```
//S Harshini-185001058
//1
import java.util.Scanner;
import java.lang.*;
interface stackADT
{
	int maxsize=15;
	public boolean isFull();
	public boolean isEmpty();
	public void push(int element);
	public int pop();
	public int peep();
}
class Stack implements stackADT
{
	int top=-1,i;
	public int []arr=new int[maxsize];
	public boolean isFull()
	{
		if(top+1==maxsize)
			return true;
		else
			return false;
	}
	public boolean isEmpty()
	{
		if(top==-1)
			return true;
		else
			return false;
	}
	public void push(int element)
	{
		if(isFull())
			System.out.println("the stack is full");
		else
			arr[++top]=element;
	}
	public int pop()
	{
		if(isEmpty())
		{
			return -1;
```

```
                }
                else
                {
                        i=arr[top];
                        top--;
                        return i;
                }
        }
        public int peep()
        {
                if(isEmpty())
                {
                        return -1;
                }
                else
                        return arr[top];
        }
}
class balance_comp
{
        boolean balanceParanthesis(String expression)
        {
                Stack s=new Stack();
                int p;
                char []str=expression.toCharArray();
                int l=str.length;
                int []exp=new int[l];
                for(int j=0;j<l;j++)
                {
                        exp[j]=(int)str[j];
                        System.out.println(exp[j]);
                }
                for(int k=0;k<l;k++)
                {
                        if(exp[k]==(int)('(') || exp[k]==(int)('{') || exp[k]==(int)('['))
                                { System.out.println(k);
                                s.push(exp[k]);}
                        else if(exp[k]==(int)(')') && s.peep()==(int)('('))
                                p=s.pop();
                        else if(exp[k]==(int)('}') && s.peep()==(int)('{'))
                                p=s.pop();
                        else if(exp[k]==(int)(']') && s.peep()==(int)('['))
                                p=s.pop();
```

```
                }
                if(s.isEmpty())
                        return true;
                else
                        return false;
        }
        boolean checkTwoStacks(Stack s1, Stack s2)
        {
                int i,j;
                for( i=0, j=0; i<=s1.top || j<=s2.top ;i++,j++)
                        if(s1.arr[i] != s2.arr[j])
                                return false;

                if( (i-1)!=s1.top || (j-1)!=s2.top)
                        return false;
                return true;
        }
}
class Main
{
        public static void main(String arg[])
        {
                Stack s1=new Stack();
                Stack s2=new Stack();
                int ch,ele;
                boolean wh=true;
                Scanner in=new Scanner(System.in);
                balance_comp oper=new balance_comp();
                System.out.println("enter expression to check");
                String e=in.nextLine();
                wh=oper.balanceParanthesis(e);
                if(wh==true)
                        System.out.println("parenthesis is balanced");
                else
                        System.out.println("parenthesis is not balanced");
                System.out.println("enter details for stack 1");
                wh=true;
                while(wh)
                {
                        System.out.println("enter choice 1.push 2.pop 3.peep 4.exit");
                        ch=in.nextInt();
                        switch(ch)
                        {
```

```
                        case 1:System.out.println("Enter element");
                                ele=in.nextInt();
                                s1.push(ele);
                                break;
                        case 2:ele=s1.pop();
                                System.out.println("element is "+ele);
                                break;
                        case 3:ele=s1.peep();
                                System.out.println("element is "+ele);
                                break;
                        case 4:wh=false;
                                break;
                }
        }
        System.out.println("enter details for stack 2");
        wh=true;
        while(wh)
        {
                System.out.println("enter choice 1.push 2.pop 3.peep 4.exit");
                ch=in.nextInt();
                switch(ch)
                {
                        case 1:System.out.println("Enter element");
                                ele=in.nextInt();
                                s2.push(ele);
                                break;
                        case 2:ele=s2.pop();
                                System.out.println("element is "+ele);
                                break;
                        case 3:ele=s2.peep();
                                System.out.println("element is "+ele);
                                break;
                        case 4:wh=false;
                                break;
                }
        }
        wh=oper.checkTwoStacks(s1,s2);
        if(wh==true)
                System.out.println("Stacks are equal");
        else
                System.out.println("Stacks are not equal");

}
```

```
}

/*Sample input/output
C:\Users\Harshini\Desktop>java Main
enter expression to check
5+(3+5()
53
43
40
51
43
53
40
41
2
6
parenthesis is not balanced
enter details for stack 1
enter choice 1.push 2.pop 3.peep 4.exit
1
Enter element
2
enter choice 1.push 2.pop 3.peep 4.exit
3
element is 2
enter choice 1.push 2.pop 3.peep 4.exit
4
enter details for stack 2
enter choice 1.push 2.pop 3.peep 4.exit
1
Enter element
1
enter choice 1.push 2.pop 3.peep 4.exit
1
Enter element
2
enter choice 1.push 2.pop 3.peep 4.exit
1
Enter element
2
enter choice 1.push 2.pop 3.peep 4.exit
4
Stacks are not equal
```

```
*/
```