**Sri Sri University, Cuttack, Odisha.**

**Faculty of Science**

|  |  |
| --- | --- |
| **Program: B.Sc. – Computer Science, Data Science, & Environmental Science**  **(2020-23 Batch)**  **Subject Code/Subject Name: Data Structure Laboratory**  **Assignment –VIII** | |
| **Full Name of the Student:** | VINAYAK SANJAY CHAVAN |
| **Full Roll Number:** | BCS-011 |
| **Program:** | B.Sc. (Computer Sc.) / B.Sc. (Data Sc.) / B.Sc. (Env. Sc.) |
| **Date:** | 5th April, 2021 (10.00 AM – 12.00 Noon) |
| **Signature** |  |

|  |  |
| --- | --- |
| **All Questions are compulsory** | **Total Marks: 50** |

|  |  |
| --- | --- |
| **Question (s)** | **Maximum Marks** |
| 1. Defining structure of a linear queue, write functions for inserting an element, deleting an element, displaying elements, empty check, and full check of the queue. Also write a function to count the no of element present in the queue. Call all the functions from the main function based on an option.   *Note: You may declare all basic operations of queue in a header file and name it myqueue.h*  **Your code: header file:**  **#include<iostream>**  **#define max 4**  **using namespace std;**  **typedef struct queue{**  **int arr[max];**  **int back;//-1 rear**  **int front;//0**  **}QUEUE;**  **void intialize\_queue(QUEUE\*);**  **void insert(QUEUE\*,int);**  **int delete\_element(QUEUE\*);**  **int peek(QUEUE\*);**  **bool isEmpty\_queue(QUEUE\*);**  **bool isFull\_queue(QUEUE\*);**  **void display(QUEUE\*);**  **int count(QUEUE\*);**  **void intialize\_queue(QUEUE \*q1){**  **q1->front=0;**  **q1->back=-1;**  **}**  **void insert(QUEUE \*q1,int element){ //max is 4 as we defined**  **if(q1->back==max-1){//to check if queue has space or not**  **cout<<"Queue overflow"<<endl;**  **}else{**  **q1->back++;//we have to ennter the element from the back or rear towards front**  **q1->arr[q1->back]=element;**  **}**  **}**  **int delete\_element(QUEUE \*q1){**  **int element;**  **if(q1->front > q1->back){//if front is 0 and rear -1 then only it will work**  **cout<<"No element present in queue.";**  **exit(0);**  **}else{**  **element=q1->arr[q1->front];**  **q1->front++;**  **}**  **return element;**  **}**  **int peek(QUEUE \*q1){**  **int res;**  **if(q1->front>q1->back){//checking if there is no element ?**  **cout<<"No element present in queue.";**  **return -1;**  **}else{**  **res=q1->arr[q1->front];/\*always remeber if u want to check the peek**  **so always check with front**  **if u want to check the peek for last element then use rear\*/**  **}**  **return res;**  **}**  **void display(QUEUE \*q1){**  **if(q1->front > q1->back) {//to check if there is a element or not**  **cout<<"Queue is empty";**  **}else{**  **cout<<"The queue elements is: "<<endl;**  **for(int i=q1->front;i<=q1->back;i++){// we need to print the first element then 2nd 3rd and so on**  **//so start the array with front and stop it with rear or back**  **cout<<q1->arr[i]<<" ";**  **}**  **cout<<endl;**  **}**  **}**  **bool isFull\_queue(QUEUE \*q){**  **bool f\_nf;**  **if(q->back==max-1){//if queue has the element till the last element**  **f\_nf=true;**  **}else{**  **f\_nf=false;//if not then return false**  **}**  **return f\_nf;**  **}**  **bool isEmpty\_queue(QUEUE \*q){**  **bool e\_ne;**  **if(q->front > q->back) {//to check if we have elements or not**  **e\_ne=true;**  **}else{**  **e\_ne=false;**  **}**  **return e\_ne;**  **}**  **int count(QUEUE \*q){**  **int count=0;**  **for(int i=q->front;i<=q->back;i++){//to cout how many element we have in it**  **count++;**  **}**  **return count;**  **}**  **CODE:**  **#include<iostream>**  **#define max 4**  **#include "QUEUE.h"**  **void intialize\_queue(QUEUE\*);**  **void insert(QUEUE\*,int);**  **int delete\_element(QUEUE\*);**  **int peek(QUEUE\*);**  **bool isEmpty\_queue(QUEUE\*);**  **bool isFull\_queue(QUEUE\*);**  **void display(QUEUE\*);**  **int count(QUEUE\*);**  **int main(){**  **QUEUE q;**  **cout<<"the given size for the queue is "<<max<<endl;**  **int choice;**  **intialize\_queue(&q);**  **do{**  **cout<< "Main menu:"<<endl;**  **cout<< "1. INSERT"<<endl;**  **cout<< "2. DELETE"<<endl;**  **cout<< "3. PEEK"<<endl;**  **cout<< "4. FULL CHECK"<<endl;**  **cout<< "5. EMPTY CHECK "<<endl;**  **cout<< "6. DISPLAY "<<endl;**  **cout<< "7. NO OF ELEMENTS "<<endl;**  **cout<< "8. EXIT "<<endl;**  **cout<<"Enter the option :";**  **cin >>choice;**  **switch(choice){**  **case 1:**  **int element;**  **cout<<"enter the element in the queue ";//to enter the element by user**  **cin>>element;**  **insert(&q,element);**  **cout<<"element entered is "<<element<<endl;**  **break;**  **case 2:**  **int deleted;**  **deleted=delete\_element(&q);**  **cout<<"deleted element is "<<deleted<<endl;**  **break;**  **case 3:**  **cout<<"top element is "<<peek(&q)<<endl;**  **break;**    **case 4:**  **bool full\_check;**  **full\_check=isFull\_queue(&q);**  **if(full\_check==true){**  **cout<<"The queue is full."<<endl;**  **}else{**  **cout<<"The queue is not full."<<endl;**  **}**  **break;**    **case 5:**  **bool empty\_check;**  **empty\_check=isEmpty\_queue(&q);**  **if(empty\_check==true){**  **cout<<"The queue is empty."<<endl;**  **}else{**  **cout<<"The queue is not empty."<<endl;**  **}**  **break;**  **case 6:**  **display(&q);**  **break;**  **case 7:**  **int result;**  **result=count(&q);**  **cout<<"The total number of elements are: "<<result<<endl;**  **break;**  **case 8:**  **exit(0);**  **}**  **}**  **while(choice=8);**  **exit(0);**  **}**  **Screenshot of output:** | 20 |
| 1. Defining structure of a circular queue (without a counter), write functions for inserting, deleting and counting no of elements present in the queue. Write functions IsFull and IsEmpty also. Write main function to call them.   **Your code:HEADER FILE:**  **#include<iostream>**  **#define size 20**  **using namespace std;**  **typedef struct cqueue{**  **int arr[size];**  **int front;**  **int rear;**  **}CQUEUE;**  **void intia\_cqueue(CQUEUE\*);**  **void insert\_cqueue(CQUEUE\*,int);**  **void display\_cqueue(CQUEUE\*);**  **int del\_cqueue(CQUEUE\*);**  **int peek\_cqueue(CQUEUE\*);**  **bool is\_full\_cqueue(CQUEUE\*);**  **bool is\_empty\_cqueue(CQUEUE\*);**  **void intia\_cqueue(CQUEUE \*q){**  **q->front=-1;**  **q->rear=-1;**  **}**  **void insert\_cqueue(CQUEUE \*q,int n){**  **if((q->front==-1)&&(q->rear==-1)){**  **q->front=q->rear=0;**  **q->arr[q->rear]=n;**  **}else if((q->rear+1)%size==q->front){**  **cout<<"The circular queue is full.";**  **}else{**  **q->rear=(q->rear+1)%size;**  **q->arr[q->rear]=n;**  **}**  **}**  **int del\_cqueue(CQUEUE \*q){**  **int x=-1;**  **if((q->front==-1)&&(q->rear==-1)){**  **x=-99;**  **}else if(q->front==q->rear){**  **x=q->arr[q->front];**  **q->front=q->rear=-1;**  **}else{**  **x=q->arr[q->front];**  **q->front=(q->front+1)%size;**  **}**  **return x;**  **}**  **bool is\_full\_cqueue(CQUEUE\* q){**  **if((q->rear+1)%size==q->front){**  **return true;**  **}else{**  **return false;**  **}**  **}**  **void display\_cqueue(CQUEUE \*q){**  **int i=q->front;**  **if((q->front==-1)&&(q->rear==-1)){**  **cout<<"The queue is empty.";**  **}else{**  **cout<<"The circular queue elements are: ";**  **while(i!=q->rear){**  **cout<<q->arr[i]<<" ";**  **i=(i+1)%size;**  **}**  **cout<<q->arr[q->rear];**  **}**  **}**  **bool is\_empty\_cqueue(CQUEUE \*q){**  **if((q->front==-1)&&(q->rear==-1)){**  **return true;**  **}else{**  **return false;**  **}**  **}**  **int peek\_cqueue(CQUEUE \*q){**  **return q->arr[q->front];**  **}**  **CODE:** **#include<iostream>**  **#include"CIRCULAR QUEUE WITHOUT COUNTER.h"**  **using namespace std;**  **int count\_cqueue(CQUEUE\*);**  **int main(){**  **CQUEUE q;**  **int choice;**  **intia\_cqueue(&q);**  **do{**  **cout<< "Main menu:"<<endl;**  **cout<< "1. INSERT"<<endl;**  **cout<< "2. DELETE"<<endl;**  **cout<< "3. PEEK"<<endl;**  **cout<< "4. FULL CHECK"<<endl;**  **cout<< "5. DISPLAY"<<endl;**  **cout<< "6. EMPTY CHECK"<<endl;**  **cout<< "7. EXIT"<<endl;**  **cout<<"Enter the option :";**  **cin >>choice;**  **switch(choice){**  **case 1:**  **int element;**  **cout<<"enter the element in the stack ";**  **cin>>element;**  **insert\_cqueue(&q,element);**  **break;**  **case 2:**  **int deleted;**  **deleted=del\_cqueue(&q);**  **cout<<"element deleted is "<<deleted<<endl;**  **break;**  **case 3:**  **cout<<"top element is "<<peek\_cqueue(&q)<<endl;**  **break;**  **case 4:**  **display\_cqueue(&q);**  **break;//changed**  **case 5:**  **bool f\_nf;**  **f\_nf=is\_full\_cqueue(&q);**  **if(f\_nf==true){**  **cout<<"The queue is full.";**  **}else{**  **cout<<"The queue is not full.";**  **}**  **break;**  **case 6:**  **bool e\_ne;**  **e\_ne=is\_empty\_cqueue(&q);**  **if(e\_ne==true){**  **cout<<"The queue is empty.";**  **}else{**  **cout<<"The queue is not empty.";**  **}**  **break;**  **case 7:**  **exit(0);**  **}**  **}**  **while(choice=7);**  **exit(0);**  **int result;**  **result=count\_cqueue(&q);**  **cout<<"The total number of elements in the circular queue is: "<<result;**  **}**  **int count\_cqueue(CQUEUE \*q){**  **int res=0;**  **if(q->front<=q->rear){**  **res=(q->rear - q->front)+1;**  **}else if(q->front>q->rear){**  **res=size - (q->rear - q->front)+1;**  **}**  **return res;**  **}**  **Screenshot of output:** | 20 |
| 1. Defining structure of a circular queue (with a counter), write functions for inserting, deleting and counting no of elements present in the queue. Write functions IsFull() and IsEmpty() also. Write main function to call them.   **Your code: header file:**  **#include<iostream>**  **#define size 5**  **using namespace std;**  **typedef struct cqueue{**  **int arr[size];**  **int rear;**  **int front;**  **int count;**  **}CQUEUE;**  **void intia\_cqueue\_count(CQUEUE\*);**  **void insert\_cqueue\_count(CQUEUE\*,int);**  **void display\_cqueue\_count(CQUEUE\*);**  **int del\_cqueue\_count(CQUEUE\*);**  **int peek\_cqueue\_count(CQUEUE\*);**  **bool is\_full\_cqueue\_count(CQUEUE\*);**  **bool is\_empty\_cqueue\_count(CQUEUE\*);**  **void intia\_cqueue\_count(CQUEUE \*q){**  **q->count=0;**  **q->front=0;**  **q->rear=-1;**  **}**  **void insert\_cqueue\_count(CQUEUE \*q,int n){**  **if(q->count==size){**  **cout<<"The circular queue is full.";**  **}else{**  **q->rear=(q->rear+1)%size;**  **q->arr[q->rear]=n;**  **q->count++;**  **}**  **}**  **int del\_cqueue\_count(CQUEUE \*q){**  **int ele;**  **if(q->count==0){**  **return -99;**  **}else if(q->front==q->rear){**  **ele=q->arr[q->front];**  **q->count=0;**  **q->front=0;**  **q->rear=-1;;**  **}else{**  **ele=q->arr[q->front];**  **q->front=(q->front+1)%size;**  **q->count--;**  **}**  **return ele;**  **}**  **bool is\_full\_cqueue\_count(CQUEUE \*q){**  **if(q->count==size){**  **return true;**  **}else{**  **return false;**  **}**  **}**  **bool is\_empty\_cqueue\_count(CQUEUE \*q){**  **if(q->count==0){**  **return true;**  **}else{**  **return false;**  **}**  **}**  **void display\_cqueue\_count(CQUEUE \*q){**  **int i=q->front;**  **if((q->front==-1)&&(q->rear==-1)){**  **cout<<"The queue is empty.";**  **}else{**  **cout<<"The circular queue elements are: ";**  **while(i!=q->rear){**  **cout<<q->arr[i]<<" ";**  **i=(i+1)%size;**  **}**  **cout<<q->arr[q->rear];**  **}**  **}**  **int peek\_cqueue\_count(CQUEUE \*q){**  **int res;**  **res=q->arr[q->front];**  **return res;**  **}**  **Code:**  **#include<iostream>**  **#include"CIRCULAR QUEUE COUNTER.h"**  **using namespace std;**  **int main(){**  **CQUEUE q;**  **int choice;**  **intia\_cqueue\_count(&q);**  **do{**  **cout<< "Main menu:"<<endl;**  **cout<< "1. INSERT"<<endl;**  **cout<< "2. DELETE"<<endl;**  **cout<< "3. PEEK"<<endl;**  **cout<< "4. DISPLAY "<<endl;**  **cout<< "5.FULL CHECK"<<endl;**  **cout<< "6. EMPTY CHECK"<<endl;**  **cout<< "7. EXIT"<<endl;**  **cout<<"Enter the option :";**  **cin >>choice;**  **switch(choice){**  **case 1:**  **int element;**  **cout<<"enter the element in the stack ";**  **cin>>element;**  **insert\_cqueue\_count(&q,element);**  **break;**  **case 2:**  **int deleted;**  **deleted=del\_cqueue\_count(&q);**  **cout<<"element deleted is "<<deleted<<endl;**  **break;**  **case 3:**  **cout<<"top element is "<<peek\_cqueue\_count(&q)<<endl;**  **break;**  **case 4:**  **display\_cqueue\_count(&q);**  **break;//changed**  **case 5:**  **bool f\_nf;**  **f\_nf=is\_full\_cqueue\_count(&q);**  **if(f\_nf==true){**  **cout<<"The queue is full.";**  **}else{**  **cout<<"The queue is not full.";**  **}**  **break;**  **case 6:**  **bool e\_ne;**  **e\_ne=is\_empty\_cqueue\_count(&q);**  **if(e\_ne==true){**  **cout<<"The queue is empty.";**  **}else{**  **cout<<"The queue is not empty.";**  **}**  **break;**  **case 7:**  **exit(0);**  **}**  **}**  **while(choice=7);**  **exit(0);**  **}**  **Screenshot of output:** | 20 |
| 1. *Optional question:* Simulate a queue of integers using two stacks of integers writing insert and delete operations of queue using push and pop functions of stack.   **Your code:**  **Screenshot of output:** | Practice Only  (Optional) |