FIRST DAY REVIEW MATERIAL. This quick assignment is to help you review CSC 20 and help me understand where you need help for CSC 130.

Answer the following questions and submit a pdf or doc file with your answers.
You may use your review material from CSC 20. The Weiss chapters 1-3, CSC130FIrstDay, and FirstDayContinued lecture slides have relevant material as well.

1. Have you taken CSC 130 before?

- ***No, I have not taken CSC 130 before.***

2. Do you know Java programming? If not: http://horstmann.com/ccc/c_to_java.pdf

- ***Yes, I have some experience with Java programming.***

   a. At home: Review Java & install IDE

- ***I installed Eclipse.***

3. Push the following elements into the empty stack below: 1,2,4,6,7,5,2

4. Pop three elements. Which elements are in the stack now? → 1,2,4,6

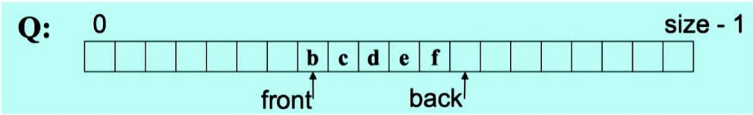
5. What are other Stack operations?

- ***peek***
- ***empty***
- ***search***

6. Enqueue the following elements into the empty queue below: 3, 2, 0, 1.

7. Dequeue three elements. Which elements are in the queue now? → 1

8. The following is an example of a circular array queue data structure:



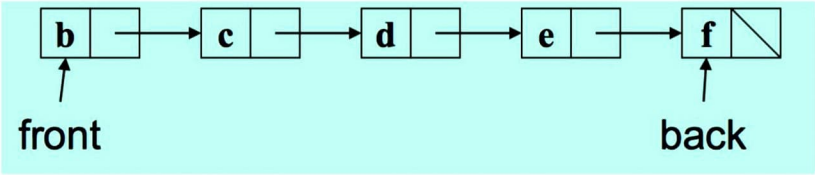
The queue implementation has the following instance variables:

- char array – holds the items
- int front, back – shows the front and back of the circular array queue.

Implement in Java the following operations:
enqueue(char x): inserts a new char elements into the back of the queue.
isEmpty(): returns true or false if the queue is empty

Include your Java code.

9.



The above is a linked list implementation of a queue.

In a linked list implementation of a queue, how do you check if the queue is empty?
What is the basic idea of enqueuing a new item into the list? (ie. Describe how you would implement enqueue).

- ***In a linked list implementation of a queue, we can check if the queue is empty by simply checking if the address of the head node is the same as the tail node. If the head node has the same address as the tail node, the queue is empty.***

10. The following is a snippet of code. What is the worst case runtime in O(n)?

```
b = b + 5
c = b / a
b = c + 100
```

O(1)

```
for (i = 0; i < n; i++) {
    sum++;
}
```

O(N)

O(N)

```
if (j < 5) {
   sum++;
} else {
  for (i = 0; i < n; i++) {
     sum++;
  }
}
```

O(1)

O(N)

11. Given an array in int, implement the following boolean Java function (include your Java code):

```
//Find an integer k in a sorted array
//requires array is sorted
//returns if k is in the array
boolean find(int[] arr, int k)
```