```
//PROBLEM STATEMENT :
/* Implement a factory design pattern for the given context . Consider Car building process ,
 *  which requires many steps from allocating accessories to final makeup. These steps should
 *  be written as methods and should be called while creating an instance of specific  car type.
 *  Hatchback, Sedan, SUV , could be the subclasses Car class. Car class and Car class its subclasses
 *   , CarFactory and Test Factory Pattern should be implemented */


//package assignment;
import java.util.Scanner;
// ============================== ABSTRACT CLASS Car_Factory ==============================//
abstract class Car_Factory{

	//declaration of data member
	String compnay,car_name;
	double budget;

	//declaration of abstract methods
	abstract void getprice(double price);
	abstract void detail(String company_name,String car_name);
	abstract void accessories();

	//declaration and implentation of input method
	void input() {
		Scanner scan =new Scanner (System.in);//creating object of scanner class
		System.out.print("Company- ");
		 compnay=scan.next();//taking input from user
		System.out.print("Car- ");
		 car_name=scan.next();//taking input from user
```

```
                System.out.print("Rough Budget(in Lakhs)- ");
                 budget=scan.nextDouble();//taking input from user
        }
        void display(Car_Factory obj1) {
                //calling the methods//
                obj1.getprice(budget);//calling getprice method
                System.out.println("\n----------------------------------");
                obj1.detail(compnay, car_name);//calling detail method
                System.out.println("\n----------------------------------");
                obj1.accessories();//calling accessories method
                System.out.println("\n----------------------------------");
        }

}

//============================  CLASS Small_car ============================//
class Small_car extends Car_Factory{
        String Ans;//declaration of data member

        //method for getprice
        public void getprice(double price) {
                if(price>2&&price<5)
                        Ans="No";       //modify Ans
                else
                        Ans="Yes";      //modify Ans

        }
        //method for displaying car detail//
        public void detail(String company_name,String car_name) {
                System.out.println("Company- "+company_name);
                System.out.println("Name of Car- "+car_name);
```

```
            System.out.println("Color- Black/White/Orange/Red");
            System.out.println("Fuel- Petrol");
            System.out.println("Gears- Manual");
        }

        //method to display accessories of car//
        public void accessories() {
            System.out.println("Types of Tyres- Alloy Wheels");
            System.out.println("Airbags- "+Ans);
            System.out.println("Back Wiper- "+Ans);
            System.out.println("Side Mirror- Two");
            System.out.println("Touch Screen Music Player- "+Ans);
        }
}

//============================  CLASS Sedan ============================//
class Sedan extends Car_Factory{
        String Ans;//declaration of data member

        //method for getprice
        public void getprice(double price) {
            if(price>6&&price<10)
                Ans="No";       //modify Ans
            else
                Ans="Yes";      //modify Ans
        }

        //method for displaying car detail//
        public void detail(String company_name,String car_name) {
            System.out.println("Company- "+company_name);
            System.out.println("Name of Car- "+car_name);
```

```
		System.out.println("Color- Black/White/Orange/Red");
		System.out.println("Fuel- Petrol/Diesel");
		System.out.println("Gears- Auto/Manual");
	}
	//method to display accessories of car//
	public void accessories() {
		System.out.println("Types of Tyres- Alloy Wheels");
		System.out.println("Airbags- YES");
		System.out.println("Back Wiper- YES");
		System.out.println("Side Mirror- Two");
		System.out.println("Touch Screen Music Player- YES");
		System.out.println("Roof Window- "+Ans);
	}
}

//============================  CLASS Small_car ============================//
class Luxary extends Car_Factory{
	String Ans;//declaration of data member

	//method for getprice
	public void getprice(double price) {
		if(price>10&&price<14)
			Ans="No";	//modify Ans
		else
			Ans="Yes";	//modify Ans
	}

	//method for displaying car detail//
	public void detail(String company_name,String car_name) {
		System.out.println("Company- "+company_name);
		System.out.println("Name of Car- "+car_name);
```

```
		System.out.println("Color- Black/White/Orange/Red");
		System.out.println("Fuel- Diesel");
		System.out.println("Gears- Auto");
	}

	//method to display accessories of car//
	public void accessories() {
		System.out.println("Types of Tyres- Alloy Wheels");
		System.out.println("Airbags- YES");
		System.out.println("Back Wiper- YES");
		System.out.println("Side Mirror- Two");
		System.out.println("Touch Screen Music Player- YES");
		System.out.println("Roof Window- YES");
		System.out.println("Automotive Garbage Cans- "+Ans);
		System.out.println("Automotice Air Freshner- "+Ans);
		System.out.println("Button Start- "+Ans);
	}
}

//============================== MAIN CLASS ==============================//
public class Practi10{

	//ststic main method
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner scan = new Scanner(System.in);//creating object of scanner class
		int ch;
		//double price;
		Car_Factory obj;// object of reference Car_Factory
		while(true){
			//menu driven
```

```
System.out.println("Which Car you want to See?- ");
System.out.println("\n\t1.Small Car\n\t2.Sedan Car\n\t3.Luxary
Car\n\t4.Exit");
ch=scan.nextInt();//taking input from user
System.out.println();
//switch case
switch(ch) {

    case 1:
        obj= new Small_car(); //creating object of Small_car
        obj.input();//calling input method
        obj.display(obj);//calling display method
        break;

    case 2:
        obj= new Sedan();//creating object of Sedan
        obj.input();//calling input method
        obj.display(obj);//calling display method
        break;

    case 3:
        obj= new Luxary();//creating object of Luxary
        obj.input();//calling input method
        obj.display(obj);//calling display method
        break;

    case 4:
        System.out.println("\n-----------------------------------");
        return;//stop execution of program

    default:
```

```
                                        System.out.println("INVALID CHOICE !!");//default
                                        System.out.println("\n-----------------------------------");
                                        break;

                                }
                        }
                }
}
```

OUTPUT:

Which Car you want to See?-

1.Small Car

2.Sedan Car

3.Luxary Car

4.Exit

1

Company- Maruti

Car- Baleno

Rough Budget(in Lakhs)- 1000000

Company- Maruti

Name of Car- Baleno

Color- Black/White/Orange/Red

Fuel- Petrol

Gears- Manual

Types of Tyres- Alloy Wheels

Airbags- Yes

Back Wiper- Yes

Side Mirror- Two

Touch Screen Music Player- Yes

Which Car you want to See?-

1.Small Car

2.Sedan Car

3.Luxary Car

4.Exit

2

Company- Hyundai

Car- Verna

Rough Budget(in Lakhs)- 1500000

Company- Hyundai

Name of Car- Verna

Color- Black/White/Orange/Red

Fuel- Petrol/Diesel

Gears- Auto/Manual

Types of Tyres- Alloy Wheels

Airbags- YES

Back Wiper- YES

Side Mirror- Two

Touch Screen Music Player- YES

Roof Window- Yes

Which Car you want to See?-

1.Small Car

2.Sedan Car

3.Luxary Car

4.Exit

3

Company- BMW

Car- Q7

Rough Budget(in Lakhs)- 4000000

Company- BMW

Name of Car- Q7

Color- Black/White/Orange/Red

Fuel- Diesel

Gears- Auto

Types of Tyres- Alloy Wheels

Airbags- YES

Back Wiper- YES

Side Mirror- Two

Touch Screen Music Player- YES

Roof Window- YES

Automotive Garbage Cans- Yes

Automotice Air Freshner- Yes

Button Start- Yes

Which Car you want to See?-

1.Small Car

2.Sedan Car

3.Luxary Car

4.Exit

4

//problem Statement

/*

Implement and apply Strategy Design pattern for simple Shopping Cart where three payment strategies are used such as Credit Card, PayPal, Bit Coin. Create an interface for strategy pattern and give concrete implementation for payment.

*/

```
//package assignment;
import java.util.Scanner;

//======================== INTERFACE PaymentProcessor ========================//
interface PaymentProcessor {

        void pay(int amount);//interface method pay

}
```

```
//====================== CLASS CreditCard ======================//
//implementing PaymentProcessor interface
class CreditCard implements PaymentProcessor {
	Scanner sc =new Scanner (System.in);//creating object of scanner class
	String name,ExpDate;//declaration of name,ExpDate
	double CardNo;//declaration of CardNo

	//Constructor of CreditCard class
	CreditCard(){
		super();//calling parent class constructor
		System.out.println("-------------------------------------------------------");
		System.out.print("\tCard holder Name :: ");//printing on console
		this.name =sc.next();//taking Card holder Name as input from user
		System.out.print("\tCard Number :: ");//printing on console
		this.CardNo =sc.nextDouble();//taking Card Number as input from user
		System.out.print("\tCard Expire Date :: ");//printing on console
		this.ExpDate =sc.next();//taking Card Expire Date as input from user
		System.out.println("-------------------------------------------------------");
	}

	@Override
	public void pay(int amount) {    //method for payment
		System.out.println("-------------------------------------------------------");
	   System.out.println("Paying through CreditCard payment: Charging $" + amount);
	   System.out.println("-------------------------------------------------------");
	}


}
```

```
//====================== CLASS PayPal ======================//
//implementing PaymentProcessor interface
class PayPal implements PaymentProcessor {

	//Constructor of PayPal class
	PayPal(){
		super();//calling parent class constructor
		System.out.println("\nChecking Internet Connection.....");
	  }

	@Override
	 public void pay(int amount) {     //method for payment
		System.out.println("-----------------------------------------------------------");
		System.out.println("Paying through PayPal payment: Charging $" + amount);
		System.out.println("-----------------------------------------------------------");
	 }

	}

//====================== CLASS BitCoin ======================//
//implementing PaymentProcessor interface
class BitCoin implements PaymentProcessor {
	Scanner sc =new Scanner (System.in);//creating object of scanner class
	String add;//declaration of add

	//Constructor of BitCoin class
	BitCoin(){
		  super();//calling parent class constructor
		  System.out.print("\nEnter Transaction 'Input Address' :: ");//asking user of address
		  add= sc.next();//taking 'INPUT ADDRESS' as input from user
```

```
    }

    @Override
    public void pay(int amount) {      //method for payment
            System.out.println("------------------------------------------------------------");
             System.out.println("Paying through BitCoin payment: Charging $" + amount);
             System.out.println("------------------------------------------------------------");
     }

}


//======================== CLASS Order ======================//

class Order {

    private final PaymentProcessor paymentProcessor;//declaration of paymentProcessor object
    private final int amount;//declaration of amount

    //Order Method
    public Order(int amount, PaymentProcessor paymentProcessor) {
       this.amount = amount;//storing value
       this.paymentProcessor = paymentProcessor;//storing value
     }

    //process Method
    public void process() {
       paymentProcessor.pay(amount);//calling pay method
     }
```

```
}


//====================== CLASS Main ======================//
public class Practi11 {
	//calling static void main method
	public static void main(String[] args) {
		int c,amt=0;//declaration of c, amt
		Order order;//reference of order assign to order obj
		Scanner sc = new Scanner(System.in);//creating object of scanner class
		while(true) {//while loop for menu driven
			System.out.println();
			//menu bar
			System.out.println("**** SHOPING CART ****");
			System.out.print("1.Credit Card \n2.PayPal \n3.BitCoin \n4.Exit");
			System.out.print("\n\nEnter the Choice ::");
			c=sc.nextInt();//taking input from user
			System.out.println("-----------------------------------------------------------");
			if(c==1||c==2||c==3) {//check whether 0<c<4
				System.out.print("\nEnter amount tobe Tranfer :: ");
				amt = sc.nextInt();//taking amt as input from user
				System.out.println("-----------------------------------------------------------");
			}
			//switch case
			switch(c) {
			 case 1://for input c ==1
				 order = new Order(amt, new CreditCard());//creating obj of order
class
				 order.process();//calling process method of order class
				break;
```

```
                        case 2://for input c == 2
                                order = new Order(amt, new PayPal());//creating obj of order class
                                order.process();//calling process method of order class
                                break;

                        case 3://for input c == 3
                                order = new Order(amt, new BitCoin());//creating obj of order class
                                order.process();//calling process method of order class
                                break;

                        case 4:
                                System.out.println("\nThank you For Shopping !!!! ");//printing on
console
                                System.out.println("-------------------------------------------------------");
                                return;//stop execution of program

                       default:
                                System.out.println("Invalid Payment Mode !!!");// default
                                System.out.println("-------------------------------------------------------");
                        }
                }
        }
}
```

OUTPUT:

**** SHOPING CART ****

1.Credit Card

2.PayPal

3.BitCoin

4.Exit

Enter the Choice ::1

Enter amount tobe Tranfer :: 5000

Card holder Name :: deepak

Card Number :: 123512541256

Card Expire Date :: 21/10/2025

Paying through CreditCard payment: Charging $5000

**** SHOPING CART ****

1.Credit Card

2.PayPal

3.BitCoin

4.Exit

Enter the Choice ::2

Enter amount tobe Tranfer :: 3000

Checking Internet Connection.....

Paying through PayPal payment: Charging $3000

**** SHOPING CART ****

1.Credit Card

2.PayPal

3.BitCoin

4.Exit

Enter the Choice ::3

Enter amount tobe Tranfer :: 6000

Enter Transaction 'Input Address' :: pune

Paying through BitCoin payment: Charging $6000

**** SHOPING CART ****

1.Credit Card

2.PayPal

3.BitCoin

4.Exit

Enter the Choice ::4

Thank you For Shopping !!!!
