CS 3050 Homework # 1. **Name : KEY**

Submitted to Canvas, due at 11:59pm on Feb. 6, 2018.

1. Use Insertion Sort on array A and show the numbers and their position in the Array for each Iteration; also show the "key" position in each iteration.

A = {3,1,4,1,5,9,2,6}

Answer:

3	1	4	1	5	9	2	6	<- Swap A[0] and A[1]
1	3	4	1	5	9	2	6	<- No swap needed
1	3	4	1	5	9	2	6	<- Swap A[2] & A[3] and insert A[2] into A[1] and shift elements
1	1	3	4	5	9	2	6	<- No swap needed
1	1	3	4	5	9	2	6	<- No swap needed
1	1	3	4	5	9	2	6	<- Swap A[5] and A[6] and insert A[6] into A[2] and shift elements
1	1	2	3	4	5	9	6	<- Swap A[6] and A[7], no insertion necessary.
1	1	2	3	4	5	6	9	<- The list is now sorted

2. Use Bubble Sort on the following letters to sort it alphabetically and give each outer step with i and j.

{S T R A I G H T}

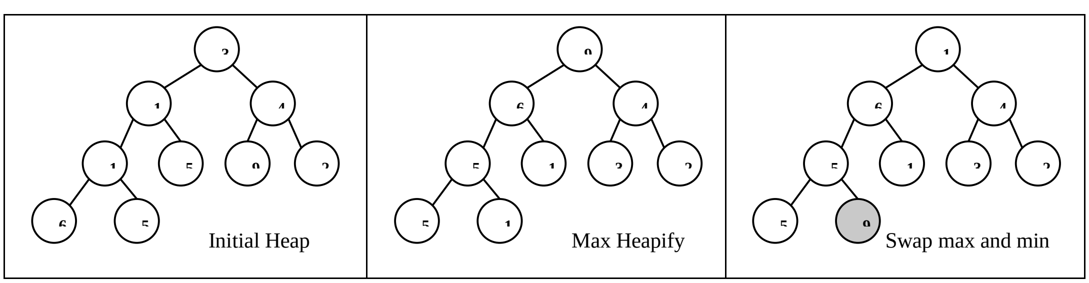
Before	*i*	*j*	
S T R A I G H T	S	T	No swap
S T R A I G H T	T	R	Swap
S R T A I G H T	T	A	Swap
S R A T I G H T	T	I	Swap
S R A I T G H T	T	G	Swap
S R A I G T H T	T	H	Swap
S R A I G H T T	T	T	Swap
S R A I G H T T	S	R	Swap
R S A I G H T T	S	A	Swap
R A S I G H T T	S	I	Swap
R A I S G H T T	S	G	Swap
R A I G S H T T	S	H	Swap
R A I G H S T T	S	T	No swap
R A I G H S T T	R	A	Swap
A R I G H S T T	R	I	Swap
A I R G H S T T	R	G	Swap

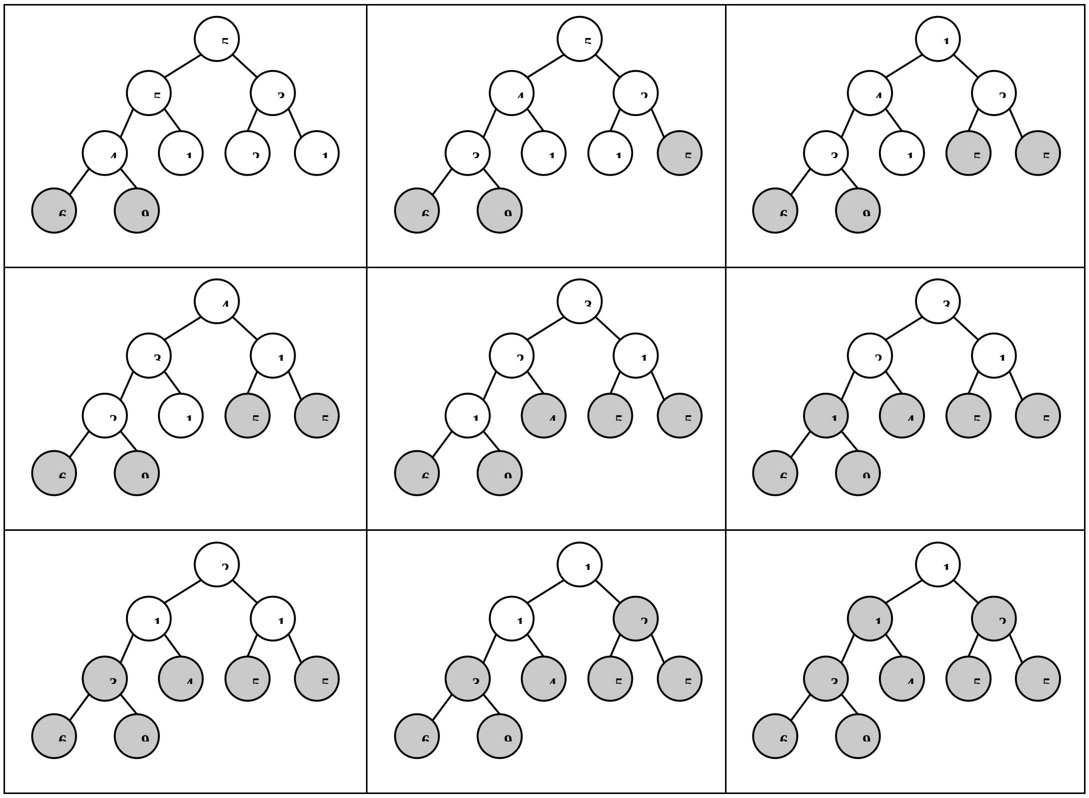
A I G R H S T T	R	H	Swap
A I G H R S T T	R	S	No swap
A I G H R S T T	A	I	No swap
A I G H R S T T	I	G	Swap
A G I H R S T T	I	H	Swap
A G H I R S T T	I	R	No swap
A G H I R S T T	A	G	No swap
A G H I R S T T	G	H	No swap
A G H I R S T T	H	I	No swap
A G H I R S T T	I	R	No swap
A G H I R S T T	R	S	No swap
A G H I R S T T	S	T	No swap
A G H I R S T T	T	T	No swap
A G H I R S T T	-	-	Sorted

3. What is the number of basic steps executed by the following method (as a function of n)? What is the big-O of the method?

```
Public int howLongA(int n) {
        int k = 0, kk = 0;
        for(int i = 0; i < n/2 ; i++) {
                k++;
                for(int j = 0; j < n/2; j++)
                        kk++;
        }
        return k*kk;
}
```

4. Using Figure 6.4 in the textbook as a model, illustrate the operation of HEAPSORT on the array:
A = {3,1,4,1,5,9,2,6,5}





5. **Analysis of tertiary heaps:** A ternary heap is like a binary heap, non-leaf nodes have 3 children instead of 2 children.

a. How would you represent a ternary heap in an array?

Solution #1(Index starts a 1):

A binary heap is a special case of a d-ary heap where d=2. We can extend the properties of d-ary heap to any abitrary number. In this case where d=3, we can then use the following formulae for representing the ternary heap in an array:

- Root = Floor[(k+1)/3]
- Left Child = 3*k-1
- Center Child = 3*k
- Right Child = 3*k+1

Solution #2(Index starts at 0):

- Root = ((i-2)/3) + 1
- Left Child = 3(i-1)+2
- Center Child = 3(i-1)+3
- Right Child = 3(i-1)+4

a. What is the height of a ternary heap of n elements in terms of n?
Answer:
$h = \log_3 n$

b. What is the height of a ternary heap of n elements in terms of n?
c. Give an efficient implementation (in pseudo code) of EXTRACT-MAX in a tertiary max-heap. Analyze its running time in terms of n.
Answer:

HeapExtract-Max(A)	
if A.heap-size < 1	O(1)
error "heap underflow"	O(1)
max = A[1]	O(1)
A[1] = A[A.heap-size]	O(1)
A.heap-size = A.heap-size-1	O(1)
Max-Heapify(A,1)	$O(\log_3 n)$
return max	Total $O(\log_3 n)$

d. Give an efficient implementation (in pseudo code) of INSERT in a ternary max-heap. Analyze its running time in terms of n.
Answer:

Heap-Insert(A, key)	
A.heap-size++	O(1)
i = A.heap-size(A)	O(1)
A[i] = -∞	O(1)
HeapChangeKey(A, i, key)	$O(\log_3 n)$
	Total $O(\log_3 n)$

e. Give an efficient implementation (in pseudo code) of INCREASE-KEY(A,i,k), which flags an error if k < A[i], but otherwise sets A[i] = k and then updates the ternary maxheap structure appropriately. Analyze its running time in terms of n.

Answer:

HeapIncrease-Key(A, i, key)	
A[i] = key	O(1)
while (i > 1 and A[parent(i)] < A[i])	O(1)
swap(A[i], A[parent(i)])	$O(\log_3 n)$
	Total $O(\log_3 n)$