Foundation for Advancement of Science and Technology

National University of Computer & Emerging Sciences (NUCES), Islamabad

School of Computing

DATA STRUCTURES
FALL 2021
LAB 01

Learning Outcomes

In this lab you are expected to learn the following:

- Abstract Data Types
- Templates

In this laboratory, you will implement ADT using Templates.

Lab Task 1

Writing three similar functions can be avoided if we use templates. Write only one Function which is generic for all data types shown in figure.

```
1  /*************************************************************
2   * A program to find the smaller between three types of data *
3   *************************************************************/
4  #include <iostream>
5  using namespace std;
6
7  // Function to find the smaller between two characters
8  char smaller (char first, char second)
9  {
10   If (first < second)
11   {
12        return first;
13   }
14   return second;
15 }
16 // Function to find the smaller between two integers
17 int smaller (int first, int second)
18 {
19   If (first < second)
20   {
21        return first;
22   }
23   return second;
24 }
25 // Function to find the smaller between two doubles
26 double smaller (double first, double second)
27 {
28   If (first < second)
29   {
30        return first;
31   }
32   return second;
33 }
34
35 int main ()
36 {
```

```
37      cout << "Smaller of 'a' and 'B': " << smaller ('a', 'B') << endl;
38      cout << "Smaller of 12 and 15: " << smaller (12, 15) << endl;
39      cout << "Smaller of 44.2 and 33.1: " << smaller (44.2, 33.1) << endl;
40      return 0;
41  }
```

```
Run:
Smaller of 'a' and 'B': B
Smaller of 12 and 15: 12
Smaller of 44.2 and 33.1: 33.1
```

Lab Task 2

You have studied overloading of regular (non-template) functions. We can apply the same concept to function templates. We can overload a function template to have several functions with the same name but different signatures. Normally, the template type is the same, but the number of parameters is different.

You are required to overload the **smaller** template function that you have implemented in task-1 to accept two or three parameters

(You can rename the function to smallest because it uses more than two arguments).

Lab Task 3

- Write a program that take two arrays as input and find the same elements from the arrays.
- Write a program that take two arrays as input and concatenate the arrays but the condition is that there are no same elements in the resultant array

Note: input of the array must be of the type int, char, and float.

Lab Task 4

Define a template class **Container** for storing different type of values e.g. int, double, float etc. This container would keep same type of values/elements at a time. The **Container** class should consist of the following member variables:

T * values; A pointer to set of values it contains. This member points to the dynamically allocated array (which you will be allocating in constructor).

capacity; An integer that shows total capacity of the container

counter; An integer counter variable which increments upon every insertion operation and decrements upon removal of any element; representing total number of elements currently present in the container.

The container should consist of following functions:

- **constructor** with capacity as parameter
- **isFull** which will return true if counter reaches capacity otherwise false
- **insert** which will receive a value of some type (int/double/float) and insert into the container if it's not already full
- **search** which will return true if the container contains value passed as parameter

- **remove** which will remove a value, if exists, from the container and shifts all subsequent values one index back.
- **print** which will print all values contained in the container

Now create a test class to create three instances of Container for types: **int, float, double** and call all functions separately for each instance. Your output must be properly formatted.