**Mid-Term Exam Fall - 2022**

# Data Structures and Algorithms – Lab

**Instructions for Invigilators:**

1. Students will have total 90 minutes to finish the whole exam. It is up to the students to manage their time.

**Instructions for Students:**

1. Please create file with appropriate name
2. Submit only **.h** and **.cpp** files with output screenshot on portal
3. Late submissions will **NOT** be considered
4. Create as many classes and functions as required. Remember one function for one functionality.
5. Take care, plagiarism will not be tolerated at any case.
6. No .rar files are accepted
7. The paper is close book and close notes. No cheat sheet allowed.
8. Use meaningful variable names, take care of naming conventions and indentation. **5 Marks will be deducted for each thing if not followed.**

# Question 1 – 30 Marks

Implement the Linked List using **head** pointer only (you are not allowed to use **tail pointer). LinkedList.h** class is shown below. Your task is to provide **implementation** for **MyLinkedList** and **Node**.

template<class Type> class LinkedList

{

protected:

Node<Type>\* head;

public:

LinkedList<Type>();

LinkedList<Type>(const LinkedList<Type> &obj); virtual void insertAtEnd(Type \_value) = 0; virtual Type deleteFromHead() = 0;

virtual bool isEmpty() = 0; virtual void print() = 0;

~LinkedList();

};

**Note:** You can implement either a singly or doubly linked list. You can also make the linked list either circular or non-circular. It is completely your choice.

# Question 2 – 30 Marks

Implement FIFO algorithm using **MyLinkedList** implemented in *task 1***. Queue.h** class is given below. Your task is to provide **implementation** for **MyQueue** class.

# Interface:

template<class Type> class Queue

{

protected:

MyLinkedList<Type> list; int currentSize;

public:

Queue<Type>();

Queue<Type>(const Queue<Type> &obj); virtual void enqueue(Type \_value) = 0; virtual Type dequeue() = 0;

virtual void display() = 0; virtual int size() = 0;

~Queue();

};

# Question 3 – 30 Marks

Now write a global function **reverseQueue** which should reverse all the contents of the **Queue**.

template<class Type>

MyQueue <Type> reverseQueue (MyQueue <Type> obj);

**Remember:** You are not allowed to use any other data structure other than **MyQueue** in the above function. For eg: You cannot use any array, stack or linked list etc in this function.

# Question 4 – 10 Marks

Now test the main function and produce the exact output given below. **Do attach the screen shot of your output in your submission.**

int main()

{

cout << "\n\n---------- Best of Luck for the Exam \n\n";

MyQueue<char> q1; q1.enqueue('D');

q1.enqueue('S');

q1.enqueue('A');

q1.enqueue(' ');

q1.enqueue('L');

q1.enqueue('A');

q1.enqueue('B'); q1.display();

MyQueue<char> reverseQ1 = reverseQueue(q1); reverseQ1.display();

return 0;

}

**Output:**

