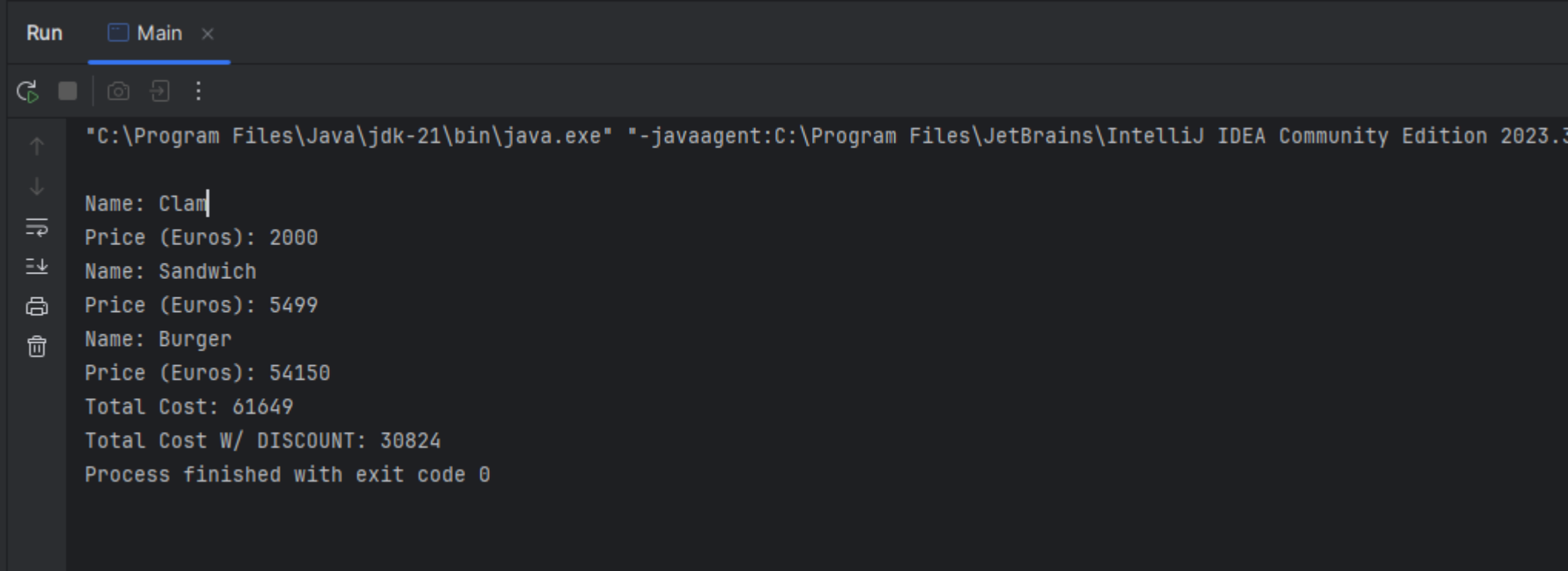
Marvin Sevilla

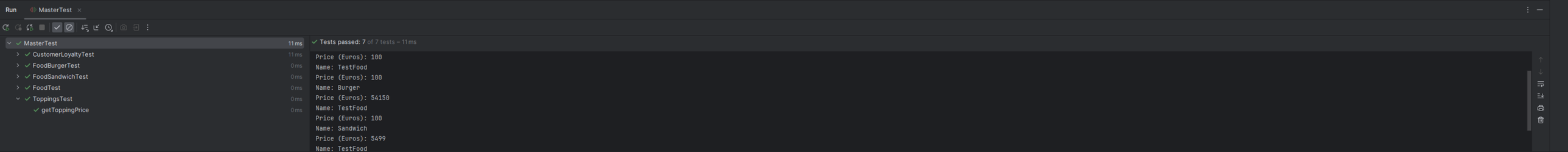
CS 4800.01

29 March, 2024

GitHub Link

https://github.com/LoloMarty/2024-/tree/main/CS4800/Homework5


Run
Main
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3
Name: Clam
Price (Euros): 2000
Name: Sandwich
Price (Euros): 5499
Name: Burger
Price (Euros): 54150
Total Cost: 61649
Total Cost W/ DISCOUNT: 30824
Process finished with exit code 0


Run
MasterTest
MasterTest 11 ms
CustomerLoyaltyTest 11 ms
FoodBurgerTest 0 ms
FoodSandwichTest 0 ms
FoodTest 0 ms
ToppingsTest 0 ms
getToppingPrice 0 ms
Tests passed: 7 of 7 tests – 11 ms
Price (Euros): 100
Name: TestFood
Price (Euros): 100
Name: Burger
Price (Euros): 54150
Name: TestFood
Price (Euros): 100
Name: Sandwich
Price (Euros): 5499
Name: TestFood
omework5
src
13:28
CRLF
UTF-8
4 spaces

SOURCE CODE

```
1  public class Main {
2      public static void main(String[] args)
3      {
4          //IFood discount = new CustomerLoyalty(new FoodBurger( new FoodSandwich( new Food(2000, "Clam", r
5
6          IFood baseFood = new Food(2000, "Clam", new String[] {});
7          IFood sandwich = new FoodSandwich(baseFood);
8          IFood burger = new FoodBurger(sandwich);
9          CustomerLoyalty discount = new CustomerLoyalty(burger);
10
11
12         discount.calculateCost(2);
13     }
14 }
```

```java
public interface IFood {
    public Integer getToppingPrice(String topping);
    public int calculateCost();
}
```

```java
import java.util.Hashtable;

class Toppings {
    private static Toppings toppingsInstance;
    private static Hashtable<String, Integer> allPossibleToppings;

    private Toppings()
    {
        allPossibleToppings = new Hashtable<>();
        allPossibleToppings.put("Onions", 120);
        allPossibleToppings.put("Tomatoes", 50);
        allPossibleToppings.put("Gold Flakes", 50000);
        allPossibleToppings.put("Bacon Bits", 200);
        allPossibleToppings.put("Ranch", 150);
        allPossibleToppings.put("Vegan Ranch", 149);
        allPossibleToppings.put("Fake Ranch", 151);
        allPossibleToppings.put("Mushrooms", 30);
    }

    public static Toppings getInstance()
    {
        if(toppingsInstance == null)
        {
            toppingsInstance = new Toppings();
        }

        return toppingsInstance;
    }

    public Integer getToppingPrice(String topping)
    {
        return allPossibleToppings.get(topping);
    }
}
```

```java
import java.util.Hashtable;

public abstract class FoodBase implements IFood{
    protected int basePrice;
    protected String foodName;
    protected String[] addedToppings;
    private final IFood wrapped;

    public FoodBase(IFood givenWrapped)
    {
        this.wrapped = givenWrapped;
    }

    public Integer getToppingPrice(String topping)
    {
        return Toppings.getInstance().getToppingPrice(topping);
    }

    @Override
    public int calculateCost()
    {
        return wrapped.calculateCost();
    }

}
```

```java
import java.util.Hashtable;

public class Food implements IFood {
    private int basePrice;
    private String foodName;
    private String[] addedToppings;

    public Food(int basePrice, String foodName, String[] addedToppings)
    {
        this.basePrice = basePrice;
        this.foodName = foodName;
        this.addedToppings = addedToppings;
    }

    public Integer getToppingPrice(String topping)
    {
        return Toppings.getInstance().getToppingPrice(topping);
    }

    @Override
    public int calculateCost() {
        int additionalPrice = 0;
        for(String topping: this.addedToppings)
        {

            additionalPrice += this.getToppingPrice(topping);
        }

        System.out.printf("\nName: %s\nPrice (Euros): %d", this.foodName, this.basePrice+additionalPrice
        return this.basePrice + additionalPrice;
    }
}
```

```
public class CustomerLoyalty extends FoodBase{
    final double oneYearDiscountRate = 0.15;
    final double twoYearDiscountRate = 0.5;
    final double threeYearDiscountRate = 0.9;
    public CustomerLoyalty(IFood wrapped)
    {
        super(wrapped);
    }


    public int calculateCost(int CustomerLoyaltyYear) {
        int totalCost = super.calculateCost();
        double appliedDiscountRate = 0;

        if(CustomerLoyaltyYear == 3)
        {
            appliedDiscountRate = 1-this.threeYearDiscountRate;
        }else if (CustomerLoyaltyYear == 2)
        {
            appliedDiscountRate = 1-this.twoYearDiscountRate;
        }else if(CustomerLoyaltyYear == 1)
        {
            appliedDiscountRate = 1-this.oneYearDiscountRate;
        }else{
            appliedDiscountRate = 1;
        }

        System.out.printf("\n\nTotal Cost: %d\nTotal Cost W/ DISCOUNT: %d", (int)totalCost, (int)(totalC

        return (int)(totalCost * appliedDiscountRate);
    }
}
```

```java
import java.sql.Array;

public class FoodBurger extends FoodBase{

    public FoodBurger(IFood wrapped)
    {
        super(wrapped);
        this.basePrice = 4000;
        this.foodName = "Burger";
        this.addedToppings = new String[]{"Onions", "Gold Flakes", "Mushrooms"};
    }

    @Override
    public int calculateCost() {
        int carriedPrice = super.calculateCost();
        int additionalPrice = 0;

        for(String topping: this.addedToppings)
        {

            additionalPrice += this.getToppingPrice(topping);
        }

        System.out.printf("\nName: %s\nPrice (Euros): %d", this.foodName, this.basePrice+additionalPrice
        return this.basePrice + additionalPrice + carriedPrice;
    }
}
```

```java
public class FoodSandwich extends FoodBase{
    public FoodSandwich(IFood wrapped)
    {
        super(wrapped);
        this.basePrice = 5000;
        this.foodName = "Sandwich";
        this.addedToppings = new String[]{"Ranch", "Vegan Ranch", "Bacon Bits"};
    }

    @Override
    public int calculateCost() {
        int carriedPrice = super.calculateCost();
        int additionalPrice = 0;

        for(String topping: this.addedToppings)
        {

            additionalPrice += this.getToppingPrice(topping);
        }

        System.out.printf("\nName: %s\nPrice (Euros): %d", this.foodName, this.basePrice+additionalPrice
        return this.basePrice + additionalPrice + carriedPrice;
    }
}
```

TESTS

```
1
2  import org.junit.Test;
3  import static org.junit.Assert.*;
4  import org.junit.runner.RunWith;
5  import org.junit.runners.Suite;
6
7  @RunWith(Suite.class)
8  @Suite.SuiteClasses({
9          CustomerLoyaltyTest.class,
10         FoodBurgerTest.class,
11         FoodSandwichTest.class,
12         FoodTest.class,
13         ToppingsTest.class,
14 })
15
16 public class MasterTest {
17 }
```

```java
import org.junit.Test;

import static org.junit.Assert.*;

public class CustomerLoyaltyTest {

    @Test
    public void calculateCost() {
        CustomerLoyalty testCustomer = new CustomerLoyalty(new Food(100, "TestFood", new String[]{}));

        int expected = 9;
        int actual = testCustomer.calculateCost(3);

        assertEquals(expected, actual);
    }

    @Test
    public void getToppingsPrice()
    {
        CustomerLoyalty testCustomer = new CustomerLoyalty(new Food(100, "TestFood", new String[]{}));

        int expected = 151;
        int actual = testCustomer.getToppingPrice("Fake Ranch");

        assertEquals(expected, actual);
    }

}
```

```java
import static org.junit.Assert.*;

public class FoodBurgerTest {

    @org.junit.Test
    public void calculateCost() {
        FoodBurger testCustomer = new FoodBurger(new Food(100, "TestFood", new String[]{}));

        int expected = 54250;
        int actual = testCustomer.calculateCost();

        assertEquals(expected, actual);
    }
}
```

```java
import org.junit.Test;

import static org.junit.Assert.*;

public class FoodSandwichTest {

    @Test
    public void calculateCost() {
        FoodSandwich testCustomer = new FoodSandwich(new Food(100, "TestFood", new String[]{}));

        int expected = 5599;
        int actual = testCustomer.calculateCost();

        assertEquals(expected, actual);
    }
}
```

```java
import org.junit.Test;

import static org.junit.Assert.*;

public class FoodTest {

    @Test
    public void getToppingPrice() {
        Food testFood = new Food(100, "TestFood", new String[]{});

        int expected = 120;
        int actual = testFood.getToppingPrice("Onions");

        assertEquals(expected, actual);
    }

    @Test
    public void calculateCost() {
        Food testFood = new Food(100, "TestFood", new String[]{"Onions"});

        int expected = 220;
        int actual = testFood.calculateCost();

        assertEquals(expected, actual);
    }
}
```

```java
import org.junit.Test;

import static org.junit.Assert.*;

public class ToppingsTest {

    @Test
    public void getToppingPrice() {
        int expected = 200;
        int actual = Toppings.getInstance().getToppingPrice("Bacon Bits");

        assertEquals(expected, actual);
    }
}
```