LAB-04

2100030059
B. Revathi
S-12

Stack and Queue Applications :-

PRE LAB:

1. Game of Numbers :-

```
import java.io.BufferedReader;
import java.util.Scanner;
class Test class {
    Public static Void main (String[] args)
{
Buffered Reader br= new Buffered Reader
                    newinput stream Reader(System.in));
int N= integer Parse int (br.readline ));
 long [] arr = new long [10];
int [] next gender = new int [N];
  next lower [N-1] =1;
 for (int i=0; i<N; i++) {
arr[i]= long.parse long (br.read line());
}
for (int i=0; i<N; i++) {
   if (arr(i)<arr[j]) {
    next Greater [i]= i;
    break;
  }
  if (j==N-1)
    next Greater [i]=-1;
  }
}
  for( int i=0; i<N; i++) {
  for(int j= j+1; j<N; j++) {
   if (arr[i] > arr [j]) {
     next lower[i]=j;
     break;
  } if(j==N-1)
```

2100030059
B. Revathi
S-12

```
        nextLower[i] = -1;
    }
}
StringBuilder sb = new StringBuilder(N*2);
    for(int i=0; i<N; i++) {
if (nextGrender[i] != -1 && nextLower[nextGrender[i]] != -1)
 sb.append(arr[nextLower[nextGreater[i]]].append(" ");
 else
    sb.append("-1").append(" ");
    }
    System.out.println(sb);
}}
```

Output:-

1 H H 4 -12 -1 -1.

2. Tower of Hanoi:-

```
import java.util.Scanner;
import java.math.*;
Class Demo {
    Static Void towerofHanoi(int n,
    char from_rod, char to_rod, char
                        aux_rod).

    {
    if(n==0)
    {
    return;
    }
towerofHanoi(n-1, from_rod, aux_rod, to
                                    ro
System.out.println("move disk"+n+"promrod"+from_rod+
                        "torod"+to_rod);
towerofHanoi(n-1, aux_rod, to_rod, from_rod
}
Public Static Void main(String[] args) {
        int n=4;
        towerofHanoi(n,'A','C','B');
towerofHanoi(n-1, from_rod, aux_rod,t
System.out.println("move disk"+n+"from rod"+from_rod+to
                        to_rod);
```

```
tower of Hanoi(n-1, aux-rod, to-rod, from-rod);
}
Public static void main (String[] args)
{
    int n=4
    tower of Hanoi (n, 'A', 'B', 'C');
}
}
```

Output:-

disk 1 moved from A to C
disk 2 moved from A to B

IN-LAB:-

```
import java.util.Stack;
Public class Test
{ static int evaluate postfix (String exp)
{ Static <Integer> static = newstack <>();
for (int i=0; i<exp.length(); i++)
{ char c=exp.charAt(i);
    if (Character.isDigit()){
      stack.push (c-'0');
  else {
      int val1 =stack.pop();
      int val2 = stack.pop();
      Switch (c)
      { case '+':
          stack.push (val2+val1);
          break;
        Case '-':
          stack.Push (val2-val1);
          break;
        Case '*':
          stack.push (val2 *val1);
          break;
```

9100030059
B. Revathi
S-12

```
case '/':
    stack.push(val2/val1);
        break;
    }
  }
 return stack.pop()
}
Public static void main (String[] args) {
        String exp = "231*+9-";
System.out.println("Postfix evaluation :"+evaluate
                                    fix exp);

  }
}
```

Output:- "1 2 +".

2.
```
import java.util.*;
import java.util.io;
class Test {
    static void stack.push (stack<string>stack)
    { System.out.println("Pop operation :");
      for (int i=0; i<5; i++)
      { integer v=(integer) stack.pop();
        System.out.println(y);
      }
      Public static void main (String[] args)
      { Stack<string>stack = new stack<string>;
        stack.Push(stack);
      }
    }
```

Output:- Hi

3. public class Solution {
Public int completeCircuit (int[] gas,
int[] cost)
{
int co, Remaining=0;
int totalRemaining=0;
int start=0;
for (int i=0; i<gas.length; i++)
{
int remaining = gas[i]-cost[i];
if (currRemaining<0)
{
start=i;
currRemaining > Remaining;
}
else
{
currRemaining = Remaining;
}
totalRemaining+= Remaining;
}
if (totalRemaining<0)
return -1;
else
return start;
}}

output:- 3.

Post LaB:-

1. import java.util.*;
Public class Test
{
Public static Void main (String[] args)
{
Scanner sc = new Scanner (System.in);
int n = sc.nextInt();
reverse<integer>q= new linked list<>
int[] arr = new int[n];

```
for (int i=0; i<n ;i++) {
    int x=sc.next();
    q.add(x);
}
for(int i=0;i<n;i++)
{ arr[i] = sc.nextint();
}
int result =0;
   int j=0;
while (!q.isempty())
{ int front =q.peek();
    if (front == arr[j])
    { q.remove();
      result ++;
       j++;
    }
    else
    {
      q.remove();
       q.add (front);
        result ++;
    }
    System.out.println (result);
}}
```

Output :- 6:

2. Public class abc

```
{ static int minParanthesis (String P)
  { int bal=0;
    int ans=0;
    for(int i=0; i<P.length(); i++) {
    bal +P.charAt(i) == ('(') ? : -1;
       if (bal==-1)
       { ans+=1;
         bal+=1;
       }
  }
```

```
        return bal_pans;
    }
    Public static void main (String[] args)
    {
        string P = "(())";
        System.out.println (min Paranthesis(P));
    }
}
```

Output:-

```
1
one :
```