|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***National University of Computer and Emerging Sciences, Lahore Campus*** | | | | |
| C:\Users\saif\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\final design.jpg | **Course:** | Data Structures | **Course Code:** | CS 201 |
| **Program:** | BS(CS) | **Semester:** | Fall 2019 |
| **Duration:** | 15 Minutes | **Total Marks:** | 10 |
| **Paper Date:** | 7 Oct 2018 | **Exam** | Quiz 2 |
| **Section:** | **A** |  |  |
| **Instruction/Notes:** | Solve the exam on this question paper. | | | |

**We are designing a mobile app and we want to remember last N operations performed in our app. That is when (N+1)th operation is performed by the user we will remove the first operation and remember 2nd till (N+1)th last operations. The stack is the natural choice to implement the above requirement. The stack is implemented using circular array.**

|  |  |  |
| --- | --- | --- |
| Stack after push(2), push(4) and  push(8) operation | Stack is full | Push(9) overwrites the first element in the stack. |

|  |
| --- |
| **template <class T>**  **void CircularStack {**  **... You can assume the constructors are implemented in the class**  **private:**  **T \* array;**  **int maxSize; // maximum size of the array**  **int currSize; // current size of the stack**  **int head; //contains the index of top element of the stack**  **};** |

**Write the Push and Pop functions for the class CircularStack.**

|  |  |
| --- | --- |
| **template <class T>**  **void CircularStack<T>::Push (const T&val){**  // updating head position  **head = ( head +1 ) % maxSize;** ---------------------- 2 mark  **array[head] = val;** -------------------- 0.5 mark  // if circular stack has spaces  **if (currSize != maxSize)** ---------------------- 1 mark  **currSize++;** -------------------- 0.5 mark  // if circular stack is filled then there is no need to update currSize because one element of stack is being added and one is being override  **}** | **template <class T>**  **T CircularStack<T>::Pop(){**  // if circular stack is not empty  **if ( currSize ! = 0)** --------------------------------- 1 mark  **{**  **T data = array[head]; -**-------------------------- 0.5 marks  **currSize --;** -------------------------------- 0.5 marks  // updating head  **if ( currSize == 0)** --------------------------------- 1 mark  **head = 0;**  **else**  **{**  **head -- ;** -------------------------------- 0.5 marks  // if circular stack pop (N+1)thelement  **if (head <0)** --------------------------------- 2 mark  **head =maxSize-1;**  ***OR instead of above three statement you can write***  **head = ( head - 1 + maxSize ) % maxSize;**  **}**  **return data;** -------------------------------- 0.5 marks  **} return null;**  **}** |