**Name: Tayyab Zain**

**Reg No: 4293-FBAS/BSCS/F20**

**Subject:DSA**

**Submitted to: Sir Muhsaraf**

**Question No 1:**

**Consider the following statements: queueType queue; int x, y; Show what is output by the following segment of code: x = 4; y = 5; queue.addQueue(x); queue.addQueue(y); x = queue.front(); queue.deleteQueue(); queue.addQueue(x + 5); queue.addQueue(16); queue.addQueue(x); queue.addQueue(y - 3); cout << "Queue Elements: "; while (!queue.isEmptyQueue()) { cout << queue.front() << " "; queue.deleteQueue(); } cout << endl;**

**Output:**

5 9 16 4 2

**Question No 2:**

**Consider the following statements: stackType stack; queueType queue; int x; Suppose the input is: 15 28 14 22 64 35 19 32 7 11 13 30 -999 Show what is written by the following segment of code: stack.push(0); queue.addQueue(0); cin >> x; while (x != -999) { switch (x % 4) { case 0: stack.push(x); break; case 1: if (!stack.isEmptyStack()) { cout << "Stack Element = " << stack.top() << endl; stack.pop(); } else cout << "Sorry, the stack is empty." << endl; break; case 2: queue.addQueue(x); break; case 3: if (!queue.isEmptyQueue()) { cout**

**<< "Queue Element = " << queue.front() << endl; queue.deleteQueue(); } else cout << "Sorry, the queue is empty." << endl; break; } //end switch cin >> x; } //end while cout << "Stack Elements: "; while (!stack.isEmptyStack()) { cout << stack.top() << " "; stack.pop(); } cout << endl; cout << "Queue Elements: "; while (!queue.isEmptyQueue()) { cout << queue.front() << " "; queue.deleteQueue(); } cout << endl;**

**Output:**

Sorry the queue is empty

Stack:

28 64

Queue:

30

Sorry the queue empty

Sorry the queue empty

Output:

"Stack Elements: " 28

64

"Queue Elements: " 30

**3. What does the following function do?**

**void mystery(queueType& q)**

**{**

**stackType s;**

**while (!q.isEmptyQueue())**

**{**

**s.push(q.front());**

**q.deleteQueue();**

**}**

**while (!s.isEmptyStack()) {**

**q.addQueue(2 \* s.top());**

**s.pop();**

**}**

**}**

**Ans:**

The function mystery reverses the elements of a queue and also doubles the values

of the queue elements.

The function mystery reverses the elements of a queue and also doubles the values

of the queue elements.

The function mystery reverses the elements of a queue and also doubles the values

of the queue elements.

The function mystery reverses the elements of a queue and also doubles the values

of the queue elements.

The function **mystery** reverse the queue and also doubles the value of the queue element

**5. What is the output of the following program segment?**

**Output:**

10

20 40 20 5 3

20 3

For all the upcoming questions

Assuming queueFront points to the first element of the array ad queueRear points to the last element of the array

**6. Suppose that queue is a queueType object and the size of the array implementing queue is 100. Also, suppose that the value of queueFront is 50 and the value of queueRear is 99.**

**a. What are the values of queueFront and queueRear after adding an element to queue?**

**b. What are the values of queueFront and queueRear after removing an element from queue?**

**Ans:**

**(a):**

Value of queueFront = 50

Value of queueRear = new added element

**(b):**

Value of queueFront = next element in the queue

Value of queueRear = 99

**7. Suppose that queue is a queueType object and the size of the array implementing queue is 100. Also, suppose that the value of queueFront is 99 and the value of queueRear is 25.**

**a. What are the values of queueFront and queueRear after adding an element to queue?**

**b. What are the values of queueFront and queueRear after removing an element from queue?**

**Ans:**

**(a):**

queueFront = 99

queueRear = new Added element

**(b):**

queueFront = next element in the queue

queueRear = 25

**8. Suppose that queue is a queueType object and the size of the array implementing queue is 100. Also, suppose that the value of queueFront is 25 and the value of queueRear is 75.**

**a. What are the values of queueFront and queueRear after adding an element to queue?**

**b. What are the values of queueFront and queueRear after removing an element from queue?**

**Ans:**

**(a):**

queueFront = 25

queueRear = new added element

**(b):**

queueFront = next element in the queue

queueRear = 75

**9. Suppose that queue is a queueType object and the size of the array implementing queue is 100. Also, suppose that the value of queueFront is 99 and the value of queueRear is 99.**

**a. What are the values of queueFront and queueRear after adding an element to queue?**

**b. What are the values of queueFront and queueRear after removing an element from queue?**

**Ans:**

**(a):**

queueFront = 99

queueRear = new element

**(b):**

queueFront = next element in the queue

queueRear = 99

**10. Suppose that queue is implemented as an array with the special reserved slot, as described in this chapter. Also, suppose that the size of the array implementing queue is 100. If the value of queueFront is 50, what is the position of the first queue element?**

**Ans:**

Position value is 51.

**11. Suppose that queue is implemented as an array with the special reserved slot, as described in this chapter. Suppose that the size of the array implementing queue is 100. Also, suppose that the value of queueFront is 74 and the value of queueRear is 99.**

**a. What are the values of queueFront and queueRear after adding an element to queue?**

**b. What are the values of queueFront and queueRear after removing an element from queue? Also, what is the position of the removed queue element?**

**Ans:**

**(a):**

QueueFront = 74

QueueRear = new added element

**(b):**

QueueFront = next element in the array

QueueRear = 99