QUEUE

Solutions

1. A linear list of elements in which deletion can be done from one end (front) and insertion can take place only at the other end (rear) is known as a ?

(a) Queue (b) Stack
(c) Tree (d) Linked list

Solution: Option (a)

2. The data structure required for Breadth First Traversal on a graph is?

(a) Stack (b) Array
(c) Queue (d) Tree

Solution: Option (c)

3. Let the following circular queue can accommodate maximum six elements with the following data:

front = 2 rear = 4
queue = _______; L, M, N, ___, ___

What will happen after ADD O operation takes place?

a) front = 2 rear = 5
queue = ______; L, M, N, O, ___

b) front = 3 rear = 5
queue = L, M, N, O, ___

c) front = 3 rear = 4
queue = ______; L, M, N, O, ___

d) front = 2 rear = 4
queue = L, M, N, O, ___

Solution: Option (a)

4. A queue is a ?

(a) FIFO (First In First Out) list (b) LIFO (Last In First Out) list
(c) Ordered array (d) Linear tree

Solution: Option (a)

Explanation:

Elements are inserted from rear and they are dequeued from the front. Hence the first element to enter is the first element to be dequeued.

5. If the elements "A", "B", "C" and "D" are placed in a queue and are deleted one at a time, in what order will they be removed?

(a) ABCD (B) DCBA
(c) DCAB (d) ABCD

Solution: Option (a)

Explanation:

Queue follows FIFO.

6. In linked list implementation of a queue, where does a new element be inserted?

(a) At the head of link list
(b) At the tail of the link list
(c) At the centre position in the link list
(d) None

Solution: Option (b)

7. Suppose implementation supports an instruction REVERSE, which reverses the order of elements on the stack, in addition to the PUSH and POP instructions. Which one of the following statements is TRUE with respect to this modified stack?

(a) A queue cannot be implemented using this stack.

(b) A queue can be implemented where ENQUEUE takes a single instruction and DEQUEUE takes a sequence of two instructions.

(c) A queue can be implemented where ENQUEUE takes a sequence of three instructions and DEQUEUE takes a single instruction.

(d) A queue can be implemented where both ENQUEUE and DEQUEUE take a single instruction each.

Solution: Option (c)

Explanation:
To DEQUEUE an item, simply POP.

To ENQUEUE an item, we can do following 3 operations:
1) REVERSE
2) PUSH
3) REVERSE

8. Following is C like pseudo code of a function that takes a Queue as an argument, and uses a stack S to do processing.

```
void fun(Queue *Q)
{
    Stack S;  // Say it creates an empty stack S
     // Run while Q is not empty
    while (!isEmpty(Q))
{
        // deQueue an item from Q and push the dequeued item to S
        push(&S, deQueue(Q));
}
       // Run while Stack S is not empty
       while (!isEmpty(&S))
   {
      // Pop an item from S and enqueue the poppped item to Q
      enQueue(Q, pop(&S));
     }
}
```

What does the above function do in general?

(a) Removes the last from Q
(b) Keeps the Q same as it was before the call
(c) Makes Q empty
(d) Reverses the Q

Solution: Option (d)

Explanation:
The function takes a queue Q as an argument. It dequeues all items of Q and pushes them to a stack S.

Then pops all items of S and enqueues the items back to Q. Since stack is LIFO order, all items of queue are reversed.

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

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

Solution: Option (d)

10. 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.

(a) 1 (b) 2
(c) 3 (d) 4

Solution: Option (b)

Explanation:

Let stack1 and stack2 be the two stacks using which we shall implement queue.

Here is psuedo code

enQueue(q,x)

1) While stack1 is not empty , push everything from stack1 to stack2

2) Push x to stack1

3) Push everything back to stack1.

deque(q)

1) If stack1 is empty then underflow.
2) Pop an item from stack1 and return it

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

(a) 1	(b) 2
(c) 3	(d) 4

Solution: Option (b)

Explanation: Let q1 and q2 be two queues to implement stack.

The pseudo code for operations push and pop are

```
push(s,x)
  1) Enqueue x to q2
  2) One by one dequeue everything from q1 and enqueue to q2.
  3) Swap the names of q1 and q2
pop(s)
  1) deque an item from q1 and return it.
```

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

(a) In push operation, if new nodes are inserted at the beginning of linked list, then in pop operation, nodes must be removed from end.

(b) In push operation, if new nodes are inserted at the end, then in pop operation, nodes must be removed from the beginning.

(c) Both of the above

(d) None of the above

Solution: Option (c)

Explanation:
To keep the First In First Out order, a queue can be implemented using linked list in any of the given two ways.

13. 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

(a) Full: (REAR+1) mod n == FRONT, empty: REAR == FRONT
(b) Full: (REAR+1) mod n == FRONT, empty: (FRONT+1) mod n == REAR
(c) Full: REAR == FRONT, empty: (REAR+1) mod n == FRONT
(d) Full: (FRONT+1) mod n == REAR, empty: REAR == FRONT

Solution: Option (a)

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

```
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?

(a) n+m <= x < 2n and 2m <= y <= n+m
(b) n+m <= x < 2n and 2m<= y <= 2n
(c) 2m <= x < 2n and 2m <= y <= n+m
(d) 2m <= x <2n and 2m <= y <= 2n

Solution: Option (a)

Explanation:
The order in which insert and delete operations are performed matters here.

The best case: Insert and delete operations are performed alternatively. In every delete operation, 2 pop and 1 push operations are performed. So, total m+ n push (n push for insert() and m push for delete()) operations and 2m pop operations are performed.

The worst case: First n elements are inserted and then m elements are deleted. In first delete operation, n + 1 pop operations and n push operation are performed. Other than first, in all delete operations, 1pop operation is performed. So, total m + n pop operations and 2n push operations are performed (n push for insert() and n push for delete())

15. 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(n^2)
(d) theta(nk)

Solution: Option (a)

Explanation:

Since the queue is empty initially, the condition of while loop never becomes true. So the time complexity is theta(n).

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

```
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);
      ptint(a);
     }
}
```

(a) Prints numbers from 0 to n-1
(b) Prints numbers from n-1 to 0
(c) Prints first n Fibonacci numbers
(d) Prints first n Fibonacci numbers in reverse order

Solution: Option (c)

Explanation:
The function prints first n Fibonacci Numbers. Note that 0 and 1 are initially there in q. In every iteration of loop sum of the two queue items is enqueued and the front item is dequeued.

17. A circular queue is implemented using an array of size 10. The array index starts with 0, front is 6, and rear is 9. The insertion of next element takes place at the array index.

(a) 0 (b) 7

(c) 9 (d) 10

Solution: Option (a)