Data Structure Tutorial 2

1. What does the following piece of code do?

```
public Object function()
{
      if(this.isEmpty())
          return -999;
     else
     {
          Object high;
          high = q[front];
          return high;
     }
}
```

a) Dequeue b) Enqueue c) Return the front element d) None of the mentioned

2. What does the following function do?

```
public Object some_func()throws emptyStackException
{
        if(isEmpty())
                throw new emptyStackException("underflow");
        return first.getEle();
}
```

a) pop
b) delete the top-of-the-stack element
c) retrieve the top-of-the-stack element
d) none of the mentioned

3. Consider the following doubly linked list: head-1-2-3-4-5-tail. What will be the list after performing the given sequence of operations?

```
Node temp = new Node(6,head,head.getNext());
Node temp1 = new Node(0,tail.getPrev(),tail);
head.setNext(temp);
temp.getNext().setPrev(temp);
tail.setPrev(temp1);
temp1.getPrev().setNext(temp1);
```

a) head-0-1-2-3-4-5-6-tail
b) head-1-2-3-4-5-6-tail
c) head-6-1-2-3-4-5-0-tail
d) head-0-1-2-3-4-5-tail

4. Consider the following doubly linked list: head-1-2-3-4-5-tail. What will be the list after performing the given sequence of operations?

```
Node temp = new Node(6,head,head.getNext());
head.setNext(temp);
temp.getNext().setPrev(temp);
Node temp1 = tail.getPrev();
tail.setPrev(temp1.getPrev());
temp1.getPrev().setNext(tail);
```

a) head-6-1-2-3-4-5-tail
b) head-6-1-2-3-4-tail
c) head-1-2-3-4-5-6-tail
d) head-1-2-3-4-5-tail

5. What does the following function do for a given Linked List with first node as head?

```
void fun1(struct node* head)
{
     if(head == NULL)
          return;
     fun1(head->next);
     printf("%d  ", head->data);
}
```

a) Prints all nodes of linked lists　　b) Prints all nodes of linked list in reverse order
c) Prints alternate nodes of Linked List　　d) Prints alternate nodes in reverse order

6. Let S be a stack of size n >= 1. Starting with the empty stack, suppose we push the first n natural numbers in sequence, and then perform n pop operations. Assume that Push and Pop operation take X seconds each, and Y seconds elapse between the end of one such stack operation and the start of the next operation. For m >= 1, define the stack-life of m as the time elapsed from the end of Push(m) to the start of the pop operation that removes m from S. The average stack-life of an element of this stack is
a) n(X+ Y)　　b) 3Y + 2X　　c) n(X + Y)-X　　d) Y + 2X

7. A single array A[1..MAXSIZE] is used to implement two stacks. The two stacks grow from opposite ends of the array. Variables top1 and top2 (top 1< top 2) point to the location of the topmost element in each of the stacks. If the space is to be used efficiently, the condition for “stack full” is:
a) (top1 = MAXSIZE/2) and (top2 = MAXSIZE/2+1)　　b) top1 + top2 = MAXSIZE
c) (top1= MAXSIZE/2) or (top2 = MAXSIZE)　　d) top1= top2 -1

8. Consider n elements that are equally distributed in k stacks. In each stack, elements of it are arranged in ascending order (min is at the top in each of the stack and then increasing downwards). Given a queue of size n in which we have to put all n elements in increasing order. What will be the time complexity of the best known algorithm?
a) O(n)　　b) O(n^2)　　c) O(n log (n))　　d) O(nk)

9. Given an efficient circular array-based queue Q capable of holding 10 objects. Show the final contents of the array after the following code is executed:

```
for(int k = 1; k <= 7; k++)
      Q.enqueue(k);
for(int k = 1; k <= 4; k++)
      Q.enqueue(Q.dequeue());
```

Answer : ________________

10. Assume that you have an empty circular queue (array-based implementation) which can hold only four elements. What are the array indexes of the back and the front elements after executing this series of queue operations? (indexes start with 0) enqueue(“a”), enqueue(“b”), getFront(), enqueue(“c”), enqueue(“d”), dequeue(), dequeue(), dequeue(), enqueue(“a”), enqueue(“b”), enqueue(“c”), dequeue(), getFront(), enqueue(“d”), dequeue()
front index = _________1____ back index = ___3_________ output = ___________

11. 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

12. A Priority-Queue is implemented as a Max-Heap. Initially, it has 5 elements. The levelorder 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:
a) 10, 8, 7, 5, 3, 2, 1 b) 10, 8, 7, 2, 3, 1, 5 c) 10, 8, 7, 1, 2, 3, 5 d) 10, 8, 7, 3, 2, 1, 5

13. 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?
a) Θ(n) b) Θ(n+k) c) Θ(n*k) d) Θ(n^{2})

14. Suppose you are given an implementation of a queue of integers. The operations that can be performed on the queue are:
1) isEmpty (Q) — returns true if the queue is empty, false otherwise.
2) delete (Q) — deletes the element at the front of the queue and returns its value.
3) insert (Q, i) — inserts the integer i at the rear of the queue.
Consider the following function:

```
void f (queue Q)
{
     int i ;
     if (!isEmpty(Q))
     {
          i = delete(Q);
          f(Q);
          insert(Q, i);
     }
}
```

What operation is performed by the above function f ?
a) Leaves the queue Q unchanged
b) Reverses the order of the elements in the queue Q
c) Deletes the element at the front of the queue Q and inserts it at the rear keeping the other elements in the same order
d) Empties the queue Q

Programming [stack]:-
1. Write a program to Implement stack using array.
2. Write a program to Implement stack using linked list (singly/doubly).
3. Write a program to Implement stack using queues. You have operations of queue (enqueue, dequeue) and you have to implement operations of stack (push, pop, etc…)
4. Write a program to convert a given infix expression to post fix expression.
5. Write a program to reverse a stack using recursion.
6. Write a program to check whether the given expression contains balanced paranthesis or not.

7. Write a code to evaluate an arithmetic expression containing numbers and binary operations (+, -, *, and /). The expression can contain parentheses. The expression will be provided in the form of infix notation.

Programming [Queue]

1. Write a program to Implement a queue data structure using array.
2. Write a program to Implement a queue data structure using linked list (singly/doubly).
3. Write a program to implement queue using stacks. You have only stack operations push, pop and you need to implement queue operation like enqueue dequeue etc.
4. Write a program to Implement min-heap using array. implement create heap, delete min, insert element etc..
5. Write a program to check whether a given Binary Tree is Complete or not. for this you need to use Breadth first search using queue.

Solution : 1) c. 2) c.3) c. 4) b. 5) b. 6) c. 7) d. 8) d. 11) a. 12) d. 13) a. 14) b.