**CSC221: DATA STRUCTURES & ALGORITHMS**

**BSCS 3*B***

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | LAB | | **06** | | An array based implementation of QUEUE with the help of algorithm for following functions   * enQueue() * deQueue() * displayQueue() |



**Submitted By:**

* Ahsan Ghaffar
* Reg: 48411
* BS(CS)\_3B

**Submitted to:**

* Miss Ambreen Akram(AA)

**Submission Date:**

[02/04/2018]

**DEPARTMENT OF COMPUTER SCIENCE**

**BAHRIA UNIVERSITY, KARACHI CAMPUS**

**LAB TASKS:**

|  |  |  |
| --- | --- | --- |
| 1 | Write a program as follows for QUEUE   |  | | --- | | ----------------------Array based implementation of QUEUE-----------------------------------   1. Enqueue an element to queue 2. Dequeue an element from queue 3. Display all 4. Exit   -------------------------------------------------------------------------------------------------------------  Please Enter Your Choice: | |

**SORCECODE:**

#include <iostream>

using namespace std;

void enque(int value);

void deque(int value);

bool isfull();

bool isempty();

void display();

int findsize();

int front = -1;

int rear = -1;

int const siz = 6;

int arr[siz];

int main()

{

int c, a, b;

char d;

do

{

cout << "\n1.Enqueue\n2.Dequeue\n3.Display\n4.Size\nEnter any of the above:" << endl;

cin >> c;

if (c == 1)

{

cout << "\nEnter the insertion value: " << endl;

cin >> a;

enque(a);

}

else if (c == 2)

{

cout << "\nEnter the value you want to delete: " << endl;

cin >> b;

deque(b);

}

else if (c == 3)

{

display();

}

else if (c == 4)

{

cout <<"\nThe Size of Queue is: "<< findsize();

}

cout << "\nEnter Y to continue and N to exit:" << endl;

cin >> d;

} while (d != 'n');

system("pause");

return 0;

}

bool isfull()

{

bool flag = false;

if (rear == siz - 1)

{

cout << "\nQueue is Full:" << endl;

flag = true;

}

else

{

flag = false;

}

return flag;

}

bool isempty()

{

bool flag = false;

if (front == -1 && rear == -1)

{

cout << "\nQueue is Empty:" << endl;

flag = true;

}

else

{

flag = false;

}

return flag;

}

void enque(int value)

{

if (isfull())

{

cout << "\n" << endl;

}

else if (front == -1 && rear == -1) {

front++;

rear++;

arr[rear] = value;

}

else {

rear++;

arr[rear] = value;

}

}

void deque(int value) {

if (isempty())

{

cout << "\n" << endl;

}

else if (front == rear) {

front--;

rear--;

}

else

{

front = front++;

}

}

void display()

{

if (isempty())

{

cout << "\nQueue is empty" << endl;

}

cout << "\nThe Queue Values are:"<<endl;

for (int i = front; i <= rear; i++)

{

cout << "\n" << arr[i];

}

}

int findsize()

{

int size1 = rear - front;

size1 = size1++;

return size1;

}

|  |  |
| --- | --- |
| 2 | Test the program using the following procedure: QUEUE of size N=6  Call enQueue(5)  Call enQueue(2)  Call enQueue(3)  Call deQueue ()  call deQueue()  Call enQueue(6)  Call enQueue(3)  Call Display() |

**SCREENSHOT:**

