Dt : 17/12/2020

(b)Constructors with parameters:

=>The Constructors which are declared with parameters are known as Parameterized Constructors or Constructors with parameters.

Exp program:

```
class CTest //SubClass
{
	CTest(int x)
	{
System.out.println("===Constructor===");
System.out.println("The value x:"+x);
	}
	void dis(int y)
	{
System.out.println("===Method===");
System.out.println("The value y:"+y);
	}
}
class DCon3//MainClass
{
	public static void main(String[] args)
	{
CTest ob = new CTest(101);//Con call
ob.dis(102);//method call
	}
```

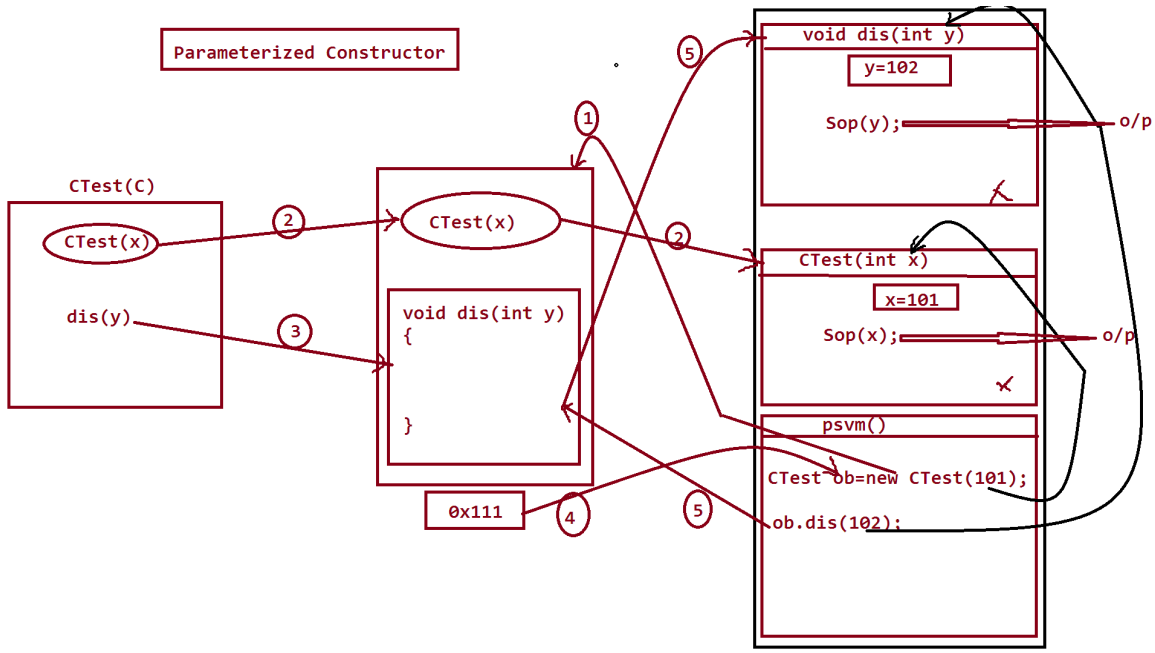
```
}
```

Execution flow of above program:

ClassFiles:

CTest.class

DCon3.class



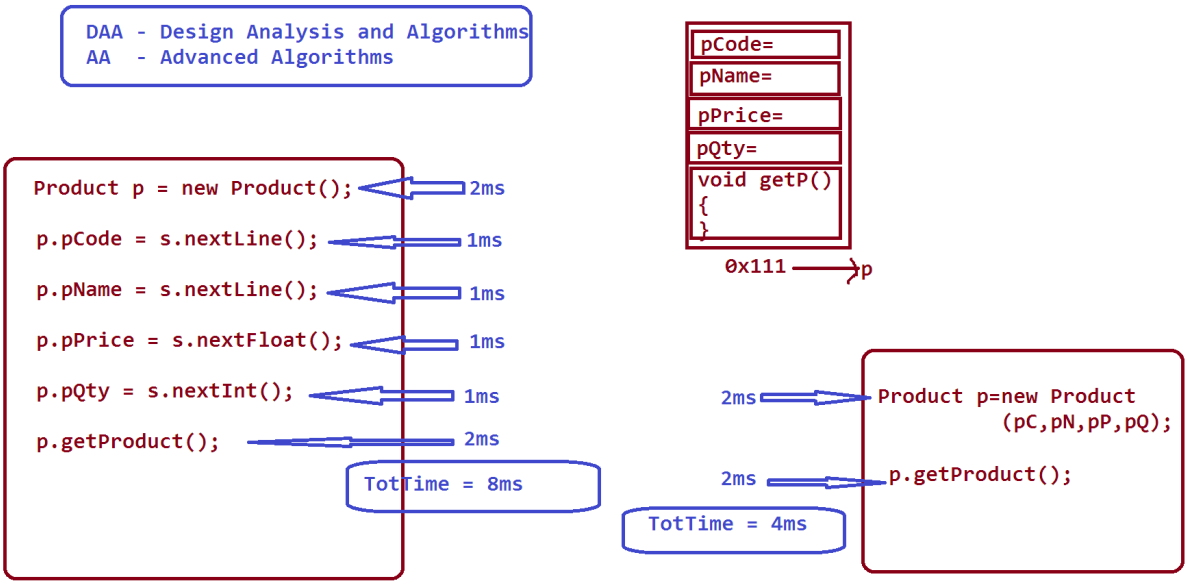
Note:

=>we pass parameters to the Constructor while Constructor call,which means while Object creation becuase the constructor call is available within the object creation syntax.

faq:

wt is the advantage of Constructor?

=>Constructors are used to initialize instance variables while object creation,which saves the execution time and generates HighPerformance.



Exp program:

```
import java.lang.String;
import java.lang.System;
import java.util.Scanner;
class Product //SubClass
{
        String pCode,pName;
        float pPrice;
        int pQty;
        Product(String pCode,String pName,float c,int d)
        {
                this.pCode=pCode;
                this.pName=pName;
```

```
                pPrice=c;
                pQty=d;
        }
  void getProduct()
        {
System.out.println("pCode:"+pCode);
System.out.println("pName:"+pName);
System.out.println("pPrice:"+pPrice);
System.out.println("pQty:"+pQty);
        }
}
class DCon4 //MainClass
{
        public static void main(String[] args)
        {
Scanner s = new Scanner(System.in);
System.out.println("Enter the ProdCode:");
String pC = s.nextLine();
System.out.println("Enter the ProdName:");
String pN = s.nextLine();
System.out.println("Enter the ProdPrice:");
float pP = s.nextFloat();
System.out.println("Enter the ProdQty:");
int pQ = s.nextInt();
```

```
Product p = new Product(pC,pN,pP,pQ);//con call

p.getProduct();//method call

        }
```

}

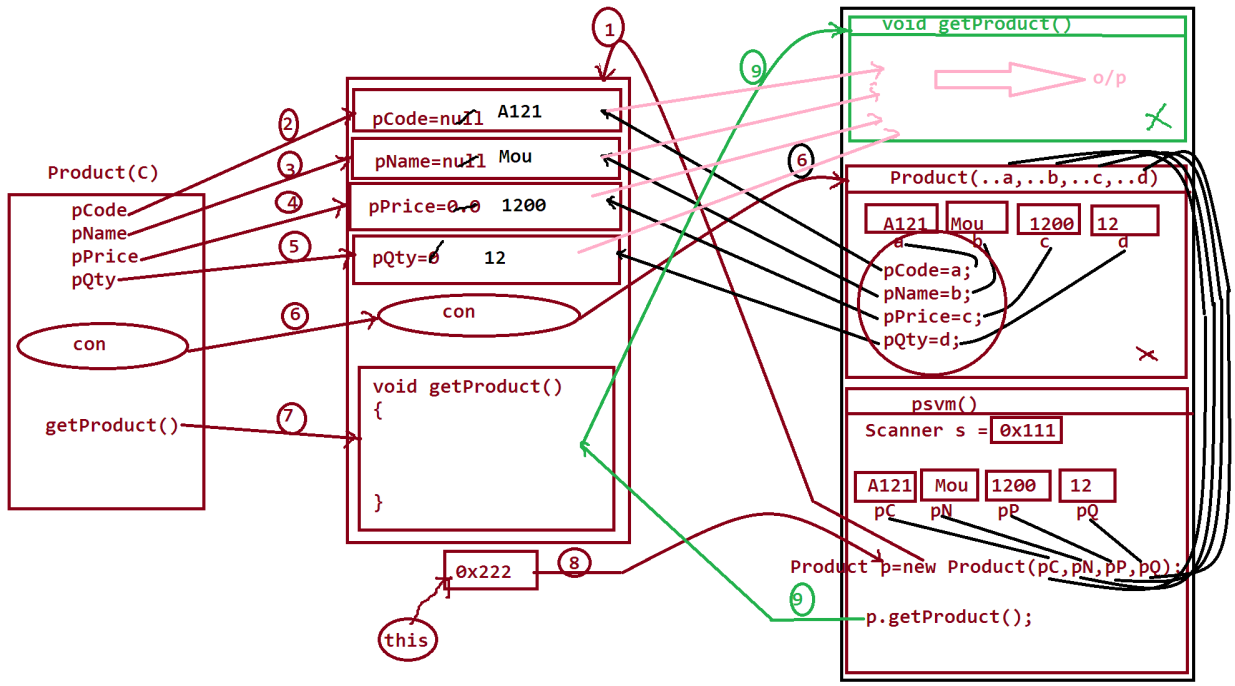
Dt : 18/12/2020

Execution flow of above program:

ClassFiles:

Product.class

DCon4.class



faq:

define 'this' keyword?

=>'this' is built-in NonPrimitive datatype variable which hold object reference.

=>'this' keyword will hold reference of object from where the current method or constructor is executing.

=>'this' keyword is used when we have same variable names in Instance variables and Local variables.

==========

faq:

define static constructor?

=>There is no concept of static constructors in java,because constructors means executed while object creation.

==========

***imp**

RelationShip b/w classes:

=>The process of establishing communication b/w classes is known as RelationShip b/w classes.

=>RelationShip b/w classes are categorized into three types:

1.References

2.Inheritance

3.InnerClasses

1.References:

=>The process of declaring NonPrimitive datatype variable part of

class and this NonPrimitive datatype variable will hold the reference of object of another classe is known as 'References concept'.

=>In this process the methods of one class can access the members of another class using reference.

Exp program:

Assignment(Solution):

Contruct 'Bank Transaction process'.

step1 : Enter the pinNo

=>pinNo must be

1111

2222

3333

step2 : If the pinNo is validated then show the following choice:

1.WithDraw

2.Deposit

1.WithDraw:

=>Enter the int amt,and the amt must be greater than zero and multiples of 100.

=>If the amt is validated then create object for 'WithDraw' class.

=>call the method of WithDraw class and pass amt as parameter.

=>Part of method check the amt is less than the bal or not.

=>If amt less than the bal then perform transaction

o/p:

Amt withDrawn :

Bal Amt :

Transaction completed

2.Deposit:

=>Enter the int amt,and the amt must be greater than zero and multiples of 100.

=>If the amt is validated then create object for 'Deposit' class.

=>call the method of Deposit class and pass amt as parameter.

o/p:

Amt Deposited :

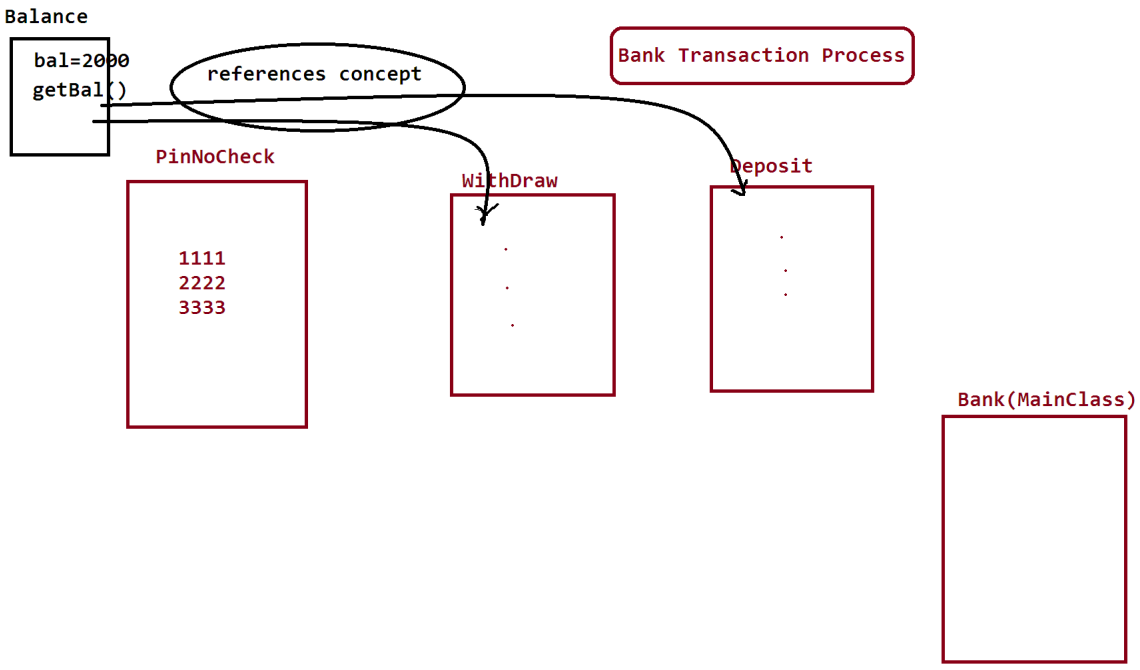
Bal Amt :

Transaction completed

Note:

=>If the pinNo entered wrongly for three times then display the msg as 'Transaction blocked Temporarly'

==========




```
import java.util.Scanner;

class Balance //SubClass

{

        double bal = 2000;

        void getBal()

        {

System.out.println("Bal Amt:"+bal);

        }

}

class PinNoCheck //SubClass

{

        boolean z=false;
```

```
        boolean verify(int pinNo)
        {
                switch(pinNo)
                {
                        case 1111:z=true;
                        break;
                        case 2222:z=true;
                        break;
                        case 3333:z=true;
                        break;
                }//end of switch
                return z;
        }
}
class WithDraw //SubClass
{
   Balance b;
   WithDraw(Balance b)
        {
          this.b=b;
        }
        void wDraw(int amt)
        {
                if(amt<=b.bal)
                {
```

```
System.out.println("Amt withDrawn:"+amt);
b.bal = b.bal-amt;
b.getBal();
System.out.println("Transaction completed..");
                }//end of if
                else
                {
System.out.println("Insufficient fund...");
                }
        }
}
class Deposit //SubClass
{
        Balance b;
        Deposit(Balance b)
        {
                this.b=b;
        }
  void deposit(int amt)
        {
System.out.println("Amt deposited:"+amt);
b.bal=b.bal+amt;
b.getBal();
System.out.println("Transaction completed...");
        }
```

```
}
class DRef1 //MainClass bank model
{
        public static void main(String[] args)
        {
Scanner s = new Scanner(System.in);
Balance b = new Balance();
int count=0;
xyz:while(true){
System.out.println("Enter the pinNo:");
int pinNo = s.nextInt();
PinNoCheck pnc = new PinNoCheck();
boolean z = pnc.verify(pinNo);
   if(z)
        {
System.out.println("===Choice===");
System.out.println("1.WithDraw\n2.Deposit");
System.out.println("Enter the choice:");
int choice = s.nextInt();
     switch(choice)
                {
                  case 1://WithDraw
System.out.println("Enter the amt:");
int a1 = s.nextInt();
        if(a1>0 && a1%100==0)
```

```
                        {
WithDraw wd = new WithDraw(b);
wd.wDraw(a1);
                        }//end of if
                        else
                        {
System.out.println("Invalid amt..");
                        }
                    break xyz;
                    case 2://Deposit
System.out.println("Enter the amt:");
int a2 = s.nextInt();
   if(a2>0 && a2%100==0)
                        {
Deposit dp = new Deposit(b);
dp.deposit(a2);
                        }//end of if
                        else
                        {
System.out.println("Invalid amt...");
                        }
                    break xyz;
         default:
System.out.println("Invalid choice...");
                    break xyz;
```

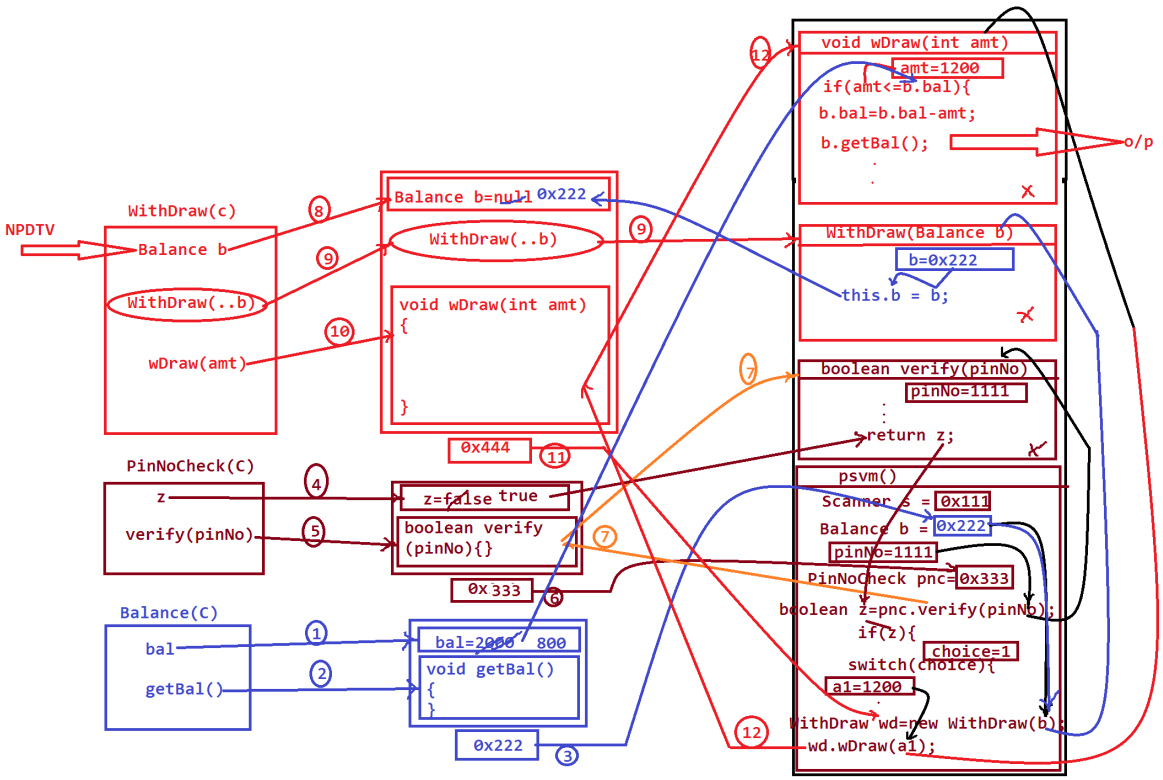
```
                }//end of switch
        }else{
System.out.println("InValid pinNo...");
count++;
        }
        if(count==3)
        {
System.out.println("Transaction blocked temporarly...");
break xyz;
        }

}//end of loop
        }
}
```

Dt : 19/12/2020

Execution flow of above program:

ClassFiles:

- **Balance.class**
- **PinNoCheck.class**
- **WithDraw.class**
- **Deposit.class**
- **DRef1.class(MainClass)**



Nore:

=>References concept means one Object is holding the reference of another object,in this process there is a link b/w objects.

=>In the above diagram WithDraw class object is holding the reference of object of Balance class,in this process the method(wDraw()) of WithDraw can access all the members of Balance class.

==========