CSCI 2270 – Data Structures and Algorithms
Instructor: Hoenigman
Midterm Review Questions

Queue for question 1

	1	2	3	4	5	6
Q						

1. Using the queue shown here, illustrate the result of each operation in the sequence:
    a. Enqueue(Q, 4), Enqueue(Q, 1), Enqueue(Q, 3), Dequeue(Q), Enqueue(Q, 8), Dequeue(Q) on an initially empty queue Q stored in array Q[1…6]. Draw the condition of the queue and the output of the Dequeue operation.

Final condition:

	1	2	3	4	5	6
Q			3	8		

First dequeue outputs 4
Second dequeue outputs 1

Queue for question 2

	1	2	3	4
Q				

2. Given the following sequence, if your Enqueue operation doesn't check if the queue is full, does data get overwritten using a circular queue stored as an array Q[1…4]. Explain your answer.
    a. Enqueue(Q, 4), Enqueue(Q, 1), Enqueue(Q, 3), Dequeue(Q), Enqueue(Q, 5), Enqueue(Q, 6)

    No, data is not overwritten. When we call dequeue that frees a space at the beginning of the array, which is where the 6 is written to on the Enqueue(Q, 6) operation.

3. Given the following array called A, what does A look like after the code below it executes.

	0	1	2	3	4	5
A =	30	35	12	15	16	0

```
int final = 5;
for(int index = 0; index < final; index++){
```

```
          if(A[index] == 12){
               int temp = A[index];
               A[index] = A[index+1];
               A[index+1] = temp;
          }
     }
```

The 12 bubbles to the end and the final array looks like:

	0	1	2	3	4	5
A =	30	35	15	16	0	12

Given the following algorithm and cost for each line, what is the cost of the code **in the for loop** for this array: A = <45, 34, 32, 34, 12, 23, 35>, and v = 45? How does the cost change when v = 34?

Pseudocode	**Cost of each line**
`findItem(A, v)`	0
`  index = -1`	1
`  for i=1 to A.length`	0
`    if A[i] == v`	1
`      index = i`	1
`  return index`	1

The conditional `if A[i] == v` executes each time we go into the for loop, which is once for every element in the array. Then, `index = i` executes when the condition is true.
The cost is 8 when v = 34, and 9 when v = 45.

4. Which of the following is the most computationally expensive (assuming each line of code has the same cost):
   a. Adding an item to the beginning of a singly linked list.
   b. Adding an item to the middle of an array (with space available).
   c. Adding an item to the middle of a linked list, after the location has been identified through a search.

b. Because the array will need to be shifted to make room for the new item. For the other option a and c, these have a constant cost and are not dependent on the size of the array.

5. Convert the number 234 to hex and then to binary.
   Hex: EA
   Binary: 1 1 1 0 1 0 1 0
6. Given the following code, what is the value of *x after the call to function didXChange(x)? What is the value of b in the main function after the call to didXChange(x)?

```
int didXChange(int *x2){
     *x2= *x2 + 1;
     return *x2;
}
int main(){
     int b;
     int *x = new int;
     *x = 5;
     b = didXChange(x);


}
```

The value of *x and b are both 6.