Name (print): ____

PID: ___________________________

COMP 410 Fall 2018

Midterm Exam

This exam is closed book, notes, calculators, cell phones, classmates, smart watches, everything but your own brain. You have 75 minutes to complete the exam. Do all your work on these exam pages. Please sign here (**and print your name up top of each page**) pledging that the work you submit is your own:

Signature: ____

Q1 (12%) True or False (T / F):

a) ______ The run-time heap is dynamic memory from which objects are allocated on calls to "new"

b) ______ An N-ary tree is a tree with height of N

c) ______ For all computations P, if P is worst case time O(2^N) then P is also worst case O(N^2)

d) ______ Making a binary heap of N items by calling the insert operation N times is sometimes as efficient as using the "magic" build operation

e) ______ Garbage collection in Java makes it impossible to run out of run-time stack space during execution.

f) ______ Array representation for a binary tree is very fast to use, but usually space inefficient

g) ______ Time/space trade off says no program can be efficient in both run time and space used

h) ______ Recursion is a useful programming technique because theoretically it allows us to compute some functions that cannot be done with looping alone.

i) ______ Items put into a priority queue might come out in LIFO order

j) ______ pre-order traversal on the tree that is a minimum binary heap always produces the elements in increasing priority sequence

k) ______ For any set of unique data elements, if we insert these elements into an empty BST in different orders, we get the same final BST

l) ______ In practical terms, it is possible for a recursive function to fail to produce results, when an iterative version of that function will succeed in producing correct results

Q2 (4%): Consider a node M in a minimum binary heap; M is stored in the heap array at index 51.

a) At what array index will we find the parent of node M ? **answer:** _____________

b) At what array index will we find the right child of node M ? **answer:** _____________

Q3 (3%): What is the minimum number of nodes that might be in a *complete binary tree* with height K, if we also have that the tree is *not a full binary tree*:

a) 2^K – 2 b) 2^K c) 2*(K-1) d) 2^(K+1) –1 e) 2^(K –1) **answer:** ___________

Name (print): ____

For Q4 through Q10, fill in the table cells with the best (tightest, closest) Big Oh time complexities. If you think an answer is "O(M log k)" (for example), you may just write "M log k", leave off O(…).

Q4 (3%) Fill in with worst-case time complexity for **binary search tree (vanilla BST)** of N items

operation	add	delete	contains
Big Oh	a)	b)	c)

Q5 (3%) Fill in with **average**-case time complexity for **binary search tree (not balanced)** of N items

operation	add	delete	contains
Big Oh	a)	b)	c)

Q6 (3%) Fill in the table with worst-case time complexity for **queue (doubly linked cells)** of N items

operation	enque	deque	front
Big Oh	a)	b)	c)

Q7 (3%) Fill in the table with worst-case time complexity for **list (array implementation)** of N items

operation	add at i	remove at i	get ith
Big Oh	a)	b)	c)

Q8 (3%) Fill in the table with worst-case time complexity for **stack (array implementation)** of N items

operation	push	pop	top
Big Oh	a)	b)	c)

Q9 (3%) Fill in the table with worst-case time complexity for **min binary heap** of N items

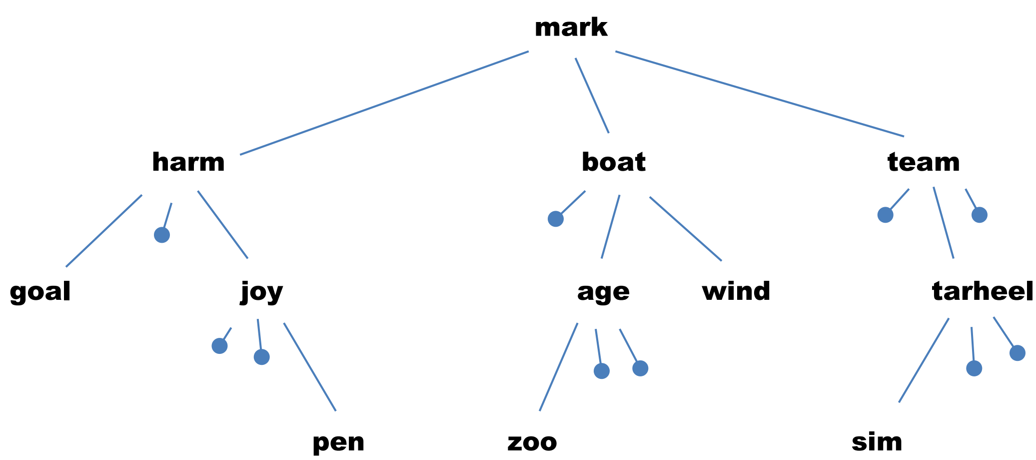
operation	add	getMin	delMin
Big Oh	a)	b)	c)

Name (print): ____

Q10 (8%) Fill in this table comparing sort methods for N items. Use theoretical Big Oh notation

Time complexity	**Worst case**	**Average case**
sort N items with a minimum binary heap	**a)**	**b)**
put N items into linked list, keep it sorted each insert (inSort operation)	**c)**	**d)**
BST sort N items (vanilla BST)	**e)**	**f)**
bubble sort (on an array) N items	**g)**	**h)**

Q11 (8%) Consider this 3-ary tree



Here are your answer choices:

1) mark harm goal joy pen boat age zoo wind team tarheel sim
2) goal harm joy pen mark boat zoo age wind team sim tarheel
3) mark harm boat team tarheel sim zoo pen goal joy age wind
4) goal pen joy harm zoo age wind boat sim tarheel team mark
5) none of the above

a) which sequence is an in-order traversal? ___________

b) which sequence is a pre-order traversal? ___________

c) which sequence is a breadth-first traversal? ___________

d) which sequence is a post-order traversal? ___________

Name (print): ____

Q12 (3%): Consider this code fragment for function **bublee**:

answer: _______________

```
public static long bublee(int N) {
  int[] arr = new int[N];
  for (int n=0; n<N; n++) { arr[n]=genRandInt(); }
  for (int i=0; i<N; i++) {
    for (int k=0; k<i*i; k++) { bubblesort( arr ); }
}
```

If we limit N to being a positive integer (not 0), and *assume bubblesort has worst case complexity* each time it runs, what is a good "Big Oh" complexity for the worst case execution time of function **bublee** ?

Q13 (3%): Consider this code fragment for function **foo**:

answer: _______________

```
public static long foo(int N) {
  int x = 2;
  for (int i=N; i>0; i--) {
    for (int j=i; j<i+5; j++) {
      for (int k=0; k<i; k++) {x *= i + j*k;} } }
}
```

If we limit N to being a positive integer (not 0), what is a good "Big Oh" complexity for the worst case execution time of function **foo** ?

Q14 (3%) Consider the program code to the right:

Which of these is most accurate when "main" is run?

a) the amount of run-time stack space that might be needed is finite
b) the amount of run-time stack space that might be needed is finite, but unbounded
c) the amount of run-time stack space that might be needed is infinite

answer: _______________

```
function main ( ) {
   var x = getUserInput();
   var result = foo(x);
   alert(result);
}
function foo ( n ) {
  if (n==1) return 1;
  return n * foo(n-1);
}
```

Q15 (3%) Consider the program code to the right:

Which of these is most accurate when "main" is run?

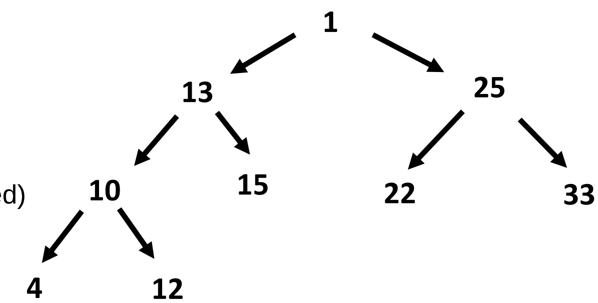
a) the amount of run-time stack space that might be needed is finite
b) the amount of run-time stack space that might be needed is finite, but unbounded
c) the amount of run-time stack space that might be needed is infinite

answer: _______________

```
function main ( ) {
   var x = 7683910024;
   var result = foo(x);
   alert(result);
}
function foo ( n ) {
  if (n==1) return 1;
  return n * foo(n-1);
}
```

Name (print): ____

Q16 (3%) Consider the data structure represented at right

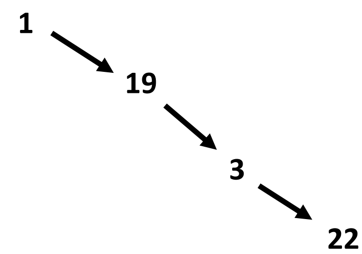


a) (T/F) ______ This could be a binary heap

b) **(T/F)** ______ This could be a BST (not being balanced)

c) **(T/F)** ______ This could be a doubly linked list

Q17 (4%) Consider the data structure/sequence represented at right



a) (T/F) ______ This could be a queue

b) (T/F) ______ This could be a stack

c) (T/F) ______ This could be a priority queue (done as list)

d) (T/F) ______ This could be a BST (not being balanced)

Q18 (10%): Binary Search Tree (not balanced)

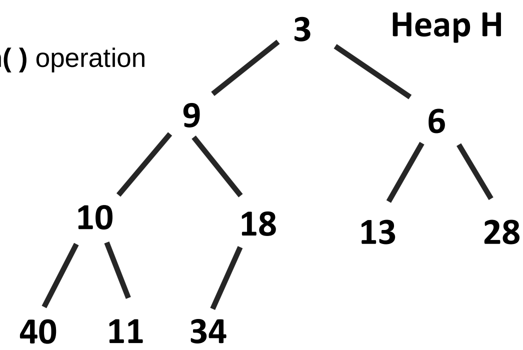
Starting with an initially empty Binary Search Tree (*vanilla, not being balanced*), show the tree that results after inserting the following string values in the order given left to right:

link, queue, heap, tree, axiom, mean, root, worst, stack, best

Name (print): ____

Q19 (5%) Consider the heap **H** shown to the right:

Show (in box below) the heap that results after a **delMin( )** operation



Q20 (5%) Consider the heap **H** shown above right (in previous question):

Show (in the box below) the heap that results after **add(7)** followed by **add(2)**

Name (print): ____

Q21 (3%): Consider this code fragment for function **recur**:

```
public static long recur(int N) {
  if (N <= 1) return 2;
  return recur(N-1) * recur(N-1);
}
```

answer: _______________

If we limit N to being a positive integer (not 0), what is a good "Big Oh" complexity for the worst case execution time of function **recur** ?

Q22 (5%): Consider the BST **B** (basic, not balanced) below. Show its structure after "**delete (18)**" is complete. Show your final tree in the box: