Dt : 19/9/2023

faq:

define Immutable Classes?

=>The classes which are constructed using the following rules are known as Immutable Classes.

Rule-1 : The class must be final class

Rule-2 : The variables in the class must be private and final variables

Rule-3 : The methods in class must be only Getter methods.

Rule-4 : The Getter methods in class must be final methods

Note:

(i)Variables in Immutable classes are initialized with Constructor

(ii)These Immutable classes will generate Immutable Objects.

---

faq:

define Immutable Objects?

=>The objects once created with data cannot be modified.

=>These Immutable Objects are also known as Secured Objects or Constant Objects.

==

Ex:

p1 : TransLog.java

```
package p1;
import java.util.Date;
public final class TransLog
{
   private final long hAccNo,bAccNo;
   private final float amt;
   private final Date dateTime;
   public TransLog(long hAccNo,long bAccNo,float amt,
              Date dateTime)
   {
        this.hAccNo=hAccNo;
        this.bAccNo=bAccNo;
        this.amt=amt;
        this.dateTime=dateTime;
   }
public final long gethAccNo() {
     return hAccNo;
}
public final long getbAccNo() {
     return bAccNo;
}
public final float getAmt() {
     return amt;
}
public final Date getDateTime() {
     return dateTime;
}

}
```

p2 : DemoPoly6.java(MainClass)

package p2;

import java.util.*;

```
import p1.*;
public class DemoPoly6 {
      public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    try(s;){
      try {
          System.out.println("Enter Home AccNo:");
          long hAccNo = s.nextLong();
          System.out.println("Enter Beneficiery AccNo:");
          long bAccNo = s.nextLong();
          System.out.println("Enter the amt to be transferred:");
          float amt = s.nextFloat();
          TransLog ob = new TransLog
                    (hAccNo,bAccNo,amt,new Date());
              //Immutable Object
          System.out.println("====Transaction Details===");
          System.out.println("HAccNo:"+ob.gethAccNo());
          System.out.println("BAccNo:"+ob.getbAccNo());
          System.out.println("Amt:"+ob.getAmt());
          System.out.println("DateTime:"+ob.getDateTime());
      }catch(Exception e) {
           e.printStackTrace();
```

```
            }
        }//end of try with resource
          }
}
```

o/p:

```
Enter Home AccNo:
6123456
Enter Beneficiery AccNo:
31313131
Enter the amt to be transferred:
1234.56
====Transaction Details===
HAccNo:6123456
BAccNo:31313131
Amt:1234.56
DateTime:Tue Sep 19 18:46:06 IST 2023
=================================================================
===
```

Note:

=>Based on Security the objects in Java are categorized into two types:

1.Mutable Objects

2.Immutable Objects

1.Mutable Objects:

=>The Objects once created can be modified are known as Mutabl Objects.

2.Immutable Objects:

=>The Objects once created cannot be modified are known as Immutable Objects.

===

faq:

define "Record"?(Java17 - new feature)

=>"Record" is an abstract class from java.lang package introduced by Java17 version and which generate Immutable Objects.

structure of "Record":

```
public abstract class java.lang.Record
{
  protected java.lang.Record();
  public abstract boolean equals(java.lang.Object);
  public abstract int hashCode();
  public abstract java.lang.String toString();

}
```

=>we use the following syntax to create Record-Objects:

```
record Record_name(para_list)

{

  //record_body

}
```

Ex:

p1 : TransLog.java

```
package p1;
import java.util.*;
public record TransLog(long hAccNo,long bAccNo,float
amt,
         Date dateTime)
{
    public TransLog
    {
    if(amt<=0)//Exception
    {
         throw new IllegalArgumentException
         ("Enter amt greater than Zero...");
    }
    }
}
```

p2 : DemoPoly7.java(MainClass)

```
package p2;

import p1.TransLog;

import java.util.*;

public class DemoPoly7 {

        public static void main(String[] args) {

      Scanner s = new Scanner(System.in);

      try(s;){

        try {

                System.out.println("Enter the hAccNo:");

                long hAccNo = s.nextLong();

                System.out.println("Enter the bAccNo:");

                long bAccNo = s.nextLong();

                System.out.println("Enter the amt:");

                float amt = s.nextFloat();

                TransLog ob = new TransLog

                                (hAccNo,bAccNo,amt,new Date());

                System.out.println("===Details====");

                System.out.println("HAccNo:"+ob.hAccNo());

                System.out.println("BAccNo:"+ob.bAccNo());

                System.out.println("Amt:"+ob.amt());

                System.out.println("DateTime:"+ob.dateTime());
```

```
    }catch(Exception e) {

        System.out.println(e.getMessage());

    }

  }///end of try with resource

    }

}
```

o/p:

```
Enter the hAccNo:

6123456

Enter the bAccNo:

313131

Enter the amt:

16000

===Details====

HAccNo:6123456

BAccNo:313131

Amt:16000.0

DateTime:Tue Sep 19 19:17:53 IST 2023
```

==

Advantage of "Record":

1.Auto-Construction of parameterized Constructor and which is known as

"Canonical Constructor".

2.Auto-Generation of Getter methods.

3.We use "Compact Constructor" to validate parameters or arguments.

==
=