United International University (UIU)

Dept. of Computer Science & Engineering (CSE)

Midterm Assessment **Summer 2020**

Course Code: CSI 217 **Course Title: Data Structures**

Total Marks: **20** Duration: **1 hour**

There are THREE Questions. Answer all of them.

1. Answer the questions given below
   a. Find out the run time complexity of the following algorithm. Clearly show the complexity of the necessary individual statements. [2]

```
1. sum = 0;
2. for( i=0; i<n; i=i+2 )
3. {
4.     for( j=0; j<n; j=j+3 )
5.     {
6.         sum++;
7.     }
8. }
```

   b. Suppose there is a **doubly linked node** declared as below, where you only have the head pointer which points to the first node of the doubly linked list: [5]

```
struct node{
        int data;
        node * next;
        node * prev;
}
```

Show the effect of each of the statements by drawing nodes and linking them.

i. a = (node*) malloc(sizeof(node));
ii. b = (node*) malloc(sizeof(node));
iii. c = (node*) malloc(sizeof(node));
iv. a→ next = b;
v. a→ data = 100;
vi. b→ next = NULL;
vii. b→ data = 5;
viii. c→ next = b;
ix. c→ data = 9;
x. a→ prev = b;
xi. b→ prev = NULL;
xii. c→ prev = a;
xiii. a→ next = c;
xiv. free(b);

2. Suppose there is a **singly linked list** node declared as below, where you only have the ***head*** pointer which points to the first node of the singly linked list:

```
struct Node {
      int data;
      struct Node * next;
}
```

   a. Write a function ***delete_all( )*** to delete the entire singly linked list. Remember that you need to bring the linked list to its initial state. [2]

   b. For the same linked list, write a code to find the maximum valued node from the list and print the value. See the example below for clarification. [2]

Sample Input	Sample Output
9 → 6 → 12 → 5 → 7	12

   c. Suppose you have a **singly linked list** and you want to print the linked list in reverse order. You also have a **stack**. Explain how you can print the values in linked list in reverse order using the stack. See the example below for clarification. [3]

Linked List	Printed in reverse order
1 → 2 → 3 → 4 → 5	5, 4, 3, 2, 1

3.

a. Suppose a **Queue** implemented using an array with dynamically increasing size. Explain briefly, how you can copy the items from the old Queue memory to the newly allocated memory when the Queue becomes full. [2]

b. Suppose a Stack is implemented by using an array and a Queue is implemented by using another array. Now consider the following sequences containing Push, Pop and Peek functions of stack, and Enqueue, Dequeue and Top function of queue. Assume that the size of both stack and queue is 4. [4]

Push(x)→ inserts value x into stack Pop()→ deletes value from stack Peek()→ returns the value at the top of stack	Enqueue(x)→ inserts value x into queue Dequeue()→ deletes value from queue Top()→ returns the value at the front of the queue

You have to draw what happens after performing each of the following operations. Clearly show the value of the top of the stack, and front and rear of the queue in each step. Also show the elements in the stack and queue after performing all the operations:

Push(100), Enqueue(20), Push(300), Enqueue(Peek()), Enqueue(Peek()), Pop(), Push(Top()), Dequeue()