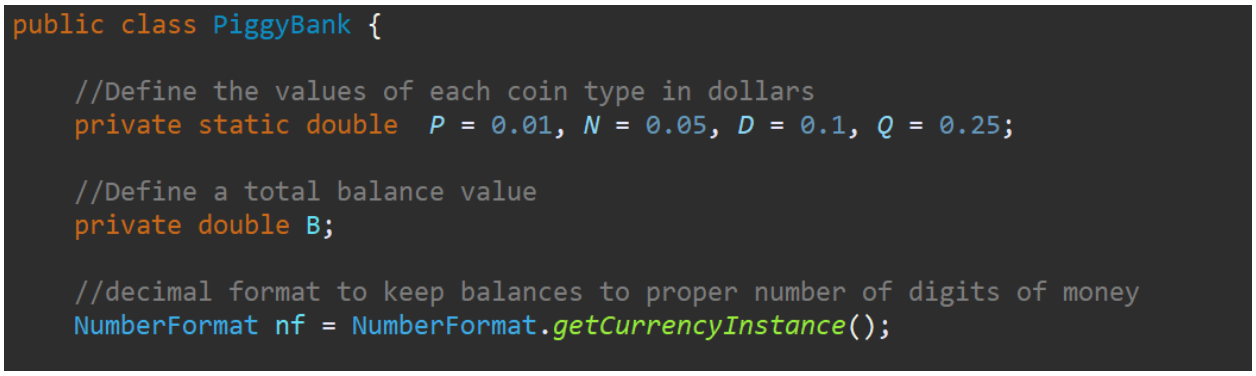
Credit Name: CS 3120 Object-Oriented Programming 1

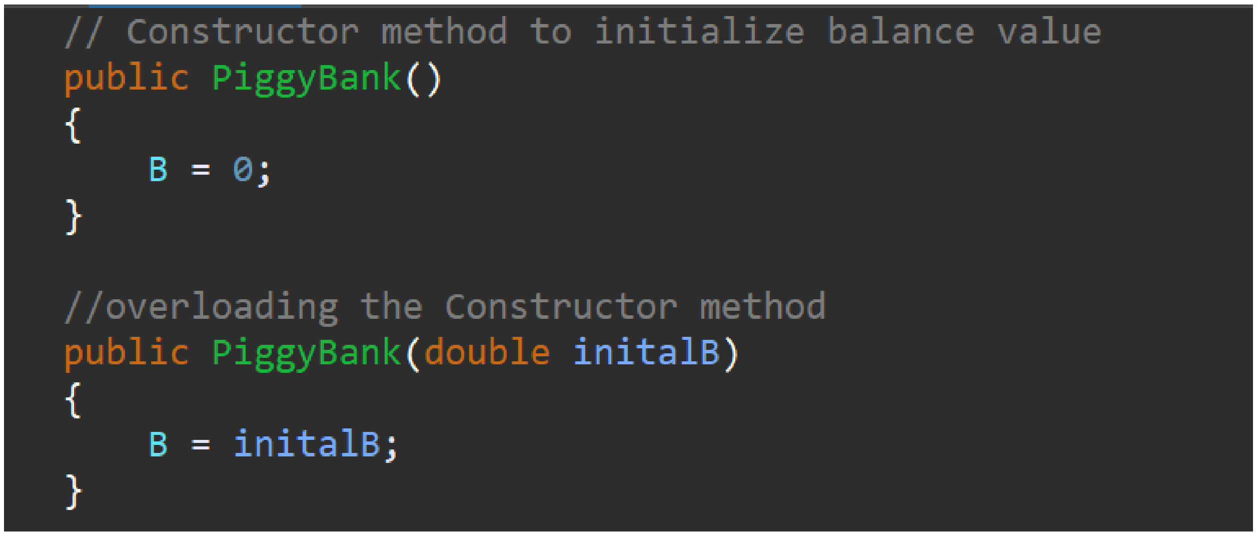
Assignment: MySavings

How has your program changed from planning to coding to now? Please Explain



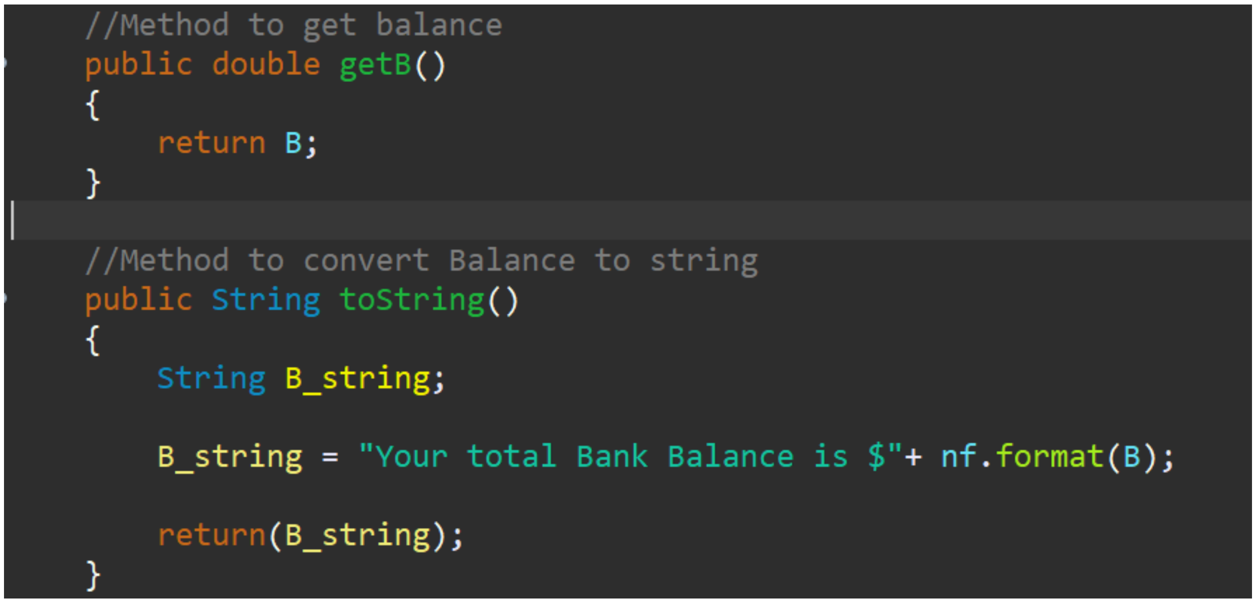
```java
public class PiggyBank {

    //Define the values of each coin type in dollars
    private static double  P = 0.01, N = 0.05, D = 0.1, Q = 0.25;

    //Define a total balance value
    private double B;

    //decimal format to keep balances to proper number of digits of money
    NumberFormat nf = NumberFormat.getCurrencyInstance();
```

Starting with my program for my Object 'PiggyBank', I defined the relevant constants which related to the value of each coin type in dollars: penny, nickel, dime and quarter. These are private and static as they are constants (related to static) and I don't want them to be called in other classes (related to private). I then defined a total balance value labelled 'B', which is a double as money can be measured to 2 decimal places. A number format was then set up to format numbers to currency standard.



```java
   // Constructor method to initialize balance value
   public PiggyBank()
   {
       B = 0;
   }

   //overloading the Constructor method
   public PiggyBank(double initalB)
   {
       B = initalB;
   }
```

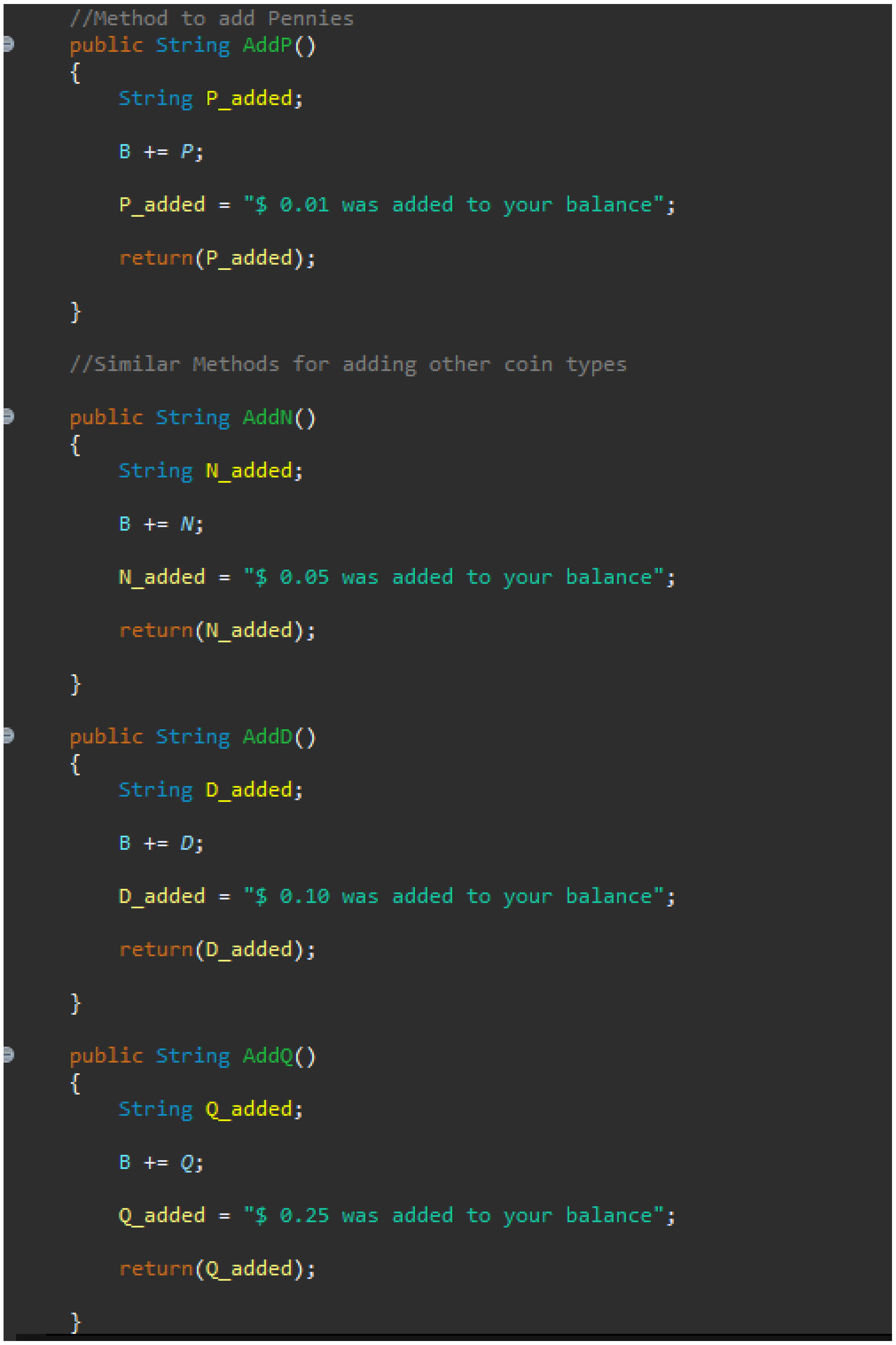
I then initialised a bank balance to equal zero by creating a constructor method for PiggyBank. Then I overloaded the constructor by adding a parameter for PiggyBank which is the initial value of the bank balance the user sets. I then set the bank balance equal to the parameter of the object.



```
    //Method to get balance
    public double getB()
    {
        return B;
    }

    //Method to convert Balance to string
    public String toString()
    {
        String B_string;

        B_string = "Your total Bank Balance is $"+ nf.format(B);

        return(B_string);
    }
```

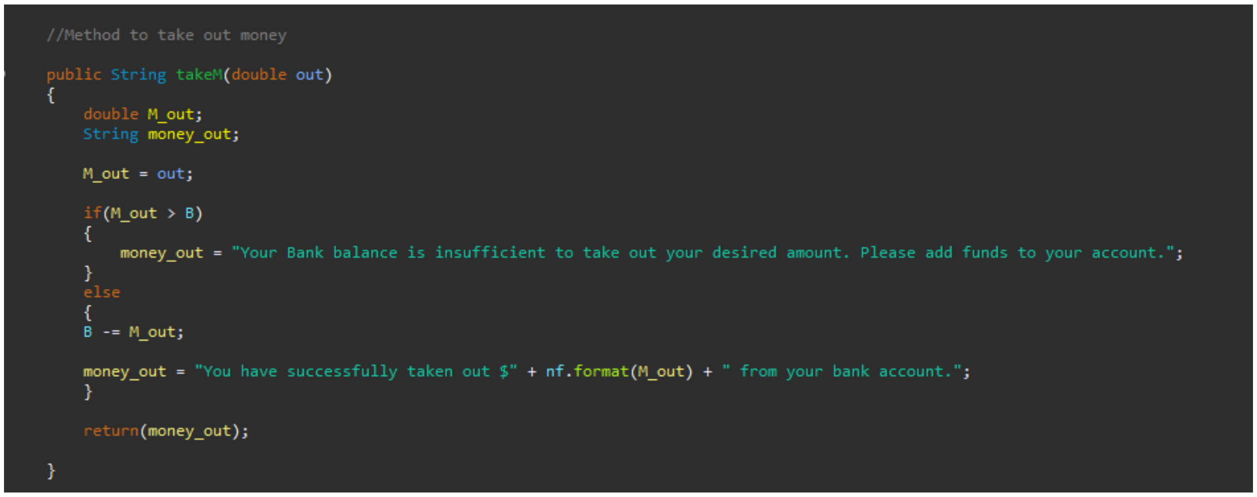
After I made an accessor method getB() which returns the value of B. Since B is a double then the method also has to be defined to return a double.

Then I made a method which returns the bank balance as a string apart of a formal statement. I defined a string variable to be the sentence to be returned and ensured the bank balance was formatted to output currency correctly.



```
//Method to add Pennies
public String AddP()
{
    String P_added;

    B += P;

    P_added = "$ 0.01 was added to your balance";

    return(P_added);

}

//Similar Methods for adding other coin types

public String AddN()
{
    String N_added;

    B += N;

    N_added = "$ 0.05 was added to your balance";

    return(N_added);

}

public String AddD()
{
    String D_added;

    B += D;

    D_added = "$ 0.10 was added to your balance";

    return(D_added);

}

public String AddQ()
{
    String Q_added;

    B += Q;

    Q_added = "$ 0.25 was added to your balance";

    return(Q_added);

}
```

I then made a method to add a penny to the bank account, and return a statement outlining that a penny was added. I made a string for the penny added statement. Which mentions that "$0.01 was added to your balance" I also added the value of the penny to the existing balance.
I repeated a similar process for the other coins.

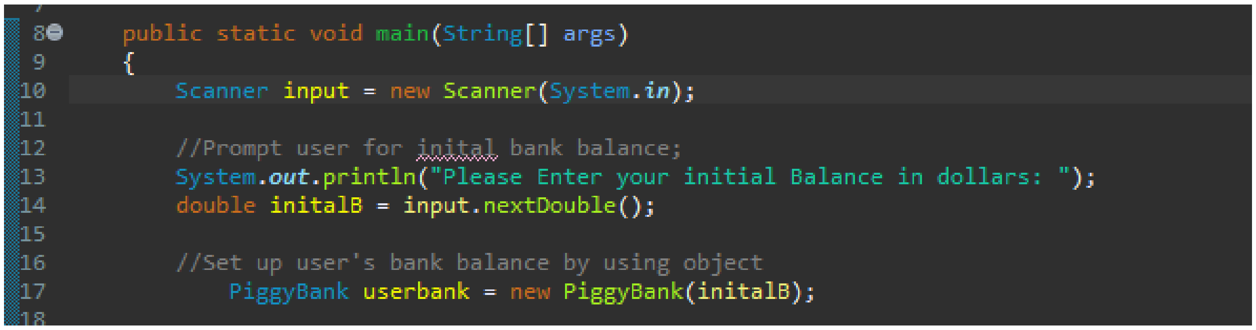


```
//Method to take out money

public String takeM(double out)
{
    double M_out;
    String money_out;

    M_out = out;

    if(M_out > B)
    {
        money_out = "Your Bank balance is insufficient to take out your desired amount. Please add funds to your account.";
    }
    else
    {
    B -= M_out;

    money_out = "You have successfully taken out $" + nf.format(M_out) + " from your bank account.";
    }

    return(money_out);

}
```

Then I defined a way to take our money by creating a method takeM. This method has 1 input which is the amount of money being taken OUT of the balance. I declared a variable M_out to be a double which is equal to the parameter given in the method. There is also a string declared which is a statement which outlines whether or not money was successfully taken out and how much money was taken out.

If the money taken out is more than what is remaining in the bank account, then I display a message showing that there is not enough money to take out. Otherwise, I subtract the money coming out from the balance and display a string showing the amount taken out from the balance.

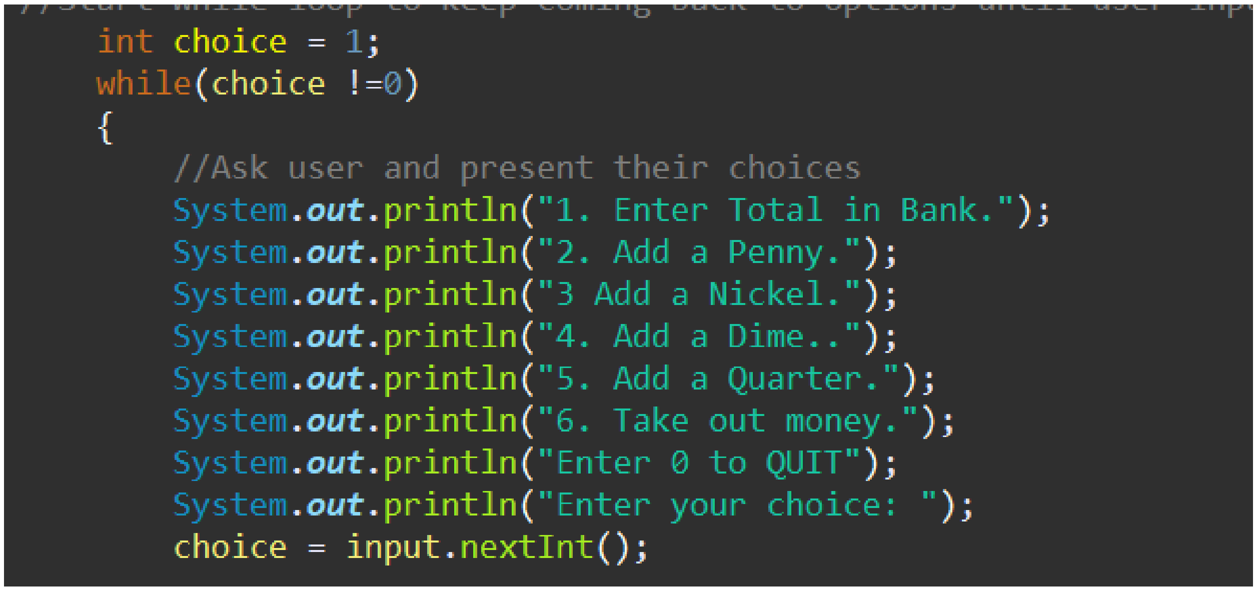
Lastly I return string output.



```
    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in);

        //Prompt user for inital bank balance;
        System.out.println("Please Enter your initial Balance in dollars: ");
        double initalB = input.nextDouble();

        //Set up user's bank balance by using object
            PiggyBank userbank = new PiggyBank(initalB);
```

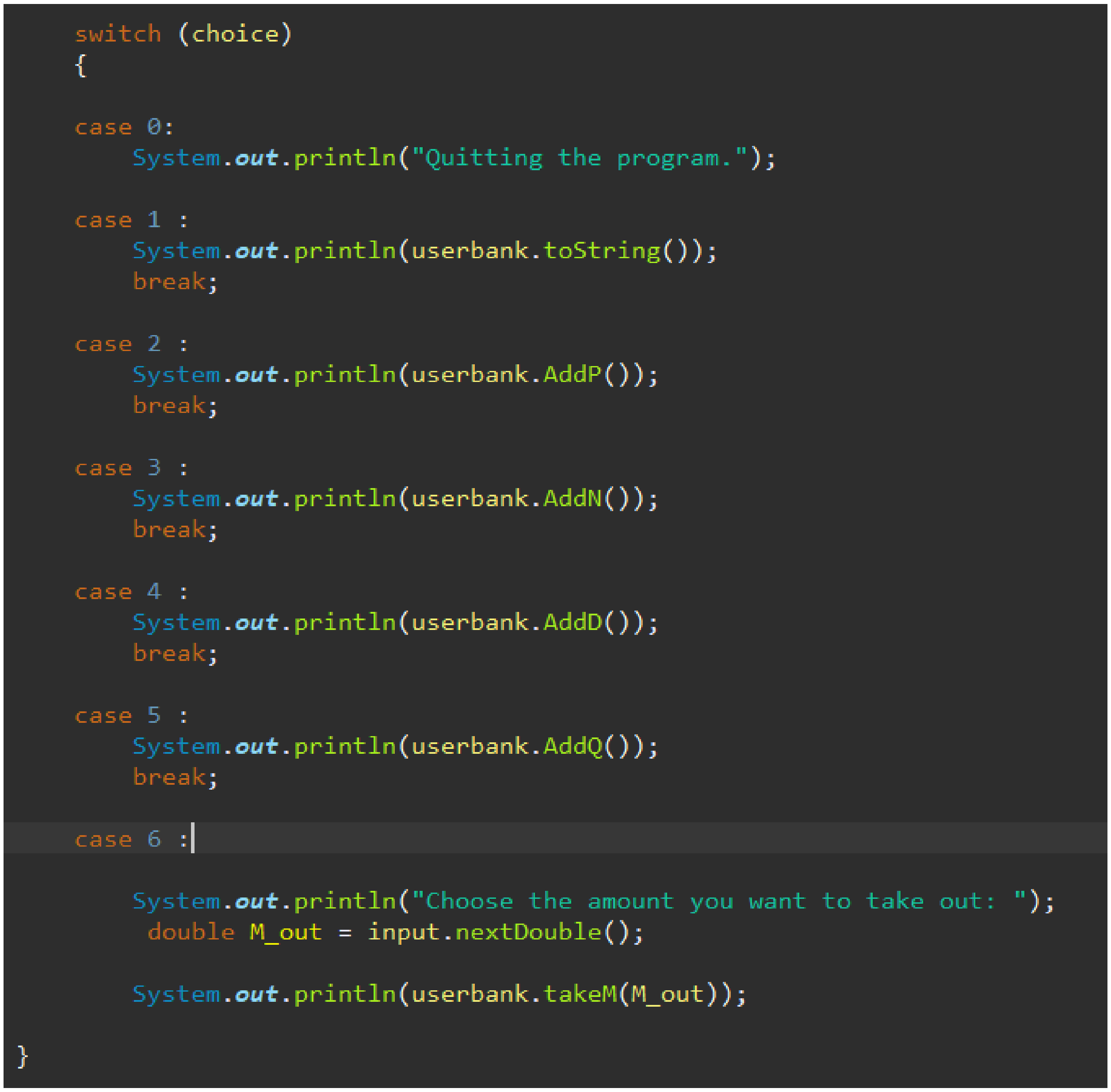
Now in my Client Code, I set up a scanner for user inputs and then ask the user to input the initial balance present in their bank account.

Then I create an object called userbank which is a new PiggyBank with an initial value of whatever the user inputs.



```
//start while loop to keep coming back to options until user inp
    int choice = 1;
    while(choice !=0)
    {
        //Ask user and present their choices
        System.out.println("1. Enter Total in Bank.");
        System.out.println("2. Add a Penny.");
        System.out.println("3 Add a Nickel.");
        System.out.println("4. Add a Dime..");
        System.out.println("5. Add a Quarter.");
        System.out.println("6. Take out money.");
        System.out.println("Enter 0 to QUIT");
        System.out.println("Enter your choice: ");
        choice = input.nextInt();
```

Then I create a integer variable which keeps track of the menu choice the user selects. It is initialised at 1 and then a while loop is made to keep the menu recurring until the user selects 0 (quit option). Then I printed out all of the options for the user to see and then allowed them to select their choice making it equal to the integer variable choice.



```
    switch (choice)
    {

    case 0:
        System.out.println("Quitting the program.");

    case 1 :
        System.out.println(userbank.toString());
        break;

    case 2 :
        System.out.println(userbank.AddP());
        break;

    case 3 :
        System.out.println(userbank.AddN());
        break;

    case 4 :
        System.out.println(userbank.AddD());
        break;

    case 5 :
        System.out.println(userbank.AddQ());
        break;

    case 6 :

        System.out.println("Choose the amount you want to take out: ");
         double M_out = input.nextDouble();

        System.out.println(userbank.takeM(M_out));

}
```

Then a switch statement organises the different cases; case 0 shows that you are quitting the program in words. Case 1 accesses the object user bank and converts the balance into a string. Case 2 to case 5 access userbank and add the respective coin to the balance. Case 6 results in asking the user the amount to be taken out and then accesses the userbank and takes out the money specified by the user just before, printing the final statement.