S.No: 28	Exp. Name: ***Stack Implementation***	Date:2023-12-25

ID: 224G1A0516

Aim:

Create an interface for stack with push and pop operations. Implement the stack in two ways fixed-size stack and Dynamic stack (stack size is increased when the stack is full).

Note: Please don't change the package name.

Source Code:

q29794/StaticAndDynamicStack.java

```
package q29794;
interface Stack {
   void push(int item);
   int pop();
}
class FixedSizeStack implements Stack {
   private int stck[];
   private int tos;
   FixedSizeStack(int size) {
      stck = new int[size];
      tos=-1;
   }
   //Push an item onto the stack
   public void push(int item) {
      if(tos==stck.length-1) //use length member
         System.out.println("Stack is full.");
      else
         stck[++tos]=item;
   }
   //pop an item from the stack
   public int pop() {
      if(tos<0) {
         System.out.println("Stack underflow");
         return 0;
      }else{
         return stck[tos--];
      }
   }
}
class DynamicStack {
   private int stck[];
   private int tos;
   DynamicStack(int size) {
      stck = new int[size];
      tos=-1;
   }
   //push an item onto the stack
   public void push(int item) {
      if(tos==stck.length-1) {
         //use length member
         System.out.println("Stack is full and increased");
```

```
            stck=doublesize(stck);
         }else{
            stck[++tos]=item;
         }
      }
      //pop an item from the stack
      public int pop() {
         if(tos<0) {
            System.out.println("Stack underflow");
            return 0;
         }else{
            return stck[tos--];
         }
      }
      int[] doublesize(int [] arr) {
         int[] newArray = new int[stck.length * 2];
         for(int i=0;i<stck.length;i++) {
            newArray[i]=stck[i];
         }
         return newArray;
      }
   }
   public class StaticAndDynamicStack {
      public static void main(String[] args) {
         FixedSizeStack mystack1 = new FixedSizeStack(5);
         DynamicStack mystack2 = new DynamicStack(5);
         //push some numbers onto the stack
         for(int i=0;i<5;i++)
            mystack1.push(i);
         for(int i=0;i<10;i++)
            mystack2.push(i);
         //pop those numbers off the stack
         System.out.println("Stack in mystack1:");
         for(int i=0;i<5;i++) {
            System.out.println(mystack1.pop());
         }
         System.out.println("Stack in mystack2 :");
         for(int i=0;i<10;i++)
            System.out.println(mystack2.pop());
      }
   }
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Stack is full and increased
Stack in mystack1:
4
3
2
1
0
Stack in mystack2 :

9
8
7
6
4
3
2
1
0
Stack underflow
0