CS50 introduction of computer science

1. Week 0

Float is not precise due to limitation of memory

Use sandbox as ide

#include <stdio.h>

Compiling the .c to .out to machine code

1. Week1

CPU

RAM

HARD DISK DRIVE

Contiguous memory (array, list), array has indexes

Prototype of function

C convention: variable capital meaning global variable

String: array of char, last one is null ‘\0’

Int main(int argc, string argv[])

Argument count from the program

Sorting: select, bubble, merge

1. Week2

Array

String is array of char, but end in \0 require extra 1 byte

1. Week3

Linear search, binary search

Big O, Omega = Lower bound

Bubble sort, selection sort omega is still O(n^2), Twist bubble sort, omega can be O(n)

Merge sort omega I still (nlogn)

1. Week4

Hex decimal

Address, pointer

Typedef:define a new type

Char \*s = “emma”

String s = “emma” the address of the first character

Malloc, free

Swap

Computer memory up to down: 1. machine code: compiling purpose 2. Global variable 3. Heap where the malloc get memory bottom: stack: local variable

|  |
| --- |
|  |
|  |
|  |
| Swap (a, b, tmp) (because swap get copies) |
| Main (x, y) |

Stack overflow, function

Heap overflow, memory, buffer overflow

Scanf(“%s, &s);

Name of array as a pointer

File, fprintf to a file

1. Week5

Insert array, O(n)

Pointer[] now pointer acting as array

realloc, resize the memory

always free memory in the end of program

typedef struct node {} node (nick name alia)

(\*pointer).sth = pointer->sth

Memory leak

Array index access can use binary search if sorted, but link list lost random access

Linked list

Tree

Binary search tree, search O(logn) comparing linked list by add one more pointer to spend more space and save time

Hash table(map) is the combination of array and linked list: use some hash function to input to unique output

If there is collision, trade of time and space to resolve collision

Trie: a tree, its children is array of 26: search O(len(word)) meaning O(1) trade off memory

Queues: First In First Out, enqueue, dequeue

Stack: push, pop

Dictionary: hashtable

1. Week6

Python. Python hello.py no need to compile

Its own tools: list, tuple, dict, set

Dict is implemented by hashtable

Python no prototype function

Python no do while

Python print automatically end with \n

Input automatically from python is string.

No overflow in python

Command line arguments can use import argv from sys

exit(1) to terminate if fail as return 1 in C

dict is a abstract data structure

in c can’t compare string because it is about address whereas python can

regular expression

1. Week7

Sorted key = func or key = lambda input: output

Command line program: sqlite

DATA :crud, create, read, update, delete vs sql: insert, select, update, delete

SQL datatype: BLOB, INTEGER, NUMERIC, REAL, TEXT

**Heap data structure** is a complete binary tree satisfy the heap property. Max heap, value of each node is always greater than its child node, the value of the root is the largest.

Heapify: turn to the binary tree to heap.

def min\_heapify(array, i):  
 left = 2 \* i + 1  
 right = 2 \* i + 2  
 length = len(array) - 1  
 smallest = i if left <= length and array[i] > array[left]:  
 smallest = left  
 if right <= length and array[smallest] > array[right]:  
 smallest = right  
 if smallest != i:  
 array[i], array[smallest] = array[smallest], array[i]  
 min\_heapify(array, smallest)

time complexity O(log(n))

def build\_min\_heap(array):  
for i in reversed(range(len(array)//2)):  
min\_heapify(array, i)

Heapsort:

Nlogn

def heapsort(array):  
 array = array.copy()  
 build\_min\_heap(array) sorted\_array = []  
 for \_ in range(len(array)):  
 array[0], array[-1] = array[-1], array[0]  
 sorted\_array.append(array.pop())  
 min\_heapify(array, 0) return sorted\_array