

1.

Valiveti manikanta bhuvanesh

Let's say you have an integer array and a string array. You have to write a single method `printArray` that can print all the elements of both arrays. The method should be able to accept both integer arrays or string arrays. (Do not use overloading, use generics). Name the file `ArrayPrint.java`

```
import java.util.*;
```

```
public class ArrayPrint{
```

```
    static <T> void printArray(T[] a){
```

```
        for(int i=0;i<a.length;i++){
```

```
            System.out.print(a[i]+" ");
```

```
        }
```

```
        System.out.println();
```

```
    }
```

```
    public static void main(String[] args) {
```

```
        Integer [] a={1,2,3,4,5,6,7,8,9};
```

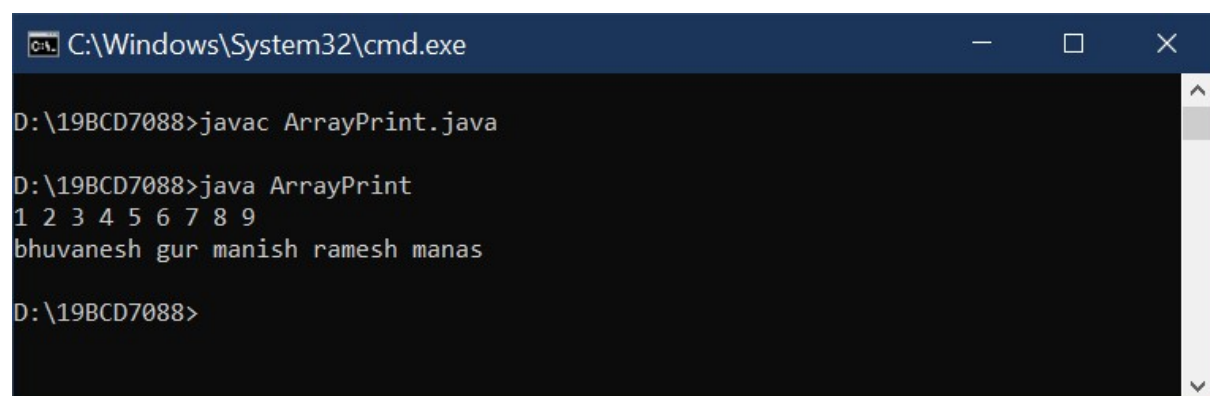
```
        String [] str = {"bhuvanesh","gur","manish","ramesh","manas"};
```

```
        printArray(a);
```

```
        printArray(str);
```

```
    }
```

```
}
```



```
C:\Windows\System32\cmd.exe

D:\19BCD7088>javac ArrayPrint.java

D:\19BCD7088>java ArrayPrint
1 2 3 4 5 6 7 8 9
bhuvanesh gur manish ramesh manas

D:\19BCD7088>
```

2.

Write a generic method to perform linearSearch on array of objects. Name the file GenericLinearSearch.java

```
import java.util.*;
```

```
public class GenericLinearSearch{
```

```
    static <T> int linearSearch(T[] a,T n){
```

```
        for(int i=0;i<a.length;i++){
```

```
            if(a[i].equals(n)){
```

```
                return i;
```

```
            }
```

```
        }
```

```
        return 0;
```

```
    }
```

```
    static <T> void printArray(T[] a){
```

```
        for(int i=0;i<a.length;i++){
```

```
            System.out.print(a[i]+" ");
```

```
        }
```

```
        System.out.println();
```

```
    }
```

```
    public static void main(String[] args) {
```

```
        Integer [] a={1,2,3,4,5,6,7,8,9};
```

```
        Integer k1=5;
```

```
        printArray(a);
```

```
        int k = linearSearch(a,k1);
```

```
        System.out.println(k1 + " is found at index " + k);
```

```
        String [] str = {"bhuvanesh","guru","manish","ramesh","manas"};
```

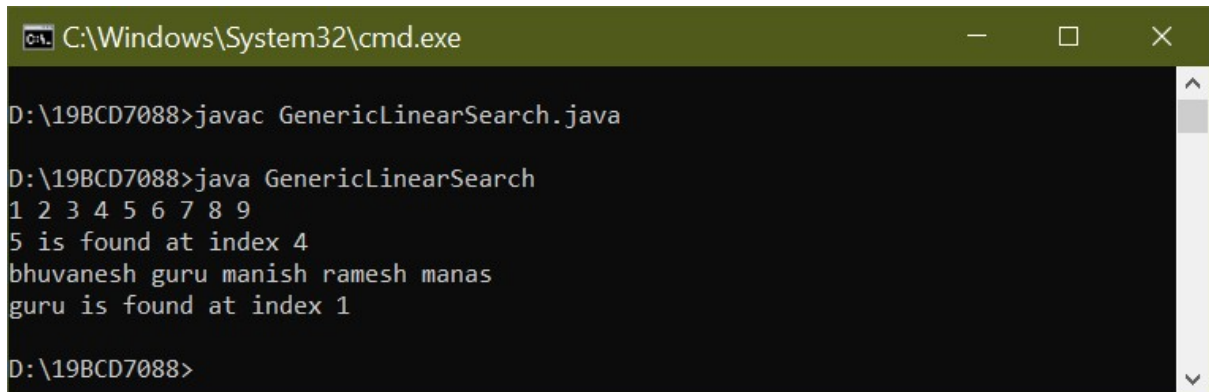
```
        String k2="guru";
```

```
        printArray(str);
```

```
        int p = linearSearch(str,k2);
```

```
        System.out.println(k2 + " is found at index " + p);
```

```
}  
  
}
```



```
C:\Windows\System32\cmd.exe  
D:\19BCD7088>javac GenericLinearSearch.java  
D:\19BCD7088>java GenericLinearSearch  
1 2 3 4 5 6 7 8 9  
5 is found at index 4  
bhuvanesh guru manish ramesh manas  
guru is found at index 1  
D:\19BCD7088>
```

3.

Write a generic class called `GenericStack<T>` that represents a stack structure. A stack structure follow the strategy last-in-first-out, which means that the last element added to the stack, is the first to be taken out.

The `GenericStack` class has the following attributes and methods:

--An attribute `ArrayList<T>` elements which represents the elements of the stack.(All of you refer collection framework for `ArrayList`. or you can use an array to hold the elements of Stack.)[Refer the following links to have intro on `ArrayList`:

https://www.w3schools.com/java/java_arraylist.asp,

<https://www.geeksforgeeks.org/arraylist-in-java/>]

--A constructor that creates the `ArrayList` or an Array

--A method `push(T e)` which adds the element to the `ArrayList<T>` or array.

--A method `pop()` which removes the last element of the `ArrayList<T>` (last element added), if the list is not already empty and returns it.

--A method `print()` which prints the elements of the stack starting from the last element to the first element.

```
import java.util.*;

class GenericStack<T>{

    int top;

    int max;

    T [] a;

    GenericStack(int size){

        a=(T[]) new Object[size];

        max=size;

        top=-1;

    }

    void push(T x){

        if(top==(max-1)){

            System.out.println("Stack is Filled");

        }

        else{

            top++;

            a[top]=x;

        }

    }

    T pop(){

        if(top==-1){

            System.out.println("Stack is empty");

            return null;

        }

        else{

            return a[top--];

        }

    }

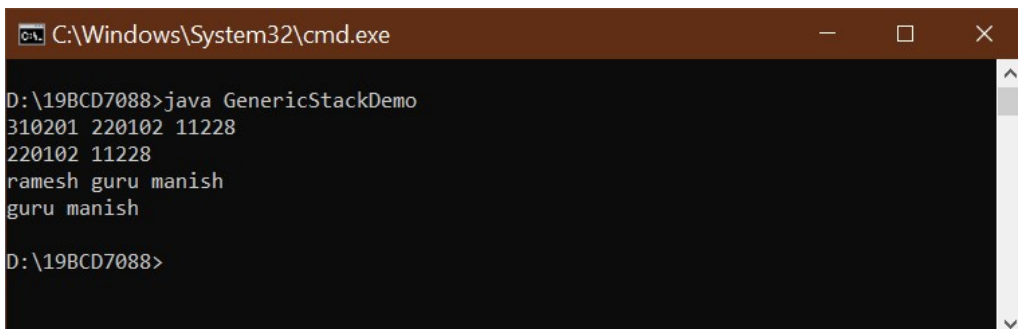
    void print(){
```

```

        for (int i=top;i>=0;i--){
            System.out.print(a[i] + " ");
        }
        System.out.println();
    }
}

public class GenericStackDemo{
    public static void main(String[] args) {
        GenericStack<Integer> s =new GenericStack<Integer>(3);
        GenericStack<String> str =new GenericStack<String>(3);
        s.push(11228);
        s.push(220102);
        s.push(310201);
        s.print();
        s.pop();
        s.print();
        str.push("manish");
        str.push("guru");
        str.push("ramesh");
        str.print();
        str.pop();
        str.print();
    }
}

```



```

C:\Windows\System32\cmd.exe
D:\19BCD7088>java GenericStackDemo
310201 220102 11228
220102 11228
ramesh guru manish
guru manish
D:\19BCD7088>

```

3. b.

Create the generic interface `GenericStackable<T>` that contains the abstract methods:

-an abstract method `push(T e)` which adds the element to the `ArrayList<T>`

-an abstract method `pop()` which remove the last element of the `ArrayList<T>` (last element added), if the list is not already empty.

-a abstract method `print()` which prints the elements of the stack starting from the last element to the first element.

-a abstract method `isEmpty()` that would return true if the stack is empty, and false otherwise.

Modify the class `GenericStack<T>` such that it implements the generic interface `GenericStackable<T>` . Create a class `GenericStackDemo2` and work with two different stacks.

```
import java.util.*;
```

```
interface GenericStackable<T>{  
    public void push(T x);  
    public T pop();  
    public void print();  
    public boolean isEmpty();  
}
```

```
class GenericStack<T> implements GenericStackable<T> {  
    int top;  
    int max;  
    T [] a;  
    GenericStack(int size){  
        a=(T[]) new Object[size];  
        max=size;  
    }  
}
```

```

        top=-1;
    }
    public void push(T x){
        if(top==(max-1)){
            System.out.println("Stack is Filled");
        }
        else{
            top++;
            a[top]=x;
        }
    }
    public T pop(){
        if(top==-1){
            System.out.println("Stack is empty");
            return null;
        }
        else{
            return a[top--];
        }
    }
    public void print(){
        for (int i=top;i>=0;i--){
            System.out.print(a[i] + " ");
        }
        System.out.println();
    }
    public boolean isEmpty(){
        if(top==-1){
            return true;
        }
    }

```

```

    }
    else{
        return false;
    }
}

}

public class GenericStackDemo2{

    public static void main(String[] args) {

        GenericStack<Integer> s =new GenericStack<Integer>(3);
        GenericStack<String> str =new GenericStack<String>(3);

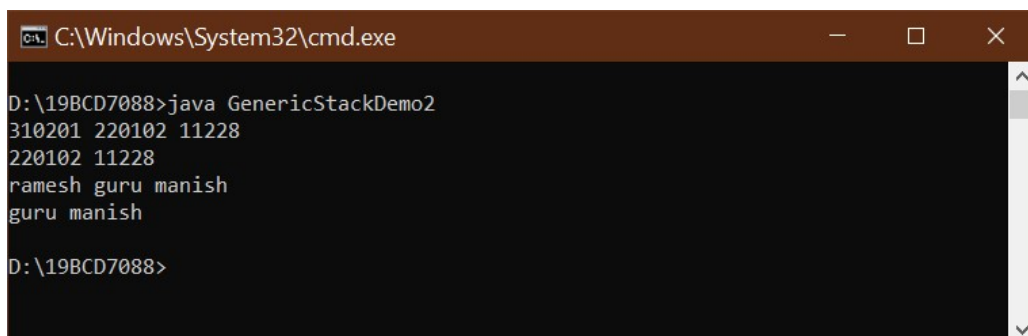
        s.push(11228);
        s.push(220102);
        s.push(310201);
        s.print();
        s.pop();
        s.print();

        str.push("manish");
        str.push("guru");
        str.push("ramesh");
        str.print();
        str.pop();
        str.print();

    }

}

```



```

C:\Windows\System32\cmd.exe
D:\19BCD7088>java GenericStackDemo2
310201 220102 11228
220102 11228
ramesh guru manish
guru manish
D:\19BCD7088>

```