Logo

Description automatically generated University of Central Punjab

**Faculty of Information Technology**

# Data Structures and Algorithms

# Spring 2022

|  |  |  |
| --- | --- | --- |
| **Lab 02** | |  |
| **Topic** | * Understanding Classes * Working on Three Different files * Abstract Classes * Templates * Arrays |
| **Objective** | The basic purpose of this lab is to revise some preliminary concepts of C++ that has been covered in the course of Introduction to Computing and Programming Fundamentals and Object Oriented Programming. |
|  | | |

**Instructions:**

* Indent your code.
* Comment your code.
* Use meaningful variable names.
* Plan your code carefully on a piece of paper before you implement it.
* Name of the program should be same as the task name. i.e. the first program should be Task\_1.cpp
* **void main() is not allowed. Use int main()**
* **You have to work in multiple files. i.e separate .h and .cpp files**
* **You are not allowed to use system**("**pause**")
* **You are not allowed to use any built-in functions**
* **You are required to follow the naming conventions as follow:**
  + **Variables:** firstName; (no underscores allowed)
  + **Function:** getName(); (no underscores allowed)
  + **ClassName:** BankAccount (no underscores allowed)

**Students are required to complete the following tasks in lab timings.**

## **Task 1**

Create a C++ generic abstract class named as **List**, with the following:

**Attributes:**

1. Type \* arr;
2. int maxSize;
3. int currentSize;

**Functions:**

virtual void addElementAtFirstIndex(Type) = 0;

* Should add the element at the first position of the **List**

virtual void addElementAtLastIndex(Type) = 0;

* Should add the element at the last position of the **List**

virtual Type removeElementFromEnd() = 0;

* Should remove the element from the last position of the **List**

virtual void removeElementFromStart() = 0;

* Should remove the element from the first position of the **List**
* Write parameterized constructor with default arguments for the above class.
* Write Copy constructor for the above class.
* Write Destructor for the above class.

## **Task 2**

Create a menu based program for the following functions, using the class made in task 1, make a class named as **MyList**, having following additional functionalities:

**bool** [**empty()**](https://www.geeksforgeeks.org/stack-empty-and-stack-size-in-c-stl/) : Returns whether the MyList is empty or not

**bool** [**full()**](https://www.geeksforgeeks.org/stack-empty-and-stack-size-in-c-stl/) **:** Returns whether the MyList is full or not  
**int** [**size()**](https://www.geeksforgeeks.org/stack-empty-and-stack-size-in-c-stl/) : Returns the current size of the MyList

**bool insertAt(int index, T value):** Adds a value at the index passed to the function, returns true if the index is present and value is added else returns false.   
**Type** [**last()**](https://www.geeksforgeeks.org/stack-top-c-stl/) : Returns the last element of the MyList

**bool search(Type):** Returns true if the searched value is present in the list else returns false

* Write parameterized constructor with default arguments for the above class.
* Write Copy constructor for the above class.
* Write Destructor for the above class.

## **Task 3**

Create a menu based program for the following functions, using the class made in task 2, make a class named as **CustomList**, having following additional functionalities:

**Type sum\_ofPrime()** : It finds prime numbers in the list, calculates and returns the sum of all prime number present in list.

**Type secondMaxEven()** : It finds and returns **second maximum even number** present in list.

**Type secondMinOdd()** : It finds and returns **second minimum odd number** present in list.

**void printDuplicates():** It finds and displays all the numbers which occurs more than once.

**void rotateClockwaise( int r)** : It divides the list into two parts (halves the list) , and rotate both parts “**r**” times in clockwise direction

**void rotateanitclockwaise( int rt)** : It divides the list into two parts (halves the list) , and rotate both parts “**rt**” times in anti-clockwise direction

* Write parameterized constructor with default arguments for the above class.
* Write Copy constructor for the above class.
* Write Destructor for the above class.