United International University (UIU)

Dept. of Computer Science & Engineering (CSE)

Midterm Exam **Trimester: Fall - 2019**

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

Total Marks: **30** Duration: **1 hour 45 minutes**

Answer all 6 questions. *Each question contains 5 marks.*

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

```
1.  int sum = 0;
2.  for ( i=0; i<n; i++ )
3.  {
4.      for ( k=1; k<n; k=k+2 )
5.      {
6.          sum++;
7.      }
8.  }
9.  printf("%d\n", sum);
```

   b. Perform **counting sort** on the following array: [3]
   You have to clearly show your work.

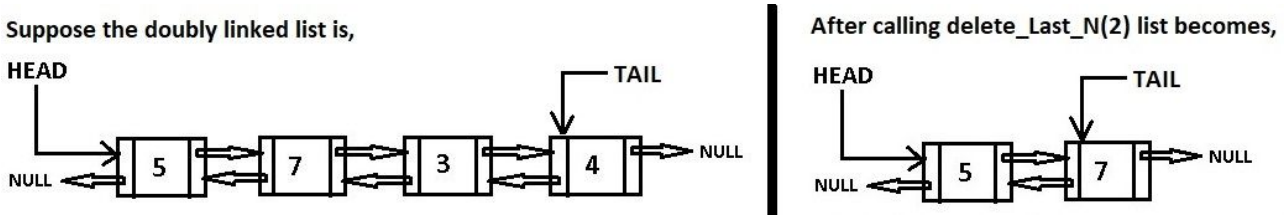
1	3	2	3	1	1	3	2	4	1	3	2	1	2	2

2. Answer the following questions on sorting algorithms.
   a. Perform **quick sort** on the following array upto the **second** partition. [3]

30	8	11	5	100	3	9	12	18

   b. What is the worst-case time complexity of Quick sort? Clearly explain your answer with examples. [2]

3.
   a. Given a **Doubly Linked List**, and an integer *n* write a function *delete_last_N(int n)* to delete last *n* nodes from the given list. If there are fewer than *n* elements return *FAILURE/FALSE*. Assume you know the pointer to the *head* and *tail* of the list. [3]



b. Given a **Circular Singly Linked List** with *tail* pointer only, write a function *shift_tail( )* to rotate the list so that *tail* pointer is shifted one node back. See the following diagram (Figure 1) for clarification. [2]

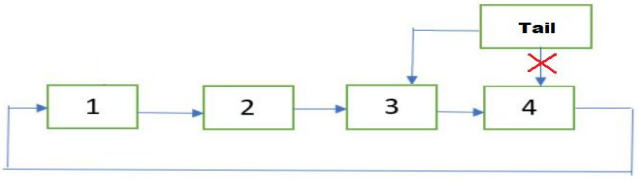


Figure 1 Question 4(b)

4. Suppose there is a **singly linked list node** declared as below, where head is the pointer to the first node of the linked list:

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

a. Given a pointer to the head of the singly linked list, write a code to insert a node after the middle element. You can assume any value for the node data. For this problem you can assume that the number of nodes in the linked list will always be greater than 5. [2]

b. Show the effect of each of the statements by drawing nodes and linking them. [3]

i. a = (node*) malloc(sizeof(node)); **OR** a = new node();
ii. b = (node*) malloc(sizeof(node)); **OR** b = new node();
iii. c = (node*) malloc(sizeof(node)); **OR** c = new node();
iv. a->data = 10;
v. b->data = 25;
vi. c->data = 7;
vii. a->next = c;
viii. c->next = b;
ix. b->next = NULL;
x. d = (node*) malloc(sizeof(node)); **OR** d = new node();
xi. d->data = 5;
xii. d->next = b;
xiii. a->next = b;
xiv. free(c);

5.

a. Write a pseudo code of a function that can reverse a Stack using a Queue. Assume that the basic operations like *push( ), pop( ), enqueue( ), dequeue( ), empty( )* etc are already implemented for you to use. [2]

b. Write *push( )* and *pop( )* function for a Stack that uses linked list implementation. [2]

c. Is Doubly Linked List a good suit for implementing a Stack? Why or why not? [1]

6.

a. Suppose Stack is implemented using an array and Queue is implemented using a circular array. Now consider the following sequences of push, pop, enqueue and dequeue. Assume that the size of stack is 6 and the size of the queue is also 6. [3]

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

   push(5), enqueue(6), push(10), enqueue(12), push(15), push(dequeue()), enqueue(24), pop(), pop(), enqueue(pop()), dequeue(), dequeue()

b. How would you implement a queue using stack(s). You don't have to write the code, explain with proper example. [2]