ECE 368. Spring 2016

Practice Midterm.

Analysis of Algorithms (20 Pts)

Function A is denoted below:

```
FUNCTION_A(int n)                                   $Cost$    $Times$

1.    int sum = 0                                   $C_1$     1
2.          for ($i = n; i > 0 ; i \;/= 2$)         $C_2$     lg(n) + 2
3.                $for(j = 0; j < n; j++)$          $C_3$     $\sum_{i=0}^{i=\lg(n)+1}(n+1)$
5.                      $sum++;$                    $C_4$     $\sum_{i=0}^{i=\lg(n)+1}(n)$
6.    return sum                                    $C_5$     1
```

a) **(5 points)** For each instruction, fill the "Times" column. Write down the expression for the number of times the instruction is executed in terms of i, j and n.

b) **(5 points)** What is the worst case time complexity of the algorithm using the big-O notation?

Given in line 3 $\sum_{i=0}^{i=\lg(n)+1}(n+1) = (\lg(n)+2)(n+1) \sim \boldsymbol{O(n\lg(n))}$

A recursive function POW is denoted by the expression below:

```
POW(int x, int n)
1.    if n == 0
2.          return 1;
3.    $else$
5.          $return\ x * pow(x, n-1)$
```

c) **(10 points)** What is the time complexity (big O) of POW?

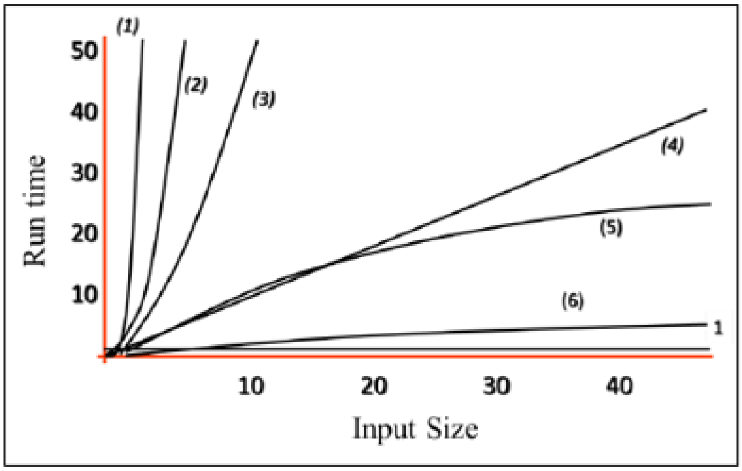
Sol: Similar to hw 2, Q1.

More details and basic recursive function analysis:

http://stackoverflow.com/questions/2709106/time-complexity-of-a-recursive-algorithm

Asymptotic Analysis (10 Pts)

Match each curve in the graph with one of the O() functions -- Nlog$_2$N, N^3, (log$_2$N)^2, N, N^2, log$_2$N.



Curve	O() function
(1)	n^3
(2)	n^2
(3)	$n \lg n$
(4)	n
(5)	$\lg n^2$
(6)	$\lg n$

Class Participation-Like Algorithms (20 Pts)

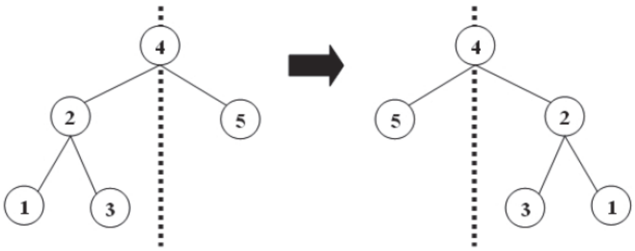
1(a) (8 points) Design an algorithm to find both the maximum and minimum of n numbers using at most $\lceil 3n/2 \rceil$ comparisons and no swaps. (To be exact, you can do that using $\lceil 3n/2 \rceil - 2$ comparisons.) You can either describe your algorithm in words or in pseudo-code. For simplicity, assume that n is even.

Sol: Compare every two elements through the array and update min and max every pairwise comparison.

Read more at:

http://stackoverflow.com/questions/13544476/how-to-find-max-and-min-in-array-using-minimum-comparisons

1(b). (12 points): You are given a binary tree. Write the pseudocode of a procedure that creates a mirror of this tree about the root. Your procedure will be called using a pointer to the root of the binary tree. For example, the original tree given on the left, when mirrored about the root, becomes the tree on the right. Use the page opposite this if you need extra space.

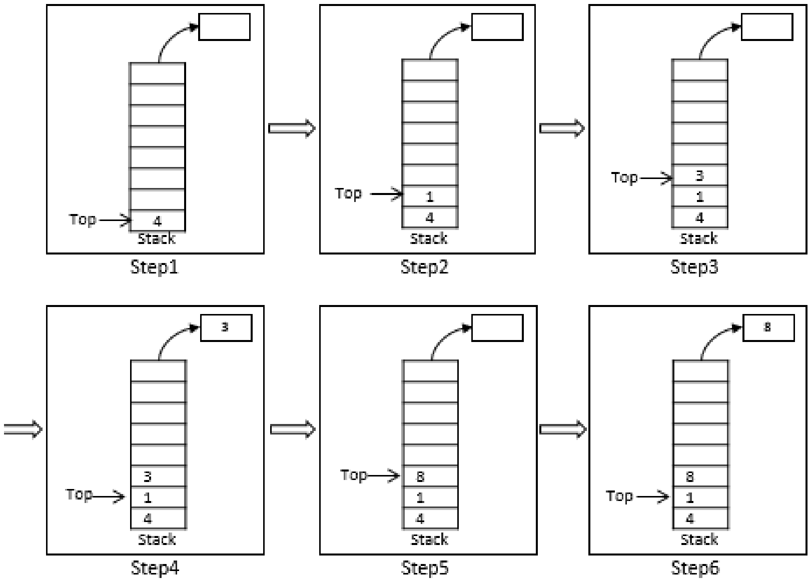


Sol:

http://stackoverflow.com/questions/4366251/mirror-image-of-a-binary-tree

Stacks and Queues (20 Pts)

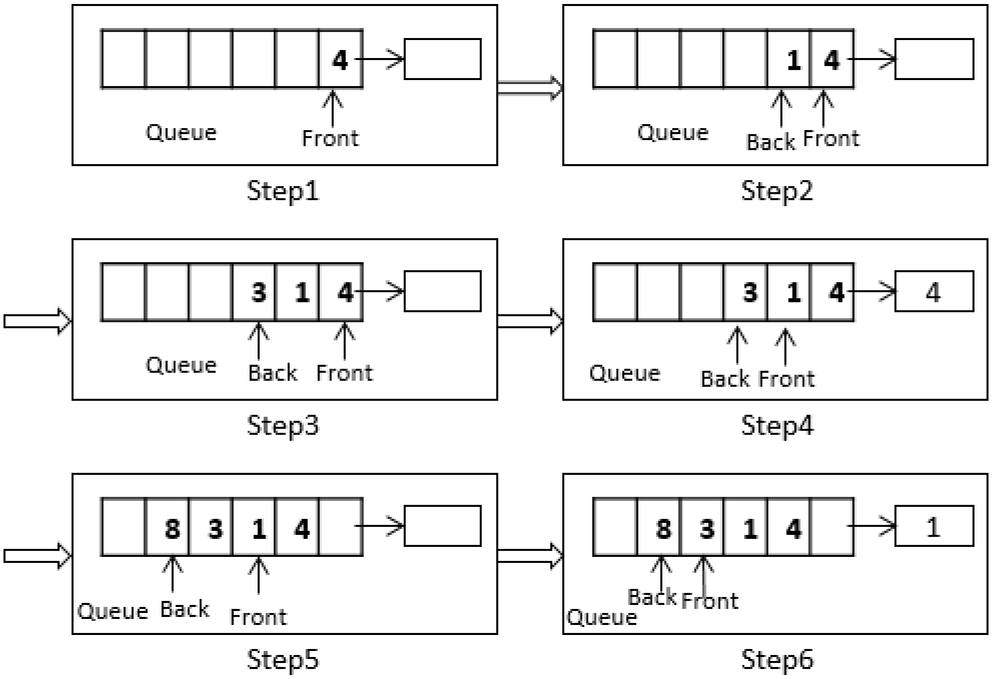
1. **(5 Pts)** Illustrate the result of *each* of the operations PUSH(S, 4), PUSH(S, 1), PUSH(S, 3), POP(S), PUSH(S, 8), and POP(S) on an initially empty stack *S* stored in array S[0..7] Indicate where the STACK_TOP index is pointing.



2. **(5 Pts)** How do you fix (time efficiently) an array-implemented stack if you run out of space?

Increase size by a percent. Allocate a new stack and copy all elements. More details Given in lecture 7

3. (5 Pts) Illustrate the result of each of the operations `ENQUEUE(Q, 4)`, `ENQUEUE(Q, 1)`, `ENQUEUE(Q, 3)`, `DEQUEUE(Q)`, `ENQUEUE(Q, 8)`, and `DEQUEUE(Q)` on an initially empty queue *Q* stored in array *Q[0..5]*. Here, `ENQUEUE` refers to inserting an element into the queue, and `DEQUEUE` refers to removing an element from the queue. Indicate where the `FRONT` and `REAR` pointers are.



4 . (5 Pts) Give a solution to fix an array-implemented Queue if you run out of space.

Implement Queue as a circular array. Move elements beyond the front to the end of the array.

Slide 28 of lecture 8.

Trees (Binary, Perfect, Complete Binary, N-ary, BST) (30 Pts total)

a. **(10 pts)** What is the relationship between the number of nodes in a full binary tree and the number of leaf nodes?

A full binary tree always has an even number of nodes, and the number of leaf nodes is always (n+1)/2.

Practice Question Taken from https://ece.uwaterloo.ca/~dwharder/aads/Lecture_materials/#introduction-and-reveiew. “Binary Trees”

b. **(5 pts)** What are the least and greatest number of leaf nodes in a binary tree with n nodes?

The least number is 1, while the greatest number is $\left\lceil \frac{n}{2} \right\rceil$

c. **(10 pts)** Use induction to prove that the number of leaves in a binary tree of heigh h is at most 2^h.

Sample Sol: https://courses.cs.washington.edu/courses/cse373/00sp/solutions/hw2-457.txt

d. **(5 pts)** Extend your proof in part c to prove that a perfect N-ary tree of height h has N^h leaf nodes.

Sample Sol: http://stackoverflow.com/questions/7842397/what-is-the-total-number-of-nodes-in-a-full-k-ary-tree-in-terms-of-the-number-o