

A CONFECTIONERY

Submitted By:

ANIKET SHRIVASTAVA (181B033)

ANIMESH GAUTAM (181B034)

ASHISH KUMAR GUPTA (181B054)

GUIDANCE FACULTY - Dr Prateek Pandey & Dr Dinesh Kumar Verma



November 2020

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

JAYPEE UNIVERSITY OF ENGINEERING & TECHNOLOGY

A-B ROAD, RAGHOGARH, DT. GUNA – 473226, M.P. INDIA

ACKNOWLEDGEMENT

We thank the almighty for giving us the courage & perseverance in completing the project. This project itself is an acknowledgement for all those who have given us their heart-felt-co-operation in making it a grand success.

We are also thankful to the project coordinator, **Dr. Prateek Pandey** for extending their sincere & heartfelt guidance throughout this project work. Without their supervision and many hours of devoted guidance, stimulating & constructive criticism, this thesis would never come out in this form.

It is a pleasure to express our deep and sincere gratitude to the project Guide **Dr. Dinesh Kumar Verma** and are profoundly grateful towards the unmatched help rendered by him. Our special thanks to all the lectures of Information Technology, for their valuable advises at every stage of this work.

Last but not the least; we would like to express our deep sense and earnest thanks giving to our dear parents for their moral support and heartfelt cooperation in doing the project. We would also like to thank our friends, whose direct or indirect help has enabled us to complete this work successfully.

Aniket Shrivastava

Animesh Gautam

Ashish Kumar Gupta

INTRODUCTION

The project helps a confectioner generate a bill. After the customer decides upon the choice of flavors he wishes to order. The program uses the **Decorator Pattern** to calculate the bill. Now the confectioner realizes that the combination he served were not working out and instead changes the menu so that the customer decides to serve only a single flavored ice – cream and that too following a **very rigid procedure**. The design also incorporates the use of the **Singleton Pattern** in baking the pastry using singleton class microwave oven.

PROBLEM STATEMENT

You start your Pastry Shop that offer different types of Pastries Pineapple with chocolate, Strawberry with apple, Vanilla 'O' custard, Orange 'E' chocolate and many more No doubt about that in the future more such delicious are expected. You also need to generate a bill to the customer this way.

1. Pineapple = 44 Rs
2. Chocolate = 24 Rs
3. Pineapple with Chocolate with Chocolate = $44 + 24 + 24 = 92$ Rs

Cost of Custard = 64 Rs

1. Strawberry = 34 Rs
2. Vanilla = 39 Rs

Write a program that can take care of present offerings and future offerings for billing. After some time, the Pastry Shop owner realised that the customer cherished single Pastry flavours more than the combination of two pastry flavours. Therefore, he came up with new menu that followed pastry rigid procedure.

Example: -

In order to prepare the Pineapple pastry, one needs to add 7ML of pineapple essence add corn starch min for about 10 min and the pastry is served by the waiter.

For Chocolate Pastry one needs to add 1ML of chocolate, add chocolate powder min for about 7 min and the pastry is served by the waiter.

For Strawberry one need to add 27 grams of strawberries add brown flavour min for about 6 min and then the pastry is served by the waiter.

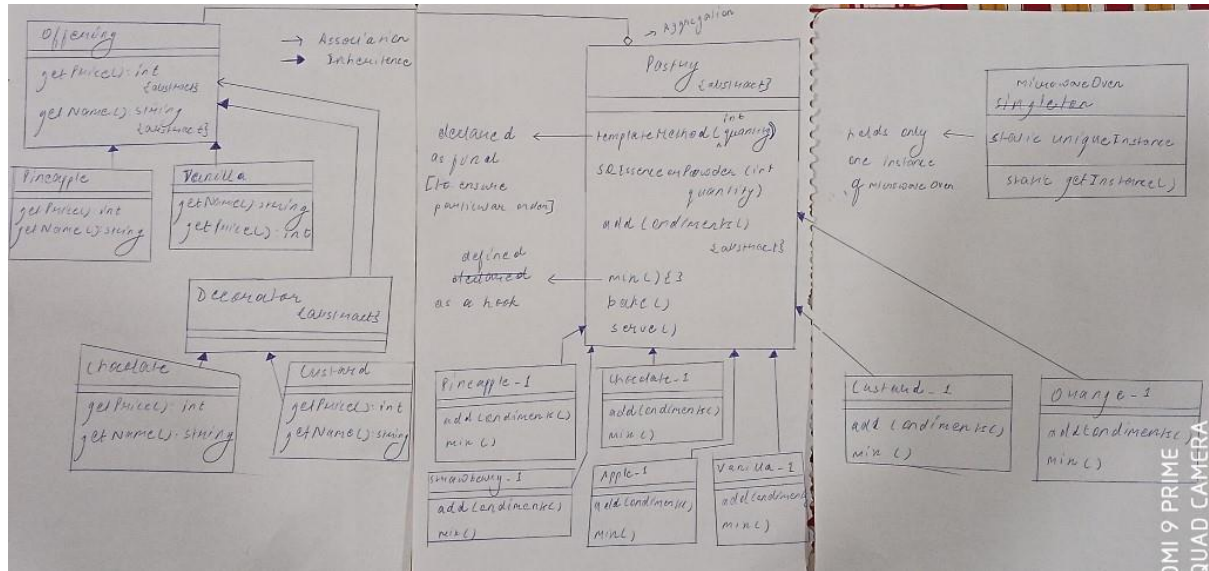
For Apple Pastry one needs to add 13 ML of apple widen vinegar add apple minutes for about 12 minutes and then the pastry is served by the waiter.

For Vanilla Pastry one needs to add 5ML of vanilla essence add wheat flavour, minutes for about 2 minutes and the pastry served by the waiter.

For Custard Pastry one needs to add 12 grams of custard powder add corn flavour, min for about 5 minutes and then the pastry is served by the waiter.

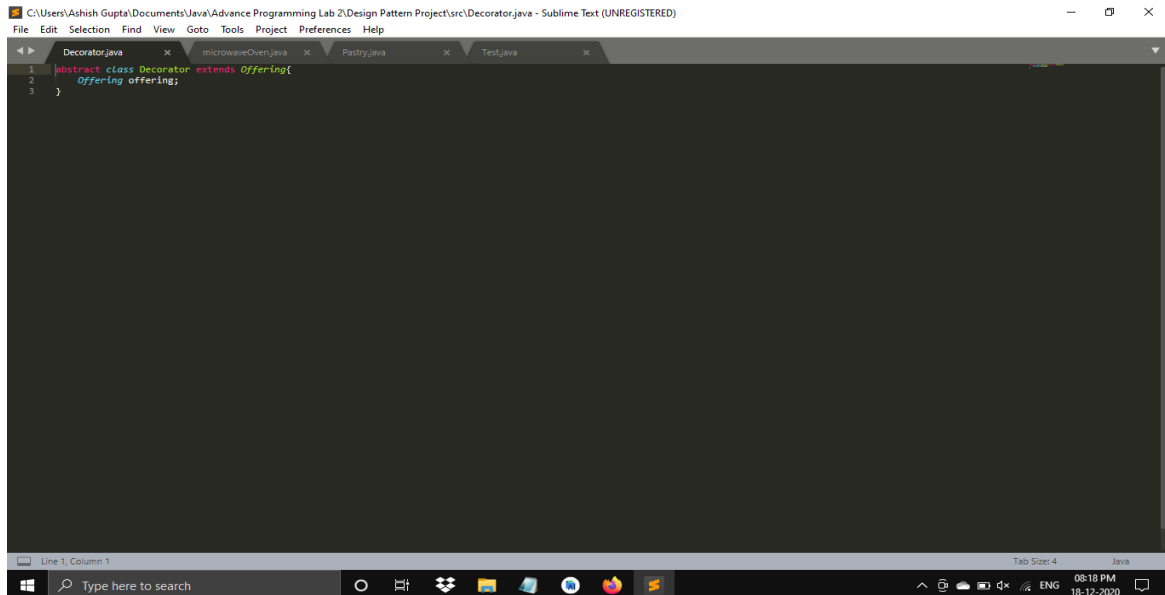
For Orange Pastry one needs to add 17 grams of orange extracts add min for about 8 minutes and the pastry is served by the waiter.

CLASSDIAGRAM



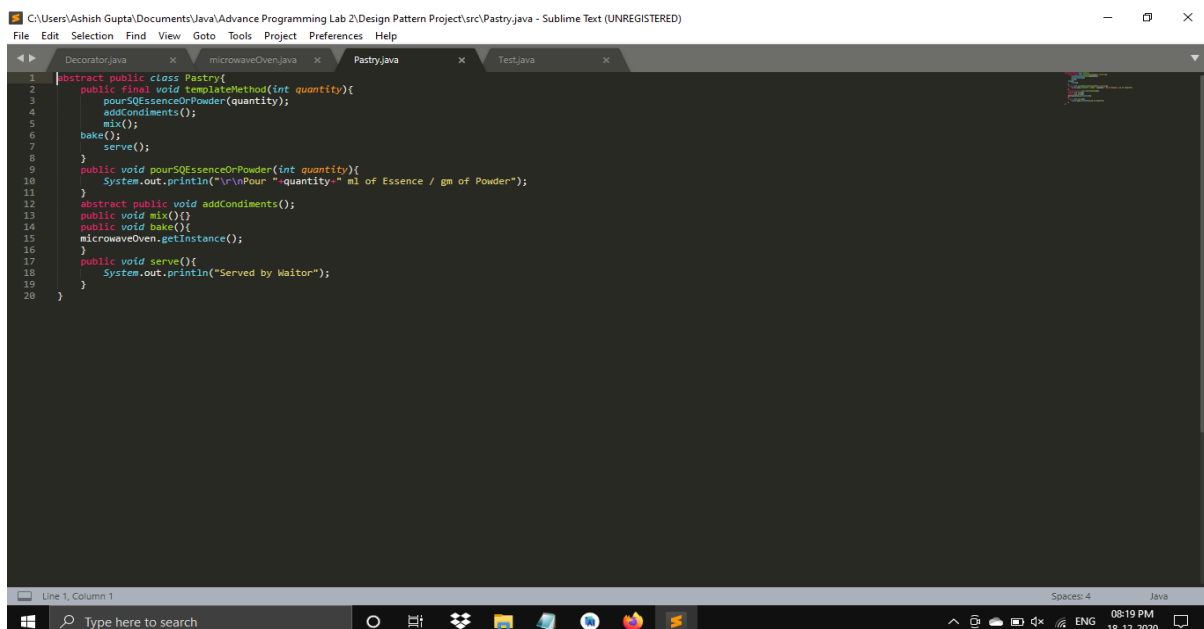
IMPLEMENTATION (CODE)

Now the **Decorator Class** is the main game changer as it enables one to add new classes in the future without affecting the behaviour of the previously created classes.



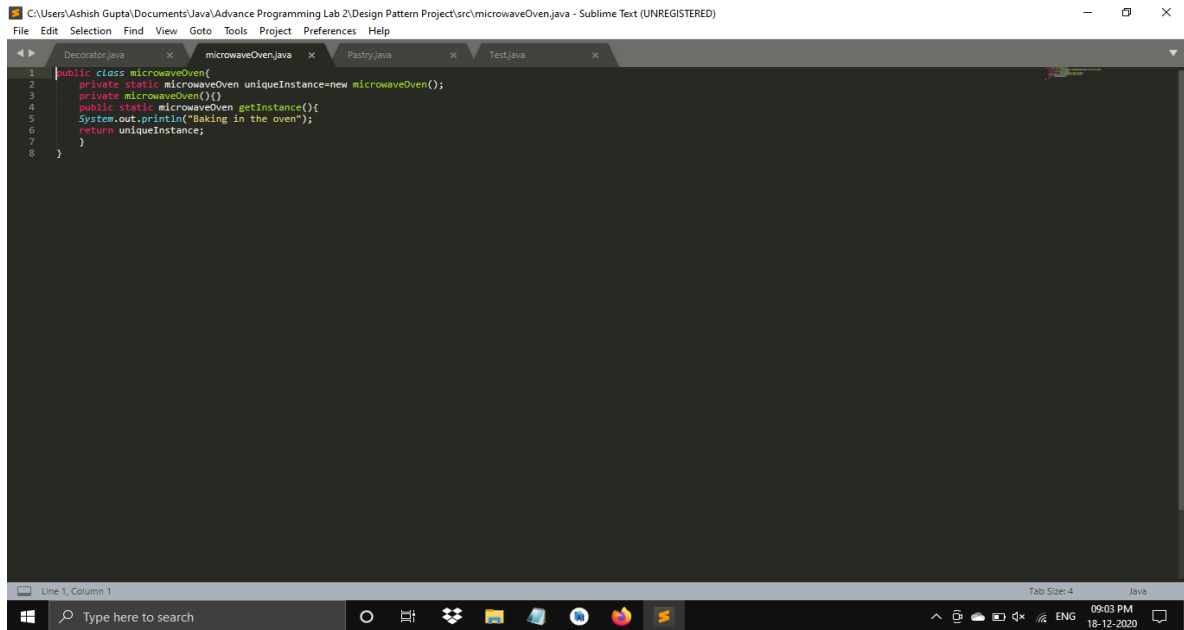
```
C:\Users\Ashish Gupta\Documents\Java\Advance Programming Lab 2\Design Pattern Project\src\Decorator.java - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
Decorator.java x microwaveOven.java x Pastry.java x Test.java x
1 abstract class Decorator extends Offering{
2     Offering offering;
3 }
Line 1, Column 1
Type here to search
Tab Size: 4
Java
08:18 PM
18-12-2020
```

The commonality between the classes is abstracted in the classes Pastry which therefore acts as a template for addition of new classes.

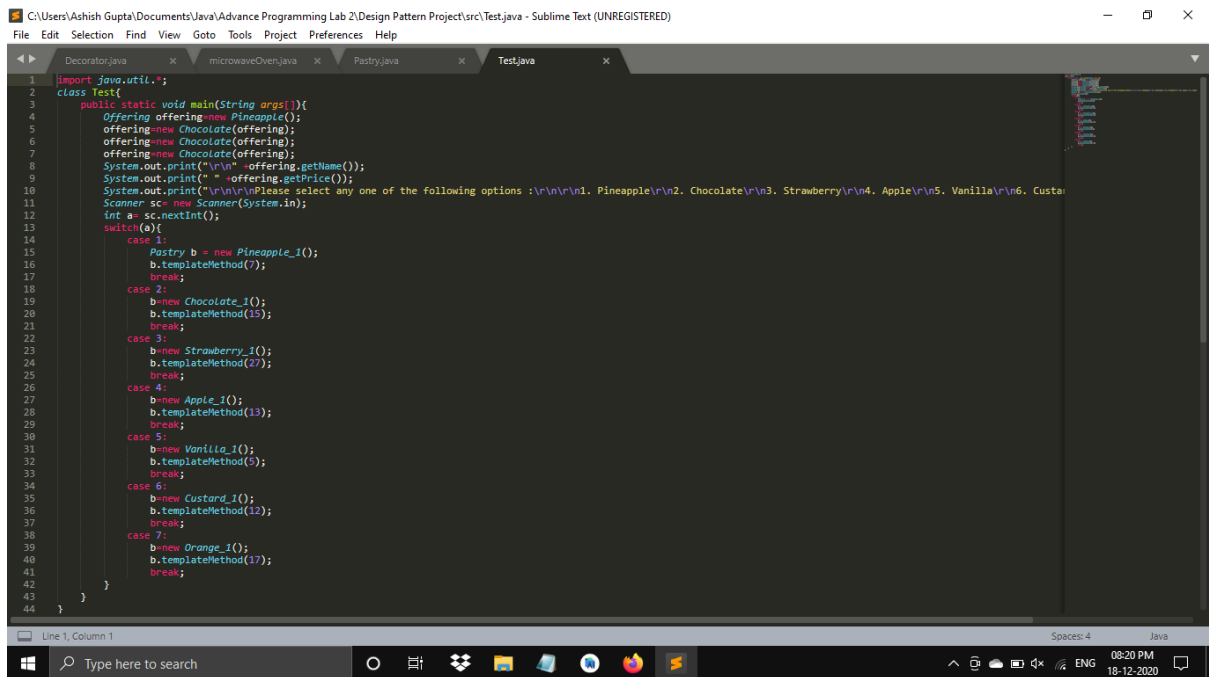


```
C:\Users\Ashish Gupta\Documents\Java\Advance Programming Lab 2\Design Pattern Project\src\Pastry.java - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
Decorator.java x microwaveOven.java x Pastry.java x Test.java x
1 abstract public class Pastry{
2     public final void templateMethod(int quantity){
3         pourSQEssenceOrPowder(quantity);
4         addCondiments();
5         mix();
6         bake();
7         serve();
8     }
9     public void pourSQEssenceOrPowder(int quantity){
10        System.out.println("\n\nPour "+quantity+" ml of Essence / gm of Powder");
11    }
12    abstract public void addCondiments();
13    public void mix(){
14    }
15    public void bake(){
16        microwaveOven.getInstance();
17    }
18    public void serve(){
19        System.out.println("Served by Waitor");
20    }
21 }
Line 1, Column 1
Type here to search
Tab Size: 4
Java
08:19 PM
18-12-2020
```

The microwaveOven class is created in order to get a unique instance of Singleton Pattern.



```
1 public class microwaveOven{
2     private static microwaveOven uniqueInstance=new microwaveOven();
3     private microwaveOven(){
4     }
5     public static microwaveOven getInstance(){
6         System.out.println("Baking in the oven");
7         return uniqueInstance;
8     }
9 }
```



```
1 import java.util.*;
2 class Test{
3     public static void main(String args[]){
4         Offering offering=new Pineapple();
5         offering=new Chocolate(offering);
6         offering=new Chocolate(offering);
7         offering=new Chocolate(offering);
8         System.out.print("\n\n" +offering.getName());
9         System.out.print(" " +offering.getPrice());
10        System.out.print("\n\nPlease select any one of the following options :\r\n\r\n1. Pineapple\r\n2. Chocolate\r\n3. Strawberry\r\n4. Apple\r\n5. Vanilla\r\n6. Custard\r\n7. Orange\r\n");
11        Scanner sc= new Scanner(System.in);
12        int a= sc.nextInt();
13        switch(a){
14            case 1:
15                Pastry b = new Pineapple_1();
16                b.templateMethod(7);
17                break;
18            case 2:
19                b=new Chocolate_1();
20                b.templateMethod(15);
21                break;
22            case 3:
23                b=new Strawberry_1();
24                b.templateMethod(27);
25                break;
26            case 4:
27                b=new Apple_1();
28                b.templateMethod(13);
29                break;
30            case 5:
31                b=new Vanilla_1();
32                b.templateMethod(5);
33                break;
34            case 6:
35                b=new Custard_1();
36                b.templateMethod(12);
37                break;
38            case 7:
39                b=new Orange_1();
40                b.templateMethod(17);
41                break;
42        }
43    }
44 }
```

Apple_1.java

```
public class Apple_1 extends Pastry{  
    public void addCondiments(){  
        System.out.println("Add Apple Pie Please");  
    }  
    public void mix(){  
        System.out.println("Mix for about 11 minutes");  
    }  
}
```

Apple_1_1.java

```
class Apple_1_1{  
    void prepareApple() {  
        pourAppleCiderVinegar();  
        addApplePie();  
        mix();  
        serve();  
    }  
    void pourAppleCiderVinegar(){  
        System.out.println("Pour 13 ml of Apple Cider Vinegar in a container");  
    }  
    void addApplePie(){  
        System.out.println("Add Apple Pie Please");  
    }  
    void mix(){  
        System.out.println("Mix for about 11 minutes");  
    }  
}
```



```
void serve() {  
    System.out.println("Waiter will serve");  
}  
}
```

Chocolate.java

```
class Chocolate extends Decorator {  
    Chocolate(Offering offering){  
        this.offering=offering;  
    }  
  
    String getName(){  
        return offering.getName()+" with Chocolate";  
    }  
  
    int getPrice(){  
        return offering.getPrice()+24;  
    }  
}
```

Chocolate_1.java

```
public class Chocolate_1 extends Pastry{  
    public void addCondiments(){  
        System.out.println("Add Chocolate Powder Please");  
    }  
  
    public void mix(){  
        System.out.println("Mix for about 7 minutes");  
    }  
}
```

Chocolate_1_1.java

```
class Chocolate_1_1{
    void prepareChocolate(){
        pourChocolateSyrup();
        addChocolatePowder();
        mix();
        serve();
    }
    void pourChocolateSyrup(){
        System.out.println("Pour 15 ml of Chocolate Syrup in a container");
    }
    void addChocolatePowder(){
        System.out.println("Add Chocolate Powder Please");
    }
    void mix(){
        System.out.println("Mix for about 7 minutes");
    }
    void serve(){
        System.out.println("Waiter will serve");
    }
}
```

Custard.java

```
class Custard extends Decorator {
    Custard (Offering offering) {
        this.offering=offering;
    }
}
```

```
String getName(){
    return offering.getName()+" with Custard";
}
```

```
int getPrice(){
    return offering.getPrice()+64;
}
}
```

Custard_1.java

```
public class Custard_1 extends Pastry {
    public void addCondiments(){
        System.out.println("Add Corn Flour Please");
    }
    public void mix(){
        System.out.println("Mix for about 5 minutes");
    }
}
```

Custard_1_1.java

```
class Custard_1_1{
    void prepareCustard(){
        pourCustardPowder();
        addCornFlour();
        mix();
        serve();
    }
    void pourCustardPowder(){
        System.out.println("Pour 12 gm of Custard Powder in a container");
    }
}
```

```

void addCornFlour{
    System.out.println("Add Corn Flour Please");
}
void mix(){
    System.out.println("Mix for about 5 minutes");
}
void serve(){
    System.out.println("Waiter will serve");
}
}

```

Decorator.java

```

abstract class Decorator extends Offering{
    Offering offering;
}

```

microwaveOven.java

```

public class microwaveOven{
    private static microwaveOven uniqueInstance=new microwaveOven();
    private microwaveOven(){}
    public static microwaveOven getInstance(){
        System.out.println("Baking in the oven");
        return uniqueInstance;
    }
}

```

Offering.java

```

abstract class Offering{
    abstract int getPrice();
    abstract String getName();
}

```

Orange_1.java

```
public class Orange_1 extends Pastry{  
    public void addCondiments(){  
        System.out.println("Add Orange Extracts Please");  
    }  
    public void mix(){  
        System.out.println("Mix for about 8 minutes");  
    }  
}
```

Orange_1_1.java

```
class Orange_1_1{  
    void prepareOrange(){  
        pourOrangeExtracts();  
        addRiceFlour();  
        mix();  
        serve();  
    }  
    void pourOrangeExtracts(){  
        System.out.println("Pour 17 gm of Orange Extracts in a container");  
    }  
    void addRiceFlour(){  
        System.out.println("Add Rice Flour Please");  
    }  
    void mix(){  
        System.out.println("Mix for about 8 minutes");  
    }  
}
```

```

void serve(){
    System.out.println("Waiter will serve");
}
}

```

Orange_1_10.java

```

public class Orange_1_10 extends Pastry_10{
    public void addCondiments(){
        System.out.println("Add Orange Extracts Please");
    }
    public void mix(){
        System.out.println("Mix for about 8 minutes");
    }
}

```

Pastry.java

```

abstract public class Pastry{
    public final void templateMethod(int quantity){
        pourSQEssenceOrPowder(quantity);
        addCondiments();
        mix();
    }
    bake();
    serve();
}
public void pourSQEssenceOrPowder(int quantity){
    System.out.println("\r\nPour "+quantity+" ml of Essence / gm of Powder");
}
abstract public void addCondiments();

```

```

public void mix(){}

public void bake(){
microwaveOven.getInstance();

}

public void serve(){

    System.out.println("Served by Waitor");

}

}

```

Pastry_10.java

```

abstract public class Pastry_10{

    public final void templateMethod(int quantity){

        pourSQEssenceOrPowder(quantity);

        addCondiments();

        mix();
bake();

        serve();

    }

    public void pourSQEssenceOrPowder(int quantity){

        System.out.println("\r\nPour "+quantity+" ml of Essence / gm of Powder");

    }

    abstract public void addCondiments();

    public void mix(){}

    public void bake(){

        microwaveOven.getInstance();

    }

    public void serve(){

        System.out.println("Served by Waitor");

        }}

```

Pineapple.java

```
class Pineapple extends Offering{

    String getName(){

        return "Pineapple";

    }

    int getPrice(){

        return 44;

    }

}
```

Pineapple_1.java

```
public class Pineapple_1 extends Pastry{

    public void addCondiments(){

        System.out.println("Add Corn Starch Please");

    }

    public void mix(){

        System.out.println("Mix for about 10 minutes");

    }

}
```

Pineapple_1_1.java

```
class Pineapple_1_1{

    void preparePineapple(){

        pourPineappleEssence();

        addCornStarch();

        mix();

        serve();

    }

}
```



```

void pourPineappleEssence(){
    System.out.println("Pour 7 ml of Pineapple Essence in a container");
}
void addCornStarch(){
    System.out.println("Add Corn Starch Please");
}
void mix(){
    System.out.println("Mix for about 10 minutes");
}
void serve(){
    System.out.println("Waiter will serve");
}
}

```

Pineapple_1_10.java

```

public class Pineapple_1_10 extends Pastry_10{
    public void addCondiments(){
        System.out.println("Add Corn Starch Please");
    }
    public void mix(){
        System.out.println("Mix for about 10 minutes");
    }
}

```

Strawberry_1.java

```

public class Strawberry_1 extends Pastry{
    public void addCondiments(){
        System.out.println("Add Brown Flour Please");
    }
}

```

```
public void mix(){  
    System.out.println("Mix for about 6 minutes");  
}  
}
```

Strawberry_1_1.java

```
class Strawberry_1_1{  
    void prepareStrawberry(){  
        pourStrawberries();  
        addBrownFlour();  
        mix();  
        serve();  
    }  
    void pourStrawberries(){  
        System.out.println("Pour 27 gm of Strawberries in a container");  
    }  
    void addBrownFlour(){  
        System.out.println("Add Brown Flour Please");  
    }  
    void mix(){  
        System.out.println("Mix for about 6 minutes");  
    }  
    void serve(){  
        System.out.println("Waiter will serve");  
    }  
}
```

Vanilla.java

```
class Vanilla extends Offering{

    String getName(){

        return "Vanilla";

    }

    int getPrice(){

        return 39;

    }

}
```

Vanilla_1.java

```
public class Vanilla_1 extends Pastry{

    public void addCondiments(){

        System.out.println("Add Wheat Flour Please");

    }

    public void mix(){

        System.out.println("Mix for about 2 minutes");

    }

}
```

Vanilla_1_1.java

```
class Vanilla_1_1{

    void prepareVanilla(){

        pourVanillaEssence();

        addWheatFlour();

        mix();

        serve();

    }

}
```

```

void pourVanillaEssence(){
    System.out.println("Pour 5 ml of Vanilla Essence in a container");
}

void addWheatFlour(){
    System.out.println("Add Wheat Flour Please");
}

void mix(){
    System.out.println("Mix for about 2 minutes");
}

void serve(){
    System.out.println("Waiter will serve");
}
}

```

Test.java

```

import java.util.*;

class Test{

    public static void main(String args[]){
        Offering offering=new Pineapple();
        offering=new Chocolate(offering);
        offering=new Chocolate(offering);
        offering=new Chocolate(offering);
        System.out.print("\r\n" +offering.getName());
        System.out.print(" " +offering.getPrice());

        System.out.print("\r\n\r\nPlease select any one of the following options :\r\n\r\n1.
Pineapple\r\n2. Chocolate\r\n3. Strawberry\r\n4. Apple\r\n5. Vanilla\r\n6. Custard\r\n7.
Orange\r\n\r\n\r\n");

        Scanner sc= new Scanner(System.in);

        int a= sc.nextInt();
    }
}

```

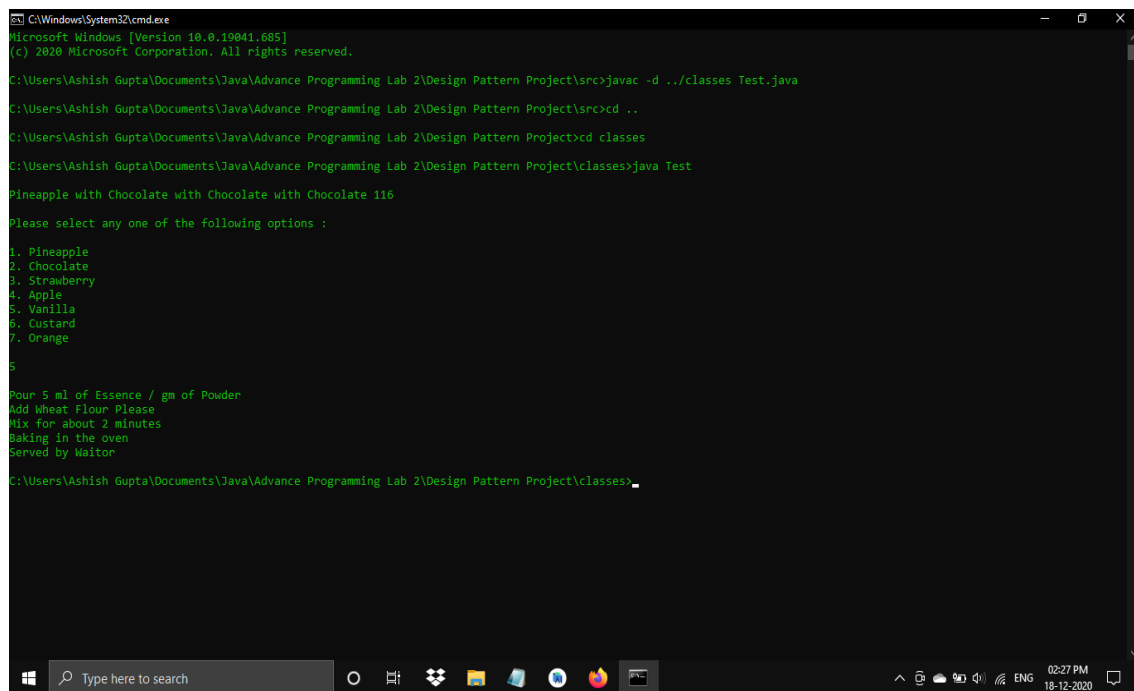
```
switch(a){  
    case 1:  
        Pastry b = new Pineapple_1();  
        b.templateMethod(7);  
        break;  
    case 2:  
        b=new Chocolate_1();  
        b.templateMethod(15);  
        break;  
    case 3:  
        b=new Strawberry_1();  
        b.templateMethod(27);  
        break;  
    case 4:  
        b=new Apple_1();  
        b.templateMethod(13);  
        break;  
    case 5:  
        b=new Vanilla_1();  
        b.templateMethod(5);  
        break;  
    case 6:  
        b=new Custard_1();  
        b.templateMethod(12);  
        break;  
    case 7:  
        b=new Orange_1();
```

```
b.templateMethod(17);
```

```
    break;
```

```
}}
```

OUTPUT



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19041.685]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\Ashish Gupta\Documents\Java\Advance Programming Lab 2\Design Pattern Project\src>javac -d ../classes Test.java
C:\Users\Ashish Gupta\Documents\Java\Advance Programming Lab 2\Design Pattern Project\src>cd ..
C:\Users\Ashish Gupta\Documents\Java\Advance Programming Lab 2\Design Pattern Project>cd classes
C:\Users\Ashish Gupta\Documents\Java\Advance Programming Lab 2\Design Pattern Project\classes>java Test
Pineapple with Chocolate with Chocolate 116
Please select any one of the following options :
1. Pineapple
2. Chocolate
3. Strawberry
4. Apple
5. Vanilla
6. Custard
7. Orange
8
Pour 5 ml of Essence / gm of Powder
Add Wheat Flour Please
Mix for about 2 minutes
Baking in the oven
Served by Waitor
C:\Users\Ashish Gupta\Documents\Java\Advance Programming Lab 2\Design Pattern Project\classes>_
```

References

<https://www.javatpoint.com/design-patterns-in-java>

https://www.tutorialspoint.com/design_pattern/design_pattern_overview.htm

<https://www.geeksforgeeks.org/design-patterns-set-1-introduction/>

https://www.youtube.com/results?search_query=design+patterns+in+java