# National University
## of Computer & Emerging Sciences

Student Name: _____ Roll No: _____

Program: BS(CS/SE) Quiz # 01 (Section B)

Semester: Spring 2022 Marks: 10

Time Allowed : 15 minutes Date: 25-02-2022

Course: CS2001 - Data Structures Instructor: Waqas Ali

1. Suppose that an **array based list** of sorted integers exists in memory. Write a function (named sortedInsert) that inserts a value in the list at its **proper position**, such that after inserting the value, the list is still sorted. The list can either be in ascending or descending order, **your function must work for both types**. You can write a separate function to determine whether the list is sorted in ascending or descending order. Consider the array list class is **already present**.

   For Example:

| 1 | 4 | 6 | 11 | | |
|---|---|---|---|---|---|

   If we want to insert 8 in to the above list, the **correct position** of 8 is between 6 and 11.

   **Note:** You are not allowed to sort the list after insertion, you have to insert the value in its proper position.

## Solution

```
bool isAscending()
{
        tail();
        if (*arr < *curr)
                return true;
        else
                return false;
}
```

```
bool compareV(int val, int curr, bool Asc)
{
        if (Asc)
        {
                if (val >= curr)
                        return true;
                else
                        return false;
        }
        else
        {
                if (val <= curr)
                        return true;
                else
                        return false;
        }
}
```

```
void sortedInsert(int val)
{
        if(size == length)
        {
                cout << ``List is full, cannot insert'' << endl;
                return;
        }

        tail();

        for (int i = length; i <= 1; i--)
        {
                if(compareV(val, *curr, isAscending()))
                        break;
                else
                {
                        *(curr + 1) = *curr;
                        back();
                }
        }
        *(curr + 1) = val;
        length++;
}
```