 4th practice if - 4 times

### Practice to win - have some strategy -

1. Using nested while loops twice, 8 out of 10, no bug with excellent code.
2. Use if, and else if 3 times, 3 out of 10, easy to create bug, hard to argue

**3. Use all if, no else, no nested if, 10 out of 10. In future practices, figure out later!**

**9th practice: (after 3+ hours study on code styles, work on issues on practice #8)**

**[2]<https://gist.github.com/jianminchen/ce79fbd9c3c97b628d8857f55e684aab>**

**End of summary of practice**

**5+ practices - a journey with online judge/ blog searching/ what to search after practice I - (try to find a smart person working on if/else statement)**

## **Leetcode 125**

1. study the code:

**[3]<http://blog.csdn.net/DERRANTCM/article/details/47651267>**

Leetcode 125 - write code based on this version

short function name - isalnum - very good name

**[4]<http://blog.csdn.net/nomasp/article/details/50623165>**

5 practices - interest journey !

**1st practice: failed static analysis - miss a bug - easy to spot -, not++**

**[5]<https://gist.github.com/jianminchen/ed12d3004dbe85815f41fef047d48823>**

there is a bug in first writing, line 62, should be right-; not right++.

static analysis did not catch the bug - be careful next time.

**2nd practice:**

**[6]<https://gist.github.com/jianminchen/1e1724c79805d2f40195f08135ec4d02>**

highlights of practice 2:

1. Fail to pass the online judge

2. line 24 - 27, logic and reasoning has flaws

### **logic with a mistake:**

both are alphabetic/ number

first one is not

else -

### **should be:**

both are ok

both are failed

A is failed

B is failed

Those are 4 cases - in the specific order as well.

### **3rd practice:**

[7]<https://gist.github.com/jianminchen/8f9ae3839499718790180aa58965d3de>

highlights of 3rd practice:

1. Fix the bug first, add these line 59 -67 two "else if", one "else", pass online judge

2. But code "else if" from line 59 -67 can be shortened, avoidable.

### **4th practice:**

[8]<https://gist.github.com/jianminchen/c6a4318a06aa271de834e1e9c1afa7e4>

~~break~~ Remove "else if" statement, make it more flat - line 60 - 65, only two if, avoid " else if" in 3rd practice.  
In other words, avoid too complicated if checking

3 cases:

### **3 if statements:**

1st if:

both are ok

Two ifs:

a is not

b is not

### **Good things** in the 4th practice:

1. Inside 1st if statement, line 51 - 56, positive checking first (line 49), then negative.



Declare explanation variable for line 49, and line 51, make it more readable.

**Good news!** Julia sets pragmatic goal - easy to control - no else statement, no nested if statement.

Practice 8 based on practice 4:

[9]<https://gist.github.com/jianminchen/54200adc1b68928b46fef5a087721f57>

*End of Good news!*

**5th** practice:

[10]<https://gist.github.com/jianminchen/8f9ae3839499718790180aa58965d3de>

**6th** practice:

**follow the flow of real processing - skip left/right if need, comparison failed, or continue to compare**

[11]<https://gist.github.com/jianminchen/71421f776948086fd653e441725b3732>

7th practice:

[12]<https://gist.github.com/jianminchen/2efaf43736f87693eb79eaf2a0d2894f>

#### **Actionable Items:**

1. If else is a maze, easy to make mistakes. Things can work on:

1. List of approaches -
2. Possible bugs -
3. What is best strategy to approach?
4. What can be trained on?

Based on practice 4, work on practice 8th, and set practical goals which are easy to control, even it takes more time to write but easy to maintain/ share the code - help to reduce bugs.

Here are practice 8:

[13]<https://gist.github.com/jianminchen/54200adc1b68928b46fef5a087721f57>

#### **Controllable goals:**

1.

##### **No else statement**

- how? practice 8, line 54 - 80, only 4 if statements (line 64, 67, 75, 78)

2. **No nested if statement** - declare short explanation variable, make a few of conditions checking. (two if statements, line 64 go first, even it is negative checking, line 67 after line 64)

3.

let return case go first

4.

hide else relationship

- but with more careful static analysis

Things to break:

1. Always check is True, no negative checking
2. Extra variable for explanation, summarize
3. Scope of variable - not ideal case
4. Avoid nested while loop, let outside while take care of business
5. ...

**Problem with practice #8:** (After 3+ hour study and study

[14]<http://juliachencoding.blogspot.ca/2016/08/leetcode-125-valid-palindrome-summary.html>  
)

study guard clauses,

[15]<http://www.refactoring.com/catalog/replaceNestedConditionalWithGuardClauses.html>

1. first, the style does not match guard clauses style,

**All guard clauses should stay together, and before the clause.**

Put line 75 if, line 78 if into guard clauses, and then, before the normal business:

line 67, if(isComparable && isSame)

2. 4 if conditions are not the same level ( **not same abstraction level!** )

1st if: isCompare && !isSame

2nd if: isCompare && isSame

3rd if: !arr[0]

4rd if: !arr[1]

1st, 3rd, 4rd if are guard clause

put the idea in next practice: 9th practice

1st if: !isCompare && !arr[0]

2nd if: !isCompare && !arr[1]

3rd if: isCompare && !isSame

4th if: isCompare && isSame

**Practice #9:**

**9th practice: (after 3+ hours study on code styles, work on issues on practice #8)**

[16]<https://gist.github.com/jianminchen/ce79fbd9c3c97b628d8857f55e684aab>

**Editorial notes:**

It is time consuming messing with if/else statement.

Facts:

1. 2+ hours programming workout
2. Over 5 practice to experience up and downs,
3. Hard to please a programmer if the code is working but take time to reasoning.
4. Celebrate 2+ hour workout on Leetcode 125 - read

De Morgan's laws

Here is the table of 8 practice:

ID Problem Practice Summary Statistics 1 a bug 1st practice increment one, mistake: decrement one 2 a bug 2nd practice run time error, logic with a mistake 3 pass Leetcode online judge 3rd practice Fix bugs in 1st and 2nd version. But code is too much with if/else if - 2 times, else if - 3 times, else - 1 time 4 pass Leetcode online judge 4th practice Work on 3rd practice, remove else if statement if - 4 times, else - 1 time 5 5th practice Remove extra checking in while statement (line 44) if - 4 times, else - 1 time 6 6th practice Break the rule, nested while loop, but it works better, code is more clean while - 2, if - 1 time 7 7th practice skip left char, or skip right char or fail because unmatched pair of chars, else statement for rest if - 1 time, else if - 2 times, else - 1 time 8 8th practice Set pragmatic goal, no nested if statement, no else, based on 4th practice if - 4 times

Practice to win - have some strategy -

1. Using nested while loops twice, 8 out of 10, no bug with excellent code.
2. Use if, and else if 3 times, 3 out of 10, easy to create bug, hard to argue
3. Use all if, no else, no nested if, 10 out of 10. In future practices, figure out!

Google search and find some related topic:

if/else statement -

1. [17]<http://programmers.stackexchange.com/questions/206816/clarification-of-avoid-if-else-advice>

1. <https://gist.github.com/jianminchen/54200adc1b68928b46fef5a087721f57>
2. <https://gist.github.com/jianminchen/ce79fbd9c3c97b628d8857f55e684aab>
3. <http://blog.csdn.net/DERRANTCM/article/details/47651267>
4. <http://blog.csdn.net/nomasp/article/details/50623165>
5. <https://gist.github.com/jianminchen/ed12d3004dbe85815f41fef047d48823>
6. <https://gist.github.com/jianminchen/1e1724c79805d2f40195f08135ec4d02>
7. <https://gist.github.com/jianminchen/8f9ae3839499718790180aa58965d3de>
8. <https://gist.github.com/jianminchen/c6a4318a06aa271de834e1e9c1afa7e4>
9. <https://gist.github.com/jianminchen/54200adc1b68928b46fef5a087721f57>
10. <https://gist.github.com/jianminchen/8f9ae3839499718790180aa58965d3de>
11. <https://gist.github.com/jianminchen/71421f776948086fd653e441725b3732>
12. <https://gist.github.com/jianminchen/2efaf43736f87693eb79eaf2a0d2894f>
13. <https://gist.github.com/jianminchen/54200adc1b68928b46fef5a087721f57>
14. <http://juliachencoding.blogspot.ca/2016/08/leetcode-125-valid-palindrome-summary.html>
15. <http://www.refactoring.com/catalog/replaceNestedConditionalWithGuardClauses.html>

16. <https://gist.github.com/jianminchen/ce79fbd9c3c97b628d8857f55e684aab>

17. <http://programmers.stackexchange.com/questions/206816/clarification-of-avoid-if-else-advice>

---

## **Confidence and determination coaching - Nishikori Kei and his coach (2016-08-20 12:47)**

August 20, 2016

Review Nishikori and his coach's video.

[1]<https://www.youtube.com/watch?v=LfoYUQIsrg4&feature=youtu.be>

### **Nishikori:**

Not tall, ...

### **Coach's talk:**

You are a little bit smaller, but ability to cover more court, quicker. Use speed to be many many advantages.

Early in the career, have a book, write down who to play with, write down strength/ weakness, why I lose, be prepared, be smart; It is not hitting a good forehand or backhand; Tennis, at the end of day, it is to find the way to win.

People interviewed you before the final match, you told that you admired Roger Federal.

First of mistake, not ready in mental to play final. Your opponent should not be your admire. A great achievement to play final; admire Roger Federal, should be off the court.

### **Mentality and determination:**

But, you are in my way. I do not care about what you have accomplished, prior achievements, no matter whoever is, find the way to beat the guy.

More videos to watch:

1. [2]<https://www.youtube.com/watch?v=s36vSwgn9A0>
  2. [3]<https://www.youtube.com/watch?v=fQBTvSqRCl>
  3. [4]<https://www.youtube.com/watch?v=zMokexfhTIY>
  4. [5]<https://www.youtube.com/watch?v=mBluyNvnSlc>
- IMG - how to build a champion?

### **Most favorite coaching - on-court coaching:**

1. [6][https://www.youtube.com/watch?v=zfZGX\\_f3BOs](https://www.youtube.com/watch?v=zfZGX_f3BOs)

Coach - great motivator/ ...

2. [7]<https://www.youtube.com/watch?v=g-wKFSkHl3Q>
3. [8]<https://www.youtube.com/watch?v=2REF6qaUSG4#t=93.0502>
4. [9]<https://www.youtube.com/watch?v=iK6MMYnNr-M>

5. Novak Djokovic

[10]<https://www.youtube.com/watch?v=9rHV9nDGHXM>

1. <https://www.youtube.com/watch?v=LfoYUQIsrg4&feature=youtu.be>
  2. <https://www.youtube.com/watch?v=s36vSwgn9A0>
  3. <https://www.youtube.com/watch?v=fQBTvSqRcI>
  4. <https://www.youtube.com/watch?v=zMokexfhTIY>
  5. <https://www.youtube.com/watch?v=mBluyNvnSlc>
  6. [https://www.youtube.com/watch?v=zfZGX\\_f3B0s](https://www.youtube.com/watch?v=zfZGX_f3B0s)
  7. <https://www.youtube.com/watch?v=g-wKFSkh13Q>
  8. <https://www.youtube.com/watch?v=2REF6qaUSG4#t=93.0502>
  9. <https://www.youtube.com/watch?v=iK6MMYnNr-M>
  10. <https://www.youtube.com/watch?v=9rHV9nDGHXM>
- 

## Leetcode 125 - Valid Palindrome - Code styles study (2016-08-21 10:41)

August 21, 2016

Here is the blog with 5+ practice:

[1]<http://juliachencoding.blogspot.ca/2016/08/leetcode-125-valid-palindrome-5-practice.html>

**8th practice code:**

[2]<https://gist.github.com/jianminchen/54200adc1b68928b46fef5a087721f57>

**Summary of study:**

**Go over blogs, read 2+ hours, and then, write 9th practice; Be able to find issues in 8th practice.**

**9th practice:**

[3]<https://gist.github.com/jianminchen/ce79fbd9c3c97b628d8857f55e684aab>

**10th practice: - best solution - extract one more function called removeNoise**

[4]<https://gist.github.com/jianminchen/765a169c1f4e3b1d1fc66f3deca16410>

**11th practice:**

[5]<https://gist.github.com/jianminchen/2e82c22b4b480a4b5eb06f9507956a60>

Julia likes to study code style blogs to help better performance on Leetcode 125:

1. [6]<http://programmers.stackexchange.com/questions/122485/elegant-ways-to-handle-if-else-else>
2. fail fast:
- [7]<http://www.martinfowler.com/ieeeSoftware/failFast.pdf>
3. [8]<http://programmers.stackexchange.com/questions/205803/how-to-tackle-a-branched-arrow-head-anti-pattern>
4. [9]<https://blog.codinghorror.com/flattening-arrow-code/>
5. Refactoring: Book: "improving the Design of Existing Code" - if-else structure sections
6. [10]<http://programmers.stackexchange.com/questions/167607/how-can-i-re-format-my-condition-to-make-it-better?noredirect=1&lq=1>
7. [11]<http://stackoverflow.com/questions/1364946/what-is-the-highest-complexity-of-any-function-you-maintain-and-how>

8. A lot of discussions:

[12]<http://stackoverflow.com/questions/36707/should-a-function-have-only-one-return-statement>

9. Guard clauses

[13]<http://www.refactoring.com/catalog/replaceNestedConditionalWithGuardClauses.html>

Try this tool - static analysis tool

8. [14]<http://www.mccabe.com/>

### Study notes:

Get some keywords for google search, or ideas to help review of function:

1. **inconsistent levels** of abstraction
2. prevent **inclusion in if**
3. two calls to one function violate **DRY principle**

### Execution paths concern:

1. **multiple returns** - avoid it
2. **only return** from a function **in one spot** - (Good ? Bad?)

Julia made a terrible mistake in 2 sum algorithm, about the return issue - took more than 20 minutes to find out return issue.

3. multiple return paths
4. early returns help to avoid the **arrowhead anti pattern**.

### Readable code

1. use a **flag variable** to avoid giant expression/ gross if statement.
2. orthogonal ?
3. early returns help to avoid the **arrowhead anti pattern**.
4. flattening arrow code
5. cyclomatic complexity value - execution paths
6. reduce a module's cyclomatic complexity make code easy to test

### Avoid If statement

1. replace conditions with guard clauses.
2. decompose conditional blocks into separate functions
3. convert negative checks into positive checks
4. always opportunistically return as soon as possible from the function.
5. the goal to make code more flat, scroll vertically not horizontally

Book "**code complete**" about cc - cyclomatic complexity value:

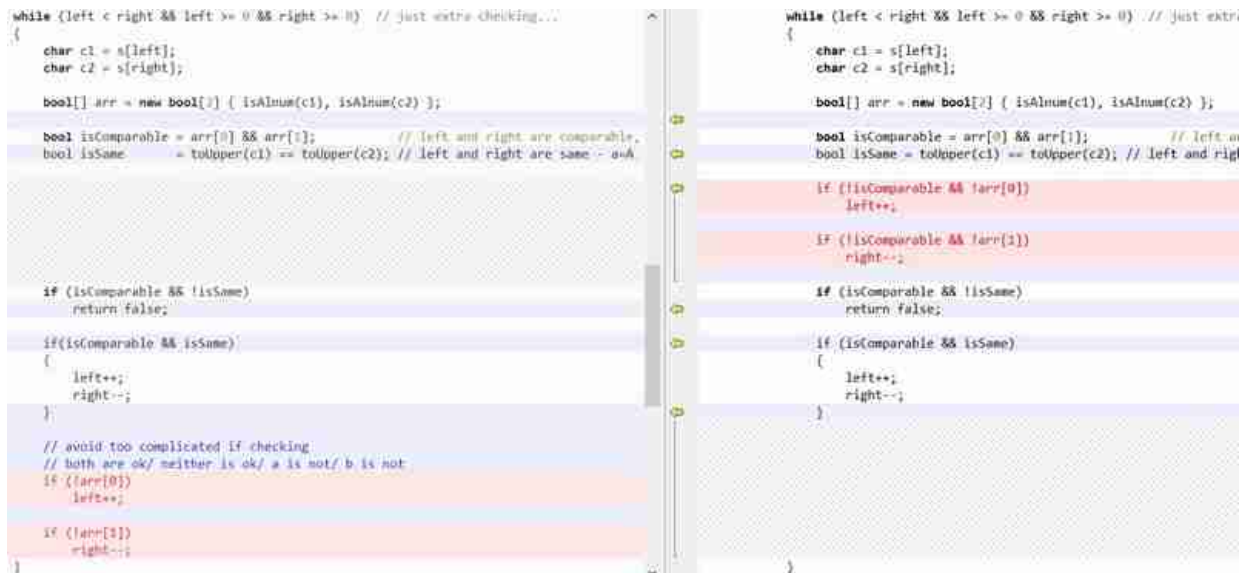
If the score is:

- 0-5 - the routine is probably fine
- 6-10 - start to think about ways to simplify the routine
- 10+ - break part of the routine into a second routine and call it from the first routine

Might be a good idea to write unit tests as you break up the original routine.

### A quick comparison image:

Leetcode 125 8th practice vs 9th practice on the change - if conditions inside a while loop:



Continue to review solutions on Leetcode 125. (August 22, 2016)

1. study code:

[15]

[https://github.com/jianminchen/leetcode-](https://github.com/jianminchen/leetcode-1/blob/master/algorithms/validPalindrome/validPalindrome.cpp)

[1/blob/master/algorithms/validPalindrome/validPalindrome.cpp](https://github.com/jianminchen/leetcode-1/blob/master/algorithms/validPalindrome/validPalindrome.cpp)

extract one more function - removeNoise - **practice #10**

C # practice

[16]<https://gist.github.com/jianminchen/765a169c1f4e3b1d1fc66f3deca16410>

Great idea to extract one more function, SRP - single responsibility principle

- make the code much more easy to follow, test, static analysis

**Two tasks - put into one function - think if the function is designed for more than 1 task.**

Both are with time complexity  $O(N)$ ,  $N$  is the string length;

Two tasks can be separated easily, task 1 can be done first.

1. remove noise

2. compare chars

Easily, combine the above 1 and 2 cases, there are 4 cases in the function to handle.

2. use continue keyword - **practice #11**

study code:

[17]<https://github.com/jianminchen/leetcode-cpp/blob/master/src/ValidPalindrome.cpp>

C # practice:

[18]<https://gist.github.com/jianminchen/2e82c22b4b480a4b5eb06f9507956a60>

1. <http://juliachencoding.blogspot.ca/2016/08/leetcode-125-valid-palindrom-5-practice.html>
  2. <https://gist.github.com/jianminchen/54200adc1b68928b46fef5a087721f57>
  3. <https://gist.github.com/jianminchen/ce79fbd9c3c97b628d8857f55e684aab>
  4. <https://gist.github.com/jianminchen/765a169c1f4e3b1d1fc66f3deca16410>
  5. <https://gist.github.com/jianminchen/2e82c22b4b480a4b5eb06f9507956a60>
  6. <http://programmers.stackexchange.com/questions/122485/elegant-ways-to-handle-ifif-else-else>
  7. <http://www.martinfowler.com/ieeeSoftware/failFast.pdf>
  8. <http://programmers.stackexchange.com/questions/205803/how-to-tackle-a-branched-arrow-head-anti-pattern>
  9. <https://blog.codinghorror.com/flattening-arrow-code/>
  10. <http://programmers.stackexchange.com/questions/167607/how-can-i-reformat-my-condition-to-make-it-better?noredirect=1&lq=1>
  11. <http://stackoverflow.com/questions/1364946/what-is-the-highest-cyclomatic-complexity-of-any-function-you-maintain-and-how>
  12. <http://stackoverflow.com/questions/36707/should-a-function-have-only-one-return-statement>
  13. <http://www.refactoring.com/catalog/replaceNestedConditionalWithGuardClauses.html>
  14. <http://www.mccabe.com/>
  15. <https://github.com/jianminchen/leetcode-1/blob/master/algorithms/validPalindrome/validPalindrome.cpp>
  16. <https://gist.github.com/jianminchen/765a169c1f4e3b1d1fc66f3deca16410>
  17. <https://github.com/jianminchen/leetcode-cpp/blob/master/src/ValidPalindrome.cpp>
  18. <https://gist.github.com/jianminchen/2e82c22b4b480a4b5eb06f9507956a60>
- 

## Leetcode 125: Valid Palindrome (III) (2016-08-22 21:27)

August 22, 2016

Continue to review solutions on Leetcode 125.

1. study code:

[1]

<https://github.com/jianminchen/leetcode-1/blob/master/algorithms/validPalindrome/validPalindrome.cpp>

extract one more function - removeNoise - **practice #10**

C # practice

[2]<https://gist.github.com/jianminchen/765a169c1f4e3b1d1fc66f3deca16410>

Great idea to extract one more function, SRP - single responsibility principle

- make the code much more easy to follow, test, static analysis

**Two tasks - put into one function - first ask if the function can be broken into two, what is the checkpoints?**

Both are with time complexity  $O(N)$ ,  $N$  is the string length;

Two tasks can be separated easily, task 1 can be done first.

1. remove noise

2. compare chars

Easily, combine the above 1 and 2 cases, there are 4 cases in the function to handle.

2. use continue keyword - **practice #11**



study code:

[3]<https://github.com/jianminchen/leetcode-cpp/blob/master/src/ValidPalindrome.cpp>

C # practice:

[4]<https://gist.github.com/jianminchen/2e82c22b4b480a4b5eb06f9507956a60>

1. <https://github.com/jianminchen/leetcode-1/blob/master/algorithms/validPalindrome/validPalindrome.cpp>

2. <https://gist.github.com/jianminchen/765a169c1f4e3bd1fc66f3deca16410>

3. <https://github.com/jianminchen/leetcode-cpp/blob/master/src/ValidPalindrome.cpp>

4. <https://gist.github.com/jianminchen/2e82c22b4b480a4b5eb06f9507956a60>

---

## **Linked List Queue (2016-08-25 21:30)**

Problem:

Implement a Queue using a linked list.

Will come back soon.

---

## **Array Queue (2016-08-25 21:30)**

August 25, 2016

Problem statement:

Implement a Queue using a circular array.

Read circular array:

[1]<http://www.mathcs.emory.edu/~cheung/Courses/171/Syllabus/8-List/array-queue2.html>

Will come back very soon!

---

1. <http://www.mathcs.emory.edu/~cheung/Courses/171/Syllabus/8-List/array-queue2.html>

---

## **System Design: Design a URL shortening service (2016-08-25 22:06)**

August 25, 2016

Problem statement:

Design a URL shortening service

- What do you think the requirements are?
- What is your high level design?
- You might be asked to implement interesting portions
- How do you test it?

Will work on the design very soon.

1. Read the website first: 5 - 10 minutes

[1]<http://stackoverflow.com/questions/1562367/how-do-short-urls-services-work>

2.

[2]<http://www.hiredintech.com/system-design/the-system-design-process/>

### **Step 1: Constraints and use cases**

### **Step 2: Abstract Design**

Start from step 1:

System constraints (Do you clarify?)

Use cases (the list to satisfy?)

Scope system (agree?)

Gather requirements -> design a solution to cover them well.

#### **Use cases:**

1. Shortening:
2. Redirection:
3. Custom Url
4. Analytics
5. Automatic link expiration
6. Manual link removal
7. UI or API

And then, discuss and exchange ideas with requirements:

For example, add one more use case 4, and move 4 - 7 out of scope, not for now.

1. Shortening:
2. Redirection:
3. Custom Url
4. High availability of the system

#### **out of scope:**

4. Analytics
5. Automatic link expiration
6. Manual link removal
7. UI or API

#### **Constraints:**

**Top 10 shorten URL services:**

**Math to figure out, for example:**

**15 billion new tweets in twitter,**

**All shortened URLs per month: 1.5BN - 10 percent**

**Sites below the top 3: shorten 300M per month**

**1. New urls per month: 100MLN - 100 millions request to shortening**

**2.**

**3.**

**4.**

20 % urls generate 80 % traffic -> 1BN request per month -> 10 % shortening, 90 % redirection.

4. Requests per second: 400+ requests per second (40: shortens, 360: redirects)

5. 6BN urls in 5 years -

6. 500 bytes per URL (100 to 1000 characters per URL)

7. 6 bytes per hash

8. 3TBs for all urls, 36GB for all hashes (over 5 years, not 10 years)

9. how much data goes through each second?

New data written per second:  $40 * (500 + 6)$ : 20K

10. Data read per second:  $360 * 506$  bytes: 180K

scope the problem beautifully - it takes a while, but 15 minutes in the video, with practice, it takes 5 minutes to clear up everything.

Sept. 3, 2016

Spend 2 hours to watch the video:

[3]<http://www.hiredintech.com/system-design/scalability-fundamentals/>

Take some notes:

1. <http://stackoverflow.com/questions/1562367/how-do-short-urls-services-work>
  2. <http://www.hiredintech.com/system-design/the-system-design-process/>
  3. <http://www.hiredintech.com/system-design/scalability-fundamentals/>
- 

## **Leetcode 348: Design Tic-Tac-Toe (2016-08-25 22:10)**

August 25, 2016

Problem statement:

Design a Tic-tac-toe game that is played between two players on a  $n \times n$  grid.

You may assume the following rules:

1. A move is guaranteed to be valid and is placed on an empty block.
2. Once a winning condition is reached, no more moves is allowed.
3. A player who succeeds in placing  $n$  of their marks in a horizontal, vertical, or diagonal row wins the game.

Will work on the algorithm very soon.

---

## **CSS - Responsive background image (2016-08-25 22:30)**

August 31, 2016

Spent over 10+ hours to work on responsive background image issues. So, Julia likes to spend time to serve herself better using CSS.

Google keyword search:

css responsive background image

---

## **Code study: Leetcode blogs (2016-08-25 22:30)**

August 25, 2016

Compile a list of Leetcode blogs to read, 10 minutes a time, look for excellence through blogs:

1. [1]<http://bangbingsyb.blogspot.ca/> (Microsoft, C++)
2. [2]<http://fisherlei.blogspot.ca/> (facebook, blogs, C++)

C++ solutions in github:

1. [3]<https://github.com/jianminchen/leetcode-1> (C++, ex-amazoner)
2. [4]<https://github.com/liaoxl/leetcode/tree/master/code> (Neteaze, C++, China)
3. [5]<https://github.com/jianminchen/Leetcode-13> (UBC, C++)
4. [6]<https://github.com/jianminchen/myleetcode> (China, Microsoft, C++)
5. [7]<https://github.com/jianminchen/LeetCode-III> (C++, HackerRank Silver medal)
6. **Inside github, search HackerRank, find this:** (MSFT, top performer, with time spent for each algorithm)
- [8]<https://github.com/jianminchen/LeetCode-17>

Java solutions in github:

1. [9]<https://github.com/jianminchen/LeetCode-Java-Solutions>
2. [10]<https://github.com/jianminchen/LeetCode-Java-Solutions> (Java, computer PH.D.)
3. [11]<https://github.com/jianminchen/LeetCode-Sol-Res> (Nine chapter etc., Java)
4. [12]<https://github.com/jianminchen/leetcode-3> (Java solution, Googler)

C # solution:

1. [13]<https://github.com/jianminchen/LeetSharp>
2. [14]<https://github.com/jianminchen/Qilu-leetcode>
2. [15]<https://github.com/jianminchen/LeetCodeInCSharp>

C++ - **Lintcode book**

- [16]<https://github.com/jianminchen/LintCodeBook>

JavaScript: Leetcode

1. [17]<https://github.com/jianminchen/leetcode-28>

### **Motivation talk:**

1. Sometime, google, bing search do not provide best source code, so, it is time to look up github forked solutions.

### **Friendly remind:**

**Julia, you should use github's search - use keyword: Leetcode to find solutions -**

2. Look for optimal solution, do not miss the best solution in practice of Leetcode algorithm. Do not rush, try to go through all kinds of practices, search for great ideas out there. Get the best idea to practice. One algorithm a time.
3. Follow more people through github, easy to fork leetcode solutions. Sometimes, it is hard to tell best code, just get some code running first, write first C # practice, and then, ideas will come, what to search, issues etc.
4. Evaluate practices (junior/middle/senior level practice), and then compare to 5-10 practices. (0-3 junior practice, 4-7 middle level, 1-2 optimal solution, senior level practice)
4. Look up stackoverflow to get more ideas, specially from high reputation/ high voted opinions, terms to know when practicing Leetcode algorithms. Try to expand the things to work on through the practice, one algorithm can bring good study about coding style. See the blog:  
[18]<http://juliachencoding.blogspot.ca/2016/08/leetcode-125-valid-palindrome-summary.html>

5. When practicing Leetcode questions, try to work on more through the practice. There is an idea that if you work on thoroughly on one problem a time, you may be able to solve unseen problems as well.

### Actionable Items:

1. Study programming principles:

#### Programming principles

[19]<https://github.com/jianminchen/programming-principles>

1. <http://bangbingsyb.blogspot.ca/>
  2. <http://fisherlei.blogspot.ca/>
  3. <https://github.com/jianminchen/leetcode-1>
  4. <https://github.com/liaoxl/leetcode/tree/master/code>
  5. <https://github.com/jianminchen/Leetcode-13>
  6. <https://github.com/jianminchen/myleetcode>
  7. <https://github.com/jianminchen/LeetCode-IIII>
  8. <https://github.com/jianminchen/LeetCode-17>
  9. <https://github.com/jianminchen/LeetCode-Java-Solutions>
  10. <https://github.com/jianminchen/LeetCode-Java-Solutions>
  11. <https://github.com/jianminchen/LeetCode-Sol-Res>
  12. <https://github.com/jianminchen/leetcode-3>
  13. <https://github.com/jianminchen/LeetSharp>
  14. <https://github.com/jianminchen/Qilu-leetcode>
  15. <https://github.com/jianminchen/LeetCodeInCSharp>
  16. <https://github.com/jianminchen/LintCodeBook>
  17. <https://github.com/jianminchen/leetcode-28>
  18. <http://juliachencoding.blogspot.ca/2016/08/leetcode-125-valid-palindrome-summary.html>
  19. <https://github.com/jianminchen/programming-principles>
- 

### Programming Principles - study (2016-08-27 11:08)

August 27, 2016

Through leetcode practice Algorithm 125 in August 2015, Julia noticed that she still stumbled badly on easy question. Because ..., **wild guess?:-)** she does not have good understanding of programming principles.

She wrote blogs about [1]the practice of Leetcode 125, 11 practice, through continuously hard word through 3 days, she finally came into realization of best solution - extract one more function called removed noise. She read the code written in C++ in her github forked solution more than 6 months ago. She could not find the best solution through google/ bing, and she studied stackoverflow related articles more than 2 hours, but she just did not come out the idea.

Apply **Separation of concerns principle?** optimal solution to Leetcode 125.

The above is enough for her to dedicate herself to read more about programming principles.

### Programming Principles

[2]<https://github.com/jianminchen/programming-principles>

Plan to work on reading 3+ hours, write down some notes here.

1. <http://juliachencoding.blogspot.ca/2016/08/leetcode-125-valid-palindrome-iii.html>
  2. <https://github.com/jianminchen/programming-principles>
- 

## **Java Design Pattern - code study (2016-08-27 11:12)**

August 27, 2016

Plan to spend 3+ hours to go over some Java Design Pattern.

[1]<https://github.com/jianminchen/java-design-patterns>

1. <https://github.com/jianminchen/java-design-patterns>
- 

## **Html, CSS style guideline study (2016-08-27 11:17)**

August 27, 2016

Always work on code guidelines first, then write more code everyday.

Plan to spend 2+ hours to study html, CSS code guide.

[1]<http://codeguide.bootcss.com/>

[2]<https://google.github.io/styleguide/htmlcssguide.xml>

1. <http://codeguide.bootcss.com/>
  2. <https://google.github.io/styleguide/htmlcssguide.xml>
- 

## **Blog writing - aiming for good writing talent (2016-08-27 11:42)**

August 27, 2016

It is a nice journey to write a lot of blogs in last 12 months. Julia learns to handle a lot of stress, spend a lot of time to proofread, feel headache about the content, organization of ideas etc. Sometimes, she notice that she writes something she does not know, she does not have best solution, since she is still developing, in the cooking process. *Or she just pushes herself, just do it.* It may be an interesting topic, start first, come back to work on.

What is about good writing talent? It is not just English writing, also about communication skills, inspiration to share, good language skills .

***A true short story about her personal experience:***

She went through 3 IELTS test in 2008 from June to August in order to score 2 of 7/9 on English (reading, writing, speaking, listening) - apply Canadian immigration. She did 3 tests in 3 months, because Fort Lauderdale test center only gives one test a month. She even filed an appeal against the scoring result after 2nd test, costed her \$160 dollars, same amount as the test fee. She practiced old tests in the time range of 3 month - 3 hours for one test, she did again and again.

More detail about IELTS:

[1]<http://www.ielts-blog.com/ielts-results-competition-winners-2008/>

***End of the true short story.***

And then, in 2015, Julia learned that in order to  
~~get best result of~~  
~~build up some writing talent continuously in long term,~~  
grow up from a junior software developer to talent programmer,  
she has to write down her coding practice everyday, speak out her unique experience, ideas/ practice log, show her progression, weakness, highlights, one step a time, one practice a time. **There is no way to avoid hard work, embrace it. She learns from her favorite tennis sports.**

**So, she try to push herself to work on writing. Being a good software programmer, writing code is much more challenging.**

**Here are some ideas:**

**From blog:**

1. [2]<https://www.elegantthemes.com/blog/tips-tricks/how-to-improve-your-blog-writing-skills>

1. Make writing a habit

As the old saying goes:

Practice Makes Perfect

.

2. Get back to basics

3. Proofread your articles

4. Remove the filler

- tighten your article and make it easier to read

5. Evaluate your writing



6. Be an avid reader  
passionate writer should be a passionate reader.
7. Establish good writing habits

**Google search keyword:**

blog writing help english writing

**Second blog:**

[3]<http://www.wordstream.com/blog/ws/2014/08/07/improve-writing-skills>

Notes:

1. Brush up on the basics
2. Write like it's your job  
Keep practicing, writing is no exception.  
diminish your fear of the blank page (or blinking cursor)  
develop a unique style  
even if nobody reads it, keep writing.
3. Read like it's Your Job  
reading on a regular basis  
pay attention to sentence structure, word choice, and how the material flows.
4. Find a writing partner  
a solitary activity - writing?  
cast an eye over your work?  
spot mistakes that you may overlooked
5. Join a workshop or take a night class - not applicable right now
6. Dissect writing that you admire
7. Imitate writers you admire  
use humor to spice up dry topics?  
use pop culture references to make their work entertaining and useful?
8. Remember that outlines are your friend - outline?  
solid outline plan? When, who, what, where, why, how? 7 W
9. Edit your work ruthlessly  
  
be your own harshest critic ?

editing is a tough skill to learn for beginner?  
writing or re writing?  
the cold, hard eye of an editor will serve you well?  
**discipline** to eliminate extraneous words?  
**resist** the temptation to wax lyrically and get to the point?

10. Accept that first draft are almost always crap

write down on paper first, clean up later. Iterative process. Do not beat yourself up.

11. Find a good patient editor

12. Eliminate unnecessary words

13. Take a stroll down memory lane

14. **Don't be afraid to say what you think**

15. Do your research

16. Don't take weeks to finish a post

**Actionable items:**

*Try 3 ideas first:*

14. *Don't be afraid to say what you think*

*testimony:*

**Share personal IELTS test experience in 2008**

1. <http://www.ielts-blog.com/ielts-results-competition-winners-2008/>

2. <https://www.elegantthemes.com/blog/tips-tricks/how-to-improve-your-blog-writing-skills>

3. <http://www.wordstream.com/blog/ws/2014/08/07/improve-writing-skills>

---

**Abbreviation - HackerRank world code sprint #6 (2016-08-28 12:10)**

August 26, 2016

Problem statement:

[1]<https://www.hackerrank.com/contests/world-codesprint-6/challenges/abbr>

C # practice:

[2]<https://gist.github.com/jianminchen/12e7c74b57c5780a069b0e9df66f77c3>

Study editorial notes about the algorithm, using dynamic programming, a DP solution.

Study other submissions through HackerRank.

1. <https://www.hackerrank.com/contests/world-codesprint-6/challenges/abbr>
  2. <https://gist.github.com/jianminchen/12e7c74b57c5780a069b0e9df66f77c3>
- 

## Bonetrousle - HackerRank world code sprint #6 - Practice 1 (2016-08-28 12:15)

August 28, 2016

Problem statement:

[1][https://www.hackerrank.com/contests/world-codesprint-6/challenges/bone trousle](https://www.hackerrank.com/contests/world-codesprint-6/challenges/bone%20trousle)

Julia was naive on planning, management of time, tried to solve this algorithm, without any hesitation. She tried to score any point above 0, found the idea to write code using stack, but with additional 2+ hours to review code, stayed overnight, from 12:00am - 2:45am, a brute force solution. She did not make any (over 10 test cases: 5+ wrong answer, 4+ time out). So, she wrote down the experience to **celebrate** her weekend, over middle night struggling with an algorithm.

Share some statistics of world code sprint #6 workout:

**Current Rank: 1112 - score 100/ 380 (score first 4 algorithms full score)**

First practice:

Use stack, brute force, each box has two choice, join or skip. She tried to implement the brute force solution first, and then, she recalled the previous work on algorithm Leetcode: phone number:

[2]<https://gist.github.com/jianminchen/872bf70039fa8c61ff208b34a591c8ec>

Only pass the test cases provided in the problem description.

[3]<https://gist.github.com/jianminchen/e3fc7d23a62274b207ddc41a06030cd4>

**Statistics:**

score 0 out of 50 points.

**Brute facts:**

A brute force solution does not score any points - even pass the basic test case - test case 1.

**Actionable Item:**

Write a blog about this algorithm, document the importance of time complexity analysis; how to plan to spend time to work on a solution.

[4][http://juliachencoding.blogspot.ca/2016/08/bonetrousle-hackerrank-world-code \\_75.html](http://juliachencoding.blogspot.ca/2016/08/bonetrousle-hackerrank-world-code-_75.html)

1. <https://www.hackerrank.com/contests/world-codesprint-6/challenges/bonetrousle>
  2. <https://gist.github.com/jianminchen/872bf70039fa8c61ff208b34a591c8ec>
  3. <https://gist.github.com/jianminchen/e3fc7d23a62274b207ddc41a06030cd4>
  4. [http://juliachencoding.blogspot.ca/2016/08/bonetrousle-hackerrank-world-code\\_75.html](http://juliachencoding.blogspot.ca/2016/08/bonetrousle-hackerrank-world-code_75.html)
- 

## **Bonetrousle - HackerRank world code sprint #6 - Practice 2 (2016-08-28 12:22)**

August 28, 2016

Problem statement:

[1][https://www.hackerrank.com/contests/world-codesprint-6/challenges/bone trousle](https://www.hackerrank.com/contests/world-codesprint-6/challenges/bone%20trousle)

Try to make any progress on timeout issue, wrong answers issue. Add additional checking to shorten the time to do processing:

[2]<https://gist.github.com/jianminchen/ff048969b7b69948eb3dc4db59119591>

**Highlights of practice:**

1. line 113

**Statistics:**

score 0 out of 50 points.

1. <https://www.hackerrank.com/contests/world-codesprint-6/challenges/bonetrousle>
  2. <https://gist.github.com/jianminchen/ff048969b7b69948eb3dc4db59119591>
- 

## **Bonetrousle - HackerRank world code sprint #6 - Practice 3 (2016-08-28 12:30)**

August 28, 2016

Practice 3:

[1]<https://gist.github.com/jianminchen/da8be63104ef404e82733669a3d60370>

**Highlights of practice:**

1. Add hashset to filter out the boxes

**Statistics:**

Score 0 out of 50.

1. <https://gist.github.com/jianminchen/da8be63104ef404e82733669a3d60370>

---

## **Bonetrroule - HackerRank world code sprint #6 - code study (2016-08-28 12:31)**

August 28, 2016

Study 5+ C # submission code, put some notes here as well.

1. Use Queue, using structure, excellent code to study -

[1]<https://gist.github.com/jianminchen/4bb8d763056684d33959d7806b471e03>

rank before 190/ 5332

Great workout using Queue, Julia came out this idea through practice, but she could not write down code. She tried to write recursive function.

2. a while loop, less than 40 lines code

[2]<https://gist.github.com/jianminchen/835965989b54834583266b6dba5fa570>

rank before 200/ 5332, score around 250/380

3.

[3]<https://gist.github.com/jianminchen/3e7b7889a179d592450a6fae7dc854bf>

4.

[4]<https://gist.github.com/jianminchen/2ab5fd357dd61a264ec918a4191b9d54>

6. Great code!

[5]<https://gist.github.com/jianminchen/18f3cd2321cfee9226d740e239ec8418>

rank in range of 270 - 300/ 5332

participants (score in range of 180- 200/380)

Time to go over C # string class:

C # string class join method:

[6][https://msdn.microsoft.com/en-us/library/tk0xe5h0\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/tk0xe5h0(v=vs.110).aspx)

**Motivation talk:**

work hard!

Julia ranks 1112 on world code sprint #6. If she can work out the algorithm - Bonetrousle, based on her current score is 100, she can score another 50, from score of 100 to 150, then, she can get into top 1100 to top 550 - 650 ranking.

1. <https://gist.github.com/jianminchen/4bb8d763056684d33959d7806b471e03>
  2. <https://gist.github.com/jianminchen/835965989b54834583266b6dba5fa570>
  3. <https://gist.github.com/jianminchen/3e7b7889a179d592450a6fae7dc854bf>
  4. <https://gist.github.com/jianminchen/2ab5fd357dd61a264ec918a4191b9d54>
  5. <https://gist.github.com/jianminchen/18f3cd2321cfee9226d740e239ec8418>
  6. [https://msdn.microsoft.com/en-us/library/tk0xe5h0\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/tk0xe5h0(v=vs.110).aspx)
- 

## C# string class - API study (2016-08-28 21:37)

August 28, 2016

Previous blog about string class study for C #, Java, JavaScript, C++:

[1]<http://juliachencoding.blogspot.ca/2016/07/c-c-javascript-java-string.html>

Go over all APIs in string class - C # programming language:

[2][https://msdn.microsoft.com/en-us/library/system.string\\_methods\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.string_methods(v=vs.110).aspx)

**40 APIs** - Go over it 30 minutes a time - start to memorize them: function signature etc.

Clone

Compare

CompareTo

Concat

Contains

Copy

CopyTo

EndsWith

Equals

Format

GetEnumerator

GetHashCode

GetType

GetTypeCode

IndexOf - (8 overloaded version)

IndexOfAny (3 overloaded version)

Insert

Intern

IsInterned

IsInterned

IsNormalized

IsNullOrWhiteSpace

Join (5 overload)  
LastIndexOf (8 overloaded version)  
Normalize  
PadLeft  
PadRight  
Remove  
Replace  
Split (6 overloaded version)

StartsWith (3 version)  
Substring  
ToCharArray  
ToLower  
ToLowerInvariant  
ToString  
ToUpper  
ToUpperInvariant  
Trim  
TrimEnd  
  
TrimStart

1. <http://juliachencoding.blogspot.ca/2016/07/c-c-javascript-java-string.html>
  2. [https://msdn.microsoft.com/en-us/library/system.string\\_methods\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.string_methods(v=vs.110).aspx)
- 

## **Bonetrousle - HackerRank world code sprint #6 - Time complexity (2016-08-28 22:13)**

August 28, 2016

Worked on the algorithm over 3 hours, Bonetrousle. HackerRank code sprint #6.

Julia started to train herself to get smart on algorithm problem solving - contest level. First and big lesson is **to put the time complexity analysis first** - she experienced the pain and valued the lesson she learned through 3+ hours.

Editorial notes from HackerRank:

[1]<https://www.hackerrank.com/contests/world-codesprint-6/challenges/bone-trousle/editorial>

Will work on this blog later!

Come back on Sept. 1, 11:15pm.

The problem is about loops, and let us work on the example used in the editorial notes:

$N = 15$ ,  $K = 8$ ,  $B = 3$ , in other words, find 3 distinct numbers from set  $\{1, 2, \dots, 8\}$ , and the sum is 15.

#### **First idea: brute force one**

Brute force, 3 number, each one is chosen from 1-8, 3 time, then, simple combinatorics:  $8 * 7 * 6$ ,  $K(K-1)(K-2)$ , , the order does not matter, so it is should  $C(8,3)$ .

As we know,  $N < 10^{18}$ ,  $K \leq 10^{18}$ , so, Julia, you like to make any points, try to avoid any brute force solution like the above.

#### **Second idea: How low it can be?**

Find B numbers, as we know, from the example,  
 $N = 15$ ,  $K = 8$ ,  $B = 3$ , there are more than 1 solution.

But, there is one definitely taking minimum time. That is to find maximum number in B number set first, and it takes  $O(1)$  time. Then, in decreasing order, find one by one. The total time can be small as  $O(B)$ .

Simple math, we can analyze first if N is in the range of minimum value and maximum value range first.

if  $B > K$ , not possible;

Assuming  $B \leq K$ ,  
Minimum value =  $1 + 2 + \dots + B$ ,  
Maximum value =  $K + (K-1) + (K-B+1)$

1. <https://www.hackerrank.com/contests/world-codesprint-6/challenges/bonetrousle/editorial>

---

## **C# StringBuilder Class - study (2016-08-28 22:51)**

August 28, 2016

Work on C # StringBuilder class APIs.

[1][https://msdn.microsoft.com/en-us/library/system.text.stringbuilder\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.text.stringbuilder(v=vs.110).aspx)

Will work on the APIs 30 minutes a time. Try to memorize all of them, understand the design in detail: function signatures, overload, optional arguments etc.



1. [https://msdn.microsoft.com/en-us/library/system.text.stringbuilder\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.text.stringbuilder(v=vs.110).aspx)

---

## Beautiful 3 set - code study (2016-08-29 23:48)

August 29, 2016

Plan to work on difficult level algorithm - world code sprint #6. Plan to work on world code sprint some day, be able to solve difficult/ advanced algorithm.

[1]<https://www.hackerrank.com/contests/world-codesprint-6/challenges/beautiful-3-set>

Will come back very soon to work on this algorithm.

### Fun facts about HackerRank world code sprint #6 :

1. Julia won her first **Bronze medal!** Excellent! a big surprise!  
Get into top 25 % - Bronze medal (Gold 4 %, Silver 8 %, Bronze 13 %)

[2]<https://www.hackerrank.com/scoring/rating>

2. Julia worked on a simple question: Flip matrix more than 1 hour  
problem statement:

[3]<https://www.hackerrank.com/contests/world-codesprint-6/challenges/flipping-the-matrix>

C # practice:

[4]<https://gist.github.com/jianminchen/856f97a7f01f049efbde50078b7699ba>

She was so interested in the flip part, later, she figured out that the algorithm should not be so complicated. It is just a medium level one.

3. Julia learned DFS/BFS algorithm through HackerRank, she found the joy to read code and try to write one for every different idea.

4. Being a programmer, it is hard to control your luck. But HackerRank contest looks like more controllable. More practice leads to more medal. Julia likes a silver one next time. <- **Nothing is impossible! top 12 %. Julia tried to [5]stay overnight Saturday night, get in top 600/ 12 %.**

### Fun facts:

1. **More than 5 computer professors / score range: around 200 score, silver medal, 230 - 270 ranking.**  
2. **Julia, work hard, try some difficult level algorithm day by day.**

1. <https://www.hackerrank.com/contests/world-codesprint-6/challenges/beautiful-3-set>

2. <https://www.hackerrank.com/scoring/rating>

3. <https://www.hackerrank.com/contests/world-codesprint-6/challenges/flipping-the-matrix>

4. <https://gist.github.com/jianminchen/856f97a7f01f049efbde50078b7699ba>

5. <http://juliachencoding.blogspot.ca/2016/08/bonetrousle-hackerrank-world-code.html>

---

## A small research on HackerRank - seek excellence (2016-08-31 00:25)

August 30, 2016

Go over HackerRank world code sprint lead board, from ranking 230 to 430, go over one by one, and see how Julia can make it in short future. To be one of them, what is missing, what should learn from those people.

From different backgrounds, computer professors, Google employees, Microsoft employees, and a lot of others, university students. But, most of them attended more than 4 contests, and showed strong interest in things:

1. Some of them take Coursera courses - More than 10 of courses
2. Some of them document their HackerRank contests on LinkedIn profile.
3. Some of them are excellent coding, one of them finishes Leetcode over 200 questions.

Julia just forked solution in the next minute, will study the code.

4. Some of them are from Intel, Intuit, Cisco, etc.

Read code, study styles, that is Julia's favorite thing to do.

Julia has to take into consideration, work on difficult level algorithm on HackerRank. She prefers to stay in easy, medium level up to August 31, 2016.

Blogs to read:

1. practice, drills, strategies - a computer science course to help

[1]<https://cs.illinois.edu/news/illinois-team-advances-icpc-world-finals>

[2]<http://goo.gl/ZdJ8C0>

2. [3]<https://algo.is/>

**competitive programming course**

[4]<https://sites.google.com/site/stevenhalim/>

Free download - version 1

[5][http://www.comp.nus.edu.sg/~stevenha/myteaching/competitive\\_programming/cp1.pdf](http://www.comp.nus.edu.sg/~stevenha/myteaching/competitive_programming/cp1.pdf)

[6][http://linc.nus.edu.sg/search\\_S16?averdu+%2CElena./averdu+elena/-3+%2C-1+%2C0+%2CB/frameset&FF=averdu+elena+%2C1+%2C](http://linc.nus.edu.sg/search_S16?averdu+%2CElena./averdu+elena/-3+%2C-1+%2C0+%2CB/frameset&FF=averdu+elena+%2C1+%2C)

Julia is looking for a book talking about suffix array etc. advanced data structure. She found one today.

2B:

[7]<http://www.stanford.edu/~liszt90/>

Interview:

[8]<https://www.youtube.com/watch?v=tZRWUesqFc&feature=youtu.be>

2C: questions and answers about practicing:

[9]<https://www.quora.com/profile/Bohdan-Pryshchenko>

2. Top 50 - blogs about programming competition:

[10]<http://sd-invol.github.io/Archives/>

[11]<http://sd-invol.github.io/>

[12]<http://sd-invol.github.io/2015/02/14/Palindromic-tree/>

Talk about algorithms - in contest - SRM641 - ?

[13]<http://sd-invol.github.io/2015/01/10/Topcoder-SRM-641-650/>

SRM

[14]<https://github.com/jianminchen/TopCoder>

3. Read the algorithm - enjoy time to read

[15]<https://threads-iiith.quora.com/>

Tutorial on Trie and example problems

4. Look into those contests the contestant chose to take part in - anything interesting?

26 gold medals - read code first

[16]<https://www.hackerrank.com/rantd>

5. Read one algorithm first: Excellent blog about algorithm - great sharing!

[17]<https://alegorithms.wordpress.com/>

[18]<https://alegorithms.wordpress.com/2016/03/11/hfcq2016-problem-analysis/>

6. Choose one of algorithms to work on this week:

[19][http://basicjahid.blogspot.ca/search/label/Algorithm %20Tutorials](http://basicjahid.blogspot.ca/search/label/Algorithm%20Tutorials)

7. MCSD programming in C #

[20]<https://www.microsoft.com/en-us/learning/exam-70-483.aspx>

1. <https://cs.illinois.edu/news/illinois-team-advances-icpc-world-finals>

2. <http://goo.gl/ZdJ8C0>

3. <https://algo.is/>

4. <https://sites.google.com/site/stevenhalim/>

5. [http://www.comp.nus.edu.sg/~stevenha/myteaching/competitive\\_programming/cp1.pdf](http://www.comp.nus.edu.sg/~stevenha/myteaching/competitive_programming/cp1.pdf)

6. <http://linc.nus.edu.sg/search~S16?/aVerdu%2C+Elena./averdu+elena/-3%2C-1%2C0%2CB/frameset&FF=averdu+elena&1%2C1%2C>

7. <http://www.stanford.edu/~liszt90/>

8. <https://www.youtube.com/watch?v=tZRWUesgqFc&feature=youtu.be>

9. <https://www.quora.com/profile/Bohdan-Pryshchenko>

10. <http://sd-invol.github.io/Archives/>

11. <http://sd-invol.github.io/>

12. <http://sd-invol.github.io/2015/02/14/Palindromic-tree/>

13. <http://sd-invol.github.io/2015/01/10/Topcoder-SRM-641-650/>

14. <https://github.com/jianminchen/TopCoder>

15. <https://threads-iiith.quora.com/>
  16. <https://www.hackerrank.com/rantd>
  17. <https://alegorithms.wordpress.com/>
  18. <https://alegorithms.wordpress.com/2016/03/11/hfcq2016-problem-analysis/>
  19. <http://basicjahid.blogspot.ca/search/label/Algorithm%20Tutorials>
  20. <https://www.microsoft.com/en-us/learning/exam-70-483.aspx>
- 

## CSS - responsive background image (2016-08-31 23:02)

August 31, 2016

Spent over 3+ hours to work on the research on CSS - responsive background image. Take time to learn CSS and enjoy the workout.

Learn to design a nice menu page with header and footer, with responsive background image, it is a fun journey. How to make it pragmatic solution? Julia has her own journey recently.

### Step 1 :

Study the idea to design CSS -

[1]<https://www.smashingmagazine.com/2013/07/simple-responsive-images-with-css-background-images/>

use a span element, and set padding top to aspect ratio of image -

**Step 2 :** box model issues - footer is cut off, and background image is cut off on mobile devices.

1. Read stackoverflow article to address the issues -

<http://stackoverflow.com/questions/8916148/ipad-iphone-full-page-background-image-shows-cut-off-screen-shot-link-includ>

<https://dev.opera.com/articles/an-introduction-to-meta-viewport-and-viewport/>

2. study the blogs:

How div container behaves - box model -

<http://stackoverflow.com/questions/9061520/auto-height-on-parent-container-with-absolute-fixed-children>

problem: here is the website:

<http://jsfiddle.net/dPCky/>

Here is the solution blog:

<http://jsfiddle.net/blowsie/dPCky/1/>

[http://www.gsmarena.com/lg\\_nexus\\_5-5705.php](http://www.gsmarena.com/lg_nexus_5-5705.php)

The lesson Julia learned is that body is the container - root of DOM tree. The height of body should be bigger

than inside elements' height. Otherwise, footer is missing since it is out of box of DOM element - body.

### Editorial Notes:

Learn how to search github more often:

1. CSS reading blog:

[2]<https://github.com/YuanXiaosong/Front-End-Interview>

1. <https://www.smashingmagazine.com/2013/07/simple-responsive-images-with-css-background-images/>

2. <https://github.com/YuanXiaosong/Front-End-Interview>

---

## 2.9 September

### A drill - Leetcode solution code study (2016-09-01 23:34)

Sept. 1, 2016

Code reading first, coding writing follows. So Julia designs a drill for her to work on Leetcode algorithm daily. Just use Visual studio to go over the best code in C++ for Leetcode solution, written by a computer science Ph.D.. Find the best code (so many solutions, this one definitely is top 1 in thousands), use the code to train herself in the thinking.

Give it a try, and see if it works out or not.

**Inside github, search HackerRank, find this:** ([1]MSFT, top performer, [2]ACM/ICPC coaches (2010 - 2012), with time spent for each algorithm)

[3]<https://github.com/jianminchen/LeetCode-17>

Julia spends hours, days to work on a solution, a best performer is there to play it with less than 20 minutes.

The practice is to find things to improve, one algorithm a time. Or find things to learn, write down one by one.

1. <http://www.cse.ust.hk/~derekhh/personal/cv.pdf>

2. [https://en.wikipedia.org/wiki/ACM\\_International\\_Collegiate\\_Programming\\_Contest](https://en.wikipedia.org/wiki/ACM_International_Collegiate_Programming_Contest)

3. <https://github.com/jianminchen/LeetCode-17>

---

### Book reading: competitive programming (2016-09-02 22:56)

Sept. 2, 2016

Plan to spend 30 minutes a time, up to 10 hours to read a short book - 152 pages first.

Competitive programming course:

Free download - version 1

[1][http://www.comp.nus.edu.sg/~stevenha/myteaching/competitive\\_programming/cp1.pdf](http://www.comp.nus.edu.sg/~stevenha/myteaching/competitive_programming/cp1.pdf)

Julia spent more than 1 hour to look into lead board of HackerRank, and then, she found the book to read:

[2]<https://algo.is/>

competitive programming course

[3]<https://sites.google.com/site/stevenhalim/>

Free download - version 1

[4][http://www.comp.nus.edu.sg/~stevenha/myteaching/competitive\\_programming/cp1.pdf](http://www.comp.nus.edu.sg/~stevenha/myteaching/competitive_programming/cp1.pdf)

[5]<http://linc.nus.edu.sg/search-S16?/aVerdu%2C+Elena./averdu+elena/-3%2C-1%2C0%2CB/frameset&FF=averdu+elena&1%2C1%2C>

Julia is looking for a book talking about suffix array etc. advanced data structure. She found one today.

Write down some great ideas after reading.

1. [http://www.comp.nus.edu.sg/~stevenha/myteaching/competitive\\_programming/cp1.pdf](http://www.comp.nus.edu.sg/~stevenha/myteaching/competitive_programming/cp1.pdf)

2. <https://algo.is/>

3. <https://sites.google.com/site/stevenhalim/>

4. [http://www.comp.nus.edu.sg/~stevenha/myteaching/competitive\\_programming/cp1.pdf](http://www.comp.nus.edu.sg/~stevenha/myteaching/competitive_programming/cp1.pdf)

5. <http://linc.nus.edu.sg/search-S16?/aVerdu%2C+Elena./averdu+elena/-3%2C-1%2C0%2CB/frameset&FF=averdu+elena&1%2C1%2C>

---

## System design - a new skill to acquire (2016-09-03 13:51)

Sept. 3, 2016

Work on the first system design blog, and learn the basics of system design:

[1]<http://juliachencoding.blogspot.ca/2016/08/system-design-design-url-shortening.html>

Continue to work on System Design day by day. (Plan to work on M. W. Fr. 8:00pm - half hour)

[2][https://github.com/jianminchen/system\\_design](https://github.com/jianminchen/system_design)

System Design:

design a Twitter website:

[3][http://www.hiredintech.com/data/uploads/hiredintech\\_system\\_design\\_the\\_twitter\\_problem\\_beta.pdf](http://www.hiredintech.com/data/uploads/hiredintech_system_design_the_twitter_problem_beta.pdf)

1. <http://juliachencoding.blogspot.ca/2016/08/system-design-design-url-shortening.html>
  2. [https://github.com/jianminchen/system\\_design](https://github.com/jianminchen/system_design)
  3. [http://www.hiredintech.com/data/uploads/hiredintech\\_system\\_design\\_the\\_twitter\\_problem\\_beta.pdf](http://www.hiredintech.com/data/uploads/hiredintech_system_design_the_twitter_problem_beta.pdf)
- 

### **A small research - project management (2016-09-03 14:23)**

Sept. 3, 2016

Julia plans to train herself to improve her skills - project management skills. As a programmer, she learns that the contributions of a single programmer can do. As an old saying, a great programmer can beat a five or six ordinary programmers, maybe a team.

To prepare herself, prepare early, prepare continuously, she likes to push herself read, memorize APIs, before she writes the code. She values the training, every lesson she learns from the training, all kinds of activities.

#### **Case study:**

One project she likes to work on is to build up a website for most of mobile users. She starts to read and write down notes about mobile phone, commercial, marketing terms about new release mobile phone.

A list of preparation for the mobile website:

1. study latest mobile phone products, learn market terms: such as retina display, human eye - 300ppi
2. a few months to learn Angular JS, MVC, entity framework etc.
3. JavaScript training
4. CSS training
5. Write bug free, solid code

Google - keyword search:

Will come back to work on this later.

---

### **Leetcode 72: Edit distance - code study (2016-09-04 12:01)**

Sept. 4, 2016

First thing in the morning, this Sunday, labor long weekend, Julia read the book about "competitive programming. She read the book -  
page 112,  
6.3 String Processing with Dynamic Programming  
6.3.1 string alignment - edit distance

Using Dynamic Programming, she was amazed that how good the solution is provided in the book. She read aloud the analysis and solution word by word, sentence by sentence, a few times. So enjoyable experience.

The book is detailed in the previous blog:

[1]<http://juliachencoding.blogspot.ca/2016/09/book-reading-competitive-programming.html>

So, she looked up google and found the similar algorithm: Leetcode 72 - edit distance

Problem statement: (Hard)

Given two words *word1* and *word2*, find the minimum number of steps required to convert *word1* to *word2*. (each operation is counted as 1 step.)

You have the following 3 operations permitted on a word:

- a) Insert a character
- b) Delete a character
- c) Replace a character

#### **Blog reading :**

1. Machine learning -

[2]<http://www.hpl.hp.com/news/2011/jul-sep/luluhe.html>

1. <http://juliachencoding.blogspot.ca/2016/09/book-reading-competitive-programming.html>

2. <http://www.hpl.hp.com/news/2011/jul-sep/luluhe.html>

---

## **Suffix Array and longest common prefix array (LCP array) - study (2016-09-05 10:56)**

June 5, 2016

It is the labor day long weekend, spent 3 hours (6:30am - 10:00am) to read suffix array from [1]this favorite competitive programming book, and try to please herself, a new goal - score any point with suffix array work or LCP (even cannot remember the full name - called longest common prefix array), HackerRank practice or code sprint.

It is a lonely journey for reading the book, but it is perfect for physical recover - muscle and bones - laying on the bed with the excellent one night sleep afterwards 3+ hour tennis sports, and do not want to move, read a book.

Yesterday, Julia warmed up more than 1+ hour, one single match, one double match lasted more than one hour, until tie break. She lost double with 5 to 7 lost the match. She suffered tennis elbow pain, slow to run issues.

The competitive book about suffix array:

6.4 Suffix Tree and Suffix Array - page 114 - 119



## Motivation talk

1. suffix array - who did the research to introduce the term - suffix array in 1993?

[2][https://en.wikipedia.org/wiki/LCP\\_array](https://en.wikipedia.org/wiki/LCP_array)

[3][https://en.wikipedia.org/wiki/Udi\\_Manber](https://en.wikipedia.org/wiki/Udi_Manber)

More reading:

Suffix array:

[4]<http://algs4.cs.princeton.edu/63suffix/>

**Play with some code first, get solid understanding suffix array - what are the benefits using suffix array?**  
**Shorten time complexity -**

suffix tree -> suffix array -> sorted array -> LCP - longest common prefix

## Arguments:

1. Building efficient Suffix Tree under contest environment is a bit complex and risky
2. Suffix Array invented by Udi Manber and Gene Myers, has similar functionalities as Suffix Tree but simpler to implement, especially in programming contest setting
- 3.

## Facts:

1. Suffix Array is an integer array that contains indices of sorted suffixes

1. Write C # version of this Java code:

[5]<http://algs4.cs.princeton.edu/63suffix/SuffixArray.java.html>

2. Suffix array - longest repeated substring - using suffix array

[6]<http://algs4.cs.princeton.edu/63suffix/LongestRepeatedSubstring.java.html>

3. Keyword in context (KWIC)

Given the suffix array, easy to search for a string or sentence via binary search. Memory is linear. Search is  $O(K \log N)$  where  $K$  is the length of the string you are searching for. (Can be done in  $K + \log N$  by using the lcp array.)

study the code:

More reading:

1. [7]<https://leetcode.com/articles/longest-common-prefix/>

2. [8]<http://www.geeksforgeeks.org/longest-common-prefix-set-1-word-by-word-matching/>

3. [9]<http://www.geeksforgeeks.org/longest-common-prefix-set-2-character-by-character-matching/>

4. [10]<http://www.geeksforgeeks.org/longest-common-prefix-set-3-divide-and-conquer/>

5. [11]<http://www.geeksforgeeks.org/longest-common-prefix-set-4-binary-search/>

- Julia likes to calm down quickly when she gets nervous. When she has a negative self-talk, she will remind herself - "**Everyone faces challenges on court and I'm no different.**" [12] Replace with positive self-talk.

Everyone faces challenges on court and I'm no different. Get out on court and overcome yours to [13]  
#CreateYourMark [14] [pic.twitter.com/RSNn7AqFch](https://pic.twitter.com/RSNn7AqFch)

— Ana Ivanovic (@Analvanovic) [15] May 21, 2016

1. <http://juliachencoding.blogspot.ca/2016/09/book-reading-competitive-programming.html>
  2. [https://en.wikipedia.org/wiki/LCP\\_array](https://en.wikipedia.org/wiki/LCP_array)
  3. [https://en.wikipedia.org/wiki/Udi\\_Manber](https://en.wikipedia.org/wiki/Udi_Manber)
  4. <http://algs4.cs.princeton.edu/63suffix/>
  5. <http://algs4.cs.princeton.edu/63suffix/SuffixArray.java.html>
  6. <http://algs4.cs.princeton.edu/63suffix/LongestRepeatedSubstring.java.html>
  7. <https://leetcode.com/articles/longest-common-prefix/>
  8. <http://www.geeksforgeeks.org/longest-common-prefix-set-1-word-by-word-matching/>
  9. <http://www.geeksforgeeks.org/longest-common-prefix-set-2-character-by-character-matching/>
  10. <http://www.geeksforgeeks.org/longest-common-prefix-set-3-divide-and-conquer/>
  11. <http://www.geeksforgeeks.org/longest-common-prefix-set-4-binary-search/>
  12. <http://juliachencoding.blogspot.ca/2016/09/8-tips-coaching-tips.html>
  13. <https://twitter.com/hashtag/CreateYourMark?src=hash>
  14. <https://t.co/RSNn7AqFch>
  15. <https://twitter.com/AnaIvanovic/status/734079097756913664>
- 

## Sports talk: onsite coaching, long hour match, interview, double partner (2016-09-05 13:39)

Sept. 5, 2016

Once a while, Julia likes to do some research on the tennis sport. She tries to educate herself, knowledge about her favorite tennis sports, and also get educated with interview talks, and smart challenges like choosing a double partner in the tennis sports.

Here are 5 things she chose to work on:

1. Angelique Kerber interview after US Open 2016 3rd round
2. coaching - onsite coaching - WTA 2008 No. 1 player
3. underdog big surprise - US open 2016 4th round - Luca Pouille
4. 5 minutes tour of central park by your favorite winner
5. how to choose competitive partner - after WTA double player top 1 made a cold call

Let us have some sports talk in this blog:

1. WTA No. 2 Angelique Kerber interview after US Open 2016 3rd round  
1 year ago, none of second week of big tournament  
last few months, going to No. 1.

**Improvement in attitude:**

Try to enjoy the game, less nervous; bring out best performance

About No. 1 ranking, how do you think about it?

Focus on next game. A long way to go.

[1]<https://www.youtube.com/watch?v=q4S7alsq0-4>

1. **How to coach best top performer in real life** ? 2008 WTA No. 1, Ana Ivanovic.

Don't know what to do?

Be clear on your head. Execute your game **plan** . ..., Let us get rhythm back. Come on.

[2][https://www.youtube.com/watch?v=qiR5M\\_7FJd8](https://www.youtube.com/watch?v=qiR5M_7FJd8)

2. Luca Pouille US Open 2016 - INTERVIEW -

[3][https://www.youtube.com/watch?v=I\\_TkcSxSyR0](https://www.youtube.com/watch?v=I_TkcSxSyR0)

**How to beat 16-14 grand slam champion in US open fourth round?**

Be aggressive all the match.

**What is the game plan?**

Just enjoy the match, as a player. It is a game, you have to enjoy it.

**How do you draw a line between enjoying something and be fierce and be competitive?**

You want to win.

I have a chance to win. Be aggressive. Otherwise, you will run, run, run to death.

Coach told him, you will make a lot of mistakes, but you will also make a lot of wins.

Take his chance to win match point in tie-break match.

3. Pouille Explores Central Park Ahead of US Open 2016

[4][https://www.youtube.com/watch?v=NOAoHhWleGk&index=6&list=PLpjoBM\\_v3S6EaPxZ2N4gZY4Zk\\_9Qo3gHH](https://www.youtube.com/watch?v=NOAoHhWleGk&index=6&list=PLpjoBM_v3S6EaPxZ2N4gZY4Zk_9Qo3gHH)

4. Reading the blog:

[5]<http://www.tennis.com/pro-game/2016/09/martina-hingis-coco-vandeweghe-doubles-2016-us-open/60686/>

[6]<https://www.quora.com/How-do-professional-tennis-players-choose-doubles-partners-to-play-with>

**Actionable Items:**

1. Study more interviews of tennis sports. And see commentator leads interview, what words, when, why he/ she do that.

21 minutes - Angelique Kerber R4 Presser - Sep 4, 2016

[7][https://www.youtube.com/watch?v=w\\_r1HzOMCnU](https://www.youtube.com/watch?v=w_r1HzOMCnU)

Question: New York vs Australia grand slam?

Keep things simple; loud everywhere.

Plan something, 2 hours extra

Question: Pressure level? After first grand slam win, before and after?

To find the middle ground. Recall the feeling of first round retiring, or get grand slam.

Question: about Ranking, approaching No. 1

**Try not to put pressure on myself** - talk about No. 1, kid's dream. Step by step, we will see.

Question:

In the past, too much pressure on myself; I lost a lot of matches; Try to focus on other things.

2. How the top performer works with the coach outside the court and on the court, live matches?

3. Study the website:

The desktop version/ mobile version, the art, layout, and also organization of structures/ different sponsors, very good design.

[8]<http://www.anaivanovic.com/profile>

1. <https://www.youtube.com/watch?v=q4S7alsq0-4>

2. [https://www.youtube.com/watch?v=qiR5M\\_7FJd8](https://www.youtube.com/watch?v=qiR5M_7FJd8)

3. [https://www.youtube.com/watch?v=I\\_TkcSxSyR0](https://www.youtube.com/watch?v=I_TkcSxSyR0)

4. [https://www.youtube.com/watch?v=NOAoHhWIEGk&index=6&list=PLpjoBM\\_v3S6EaPxZ2N4gZY4Zk\\_9Qo3gHH](https://www.youtube.com/watch?v=NOAoHhWIEGk&index=6&list=PLpjoBM_v3S6EaPxZ2N4gZY4Zk_9Qo3gHH)

5. <http://www.tennis.com/pro-game/2016/09/martina-hingis-coco-vandeweghe-doubles-2016-us-open/60686/>

6. <https://www.quora.com/How-do-professional-tennis-players-choose-doubles-partners-to-play-with>

7. [https://www.youtube.com/watch?v=w\\_r1Hz0MCnU](https://www.youtube.com/watch?v=w_r1Hz0MCnU)

8. <http://www.anaivanovic.com/profile>

---

## **Sports training - a programmer needs strong physical body to perform tasks - strong back muscle (2016-09-05 15:40)**

Sept. 5, 2016

Learn from sports training -

Do some research how professional players conduct training.

### **Personal story to warm up the topic:**

Early in 1998, in Florida state of USA, Julia suffered first back pain injury because she did not exercise regularly, she could not turn one side if she lies on the bed without using her arm to help, over 1 week; In 2001, she suffered a few back pain incidents as well.

Since 2011, Julia started to play tennis regularly, invest time to do some research on fitness, nutrition. She did not have back pain anymore, because if she sits too long for a few days, she knows that back pain will come back; she takes breaks to play tennis for a few hours, a lot of running, a lot of tennis forehand swing and backhand swing, and other conditioning exercise. Hour spent on tennis, 500+ hours (Just guess, last 5 years)

So, it is important to change the life style, play more sports, be more healthy.

Now back to the topic.

**Sports training / Coaching** - Julia always learns from professional tennis players, how they handle training, work with coaches, and handle difficult time as a professional player up - and - down in ranking etc.

She learns from sports, always prepare, get more training before she works on a new project in her career.

Through her tennis training, she learns from her most favorite tennis players - Angelique, Maria Sharapova, Ana, through training videos. She starts to examine her training, discipline herself, use a variety of tools, work on more warming up etc.



IFRAME: [1]<https://www.youtube.com/embed/lynY9CODYa8>

Sharapova training video:



IFRAME: [2][https://www.youtube.com/embed/n\\_Eo1-cnPLE](https://www.youtube.com/embed/n_Eo1-cnPLE)



IFRAME: [3]<https://www.youtube.com/embed/-406mGh5FC8>

Ana training before tournament, 15 minutes warmup using medicine balls, and all other routines, using elastic strips to stretch arm muscles etc.



IFRAME: [4]<https://www.youtube.com/embed/C8RLSvq4S1U>

More training videos: (Ana Ivanovic)

Fast activity, lower center gravity, a lot of drills - work with ladders, tones, fitness trainer, stretch etc.  
Do not over train, do not push too hard.

[5]<https://www.youtube.com/watch?v=OrXncq7dO4E>

[6]<https://www.youtube.com/watch?v=JqBbuhZEKlg>

Throw tennis balls, to sky, to forward, etc.

[7]<https://www.youtube.com/watch?v=tLyv-VXW9v4>

Fitness, coordination, strength, balance, speed, - scott byrnes - strength and conditioning coach

3 coordination - head and eye coordination

Prevent injury - structure in your training,

[8]<https://www.youtube.com/watch?v=Lm-IO7mRRC4>

How a coach helped so many WTA top players to achieve success?

[9]<http://www.wtatennis.com/news/article/5287997>

### Competitive Edge Sports Performance Tennis Training Drills -

[10]<https://www.youtube.com/watch?v=7dd07b1bnaQ>

[11]<https://www.youtube.com/watch?v=OvLicjixeoc>

Fitness Drills for Tennis Players - Tennis Now

[12]<https://www.youtube.com/watch?v=kb4IkMbNEIE>

Spider's drill is such a great idea - learn how to get down low, work on quad and string muscles.

Coach's talk - more focus, move to net, top 5 players - coaching is not an easy job.



IFRAME: [13]<https://www.youtube.com/embed/jNP4io5sAxx>

1. <https://www.youtube.com/embed/IynY9CODYa8>
2. [https://www.youtube.com/embed/n\\_Eo1-cnPLE](https://www.youtube.com/embed/n_Eo1-cnPLE)
3. <https://www.youtube.com/embed/-406mGh5FC8>
4. <https://www.youtube.com/embed/C8RLSvq4S1U>
5. <https://www.youtube.com/watch?v=OrXncq7d04E>
6. <https://www.youtube.com/watch?v=JqBbuhZEKIg>
7. <https://www.youtube.com/watch?v=tLyv-VXW9v4>
8. <https://www.youtube.com/watch?v=Lm-I07mRRC4>
9. <http://www.wtatennis.com/news/article/5287997>
10. <https://www.youtube.com/watch?v=7dd07b1bnaQ>
11. <https://www.youtube.com/watch?v=OvLicjixeoc>
12. <https://www.youtube.com/watch?v=kb4IkMbNEIE>
13. <https://www.youtube.com/embed/jNP4io5sAxx>

---

## Lecture study - Scalability Harvard Web Development (2016-09-07 20:06)

Sept. 7, 2016

Work on system design, spend 2 hours to study the lecture.

[1][https://www.youtube.com/watch?v=-W9F\\_\\_D3oY4](https://www.youtube.com/watch?v=-W9F__D3oY4)

Lecture notes:

[2]<http://cdn.cs75.net/2012/summer/lectures/9/lecture9.pdf>

Write down some keywords from the lecture, and then, google search on them.

Open courseware:

[3]<http://cs75.tv/2012/summer/>

1. [https://www.youtube.com/watch?v=-W9F\\_\\_D3oY4](https://www.youtube.com/watch?v=-W9F__D3oY4)
  2. <http://cdn.cs75.net/2012/summer/lectures/9/lecture9.pdf>
  3. <http://cs75.tv/2012/summer/>
- 

## **Longest Common Prefix - using Trie (2016-09-07 20:07)**

Sept. 7, 2016

Work on C # practice:

[1]<http://www.geeksforgeeks.org/longest-common-prefix-set-5-using-trie/>

One of longest common prefix series:

1. [2]<http://www.geeksforgeeks.org/longest-common-prefix-set-1-word-by-word-matching/>
2. [3]<http://www.geeksforgeeks.org/longest-common-prefix-set-2-character-by-character-matching/>
3. [4]<http://www.geeksforgeeks.org/longest-common-prefix-set-3-divide-and-conquer/>
4. [5]<http://www.geeksforgeeks.org/longest-common-prefix-set-4-binary-search/>
5. [6]<http://www.geeksforgeeks.org/longest-common-prefix-set-5-using-trie/>

Will work on coding very soon.

1. <http://www.geeksforgeeks.org/longest-common-prefix-set-5-using-trie/>
  2. <http://www.geeksforgeeks.org/longest-common-prefix-set-1-word-by-word-matching/>
  3. <http://www.geeksforgeeks.org/longest-common-prefix-set-2-character-by-character-matching/>
  4. <http://www.geeksforgeeks.org/longest-common-prefix-set-3-divide-and-conquer/>
  5. <http://www.geeksforgeeks.org/longest-common-prefix-set-4-binary-search/>
  6. <http://www.geeksforgeeks.org/longest-common-prefix-set-5-using-trie/>
- 

## **10 Tips to help you perform to your highest potential in coding (2016-09-07 20:17)**

Sept. 7, 2016

Borrow some tips from tennis coaching, try to apply coding practice, algorithm problem solving practice, programming contest.

**Memorize 8 tips to help you to perform to your highest potential in Tennis (? code practice, etc.):**

- 1.

Let go of what others think

2. Perform for yourself, not to impress or to "not disappoint" others
3. Accept that you will make mistakes, and let them go
4. Focus on what you can control
5. Recognize when you are using negative self-talk and replace it with positive
6. Rather than performing perfectly, perform to see improvement
7. Be objective about your performance, not subjective
8. Focus on the Journey, not the Destination

10 Tips to help you perform to your highest potential in Tennis[1] #tennistips [2] #tennis [3] #sports [4][pic.twitter.com/bVmtkwsQ2T](https://pic.twitter.com/bVmtkwsQ2T)

— Tennis Coaching™ (@tennisdothow) [5]July 21, 2016

- Julia likes to calm down quickly when she gets nervous. When she has a negative self-talk, she will remind herself - **" Everyone faces challenges on court and I'm no different. "** [6]Replace with positive self-talk.

Everyone faces challenges on court and I'm no different. Get out on court and overcome yours to [7] #CreateYourMark [8][pic.twitter.com/RSNn7AqFch](https://pic.twitter.com/RSNn7AqFch)

— Ana Ivanovic (@Analvanovic) [9]May 21, 2016

1. <https://twitter.com/hashtag/tennistips?src=hash>
2. <https://twitter.com/hashtag/tennis?src=hash>
3. <https://twitter.com/hashtag/sports?src=hash>
4. <https://t.co/bVmtkwsQ2T>
5. <https://twitter.com/tennisdothow/status/756143055116402688>
6. <http://juliachencoding.blogspot.ca/2016/09/8-tips-coaching-tips.html>
7. <https://twitter.com/hashtag/CreateYourMark?src=hash>
8. <https://t.co/RSNn7AqFch>
9. <https://twitter.com/AnaIvanovic/status/734079097756913664>

---

## Performance review - HackerRank world code sprint #4, #5, #6 (2016-09-07 22:14)

Sept. 7, 2016

Review performance of coding contest on HackerRank:



Log of performance - HackerRank world code sprint #4, #5, #6, from score 40 to 100; in other words, from nothing to the first bronze medal. Excellent experience.

Spend some time to review the performance of 3 contests Julia did from June to August 2016.

No. 1:

[1]world code sprint #4

- score 40/?, bet on algorithm [2]AorB (medium level) ,

[3] HackerRank: AorB - intense workout - 3 hours+

- score 0 of 50, 5+ hours.

No. 2

**world code sprint #5** - forgot to attend, worked on the algorithm next day. Finished first 2 questions, 40/430, spent time to read all questions - read 1 expert question, 2 advanced questions, 1 difficult, 2 medium. Just enjoy all the questions by reading the problem statement again and again, 3+ hours. See how many things she missed in every reading, every new thing she found through one more reading.

No. 3

**world code sprint #6** - score 100/380, first 4 algorithms full score, worked on [4]5th algorithm - Bnetrousel (medium level) more than 3+ hours, score 0 of 50. Bronze medal, top 25 %. Tried to get into 10 %. She did not know that time 10 % means silver medal when she worked on the algorithm.

### Analysis of performance:

Julia must have learned a lot through the practice of those 3 world code sprints; she never expects that she can be one of people having some medals with HackerRank so soon - from June to August, 2016, 3 contest practice.

1. Most valuable experience from world code sprint #4, June 26, 2016

[5]<http://juliachencoding.blogspot.ca/2016/06/hackerrank-aorb-interesting-struggle.html>

She learns that in order to perform in less than 30 minutes, she must work on the small, simple, well-defined problem from the very beginning. 30 minutes beats hard-working 8 hours. First time, she experienced two approaches with over 10 times performance difference (0.5 hour vs 8 hours). She likes to make the smart choice every time ever since.

2. Most valuable experience from world code sprint #6, August, 2016

Time complexity analysis. It does not matter if queue/ stack/ recursive function is used in the design, most important is to analyse time complexity - from brute force to optimal, try to target the optimal time complexity. Mathematics is so important to the problem solving, it does not matter if the math is so easy, how hard to figure out. Know the time complexity range, brute force vs optimal solution; aim high target! play to win!

**Most favourite tip from** [6]<http://juliachencoding.blogspot.ca/2016/09/8-tips-coaching-tips.html>:

Rather than performing perfectly, perform to see improvement (10 out of 10)

Focus on the Journey, not the Destination (9 out of 10)

Recognize when you are using negative self-talk and replace it with positive (8 out of 10)

**Actionable items:**

1. Set long time goal - attend more contests, try to score 200 points one day.
2. Finish competitive programming book - 150 pages.
3. Go over Leetcode questions one by one, by reading the solution.

1. <http://juliachencoding.blogspot.ca/2016/06/a-small-research-on-hackerrank-world.html>
  2. <http://juliachencoding.blogspot.ca/2016/06/aorb-hackerrank-workout-example.html>
  3. <http://juliachencoding.blogspot.ca/2016/06/hackerrank-aorb-interesting-struggle.html>
  4. [http://juliachencoding.blogspot.ca/2016/08/bonetrousle-hackerrank-world-code\\_75.html](http://juliachencoding.blogspot.ca/2016/08/bonetrousle-hackerrank-world-code_75.html)
  5. <http://juliachencoding.blogspot.ca/2016/06/hackerrank-aorb-interesting-struggle.html>
  6. <http://juliachencoding.blogspot.ca/2016/09/8-tips-coaching-tips.html>
-



BlogBook v0.9,  
 $\text{\LaTeX}$  2 $\epsilon$  & GNU/Linux.  
<https://www.blogbooker.com>

Edited: September 8, 2016

