Pizza Ordering System:

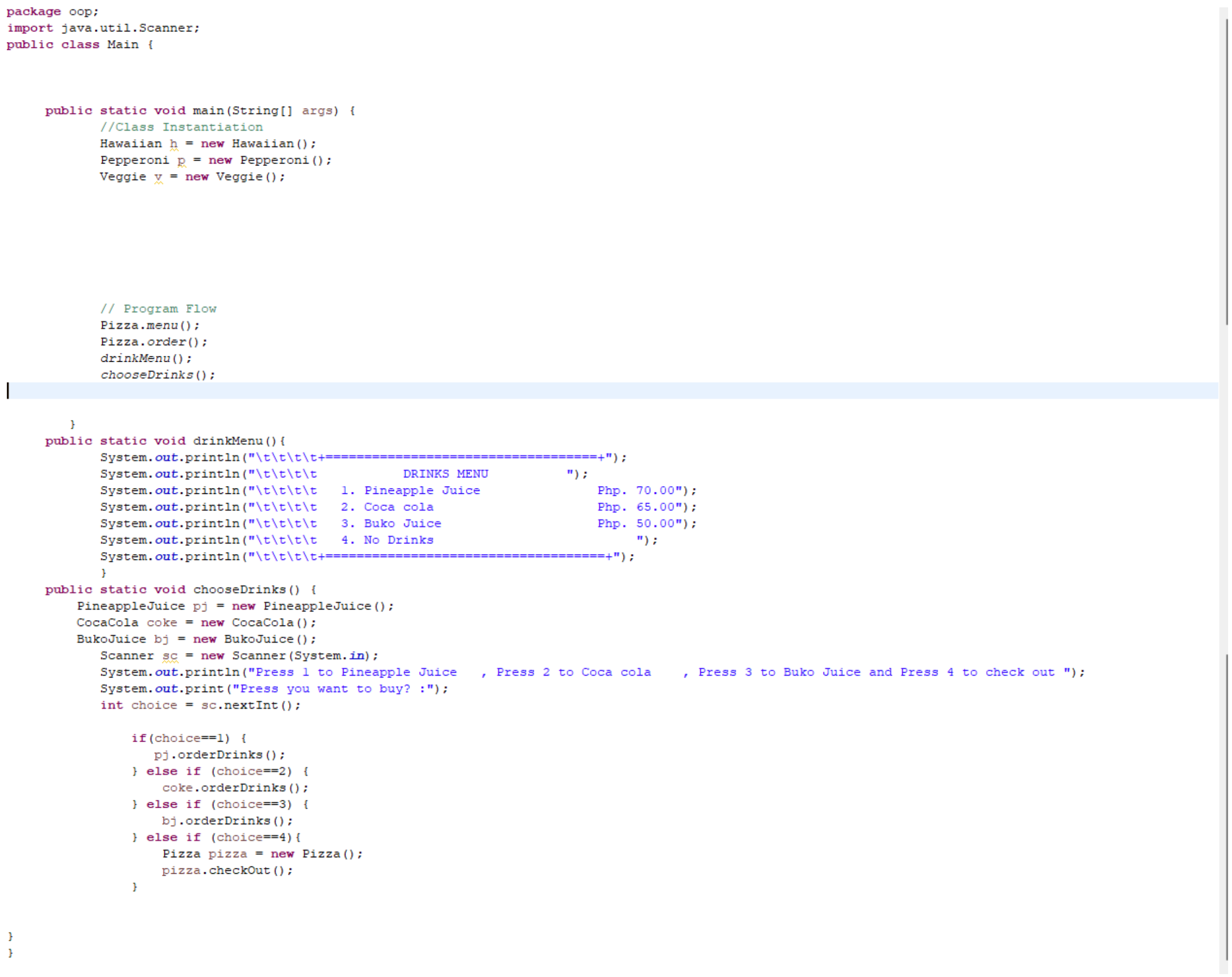
An Object-Oriented Programming Case Study

Banag, Russell D.,

Bustamante, Bea Bianca

Figure I. Class Main

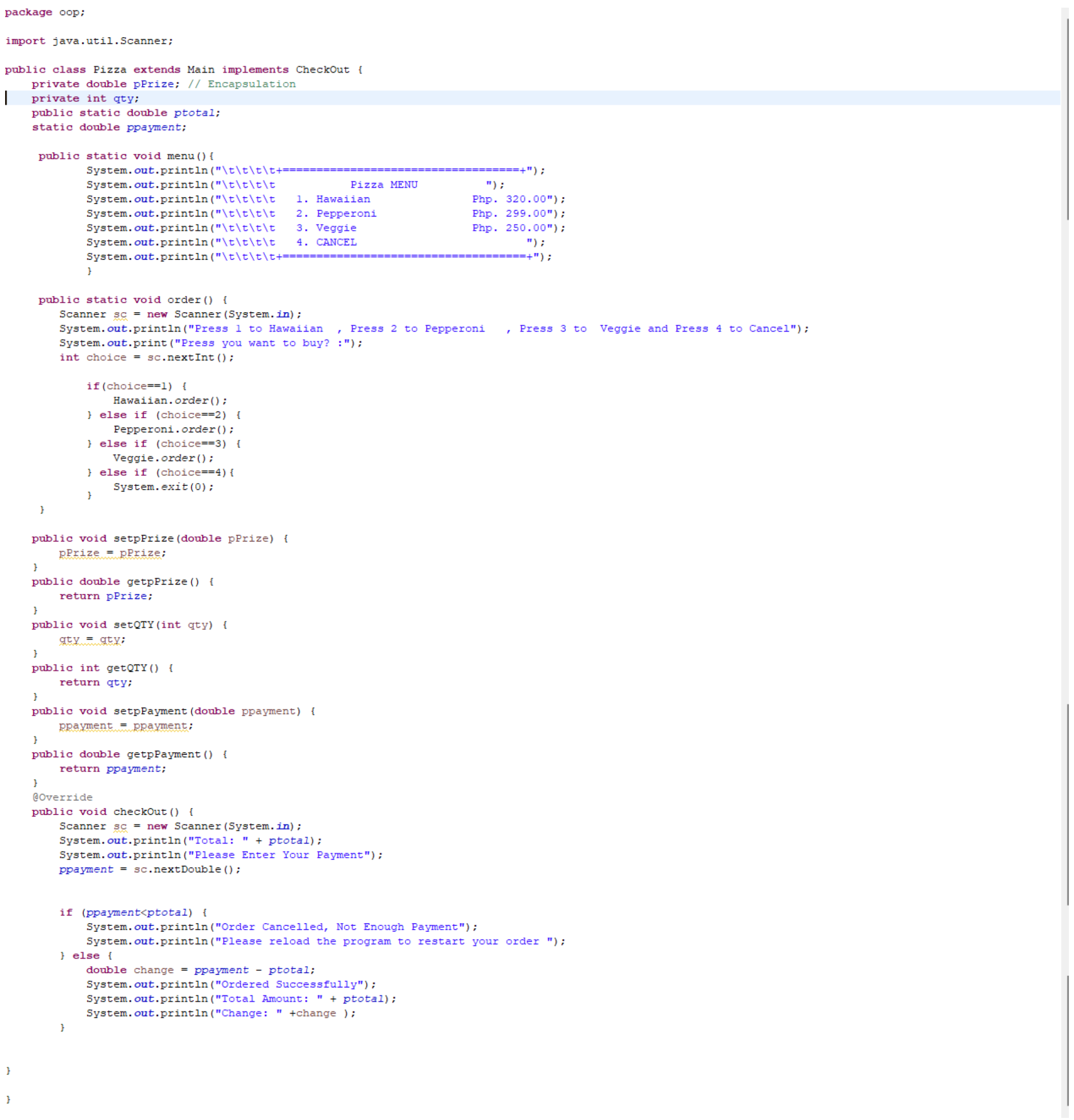
The Main Class shows the instantiation of objects, the program flow, and some methods that are supposedly for the Class Drinks which is an abstract class.



```java
package oop;
import java.util.Scanner;
public class Main {



    public static void main(String[] args) {
        //Class Instantiation
        Hawaiian h = new Hawaiian();
        Pepperoni p = new Pepperoni();
        Veggie v = new Veggie();







        // Program Flow
        Pizza.menu();
        Pizza.order();
        drinkMenu();
        chooseDrinks();


        }
    public static void drinkMenu(){
        System.out.println("\t\t\t\t+====================================+");
        System.out.println("\t\t\t\t           DRINKS MENU          ");
        System.out.println("\t\t\t\t   1. Pineapple Juice              Php. 70.00");
        System.out.println("\t\t\t\t   2. Coca cola                    Php. 65.00");
        System.out.println("\t\t\t\t   3. Buko Juice                   Php. 50.00");
        System.out.println("\t\t\t\t   4. No Drinks                         ");
        System.out.println("\t\t\t\t+=====================================+");
        }
    public static void chooseDrinks() {
        PineappleJuice pj = new PineappleJuice();
        CocaCola coke = new CocaCola();
        BukoJuice bj = new BukoJuice();
           Scanner sc = new Scanner(System.in);
           System.out.println("Press 1 to Pineapple Juice   , Press 2 to Coca cola    , Press 3 to Buko Juice and Press 4 to check out ");
           System.out.print("Press you want to buy? :");
           int choice = sc.nextInt();

               if(choice==1) {
                  pj.orderDrinks();
               } else if (choice==2) {
                   coke.orderDrinks();
               } else if (choice==3) {
                   bj.orderDrinks();
               } else if (choice==4){
                   Pizza pizza = new Pizza();
                   pizza.checkOut();
               }


}
}
```

Figure II. Class Pizza

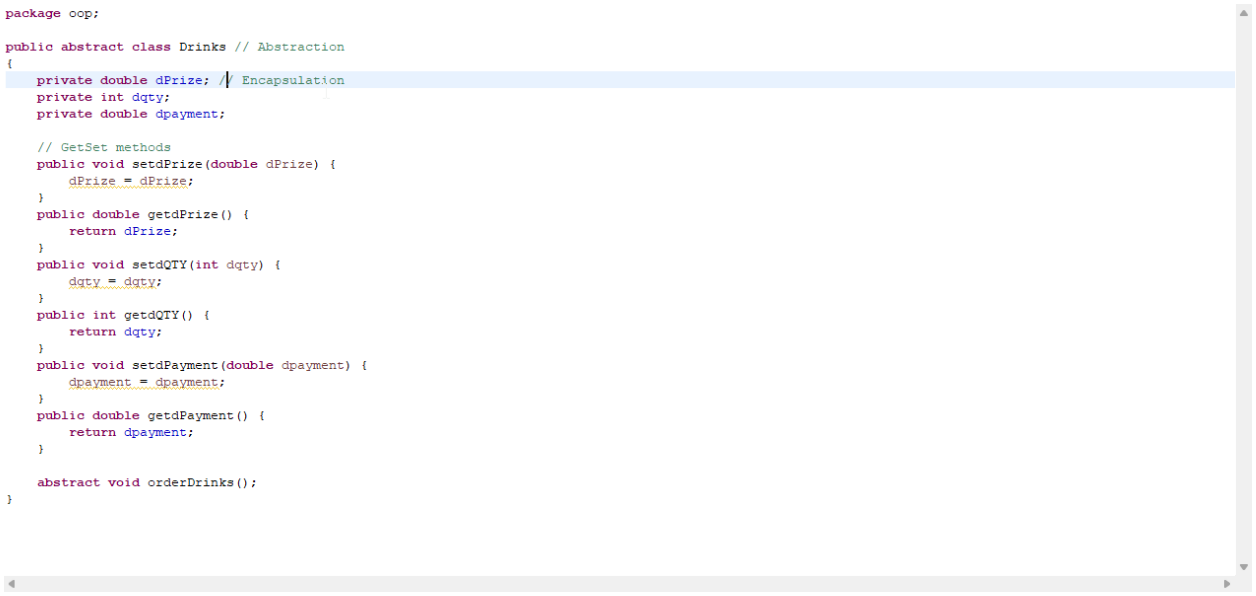
The pizza class contains the menu() for pizza, an order() method and the GetSet method for pizza price, quantity, and payment. It also implements the checkout() method from the checkout interface incase a customer would like to opt out from buying drinks.



```java
package oop;

import java.util.Scanner;

public class Pizza extends Main implements CheckOut {
    private double pPrize; // Encapsulation
    private int qty;
    public static double ptotal;
    static double ppayment;

     public static void menu(){
            System.out.println("\t\t\t\t+===================================+");
            System.out.println("\t\t\t\t           Pizza MENU          ");
            System.out.println("\t\t\t\t   1. Hawaiian               Php. 320.00");
            System.out.println("\t\t\t\t   2. Pepperoni              Php. 299.00");
            System.out.println("\t\t\t\t   3. Veggie                 Php. 250.00");
            System.out.println("\t\t\t\t   4. CANCEL                         ");
            System.out.println("\t\t\t\t+====================================+");
            }

     public static void order() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Press 1 to Hawaiian  , Press 2 to Pepperoni   , Press 3 to  Veggie and Press 4 to Cancel");
        System.out.print("Press you want to buy? :");
        int choice = sc.nextInt();

            if(choice==1) {
                Hawaiian.order();
            } else if (choice==2) {
                Pepperoni.order();
            } else if (choice==3) {
                Veggie.order();
            } else if (choice==4){
                System.exit(0);
            }
     }

    public void setpPrize(double pPrize) {
        pPrize = pPrize;
    }
    public double getpPrize() {
        return pPrize;
    }
    public void setQTY(int qty) {
        qty = qty;
    }
    public int getQTY() {
        return qty;
    }
    public void setpPayment(double ppayment) {
        ppayment = ppayment;
    }
    public double getpPayment() {
        return ppayment;
    }
    @Override
    public void checkOut() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Total: " + ptotal);
        System.out.println("Please Enter Your Payment");
        ppayment = sc.nextDouble();


        if (ppayment<ptotal) {
            System.out.println("Order Cancelled, Not Enough Payment");
            System.out.println("Please reload the program to restart your order ");
        } else {
            double change = ppayment - ptotal;
            System.out.println("Ordered Successfully");
            System.out.println("Total Amount: " + ptotal);
            System.out.println("Change: " +change );
        }


}

}
```

Figure III. Class Drinks

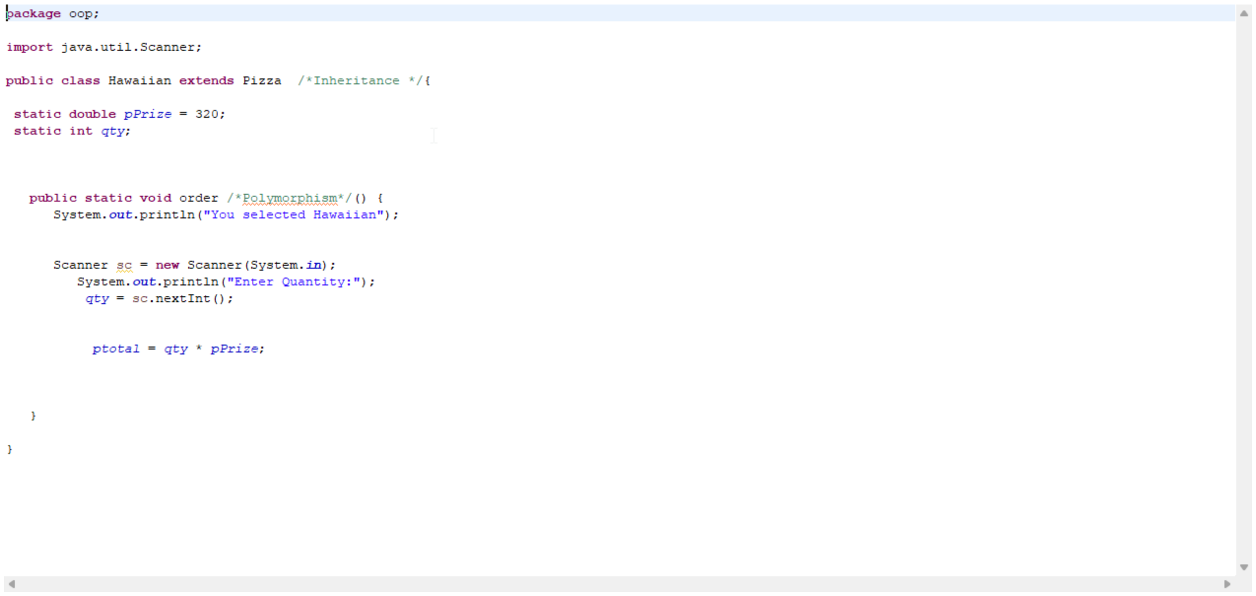
The class drinks has the declaration of drinks prices, quantity, payment and the GetSet method for each attributes. It also has an orderDrink abstact method which will be overridden later.



```java
package oop;

public abstract class Drinks // Abstraction
{
    private double dPrize; // Encapsulation
    private int dqty;
    private double dpayment;

    // GetSet methods
    public void setdPrize(double dPrize) {
        dPrize = dPrize;
    }
    public double getdPrize() {
        return dPrize;
    }
    public void setdQTY(int dqty) {
        dqty = dqty;
    }
    public int getdQTY() {
        return dqty;
    }
    public void setdPayment(double dpayment) {
        dpayment = dpayment;
    }
    public double getdPayment() {
        return dpayment;
    }

    abstract void orderDrinks();
}
```

Figure V. Hawaiian

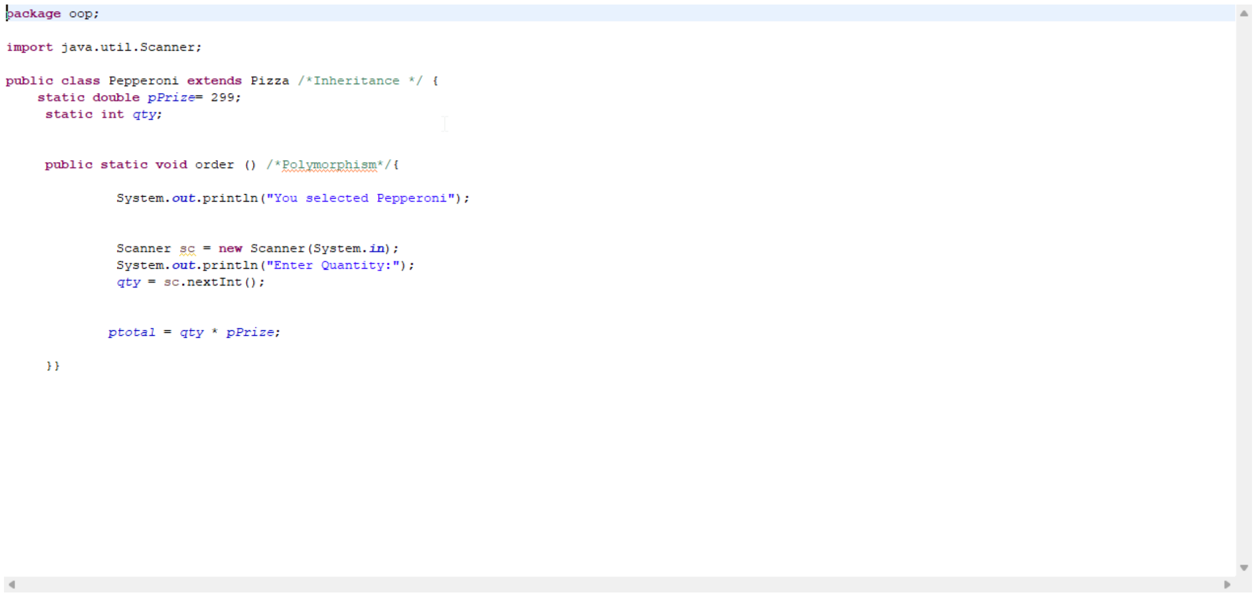
The class Hawaiian contains an order() method that is inherited from its parent class Pizza. The method body has scanner object to get the quantity input from the user and calculate the total amount of ordered pizza.



```java
package oop;

import java.util.Scanner;

public class Hawaiian extends Pizza  /*Inheritance */{

 static double pPrize = 320;
 static int qty;


   public static void order /*Polymorphism*/() {
      System.out.println("You selected Hawaiian");


      Scanner sc = new Scanner(System.in);
         System.out.println("Enter Quantity:");
          qty = sc.nextInt();


           ptotal = qty * pPrize;



   }

}
```

Figure IV. Class Pepperoni

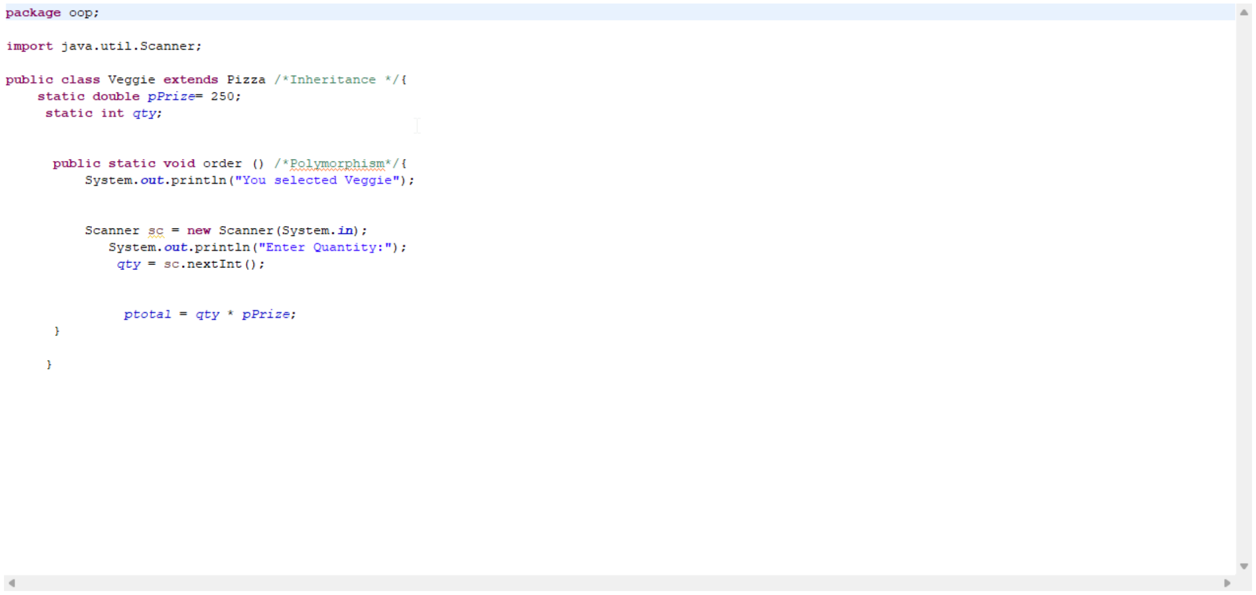
The class Pepperoni contains an order() method that is inherited from its parent class Pizza. The method body has scanner object to get the quantity input from the user and calculate the total amount of ordered pizza.



```
package oop;

import java.util.Scanner;

public class Pepperoni extends Pizza /*Inheritance */ {
    static double pPrize= 299;
     static int qty;


     public static void order () /*Polymorphism*/{

              System.out.println("You selected Pepperoni");


              Scanner sc = new Scanner(System.in);
              System.out.println("Enter Quantity:");
              qty = sc.nextInt();


             ptotal = qty * pPrize;

     }}
```

Figure VI. Veggie

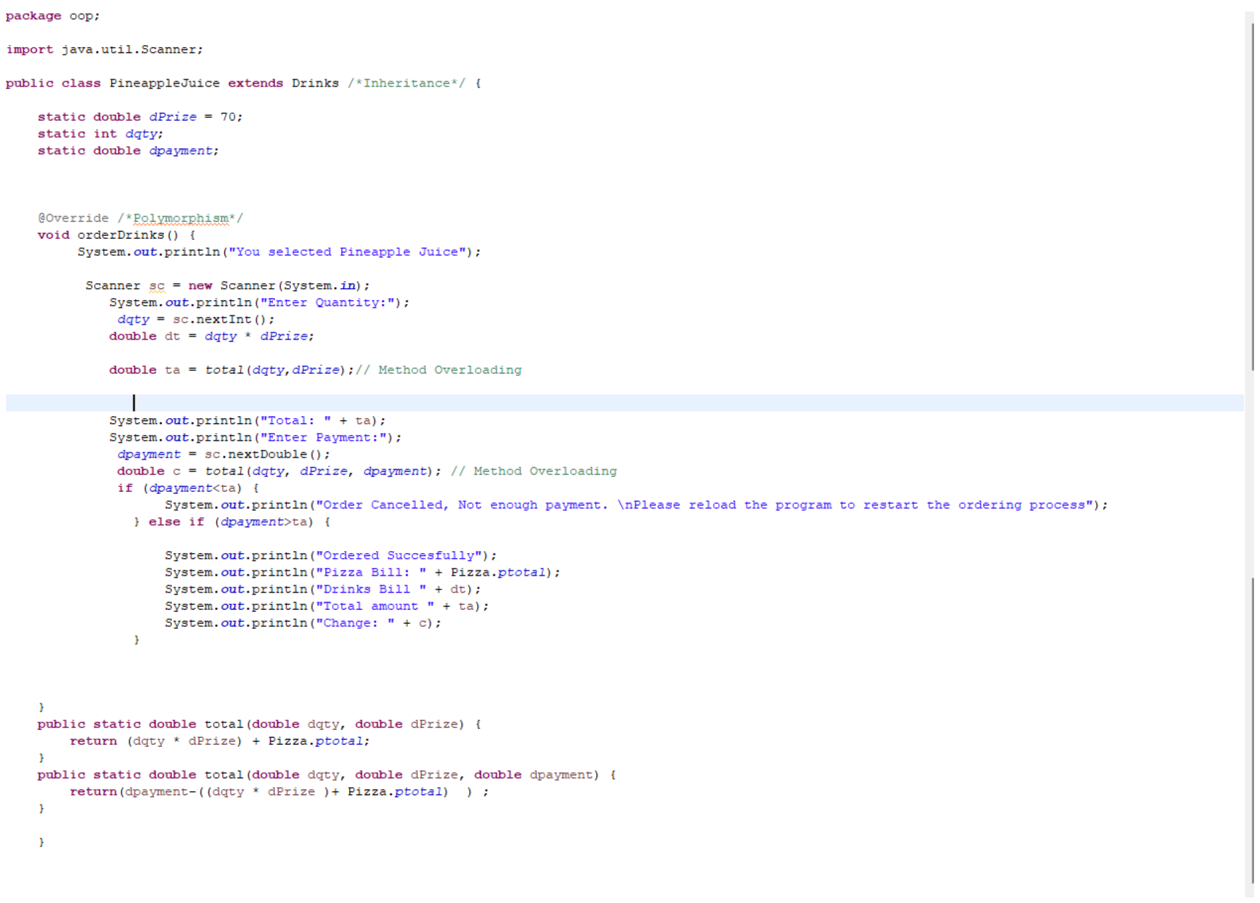
The class Veggie contains an order() method that is inherited from its parent class Pizza. The method body has scanner object to get the quantity input from the user and calculate the total amount of ordered pizza.



```
package oop;

import java.util.Scanner;

public class Veggie extends Pizza /*Inheritance */{
    static double pPrize= 250;
     static int qty;


      public static void order () /*Polymorphism*/{
          System.out.println("You selected Veggie");


          Scanner sc = new Scanner(System.in);
             System.out.println("Enter Quantity:");
              qty = sc.nextInt();


               ptotal = qty * pPrize;
      }

     }
```

Figure VII. Class PineappleJuice

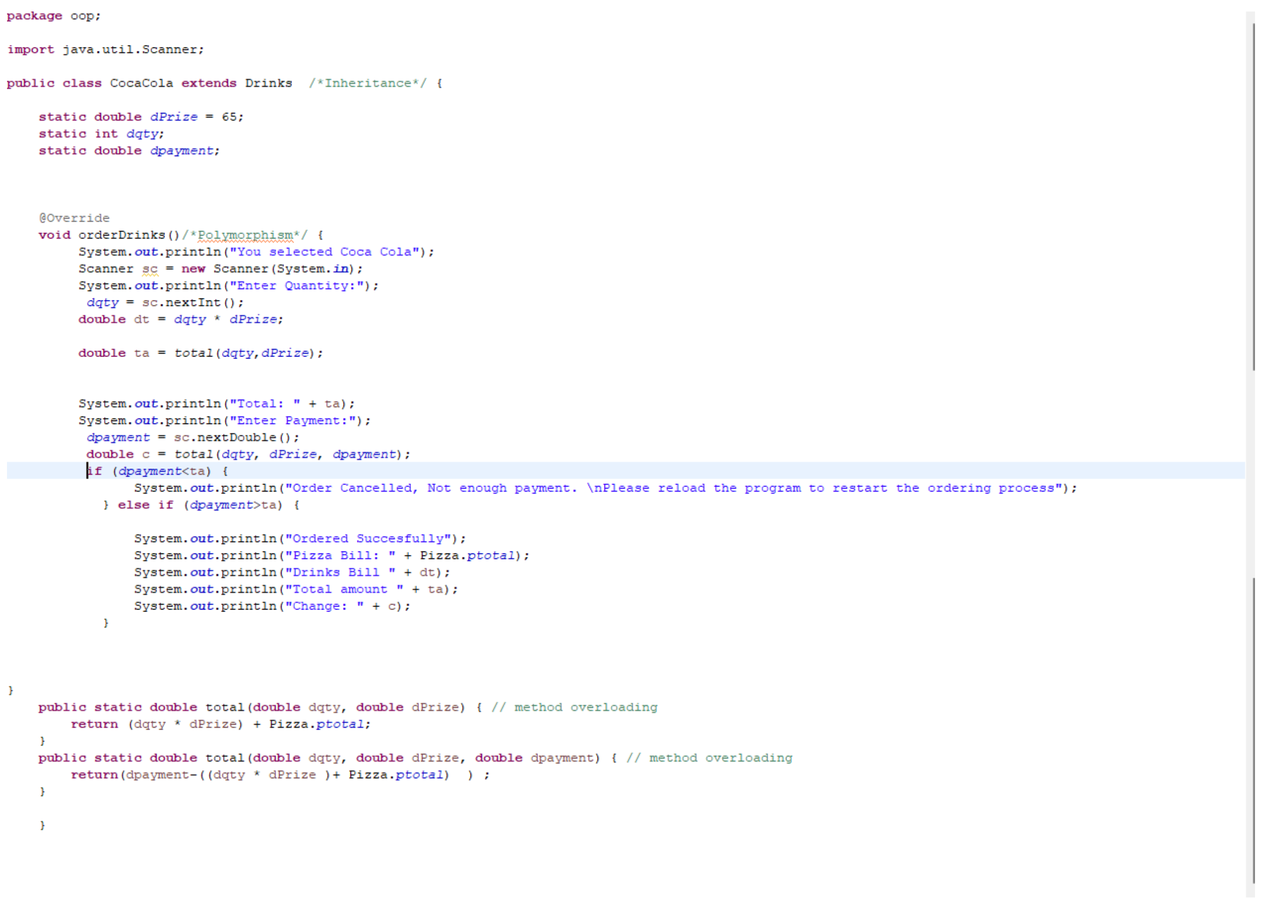
PineappleJuice contains an orderDrink() method which perform the calculation of total bill including the Pizza and Drinks total amount. At the bottom part is the overloaded total() methods. Same functionalities were applied to other child class of the class Drinks.



```java
package oop;

import java.util.Scanner;

public class PineappleJuice extends Drinks /*Inheritance*/ {

    static double dPrize = 70;
    static int dqty;
    static double dpayment;



    @Override /*Polymorphism*/
    void orderDrinks() {
         System.out.println("You selected Pineapple Juice");

          Scanner sc = new Scanner(System.in);
             System.out.println("Enter Quantity:");
              dqty = sc.nextInt();
             double dt = dqty * dPrize;

             double ta = total(dqty,dPrize);// Method Overloading


             System.out.println("Total: " + ta);
             System.out.println("Enter Payment:");
              dpayment = sc.nextDouble();
              double c = total(dqty, dPrize, dpayment); // Method Overloading
              if (dpayment<ta) {
                    System.out.println("Order Cancelled, Not enough payment. \nPlease reload the program to restart the ordering process");
                } else if (dpayment>ta) {

                    System.out.println("Ordered Succesfully");
                    System.out.println("Pizza Bill: " + Pizza.ptotal);
                    System.out.println("Drinks Bill " + dt);
                    System.out.println("Total amount " + ta);
                    System.out.println("Change: " + c);
                }



    }
    public static double total(double dqty, double dPrize) {
        return (dqty * dPrize) + Pizza.ptotal;
    }
    public static double total(double dqty, double dPrize, double dpayment) {
        return(dpayment-((dqty * dPrize )+ Pizza.ptotal)  ) ;
    }

    }
```

Figure VIII. Class CocaCola

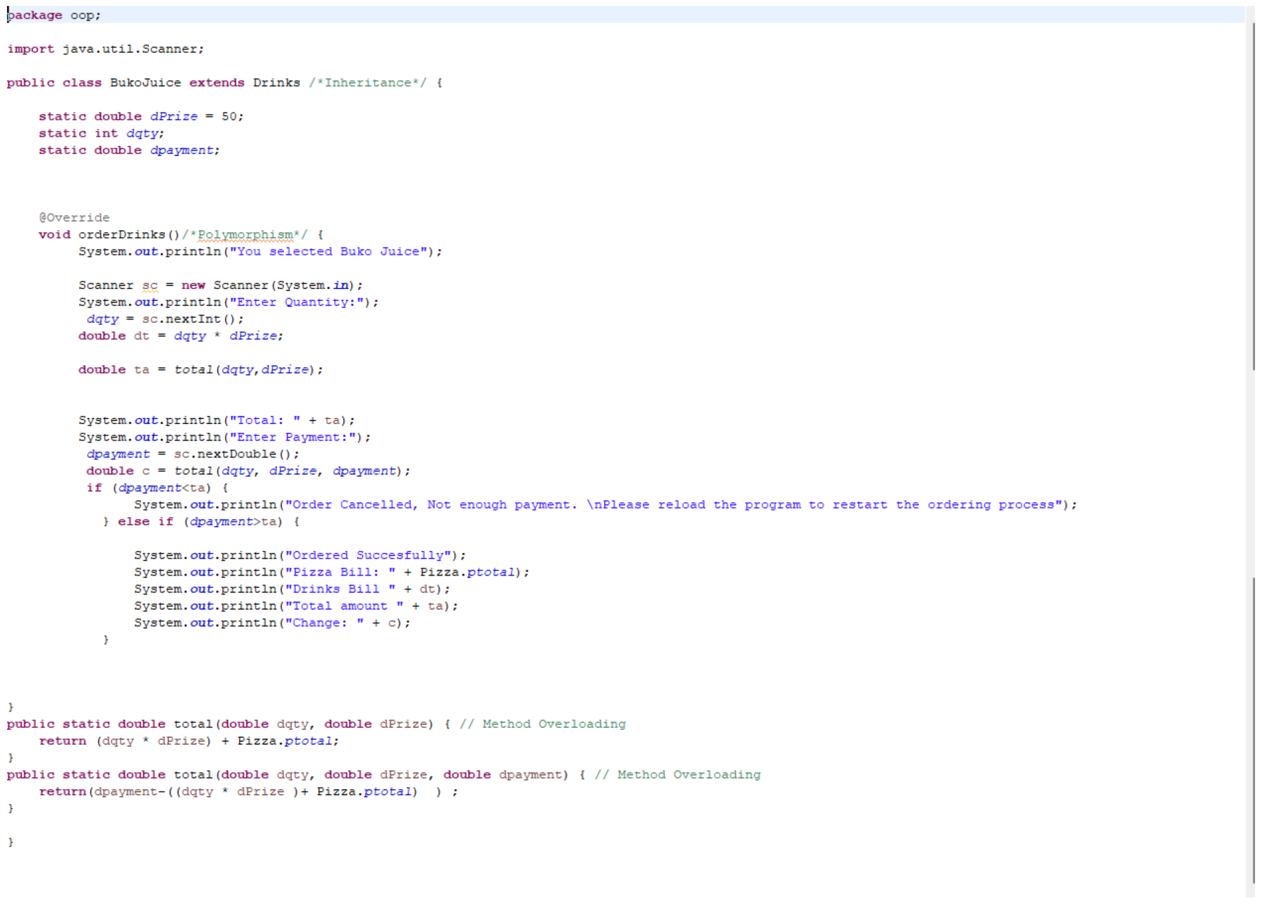
CocaCola contains an orderDrink() method which perform the calculation of total bill including the Pizza and Drinks total amount. At the bottom part is the overloaded total() methods. Same functionalities were applied to other child class of the class Drinks.



```java
package oop;

import java.util.Scanner;

public class CocaCola extends Drinks  /*Inheritance*/ {

    static double dPrize = 65;
    static int dqty;
    static double dpayment;



    @Override
    void orderDrinks()/*Polymorphism*/ {
         System.out.println("You selected Coca Cola");
         Scanner sc = new Scanner(System.in);
         System.out.println("Enter Quantity:");
          dqty = sc.nextInt();
         double dt = dqty * dPrize;

         double ta = total(dqty,dPrize);


         System.out.println("Total: " + ta);
         System.out.println("Enter Payment:");
          dpayment = sc.nextDouble();
          double c = total(dqty, dPrize, dpayment);
          if (dpayment<ta) {
                System.out.println("Order Cancelled, Not enough payment. \nPlease reload the program to restart the ordering process");
            } else if (dpayment>ta) {

                System.out.println("Ordered Succesfully");
                System.out.println("Pizza Bill: " + Pizza.ptotal);
                System.out.println("Drinks Bill " + dt);
                System.out.println("Total amount " + ta);
                System.out.println("Change: " + c);
            }



}
    public static double total(double dqty, double dPrize) { // method overloading
        return (dqty * dPrize) + Pizza.ptotal;
    }
    public static double total(double dqty, double dPrize, double dpayment) { // method overloading
        return(dpayment-((dqty * dPrize )+ Pizza.ptotal)  ) ;
    }

    }
```

Figure IX. Class BukoJuice

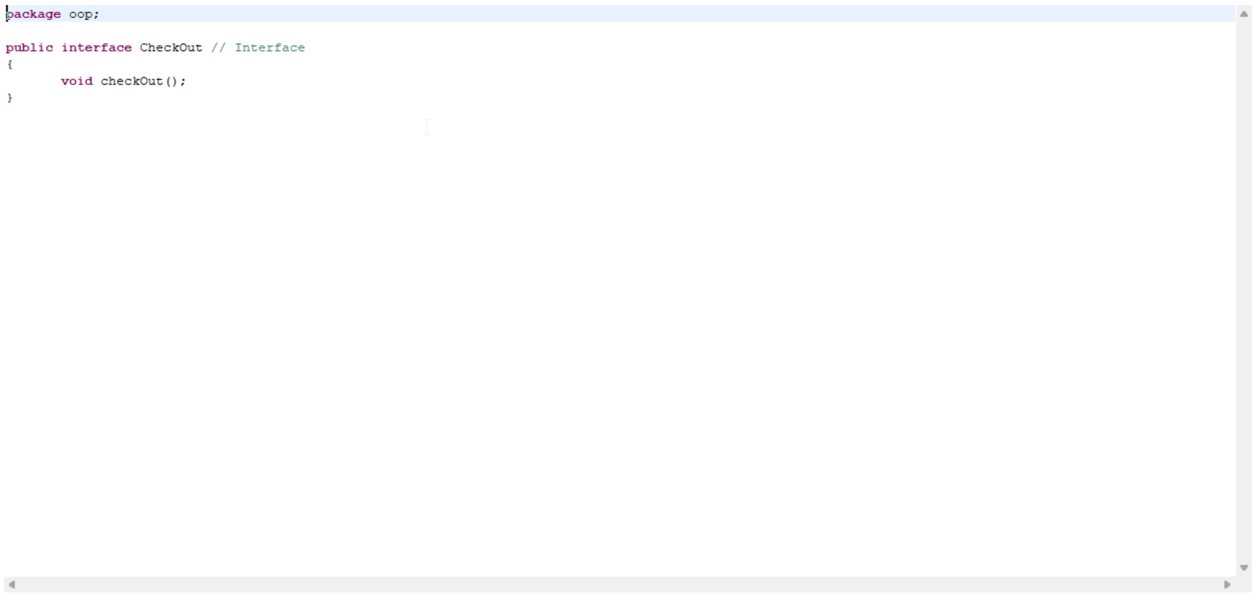
BukoJuice contains an orderDrink() method which perform the calculation of total bill including the Pizza and Drinks total amount. At the bottom part is the overloaded total() methods. Same functionalities were applied to other child class of the class Drinks.



```java
package oop;

import java.util.Scanner;

public class BukoJuice extends Drinks /*Inheritance*/ {

    static double dPrize = 50;
    static int dqty;
    static double dpayment;



    @Override
    void orderDrinks()/*Polymorphism*/ {
         System.out.println("You selected Buko Juice");

         Scanner sc = new Scanner(System.in);
         System.out.println("Enter Quantity:");
          dqty = sc.nextInt();
         double dt = dqty * dPrize;

         double ta = total(dqty,dPrize);


         System.out.println("Total: " + ta);
         System.out.println("Enter Payment:");
          dpayment = sc.nextDouble();
          double c = total(dqty, dPrize, dpayment);
          if (dpayment<ta) {
                System.out.println("Order Cancelled, Not enough payment. \nPlease reload the program to restart the ordering process");
            } else if (dpayment>ta) {

                System.out.println("Ordered Succesfully");
                System.out.println("Pizza Bill: " + Pizza.ptotal);
                System.out.println("Drinks Bill " + dt);
                System.out.println("Total amount " + ta);
                System.out.println("Change: " + c);
            }



}
public static double total(double dqty, double dPrize) { // Method Overloading
    return (dqty * dPrize) + Pizza.ptotal;
}
public static double total(double dqty, double dPrize, double dpayment) { // Method Overloading
    return(dpayment-((dqty * dPrize )+ Pizza.ptotal)  ) ;
}

}
```

Figure X. Interface CheckOut

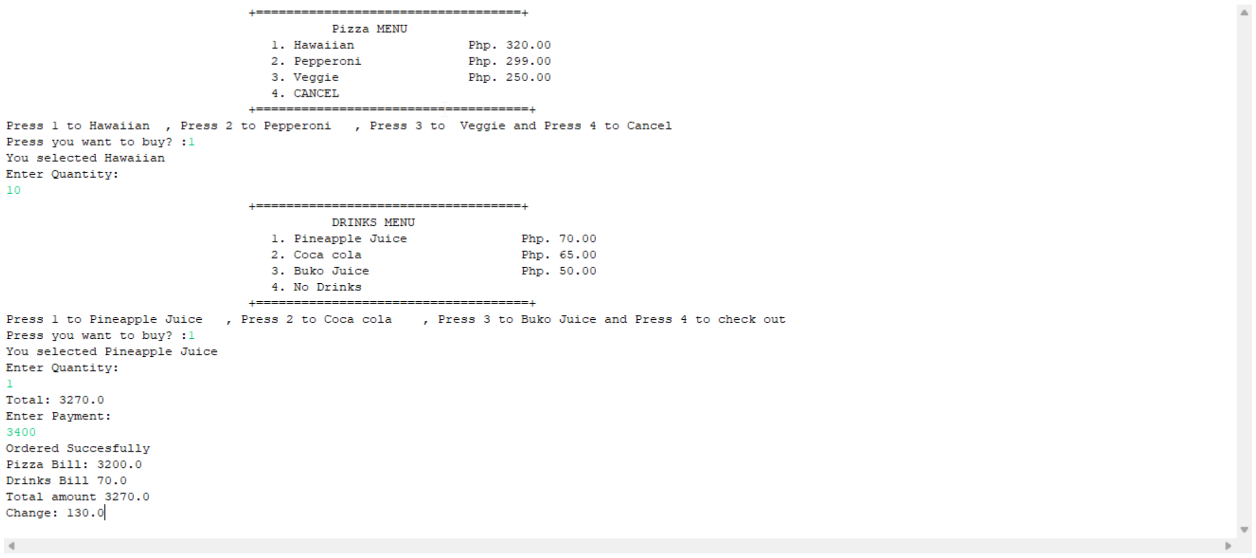
The CheckOut interface contains a checkout() method which was implemented in the class Pizza



```java
package oop;

public interface CheckOut // Interface
{
	void checkOut();
}
```

Sample Output 1

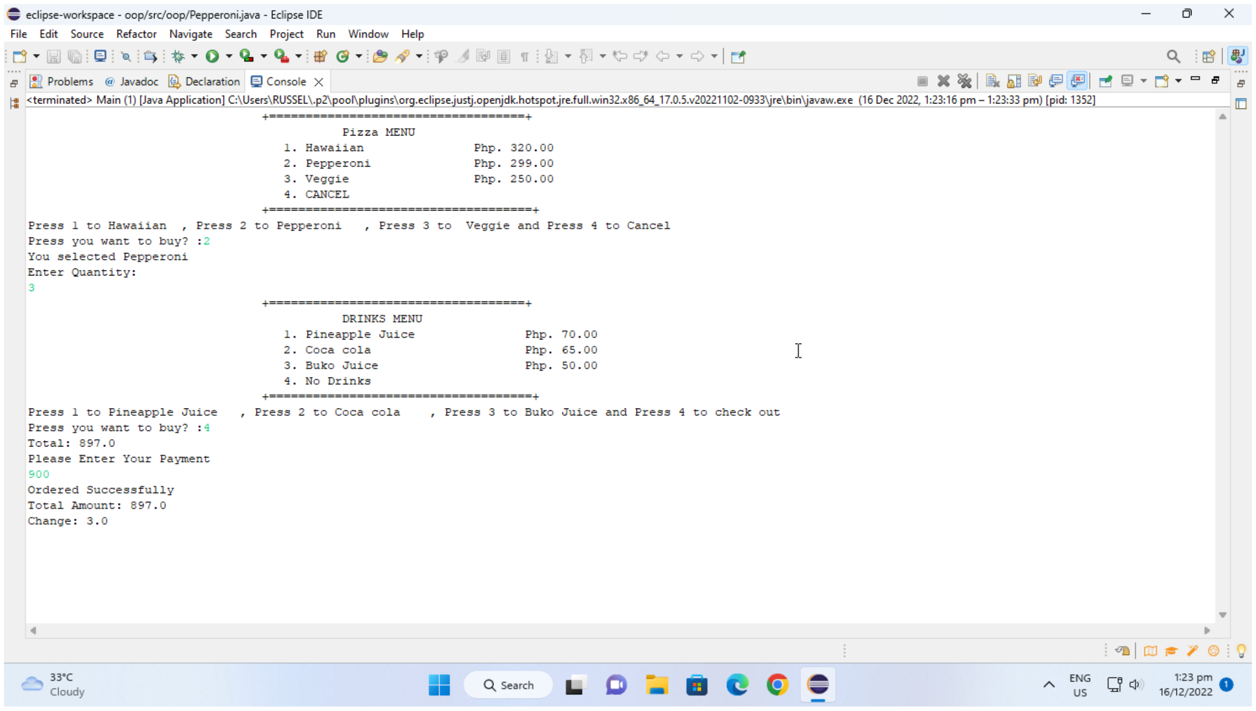
The costumer ordered both drinks and pizza successfully.



```
                                +===================================+
                                           Pizza MENU
                                   1. Hawaiian               Php. 320.00
                                   2. Pepperoni              Php. 299.00
                                   3. Veggie                 Php. 250.00
                                   4. CANCEL
                                +====================================+
Press 1 to Hawaiian  , Press 2 to Pepperoni   , Press 3 to  Veggie and Press 4 to Cancel
Press you want to buy? :1
You selected Hawaiian
Enter Quantity:
10
                                +===================================+
                                           DRINKS MENU
                                   1. Pineapple Juice               Php. 70.00
                                   2. Coca cola                     Php. 65.00
                                   3. Buko Juice                    Php. 50.00
                                   4. No Drinks
                                +====================================+
Press 1 to Pineapple Juice   , Press 2 to Coca cola    , Press 3 to Buko Juice and Press 4 to check out
Press you want to buy? :1
You selected Pineapple Juice
Enter Quantity:
1
Total: 3270.0
Enter Payment:
3400
Ordered Succesfully
Pizza Bill: 3200.0
Drinks Bill 70.0
Total amount 3270.0
Change: 130.0
```

Sample Output 2

The costumer ordered a pizza but not a drink. The program exits in advance through the checkout() method.



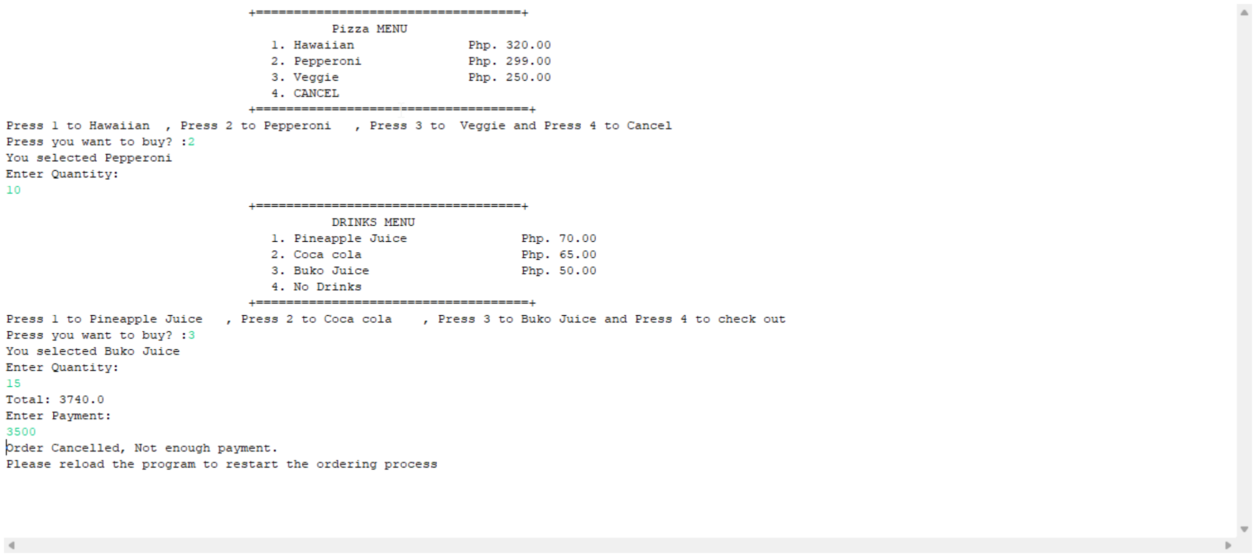
eclipse-workspace - oop/src/oop/Pepperoni.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems
Javadoc
Declaration
Console
33°C Cloudy
Search
ENG US
1:23 pm 16/12/2022

<terminated> Main (1) [Java Application] C:\Users\RUSSEL\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.5.v20221102-0933\jre\bin\javaw.exe (16 Dec 2022, 1:23:16 pm – 1:23:33 pm) [pid: 1352]

```
                                +===================================+
                                           Pizza MENU
                                   1. Hawaiian               Php. 320.00
                                   2. Pepperoni              Php. 299.00
                                   3. Veggie                 Php. 250.00
                                   4. CANCEL
                                +====================================+
Press 1 to Hawaiian  , Press 2 to Pepperoni   , Press 3 to  Veggie and Press 4 to Cancel
Press you want to buy? :2
You selected Pepperoni
Enter Quantity:
3
                                +===================================+
                                           DRINKS MENU
                                   1. Pineapple Juice               Php. 70.00
                                   2. Coca cola                     Php. 65.00
                                   3. Buko Juice                    Php. 50.00
                                   4. No Drinks
                                +====================================+
Press 1 to Pineapple Juice   , Press 2 to Coca cola    , Press 3 to Buko Juice and Press 4 to check out
Press you want to buy? :4
Total: 897.0
Please Enter Your Payment
900
Ordered Successfully
Total Amount: 897.0
Change: 3.0
```

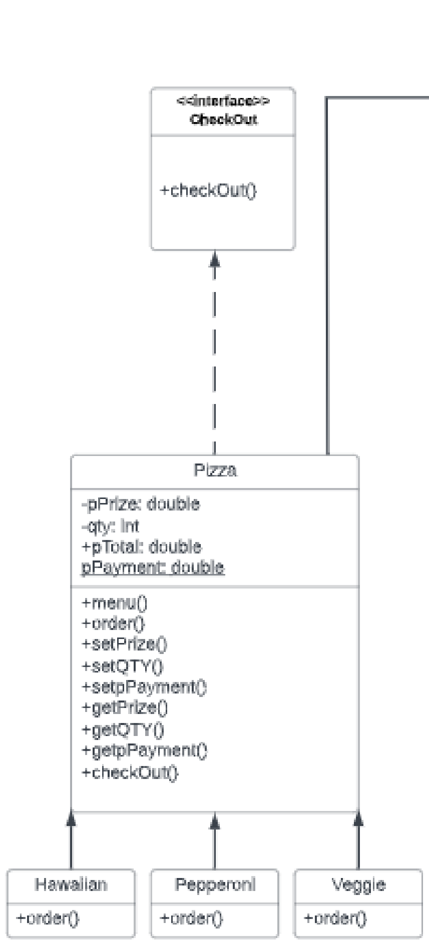
Sample Output 3

The customer payment was not enough, and the ordering process must be repeated.



```
                                +===================================+
                                           Pizza MENU
                                   1. Hawaiian              Php. 320.00
                                   2. Pepperoni             Php. 299.00
                                   3. Veggie                Php. 250.00
                                   4. CANCEL
                                +====================================+
Press 1 to Hawaiian  , Press 2 to Pepperoni   , Press 3 to  Veggie and Press 4 to Cancel
Press you want to buy? :2
You selected Pepperoni
Enter Quantity:
10
                                +===================================+
                                           DRINKS MENU
                                   1. Pineapple Juice              Php. 70.00
                                   2. Coca cola                    Php. 65.00
                                   3. Buko Juice                   Php. 50.00
                                   4. No Drinks
                                +====================================+
Press 1 to Pineapple Juice   , Press 2 to Coca cola    , Press 3 to Buko Juice and Press 4 to check out
Press you want to buy? :3
You selected Buko Juice
Enter Quantity:
15
Total: 3740.0
Enter Payment:
3500
Order Cancelled, Not enough payment.
Please reload the program to restart the ordering process
```

UML Diagram



OOP Concepts Applied:

- GetSet methods
- Encapsulation (private),
- Inheritance
- Method Overloading
- Abstraction
- Interface
- Runtime polymorphism. (Method overriding)