**BRAC University (Department of Computer Science and Engineering)**
**Summer 2022 Semester**

**CSE-220 (Data Structure)**
**Section 14**

**Quiz 3**
**17 August, 2022**

**Student ID:** **Full Marks: 30**
**Name:** **Duration: 40 minutes**

---

**[No extra sheet will be provided. Write your answer to the questions in this answer script.]**
**[Marks allocated to each question is given in the statement of corresponding question.]**

## Answer all the questions

1. Suppose, you are given a list $l$ (You do not need to take any input). Write a recursive function that reverses the list $l$.

| | |
|---|---|
| Sample Input:<br>[1,2,3,4,5]<br>Sample Output:<br>[5,4,3,2,1] | Sample Input:<br>['c', 's' ,'e' ,2, 2, 0]<br>Sample Output:<br>[0, 2, 2, 'e', 's', 'c'] |

**[6]**

```
def reverse_list(l): #wrapper method
  return reverse_list_recursion_method1(l,0,len(l)-1)

def reverse_list_recursion_method1(l,start,end):
  if start>end:
    return l

  l[start],l[end] = l[end],l[start]
  return reverse_list_recursion_method1(l,start+1, end-1)

def reverse_list_iterative(l): #equivalent iterative method
  start = 0
  end = len(l)-1
  while start<=end:
    l[start],l[end] = l[end],l[start]
    start += 1
    end -= 1
  return l

a_list = ['c', 's', 'e' ,2, 2, 0]
ans = reverse_list(a_list)
print(ans)
```

2. Suppose, you have been given a non-negative integer $n$. Write a recursive function that prints all the **Hailstone Numbers** from $n$ upto 1. Hailstone number follows Collatz conjecture. According to Collatz Conjecture, for any integer $a$

$$a_n = \begin{cases} \frac{1}{2} a_{n-1} & \text{for } a_{n-1} \text{ even} \\ 3\,a_{n-1} + 1 & \text{for } a_{n-1} \text{ odd} \end{cases}$$

Irrespective of the choice of $n$, the sequence will eventually converge to 1.

| | |
|---|---|
| Sample Input:<br>6<br>Sample Output:<br>6, 3, 10, 5, 16, 8, 4, 2, 1 | Sample Input:<br>13<br>Sample Output:<br>13, 40, 20, 10, 5, 16, 8, 4, 2, 1 |

**[7]**

```
[ ]  def hailstone_number(n):
       print(n,end=' ')
       if n == 1: #base case: sequence will converge to 1 always
         return

       if n%2 == 0:
         n = n//2
       else:
         n = 3*n+1
       hailstone_number(n)

     n = 13
     hailstone_number(n)
```

```
13 40 20 10 5 16 8 4 2 1
```

3. Suppose you are given an object q of class Queue and a number k. Queue class supports all the basic operations like enqueue(n), dequeue() and peek() and has 2 attributes, front and size. **You do not need to implement the Queue class**. Your task is to rotate the q, k times.

| | |
|---|---|
| Sample Input:<br>[1, 2, 3, 4, 5, 6]<br>3<br>Sample Output:<br>[4, 5, 6, 1, 2, 3] | Sample Input:<br>['a', 'b', 'c', 'd', 'e']<br>2<br>Sample Output:<br>['c', 'd', 'e', 'a', 'b'] |

```
def rotate_queue(q,k):
        #your code goes here
```

Hint: enqueue(n) inserts n at the end of the queue. dequeue() removes the first element of the queue. You have to use both for this task. **[6]**

```
def rotate_queue(q,k):
    for i in range(k):
        temp = q.dequeue()
        q.enqueue(temp)
```

4. 
```
def change(l,i):
    if i==len(l):
        return l
    l[i] = 5*l[i]
    change(l,i+1)
    return l
a = [10,20,30,40]
l = change(a,0)
print('l = ',l,',a = ',a)
```

```
def change(l,i):
  if i==len(l):
    return l

  l[i] = 5*l[i]
  change(l,i+1)
  return l

a = [10,20,30,40]
l = change(a,0)
print(l,a)
```

```
[50, 100, 150, 200] [50, 100, 150, 200]
```

What will be the output?

a. l = [50, 100, 150, 200] ,a = [50, 100, 150, 200]
b. l = [50, 100, 150, 200] ,a = [10,20,30,40]
c. l = [10,20,30,40] ,a = [10,20,30,40]
d. l = [10,20,30,40] ,a = [50, 100, 150, 200] **[3]**

5. 
```
def change(l,i):
    if i==len(l):
        return l
    l[i] = 5*l[i]
    change(l,i+1)
    return l
a = [10,20,30,40]
l = change(a[:],0)
print('l = ',l,',a = ',a)
```

```
def change(l,i):
    if i==len(l):
        return l
    l[i] = 5*l[i]
    change(l,i+1)
    return l

a = [10,20,30,40]
l = change(a[:],0)
print('l = ',l,',a = ',a)
```

```
l =  [50, 100, 150, 200] ,a =  [10, 20, 30, 40]
```

a. l = [50, 100, 150, 200] ,a = [50, 100, 150, 200]
b. l = [50, 100, 150, 200] ,a = [10,20,30,40]
c. l = [10,20,30,40] ,a = [10,20,30,40]
d. l = [10,20,30,40] ,a = [50, 100, 150, 200] **[3]**

6. The priority of a (PQ) priority queue is given serially in a descending order:
→ Even numbers get more priority than odd numbers.
→ Lower numbers get more priority than higher numbers.
Using the given priorities, how the given numbers will be inserted in the PQ?
4, 23, 7, 65, 12, 2 **[4]**

2 4 12 7 23 65