 University of Central Punjab

**Faculty of Information Technology**

# Data Structures and Algorithms

# Spring 2021

|  |  |  |
| --- | --- | --- |
| **Lab 06** | |  |
| **Topic** | * Abstract Classes * Templates * Circular Queue |
| **Objective** | The basic purpose of this lab is to implement ADT of Circular queue and test its applications. |
|  | | |

**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

**Queue: Queue** is a data structure that works on First in First out (FIFO) approach. In general, items are added to the end of Array (In the way that we join at the end of a queue for a bus) and items are removed from the front of an Array. (The people at the front of the queue for the bus can get on the bus first.)

**Circular Queue**: is a linear data structure in which the operations are performed based on FIFO (First In First Out) principle and the **last position** is connected back to the **first position** to make a circle. It is also called **‘Ring Buffer**

**Applications:**

1. **Memory Management:** The unused memory locations in the case of ordinary queues can be utilized in circular queues.
2. **Traffic system:** In computer controlled traffic system, circular queues are used to switch on the traffic lights one by one repeatedly as per the time set.
3. **CPU Scheduling:** Operating systems often maintain a queue of processes that are ready to execute or that are waiting for a particular event to occur.

So using the class made in last lab, make a class named as **CircularQueue**, having following additional functionalities:

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

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

**int** [**size()**](https://www.geeksforgeeks.org/stack-empty-and-stack-size-in-c-stl/) : Returns the current size of the **CircularQueue**.   
**Type** [**front ()**](https://www.geeksforgeeks.org/stack-top-c-stl/) : Returns the front element of the **CircularQueue.**

**void** [**enqueue(Type)**](https://www.geeksforgeeks.org/stack-push-and-pop-in-c-stl/) : Adds the element of type Type at the top of the **CircularQueue**.

**Type** [**dequeue()**](https://www.geeksforgeeks.org/stack-push-and-pop-in-c-stl/) : Deletes the first most element of the **CircularQueue** and returns it**Hint:** You can use removeElement function in it. You should shift left all the elements after you remove an element from **CircularQueue**.

* Write non-parameterized constructor for the above class.
* Write Copy constructor for the above class.
* Write Destructor for the above class.

Now write a global function **showCircularQueue** which should display all the contents of the **CircularQueue**.

void showCircularQueue (**CircularQueue** <Type> obj);

Instantiate several objects of **CircularQueue**, test all the functions of **CircularQueue** on them and then display them through showCircularQueue function.

## Task 2

Now write a global function **reverseCircularQueue** which should reverse all the contents of the **CircularQueue**.

**CircularQueue** <**Type**> **reverseCircularQueue** (**CircularQueue** <**Type**> obj)

Instantiate several objects of **CircularQueue**, test the function of **reverseCircularQueue** on them and then display the reversed **CircularQueue** through showCircularQueue function.

**Remember:**

You are now allowed to use any other data structure other than Circular Queue in the above function.

## Task 3

Create a function **isPalindrome()** to check weather the contents of queue are palindrome or not.