QUESTION 1

1 marks

Which one of the following is an application of Queue Data Structure?

- When a resource is shared among multiple consumers.
- When data is transferred asynchronously (data not necessarily received at same rate as sent) between two processes
- Load Balancing
- [✓ All of the above]

QUESTION 2

1 marks

How many stacks are needed to implement a queue. Consider the situation where no other data structure like arrays, linked list is available to you.

- 1
- [✓ 2]
- 3
- 4

QUESTION 3

1 marks

A priority queue can efficiently implemented using which of the following data structures? Assume that the number of insert and peek (operation to see the current highest priority item) and extraction (remove the highest priority item) operations are almost same.

- ○ Array
- ○ Linked List
- [✓ Heap Data Structures like Binary Heap, Fibonacci Heap]
- ○ None of the above

QUESTION 4

1 marks

Which of the following is true about linked list implementation of queue?

- ○ In push operation, if new nodes are inserted at the beginning of linked list, then in pop operation, nodes must be removed from end.
- ○ In push operation, if new nodes are inserted at the end, then in pop operation, nodes must be removed from the beginning.
- [✓ Both of the above]
- ○ None of the above

QUESTION 5

1 marks

Suppose a circular queue of capacity (n – 1) elements is implemented with an array of n elements. Assume that the insertion and deletion operation are carried out using REAR and FRONT as array index variables, respectively. Initially, REAR = FRONT = 0. The conditions to detect queue full and queue empty are

- [✓ Full: (REAR+1) mod n == FRONT, empty: REAR == FRONT]
- ○ Full: (REAR+1) mod n == FRONT, empty: (FRONT+1) mod n == REAR
- ○ Full: REAR == FRONT, empty: (REAR+1) mod n == FRONT
- ○ Full: (FRONT+1) mod n == REAR, empty: REAR == FRONT

QUESTION 6

1 marks

An implementation of a queue Q, using two stacks S1 and S2, is given below:

C/C++ Code

```
void insert(Q, x) {

  push (S1, x);

}



void delete(Q){

  if(stack-empty(S2)) then

    if(stack-empty(S1)) then {

       print("Q is empty");

       return;

    }

    else while (!(stack-empty(S1))){

       x=pop(S1);

       push(S2,x);

    }

  x=pop(S2);

}
```

Let n insert and m (<=n) delete operations be performed in an arbitrary order on an empty queue Q. Let x and y be the number of push and pop operations performed respectively in the process. Which one of the following is true for all m and n?

- [✓ n+m <= x < 2n and 2m <= y <= n+m]
- n+m <= x < 2n and 2m<= y <= 2n
- 2m <= x < 2n and 2m <= y <= n+m
- 2m <= x <2n and 2m <= y <= 2n

QUESTION 7

1 marks

Consider the following operation along with Enqueue and Dequeue operations on queues, where k is a global parameter.

```
MultiDequeue(Q){

   m = k

   while (Q is not empty and m  > 0) {

      Dequeue(Q)

      m = m - 1

   }

}
```

What is the worst case time complexity of a sequence of n MultiDequeue() operations on an initially empty queue? (GATE CS 2013) (A) $Theta(n)$(B) $Theta(n+k)$(C) $Theta(nk)$ (D) $Theta(n^2)$

[✓ A] B

C D

QUESTION 8

1 marks

Consider the following pseudo code. Assume that IntQueue is an integer queue. What does the function fun do?

C/C++ Code

```
void fun(int n)
{
  IntQueue q = new IntQueue();
  q.enqueue(0);
  q.enqueue(1);
  for (int i = 0; i < n; i++)
  {
    int a = q.dequeue();
    int b = q.dequeue();
    q.enqueue(b);
    q.enqueue(a + b);
    print(a);
  }
}
```

- ○ Prints numbers from 0 to n-1
- ○ Prints numbers from n-1 to 0
- [✓ Prints first n Fibonacci numbers]
- ○ Prints first n Fibonacci numbers in reverse order.

QUESTION 9

1 marks

Which of the following is NOT a common operation in a queue data structure?

- ○ Enqueue
- ○ Dequeue
- ○ Peek
- [✓ Shuffle]

QUESTION 10

1 marks

A Priority-Queue is implemented as a Max-Heap. Initially, it has 5 elements. The level-order traversal of the heap is given below: 10, 8, 5, 3, 2 Two new elements "1' and "7' are inserted in the heap in that order. The level-order traversal of the heap after the insertion of the elements is:

- ○ 10, 8, 7, 5, 3, 2, 1
- ○ 10, 8, 7, 2, 3, 1, 5
- ○ 10, 8, 7, 1, 2, 3, 5
- [✓ 10, 8, 7, 3, 2, 1, 5]