CS210 Data Structures (172) **Name________________________________** **ID_______________**

Major Exam 2 **Weight 15%** **Time: 50 minutes**

There are six questions in this exam. Answer all questions. Time is of the essence, use it wisely!

Q1. [2] In project 2, you implemented a program to process data for an online retailer using min and max binary heaps. Imagine that your program reads from a text file instead of console and processes unlimited data.

Explain what changes you would make to your program.

What would be the size of minHeap and maxHeap? How would you determine it?

What would be the runtime given that duplicates are not allowed? How can you prove it?

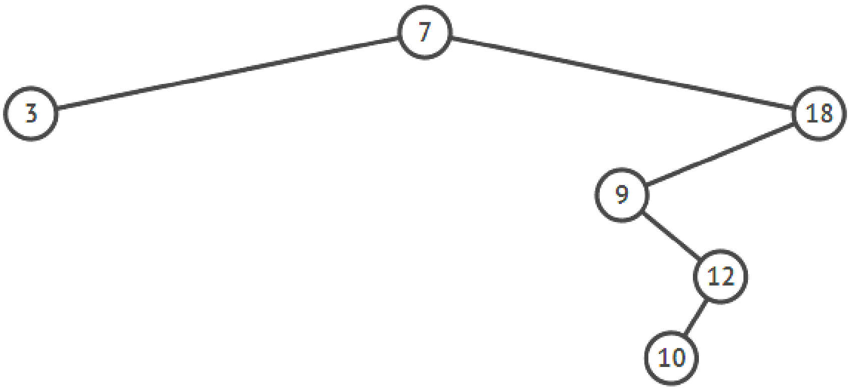
Q2. [4]. An instructor decided to sort the first names of all students in his class. Someone told him to use a priority queue (min-Heap) to sort all names in ascending order.

(a) Show how you would insert the following names in this heap. Illustrate trees/heap at each step:

“Bilal” “Bob” “John” “Jason” “Jamal” “Basel” “Bandar” “Ahmed”

(b) Remove two names from the above Priority Queue. Highlight which nodes you decide to sink or swim and re-draw the heap at each step.

[CLO-3] Q2. [3] Convert the following BST of integers to an AVL tree. Re-draw the tree every time you decide to rotate it. Highlight the rotation type (case) for each rotation.



Q4. [3] Write a method called **public boolean searchInStack(Stack <Item> S, String key)**. This method searches for String key within the Stack S of type Item. If the key matches with the **`item description`** of any item in the stack, the method returns true, false otherwise. Make sure, after the searching is complete, the stack S is restored with its original values in the same order. **`Item`** API is given in the adjacent box.

```
public class Item {
  int item_id;
  String description;
  double price;
  //constructors
}
```

public boolean searchInStack(Stack <Item> S, String key) {

[CLO-4] Q5. [3] Consider a hash table of size 9 that stores entries with integer keys. Suppose the hash function is *h(k) = k mod 7*. Insert, in the given order, entries with keys **8, 11, 18, 22, 28, 13, 25** into the hash table using linear probing to resolve collisions. Show all the work and fill the array **A** accordingly.

A

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**

k	h(k)	Collisions
5		
11		
18		
23		
28		
13		
25		

Insert the same keys in the below Hashtable using Separate Chaining method. Use the same hash function

0
1
2
3
4
5
6

Which method provides less number of collisions?

--End of Exam--