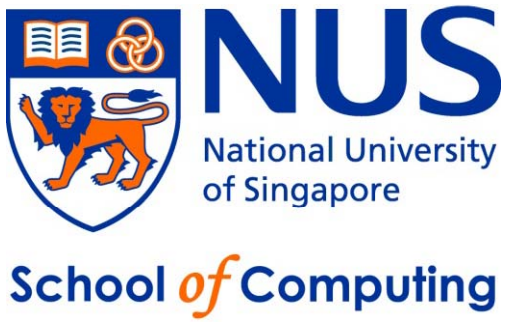
CS2020 – Data Structures and Algorithms Accelerated

Lecture 23 – Finale

stevenhalim@gmail.com



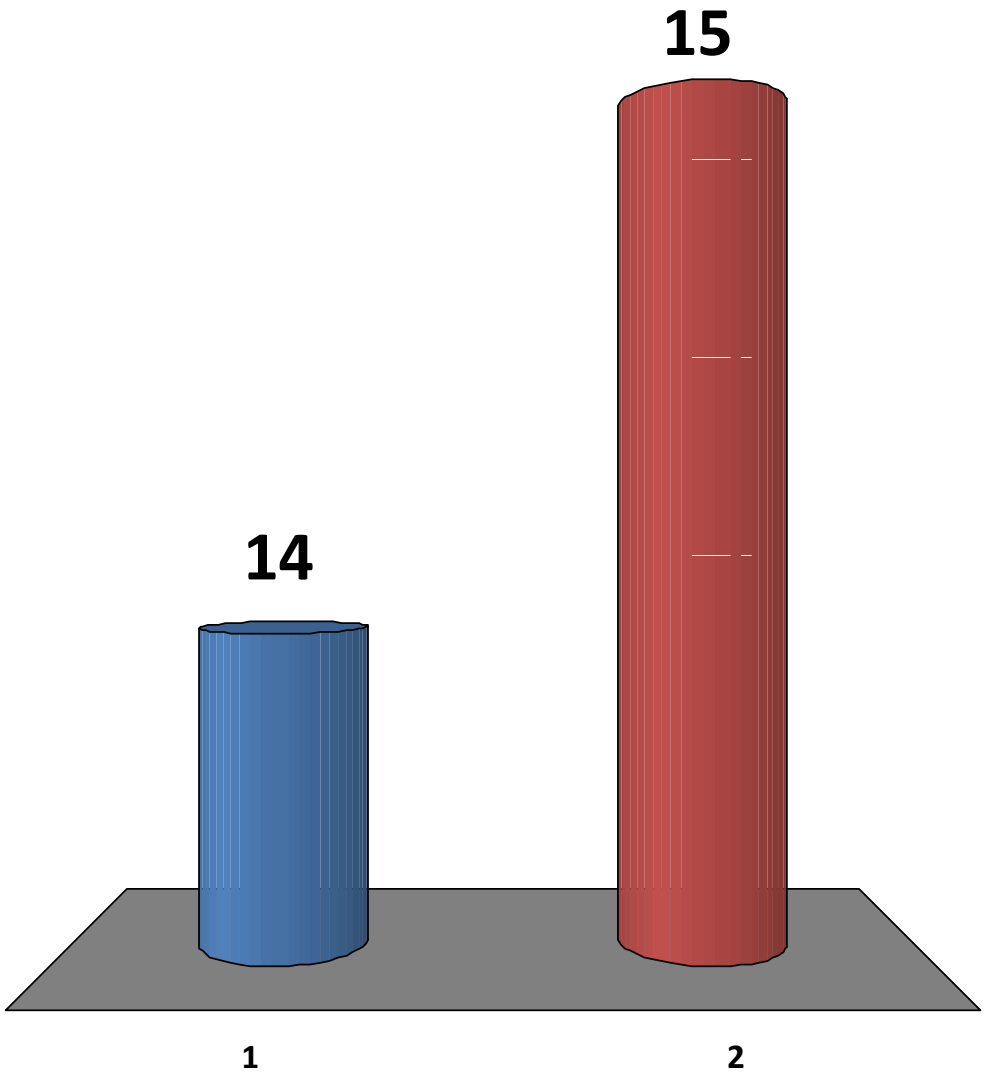
Outline

- What are we going to learn in this lecture?
  - Review of the entire CS2020 (the last time we use clickers)
    - And introducing the world beyond polynomial algorithms
  - Admin (15-20 minutes): please return your clicker!
  - Future
    - Modules in SoC beyond CS2020
    - Part time TA jobs
    - Final Exam Tips

Do you bring your clicker?

0 marks for your CS2020 final exam if you do not/cannot click yes ☹

1. Yes
2. No (do NOT click me)

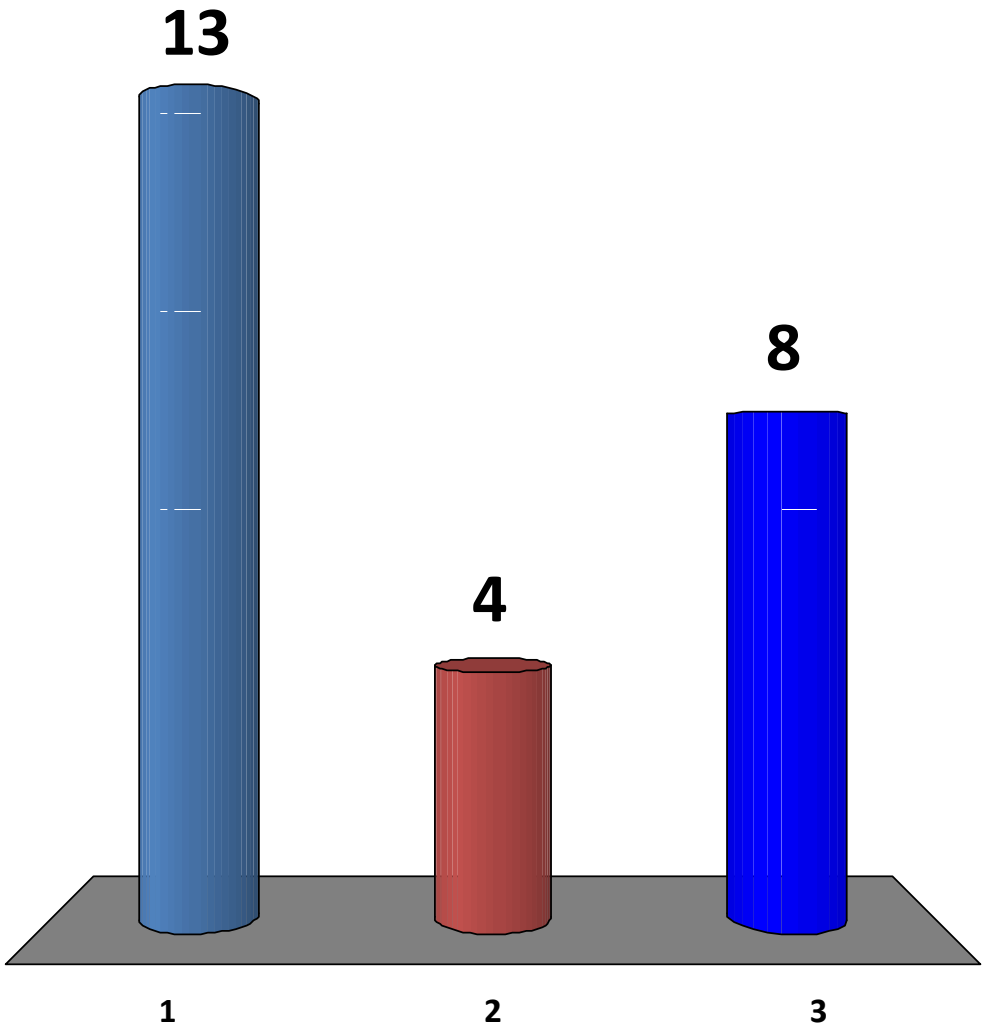


Important

- For those who click “yes” (or “no” ☺)
  - You will need to return your clicker today
  - Sign (again) the loan form that you have signed at the start of this semester
  - We will do this **later** during lecture break (10.50-11.10am)
- For those who cannot click “yes”
  - You will not get 0 marks for final if you:
    - Meet Steven or Seth during reading week and return your clicker (if you just forgot to bring it today)
    - Pay 89 SGD replacement fee (we will talk with SoC UG office for some compensation) if you really lost / broke it …

PS10

1. I have submitted my codes few days ago ☺
2. I have JUST submitted my code 1 hour++ ago, I haven't sleep >.<
3. I need more time, is it possible to extend the deadline again?



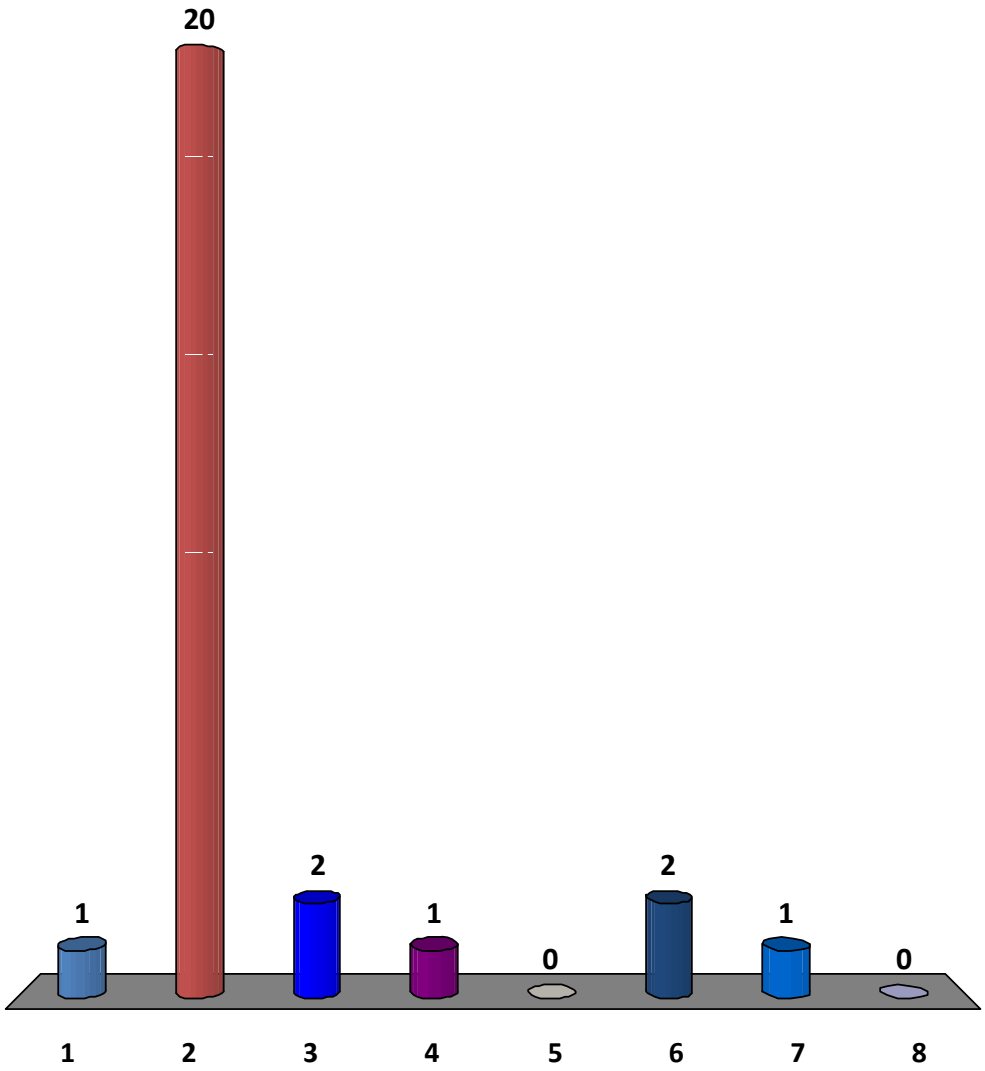
Semester Review: Topic 1

- Linear Data Structures
  - Arrays/Vectors
  - Searching
    - O(n) linear search, O(log n) binary search
  - Sorting
    - $O(n^2)$ sort, O(n log n) sort,
      Ω(n log n) lower bound of sorting, O(n) special case sort
  - Lists/Linked Lists/Skip List
  - Stacks, Queues
- Divide and Conquer Paradigm
- Notable Example: Document Distance Problem
- In red: not covered in CS1020

What is the best sorting algorithm to sort this particular sequence?

$$X = \{ 1, 1, 1, 2, 1, 3, 3, 4, 5, 6, 7, 10, 1M\}$$

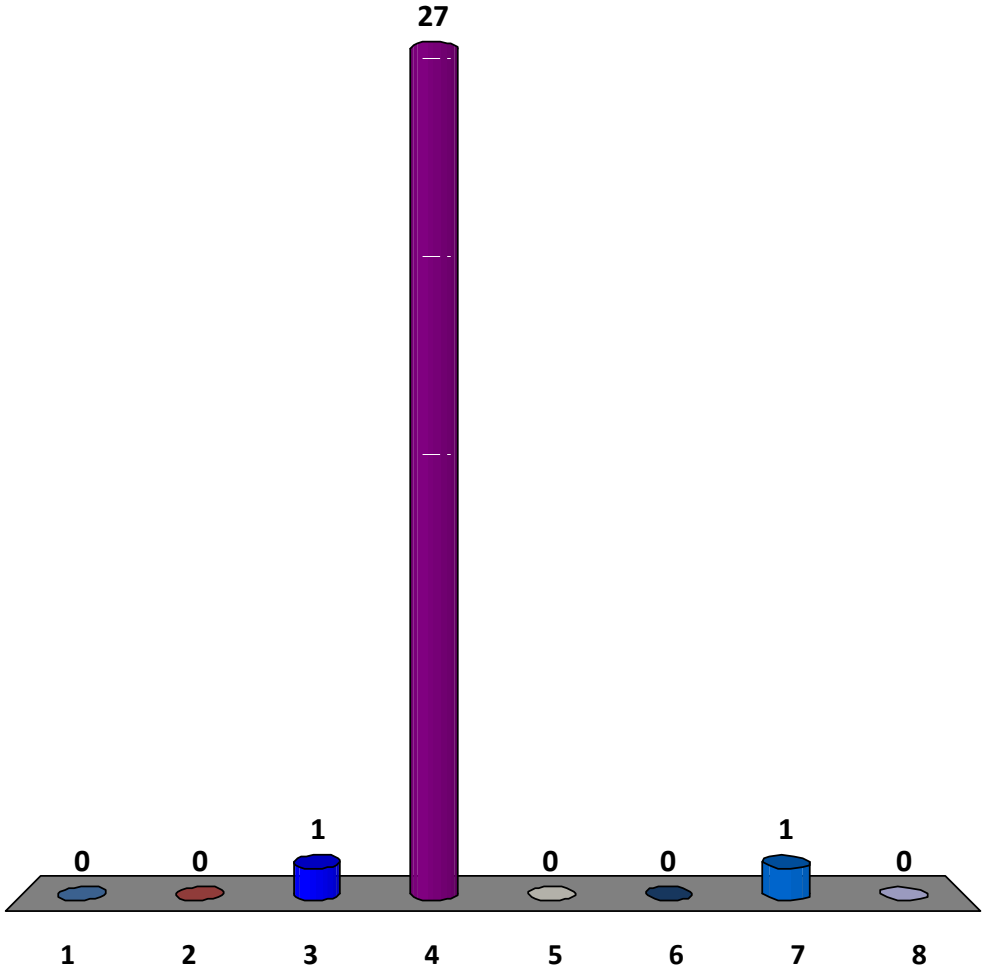
1. Bubble Sort
2. Insertion Sort
3. Quick Sort
4. Merge Sort
5. Heap Sort
6. Counting Sort
7. Radix Sort
8. Others: _____



$g(n) = 5000 \log n * \log n + 5n - n + \text{sqrt}(n^2)$

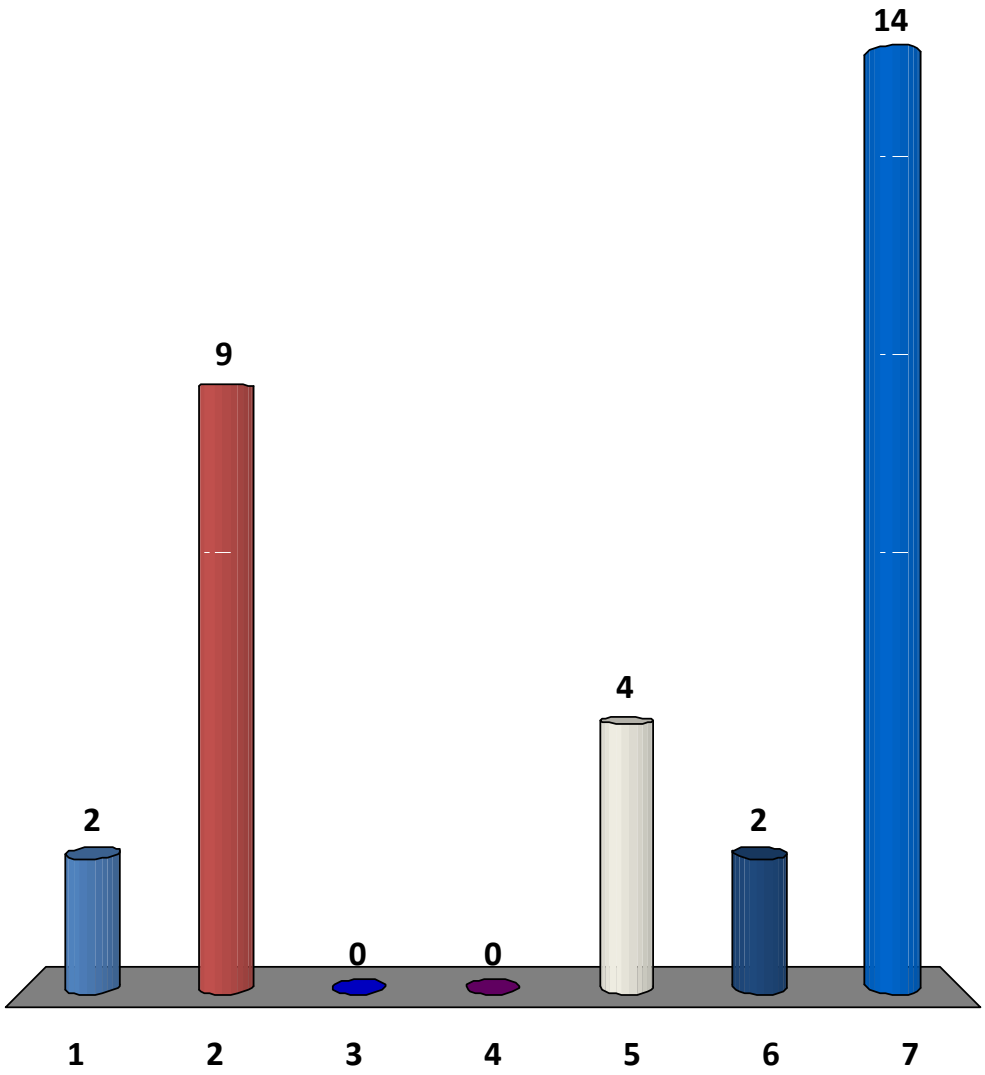
$g(n) =$

1. $O(\text{sqrt}(n^2))$
2. $O(n^2)$
3. $O(n \log n)$
4. $O(n)$
5. $O(n \log \log n)$
6. $O(\log n)$
7. $O(\log \log n)$
8. $O(1)$



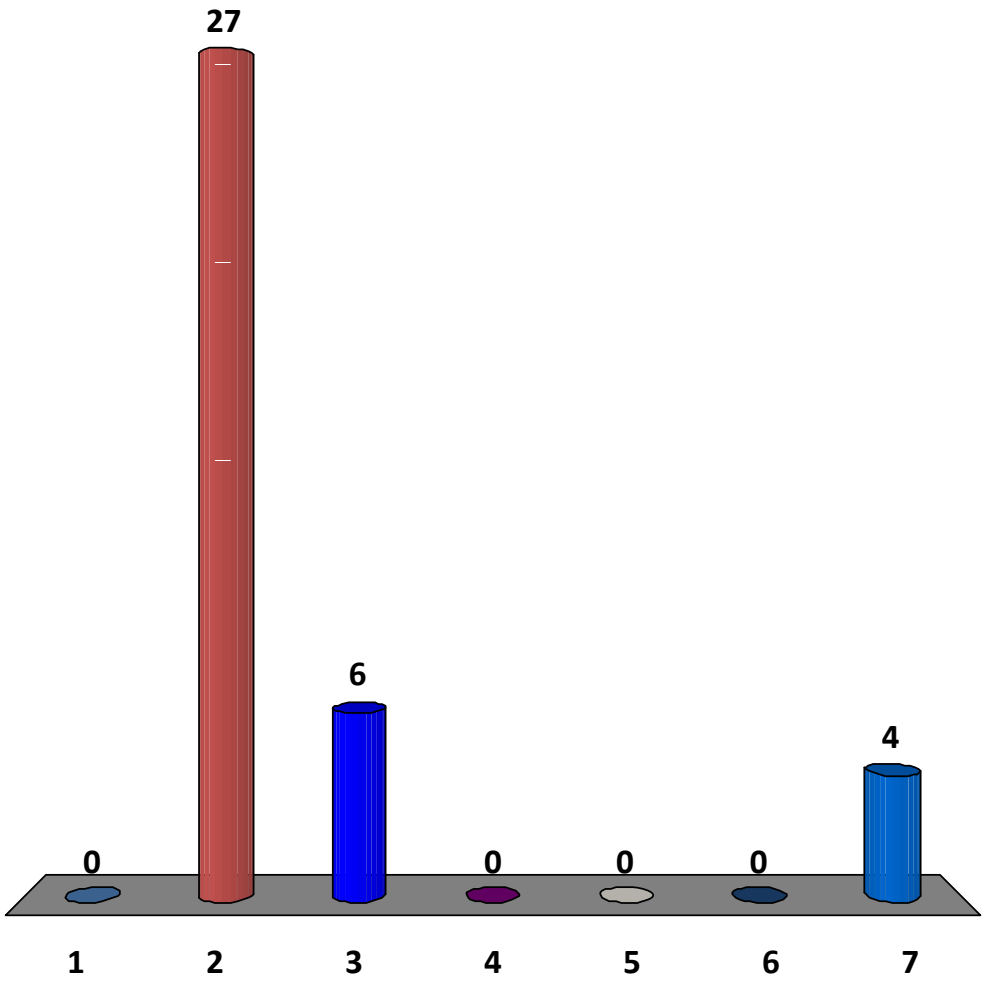
What is the worst possible input for the merge sort algorithm?

1. A nearly sorted array
2. Reverse sorted array
3. A very unsorted array
4. Array where all elements are the same
5. 1 & 2 only
6. 1 & 4 only
7. All of the above



What is the time complexity of Strassen's Algorithm for Matrix Multiplication?

1. $O(n^3)$
2. $O(n^{2.81})$
3. $O(n^{2.376})$
4. $O(n^2)$
5. $O(n \log n)$
6. $O(n)$
7. Eh, did we ever discuss this algorithm?

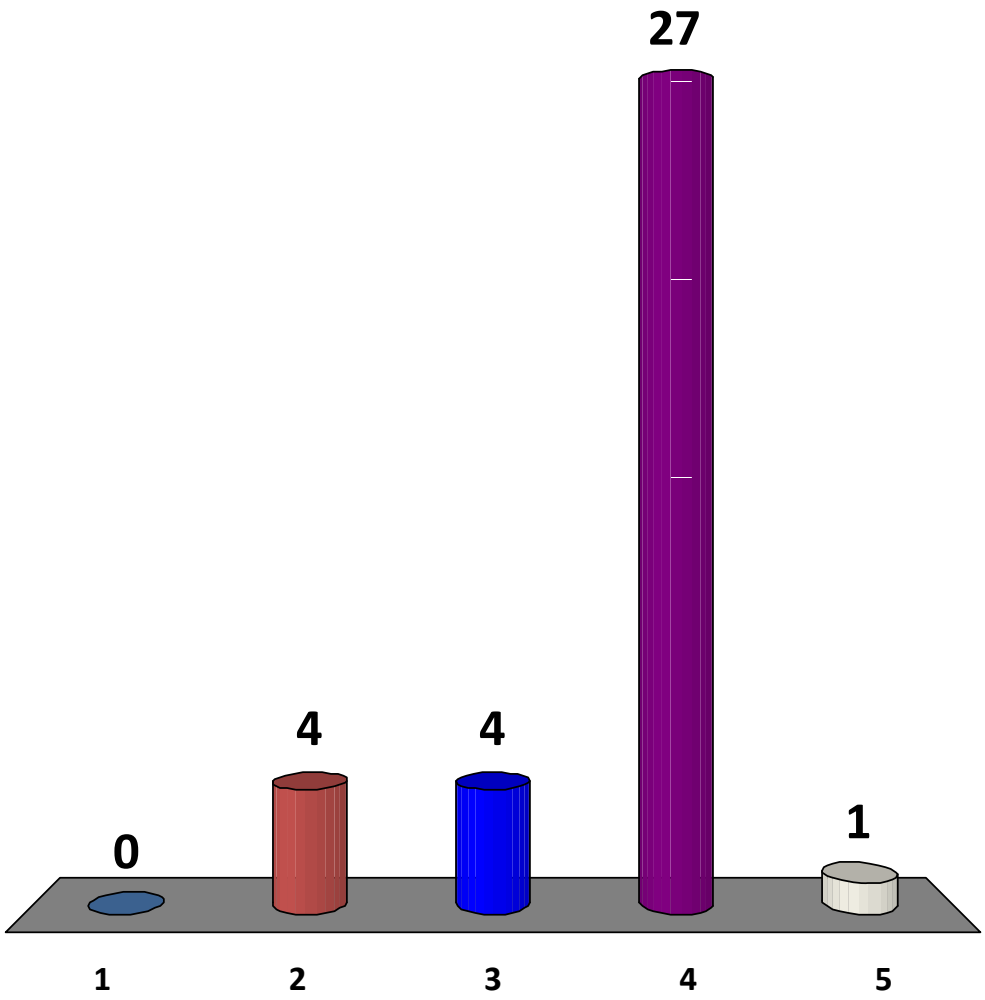


Semester Review: Topic 2

- Trees
  - Binary Search Trees
  - Balanced BSTs
  - Heaps and Priority Queues
- Notable Example: Air Traffic Controller Problem
- Green: I will likely cover them in CS2010

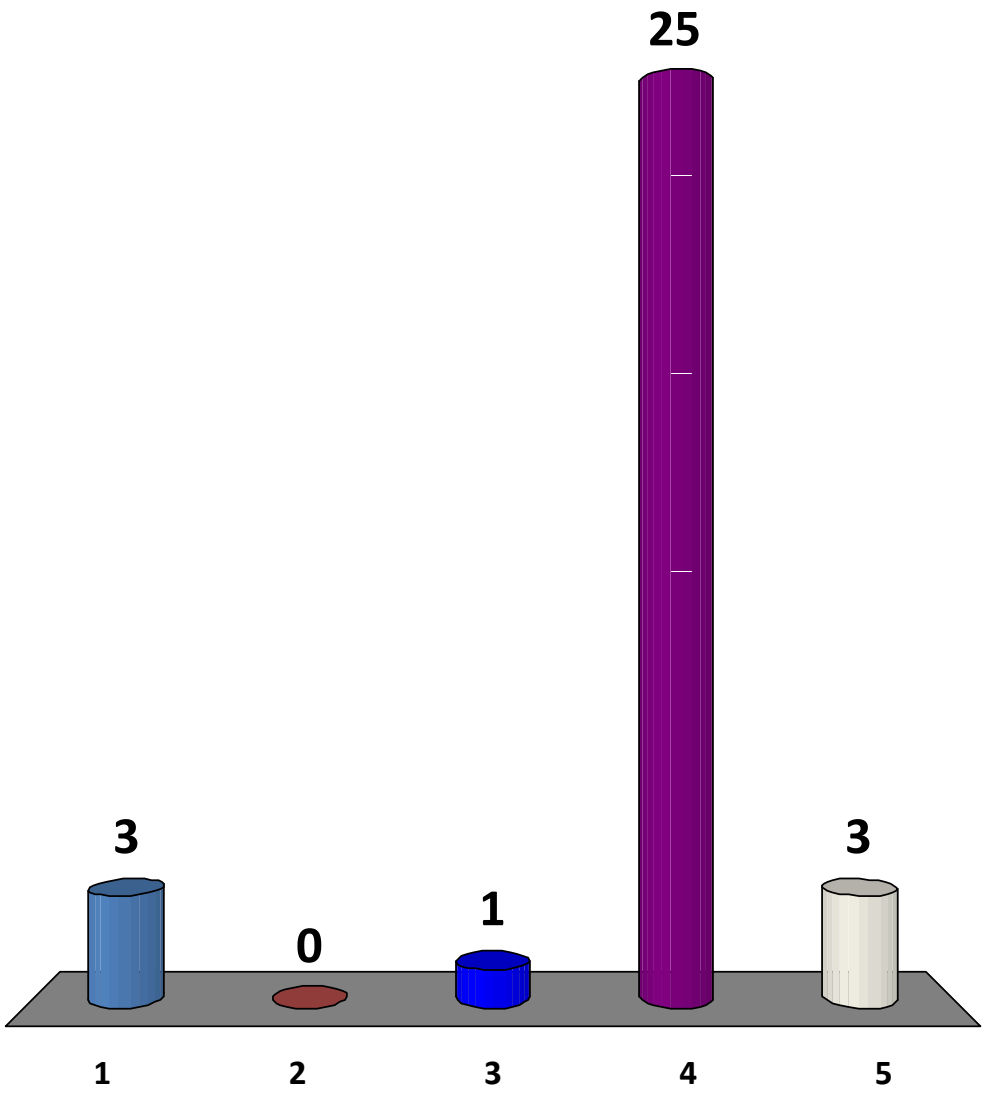
What is the right child of the root of BST if the following sequence of items are inserted to an initially empty BST: {8, 7, 2, 10, 4}

1. 2
2. 7
3. 8
4. 10
5. 4



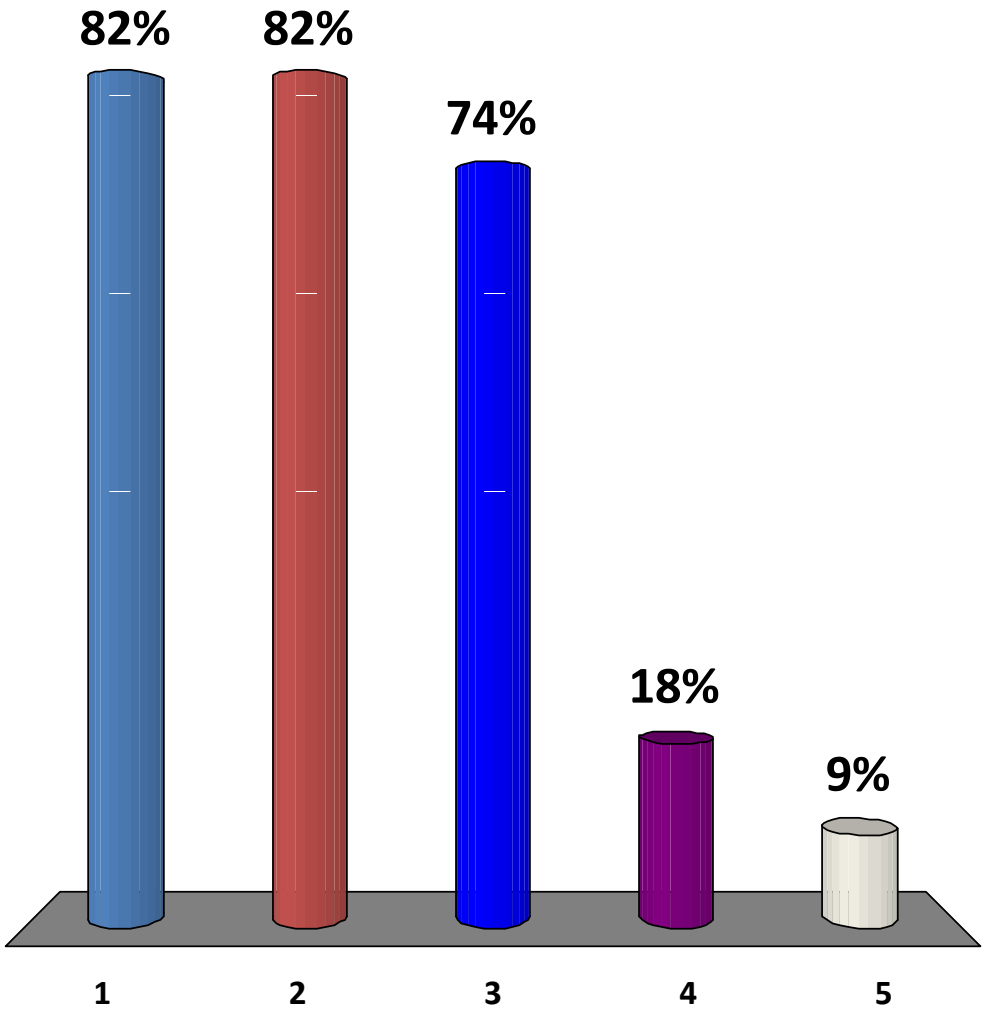
Which Statement About Heap is False?

1. Heap must be a complete binary tree
2. Heap can be implemented with array
3. Heap can be used for sorting
4. Building a heap from an unsorted array can only be done in O(n log n)
5. Heap can be used in Prim's algorithm



Which one is a **valid** partition of an array X = {8, 2, 7, 9, 10, 1, 5} around value 7? (you can select up to 5 choices)

1. X’ = {1, 2, 5, 7, 9, 10, 8}
2. X’ = {1, 2, 5, 7, 8, 9, 10}
3. X’ = {5, 2, 1, 7, 8, 9, 10}
4. X’ = {5, 2, 1, 8, 7, 9, 10}
5. X’ = {2, 1, 5, 8, 7, 9, 10}

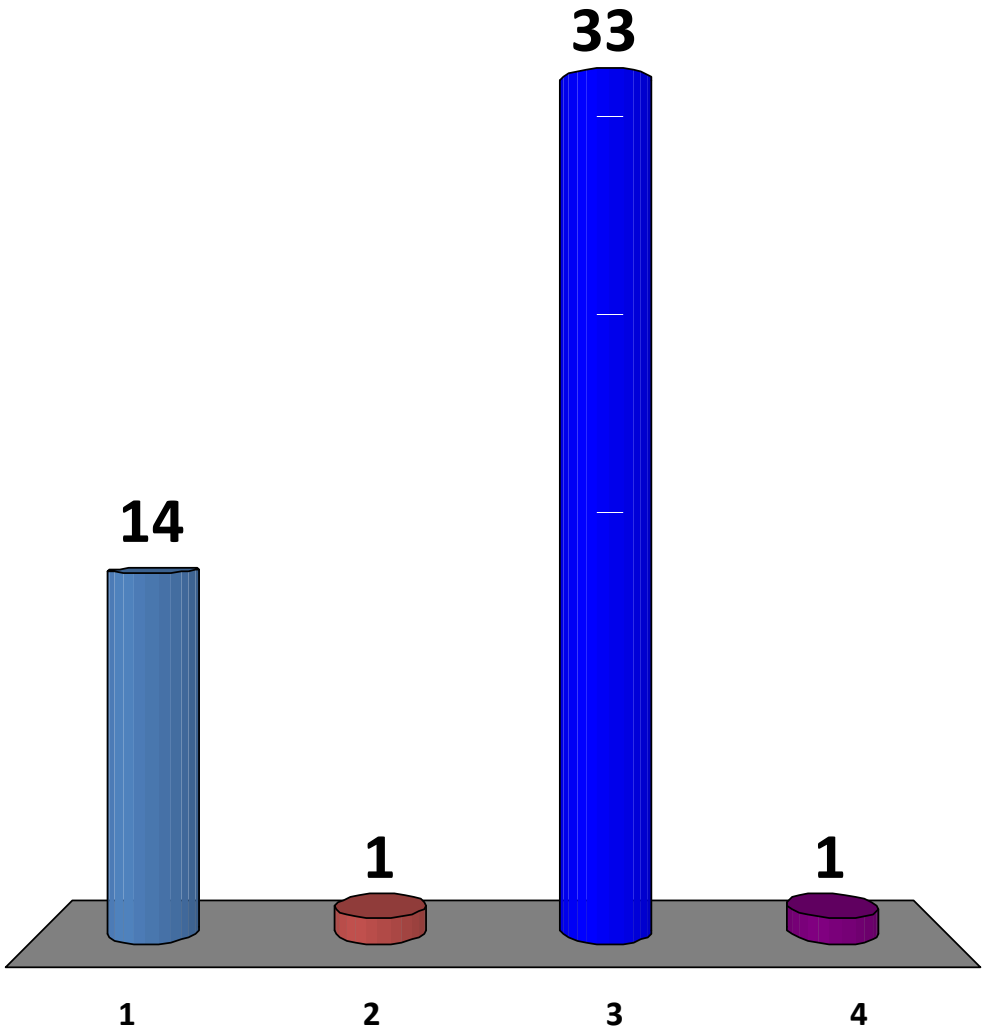


Semester Review: Topic 3

- Hash Tables
  - Dictionaries
  - Hash functions
  - Chaining
  - Amortized Analysis
- Notable Example: DNA Analysis

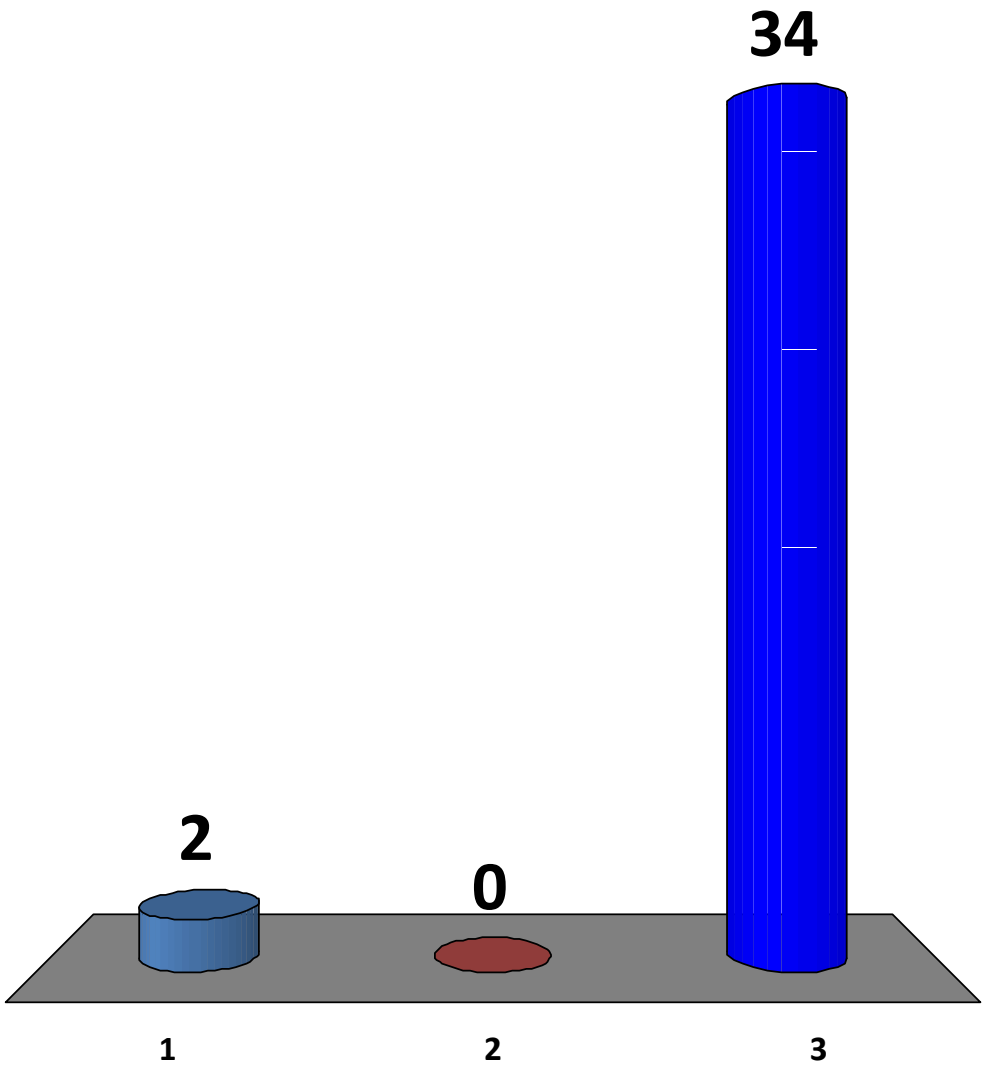
Give me a pair of numbers x1 and x2 so that h(x1) = h(x2) for h(x) = (x * x) % 7 (you can select up to 4 options)

1. x1 = 71, x2 = 55
2. x1 = 77, x2 = 66
3. x1 = 7, x2 = 14
4. x2 = 9, x3 = 147



In which scenario that hash table is a worse choice than BST?

1. When n is big, there will be lots of collision, so hash table can be slower than BST
2. When n is very small, it is faster to use BST
3. When you want to list down the items in sorted order



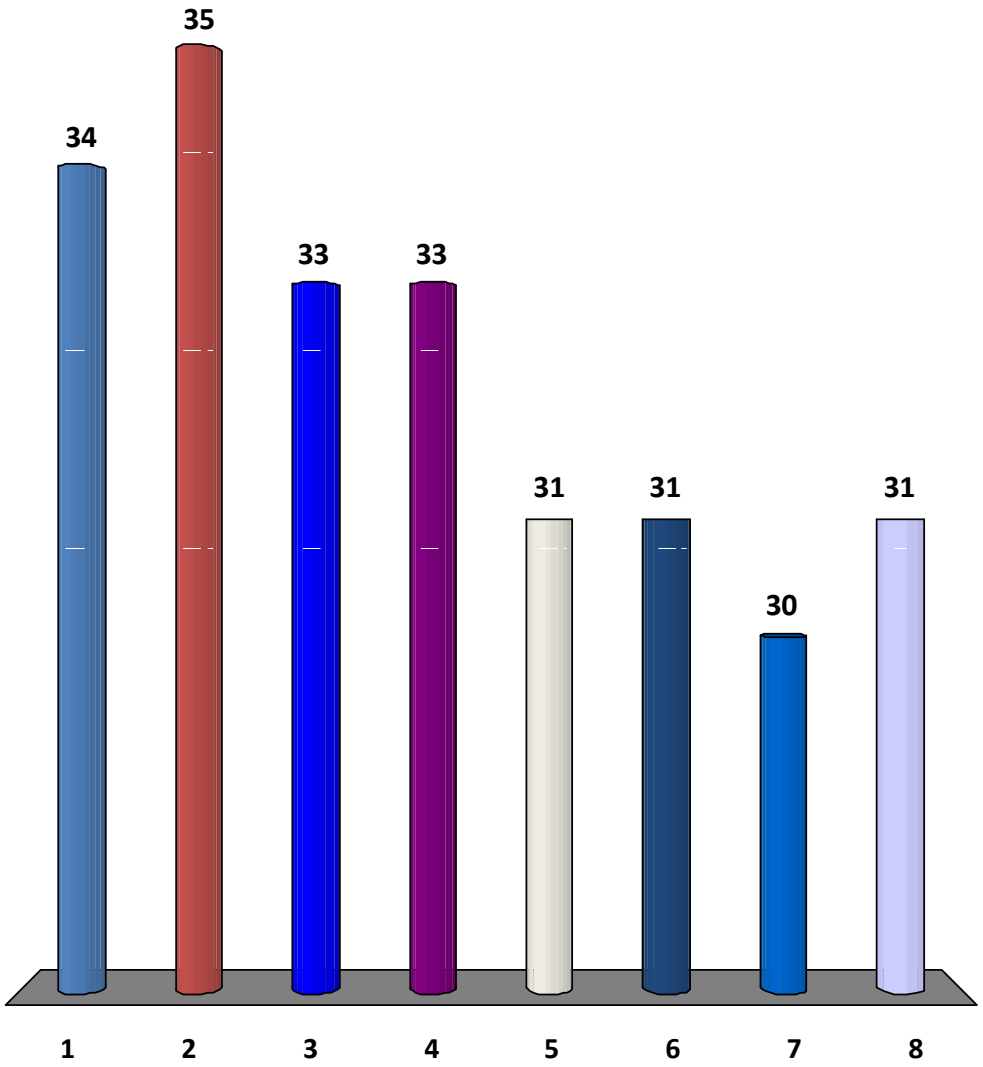
2ND HALF

Semester Review: Topic 4

- Graphs
  - Graph Data Structures: AdjMat/List/EdgeList/Implicit Graph, etc
  - Graph Traversal: DFS/BFS
  - Minimum Spanning Trees: Kruskal's/Prim's
  - Single Source Shortest Paths: Bellman Ford's/Dijkstra's
- Notable Examples: McDonalds, Parity Networks

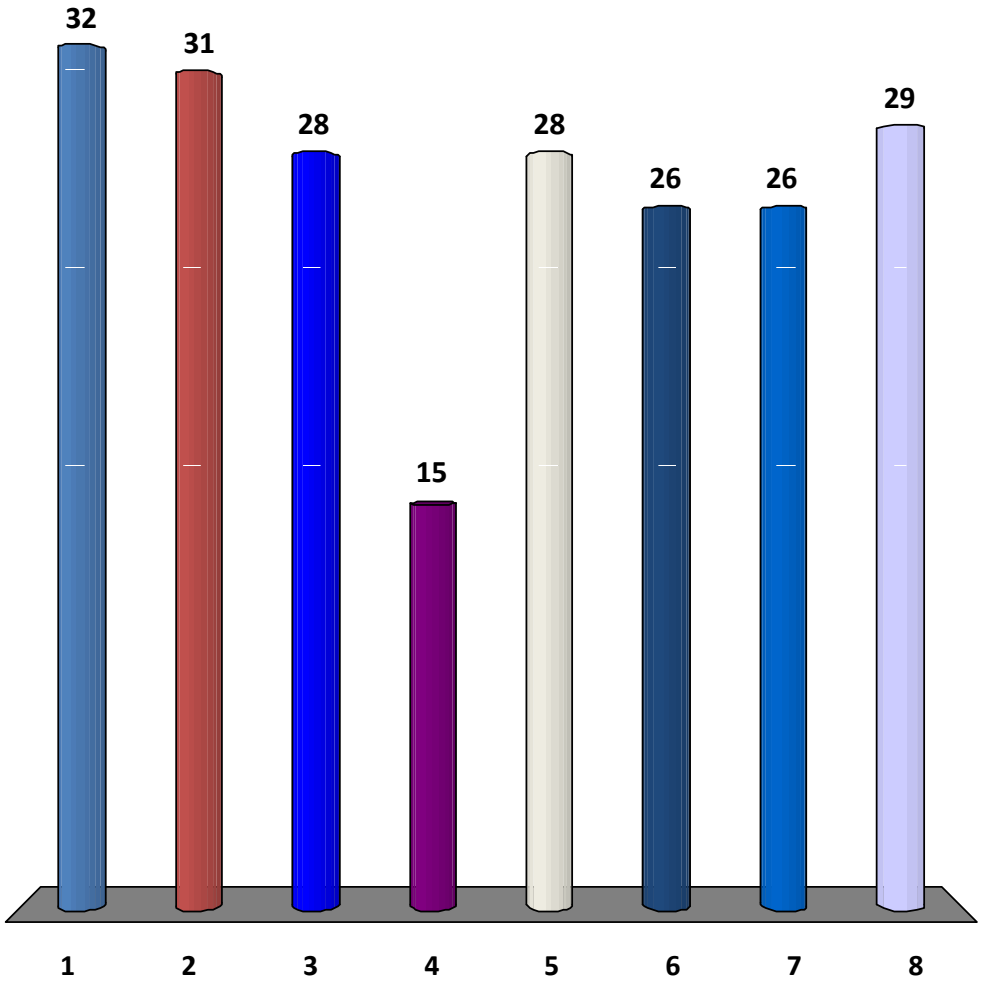
Select graph terminologies that you know **now**… (can select up to 8/clicker)

1. AdjMatrix/List/EdgeList
2. DFS/BFS
3. Topological Sort
4. MST/Prim's
5. MST/Kruskal's
6. SSSP/Bellman Ford's
7. SSSP/Dijkstra's
8. APSP/Floyd Warshall's



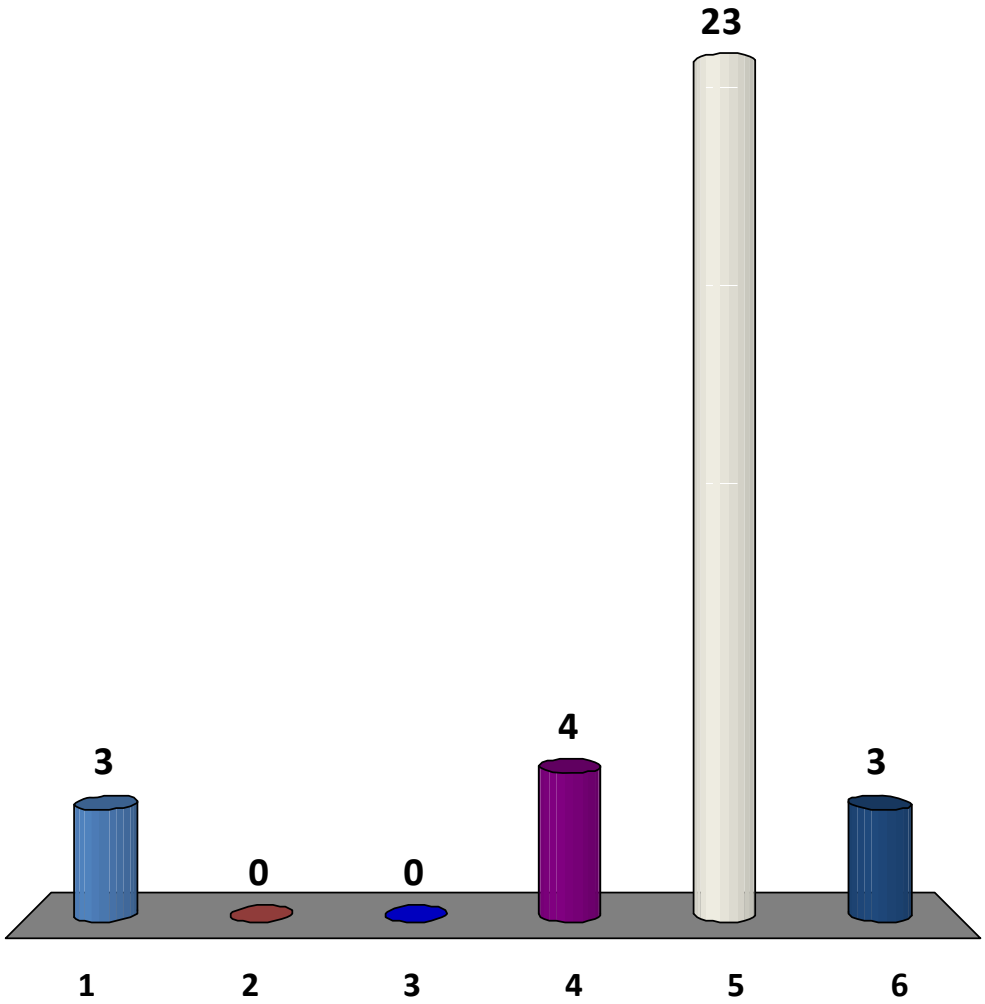
Select DS/algorithms that you have already **implement now**… (can select up to 8/clicker)

1. AdjMatrix/List/EdgeList
2. DFS/BFS
3. Topological Sort
4. MST/Prim's
5. MST/Kruskal's
6. SSSP/Bellman Ford's
7. SSSP/Dijkstra's
8. APSP/Floyd Warshall's



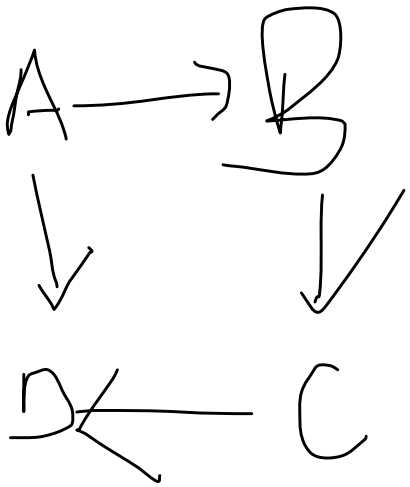
DFS/BFS can run in $O(V^2)$ on

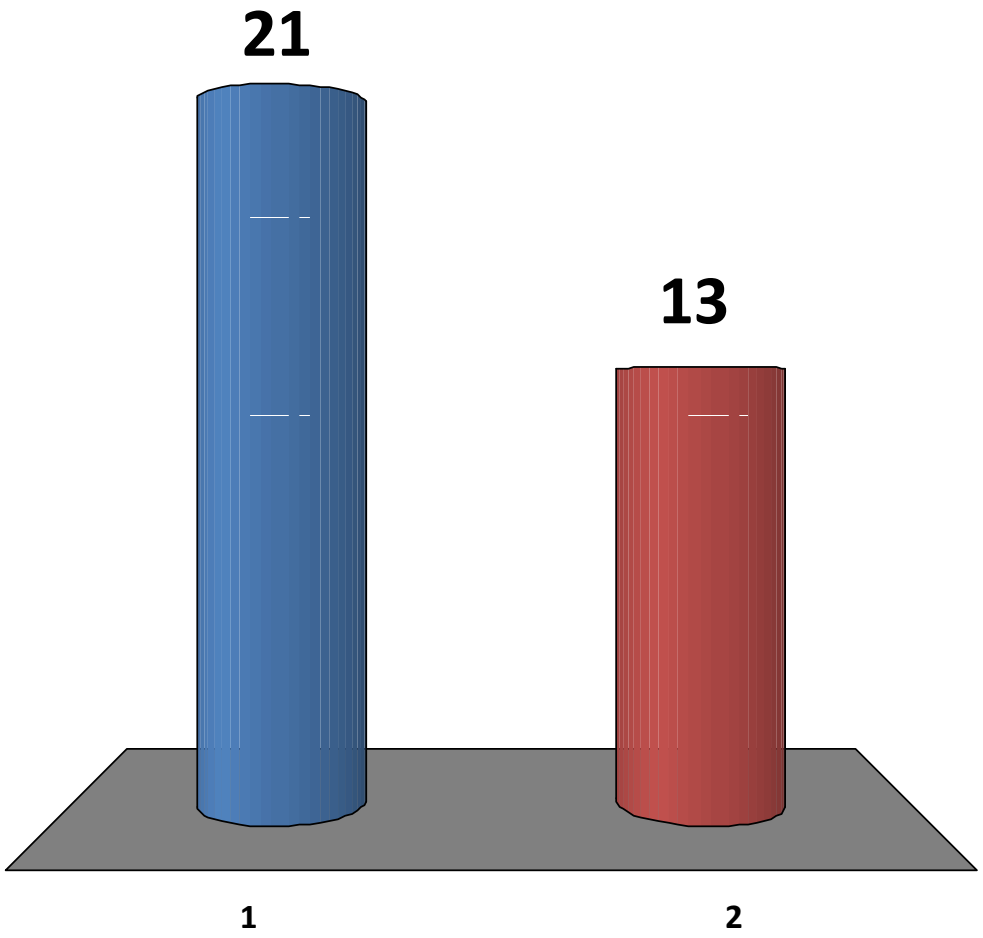
1. Tree
2. Directed Acyclic Graph
3. Bipartite Graph
4. General Graph
5. Complete Graph
6. Impossible, it is O(V+E)



Hm… Can I use BFS to find toposort?

1. Yes, why not?
2. No, I think BFS will have a problem because _____





Semester Review: Topic 5

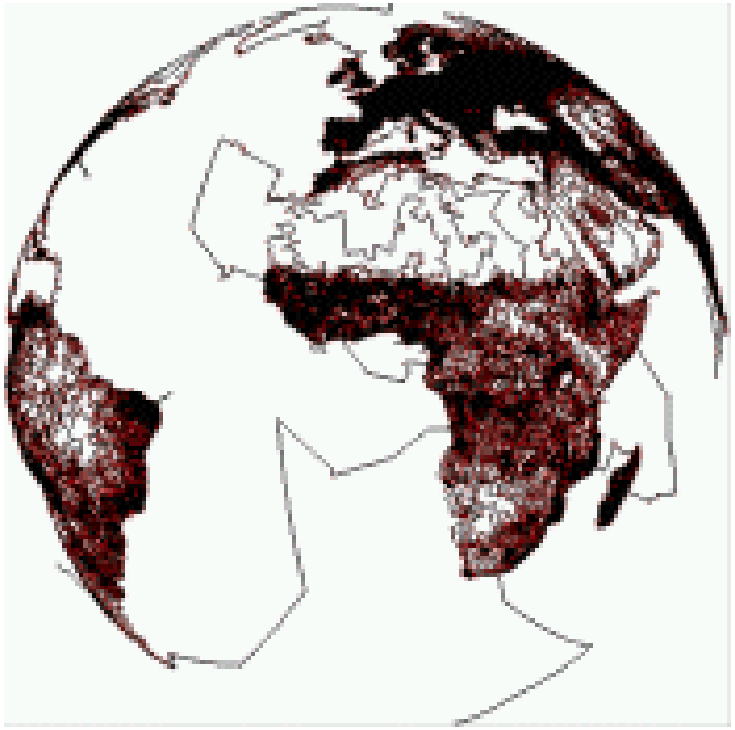
- Dynamic Programming (mainly on Graph)
  - Algorithms on DAG
  - Algorithms on Tree
  - Algorithms on General Graph
  - All Pairs Shortest Paths: Floyd Warshall's
  - DP, the true form
- Notable Example: Traveling Salesman Problem (TSP)

Larger TSP Instances

N	N!	$N^2 * 2^N$	Improvement
9	362,880	81 * 512 = 41,472	8.75 x
10	3,628,800	100 * 1024 = 102,400	35.44 x
11	39,916,800	121 * 2048 = 247,808	161.07 x
12	479,001,600	144 * 4096 = 589,824	812.11 x
13	6,227,020,800	169 * 8192 = 1,384,448	4,497.84 x
14	87,178,291,200	196 * 16384 = 3,211,264	27,147.66 x
15	1,307,674,368,000	225 * 32,768 = 7,372,800	177,364.69 x
16	20,922,789,888,000	256 * 65,536 = 16,777,216	1,247,095.46 x
17	355,687,428,096,000	289 * 131,072 = 37,879,808	9,389,895.22 x
51	Hm…	5,856,931,315,395,330,048 ☹	…
105?		4.472271317605193328480 36e+35 ☹	
		TSP is NP-Complete	

How to solve this?

- http://www.tsp.gatech.edu/world/images/anim1a.html



Research on Optimization Problems

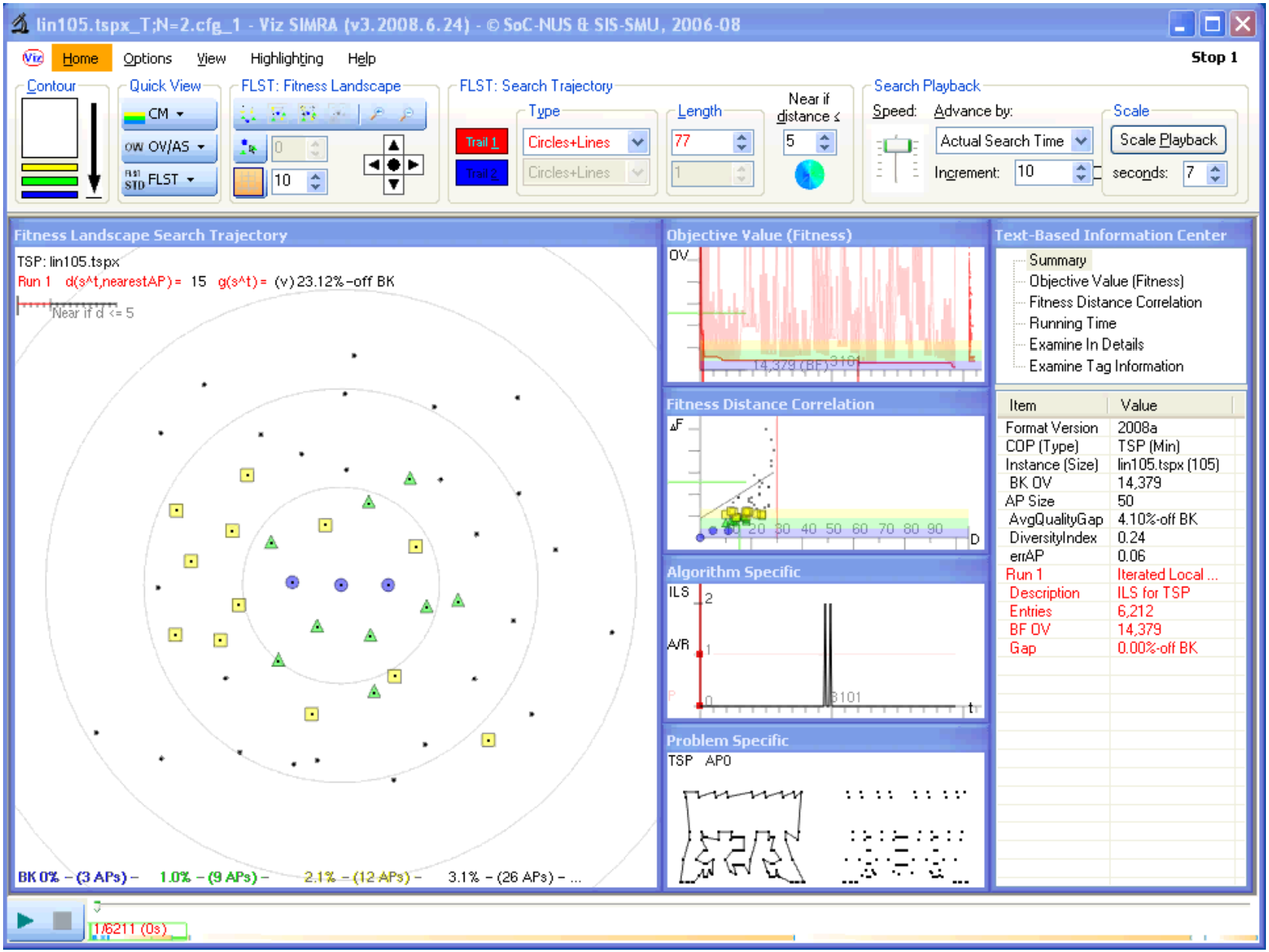
- What if we can accept **near optimal** solution?
- Other options are now available:
  - Approximate/Heuristics Solution
  - Local Search

Euclidean TSP Approximation with MST

- http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/AproxAlgor/TSP/tsp.htm
- We can get MST of a complete graph with N vertices (N^2 edges) in $O(N^2 \log N)$
- We can then get a TSP tour that is at worst twice the length of the optimal tour…

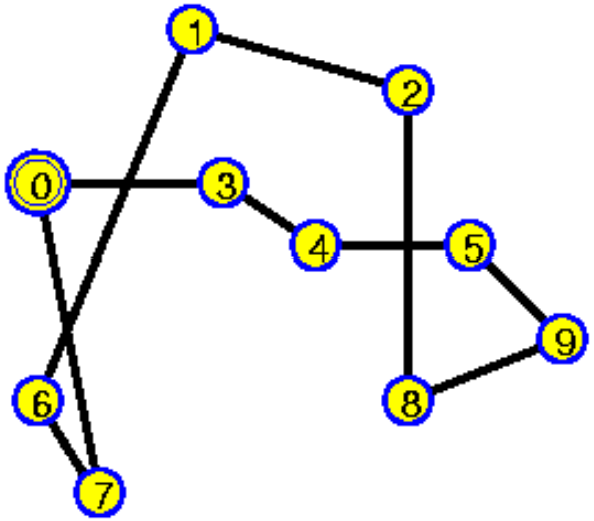
Local Search Solution

From Steven's PhD thesis (2004-2010)



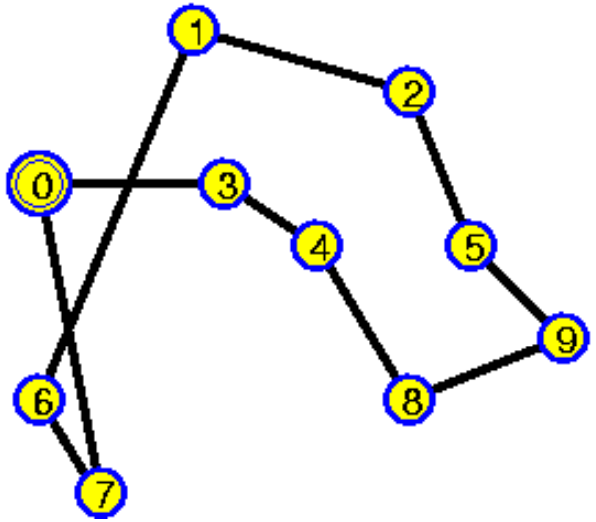
Local Search Solution

- Start from any permutation that cycles back (a tour)
- Use (2-Opt) Swap-Edge Heuristics
- Locally optimize it
- Iteratively repeat this process with other random tours



Solution: (0, 3, 4, 5, 9, 8, 2, 1, 6, 7, 0)

AFTER TWO–SWAP((4,5),(8,2))



Some Demo ☺

- http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/STSP.html

SUMMARY OF 2ND HALF

Graph DS

- L13: AdjMatrix, AdjList
- RecitationWk8: Implicit Graph
- L15: EdgeList
- L20: special graph that has 2 edges/vertex
- Space Complexity
- Benefits/Disadvantages of using them

MST

- L15: Kruskal's Algorithm
- L15: Union Find Data Structure
- L15: Prim's Algorithm (brief)
- Uses L6: Heap Data Structure
- Model problem as an MST problem

SSSP, Special Cases

- L16: General Shortest Paths Algorithm: O($\propto$)
- L16: Bellman Ford's: O(VE) – general
- RecWk9: SSSP (Graph) Problem Modeling
- L17: Dijkstra's: O((V + E) log V) – binary heap
- L17: BFS: O(V + E), unweighted graph
- L17: DFS/BFS: O(V), Tree
- L17: Toposort + DP: O(V + E), DAG
- RecWk10/Quiz2: Graph Problem Modeling

DP (have we test you on this?)

- L18: Algorithms on DAG
- L20: Algorithms on General Graph → implicit DAG
- RecWk11: Algorithms on Tree → implicit DAG
- L19: APSP: Floyd Warshall's DP algorithm
- L21: DP, true form
- Space Complexity: Vertices in implicit DAG/states
- Time Complexity: Edges in implicit DAG/transitions

Traveling Salesman Problem

- L20: Recursive Backtracking Algorithm: O(N!)
- L20: Dynamic Programming Algorithm: $O(N^2 \times 2^N)$
- PS10: TOUR
- L23: Approximation Algorithm/MST: $O(N^2 \log N)$
  - Only for Euclidean MST
  - The results are not guaranteed to be optimal
- L23: Local Search / Heuristics: O(Iterations * C)
  - The results are not guaranteed to be optimal

Things That Can Be Done Better

- ii.java → IntegerPair.java
- iii.java → IntegerTriple.java
- L6: Heap_DecreaseKey() → heapDecreaseKey()
- In several demos: Reduce the usage of global variables
- More Object Oriented examples
- Less typos in our lecture notes/Pses
- More balanced DGs/PSes
- Etc (we will look at your NUS Online Teaching Feedback regarding this module ☺)

Admin

- Now, let's return your clicker
- The procedure:
  - I have sorted the loan forms based on clicker ID
  - I will call your name one by one
  - Come forward, return your clicker, and sign again with red pen in your own loan form to indicate that you have returned the clicker
  - Then you can have 10-15 minutes break
    (until this process is over, we need about 54 iterations ☺)

SOME FINAL SLIDES

Have you count your own marks so far? (15% PS, 5% DG, 15% Quiz1, 10% Coding Quiz (please just use your own estimation), 15% Quiz2)

1. I have scored [55-60]%
2. [50-55]%
3. [45-50]%
4. [40-45]%
5. I haven't count
6. I have count but don't want to tell

Relevant Modules After CS2020 (1)

- CS3233: Competitive Programming
  - Pre-req: **at least** A- from CS2020, **(very)** strong interest in algorithms, programming (especially implementation), and problem solving
  - Must free-up your schedule every Wednesday 6-9pm on the 2nd semester…
  - CS3233 syllabus will be altered **a bit** because of CS2020/10
- CS3230: Design and Analysis of Algorithms
  - Discussion: take soon (next semester) or next next AY (after "distance propagation" reaches the **new** CS3230)
- CS5206: Foundations in Algorithms

CS3233: Competitive Programming

1. I will **definitely** take your challenging module next year (S2, AY2011-2012)
2. I will consider ☺, maybe I will take it…
3. I will consider, but at the moment I do not have plan to take it…
4. I know for sure that this module is not for me, thanks

CS3230: Design and Analysis of Algorithms

1. I will definitely take this module next sem (S1 AY2011-2012) to increase my chance for A before the syllabus gets harder
2. I will strategically take other modules first and will take the **updated** and proper CS3230 only next next AY
3. I do not know what to do at the moment…

Relevant Modules After CS2020 (2)

- CS3210: Parallel Computing
- CS3211: Parallel and Concurrent Programming
- CS4231: Parallel and Distributed Algorithms
  - If you like Dr Seth's mystery lecture last Tuesday
- CS2103: Software Engineering
- CS3215: Software Engineering Project
  - Larger scale software creation modules
- CS3243: Artificial Intelligence
- Enjoy ☺

Final Exam (1)

- Not a fully open book test
- Your only “data structure”: 2 sheets/4 sides A4 paper
- Either handwritten or printed
  - Use whatever font size convenient for you
  - NO magnifying glass is allowed in the exam hall ☺
- Purpose:
  - To improve your search performance during exam time from O(N) *where N is the number of books/lecture notes that you are planning to bring into exam hall* into O(1) where you can use a kind of “hashing” to identify the required information in your 4 sides A4 paper ☺

Final Exam (2)

- There will be 4 questions
  - Not of the same length
  - Not of the same marks
  - More similar to Quiz 2 style rather than Quiz 1
    - No MCQ
  - Topics: from the entire semester with focus on things that are not yet tested in Quiz 1, Coding Test, & Quiz 2 :O
    - Your job is to see those quizzes and test and see which part in the syllabus that we have not test yet ☺

Final Exam (3)

- Time Management
  - You have 20 more minutes than quiz 1 or 2
  - We assume that you have 4 sides A4 paper with O(1) performance ☺
  - We estimate that you will likely spend more time thinking than writing answers
- Study Preparation
  - In case you are not aware, few years ago I have compiled solutions for a lot of past exam papers of the (old) CS1102
    - http://www.comp.nus.edu.sg/~stevenha/myteaching/exam_hints.pdf

Consultation Slots @ Study Week

- Dr Seth Gilbert, COM2-3-23
  - Preferred time slot: Wednesday 20 Afternoon
  - Not available on study week: Monday 18/Tuesday 19
  - Email first to book a slot: seth.gilbert@comp.nus.edu.sg
- Dr Steven Halim, COM2-3-37
  - Really just opposite of Seth's office
  - Preferred time slot: Tuesday 19/Thursday 21 Afternoon
  - Not available on study week: Monday 18
  - Email first to book a slot: stevenhalim@gmail.com
- Note: next Friday is Good Friday

Part-time Teaching Assistant Job (1)

- FYI, I will teach CS2010 next semester
  - S1, AY2011/2012 (Aug-Nov 2011)
  - Estimated enrollment (if everyone pass CS1020): ~228 :O
- I estimate that if I want 1:20 staff:students ratio
  - Then I need to hire **at least** ceil(228/20.0) = 12 different TAs
  - Note: This is worse than CS2020 or even CS1101S ratio ☹…
- I prefer TAs who:
  - Like algorithms and problem solving
  - Eventually score at least A- in CS2020
    - You will know this by end of May 2011
  - Like to talk with students (to be precise: your friends)
    - FYI: these CS2010 students will be from YOUR BATCH

Part-time Teaching Assistant Job (2)

- If you are interested, drop me an email as soon as you know your CS2020 grade
  - Or if you are sure that you will get at least A-, then you can email me earlier
  - We will do some interviews, maybe during July 2011
- FYI, Dr Seth will teach CS2020 next year
  - S2, AY2011/2012 (Jan-Apr 2012)
  - Estimated enrollment next year: ~40-50
  - So if you prefer to TA CS2020 next year, drop Seth an email

Thank You ☺

- For being a good pioneer batch of CS2020
- We need your constructive feedback to further improve this module next year
- Please write those comments in your NUS Online Teaching Feedback exercise ☺