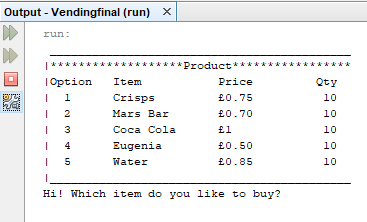
**Vending Machine**

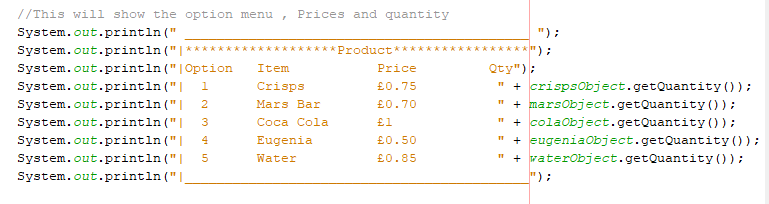
This is a vending machine created in java I will start presenting the functionality of the program and the code of each step.

This is the menu of option.

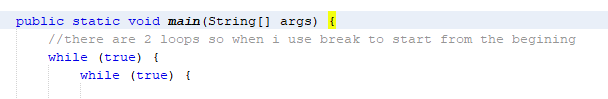


When you type one of the option will record that option as the option you want to buy

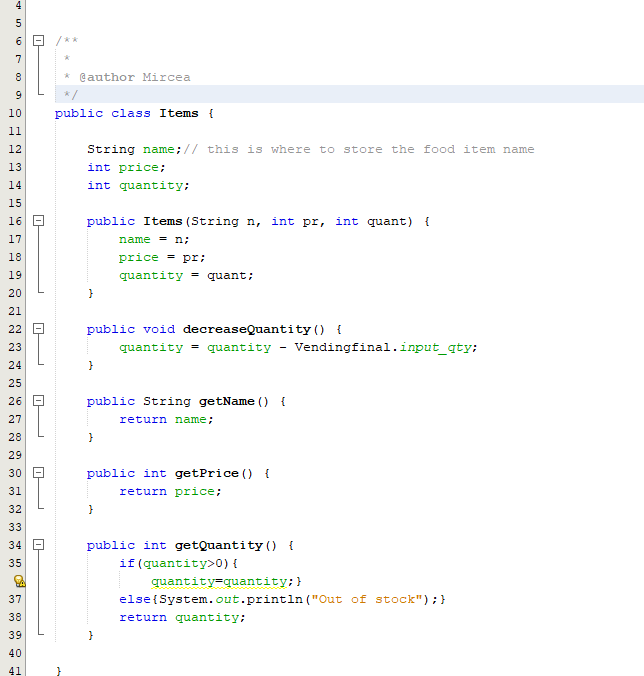
This is the code for options having the objects of items with get quantity, so we will see every time how many items are left in machine.



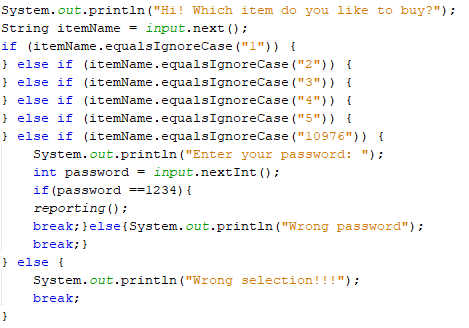
I had introduced the main method into two loops so whenever I will break the program starts from the beginning instead of closing.



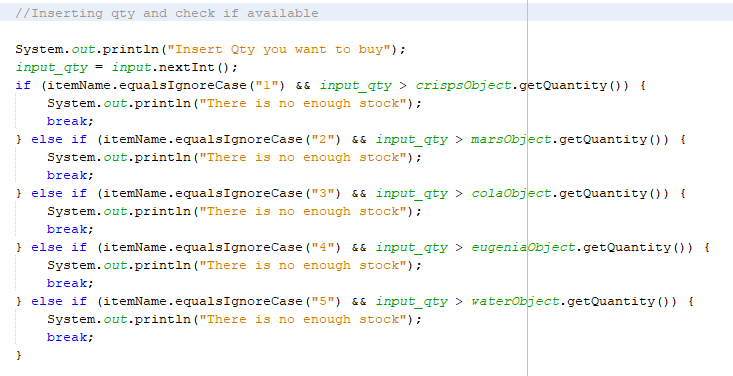
Here is the class for items the methods for them.



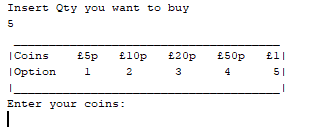
Here we have the code for option choosing and the admin reporting.



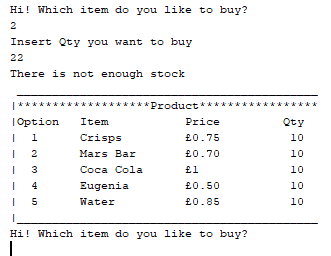
Here we insert the quantity of items we want to buy, and we check if quantity stock you entered is available



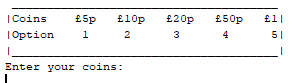
Here is an example when the quantity is enough. If the inserted quantity is correct you will see the coins menu.



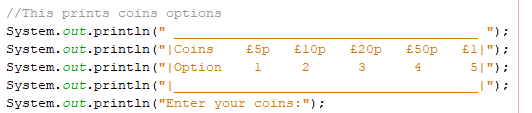
If the inserted quantity is out of stock you will see this message and you will be returned to the option choice and you will be able to see the menu and quantity remaining again.



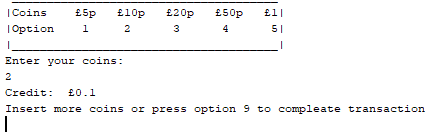
Next, we have a submenu with coins option. Option 1 inserts 5p option 2 10p and so on



And the code for this submenu



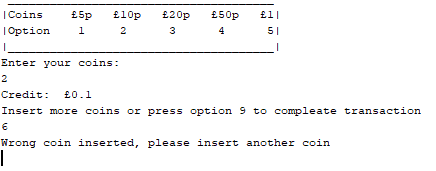
In my example I inserted 10p after that you can see the credit you have and a message telling you to add more coins or press number 9 to complete transaction (buy items).



Here we have example of adding different coins and adding them to the credit.

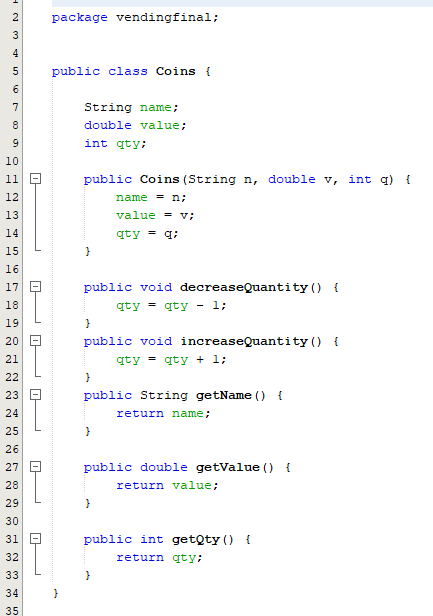


Here is another example when you inserted an unsupported coin you will get a message telling you is incorrect and to try again

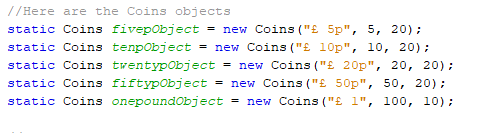


Now I will explain the code for adding coins.

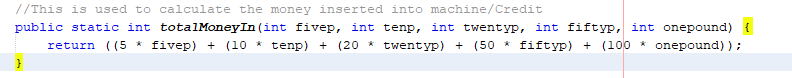
Here we have the Coins class where we store the coins and coins methods.



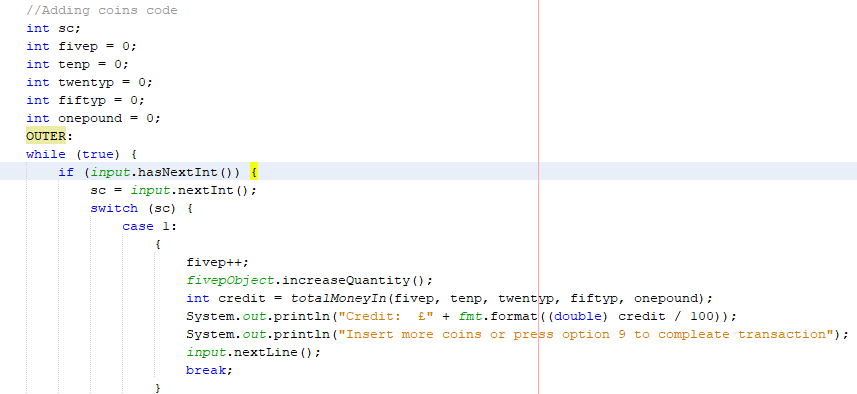
Here is where we declare the coins into the main class



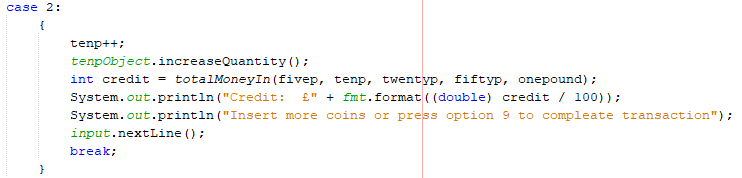
I will explain how 5p works.

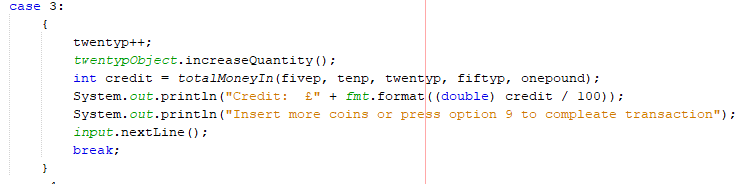
First, I had created 5 variables for coins, so I can calculate the total money inserted (Credit). In the code below, you can see when user will press option 1 the variable of 5p will get +1 and the fivepObject will increase its quantity by one using the method I had added in Coins class. Then I had created a variable for credit, so I can print the credit after each coin inserted by creating a method add all the coins as you can see in the image below. Each variable is multiplied by its value then added with other variables.

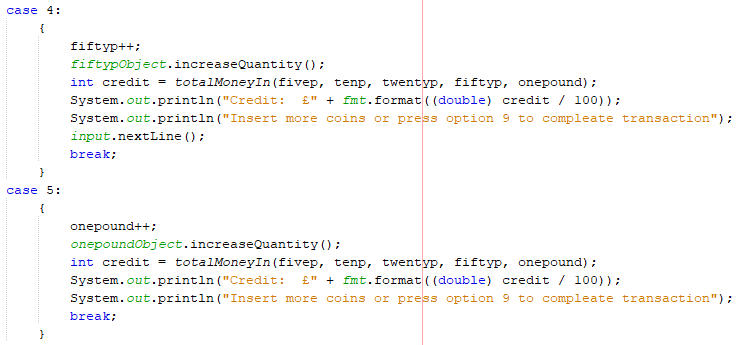
Next, we print the credit, so user can see how much money inserted after credit and after that we print a line telling the user to add more coins or press number 9 so his transaction to be completed and receive his items.



Here is the code for the rest of the cases:

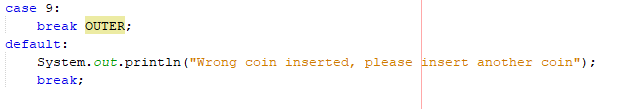






Case 9 is for when you finish adding coins and want to buy your items, pressing 9 will buy.

If the user will add the wrong coin will get the message from default.



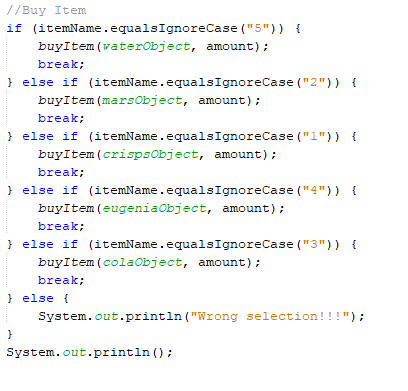
Here I used another variable called amount to store the money inserted by the user, so I can use it in buying method and a final Credit print.



Here we have the buy items depend of witch item user selected

If the user at the start selected option 5 for example the code will run to buy water.

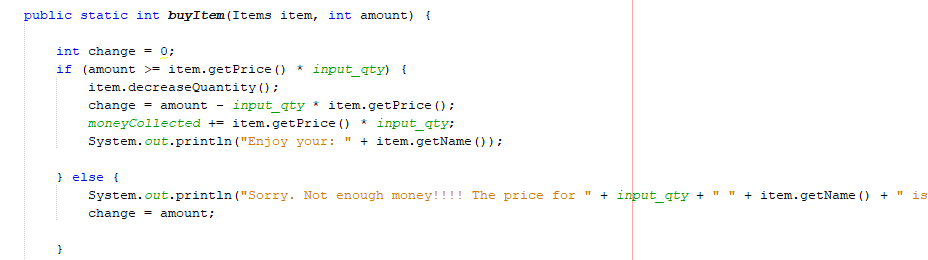
If the user selects the wrong option will receive a message with wrong selection



I will explain how the buy item method works below.

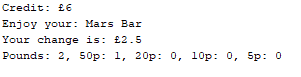
Here I have the method for buying

If the amount inserted is bigger or equal, then the item price multiply by the quantity user inserted to buy then the quantity will be decreased from the item selected stock and the items will be dispense. After that change will be calculated. If the amount will be smaller than the item price multiply by the quantity inserted the user will get a message saying not enough money and all the money inserted will be given as change



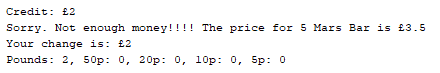
Now I will present the code for giving out change and printing it.

The 5 variables I created here are created to calculate the amount of each coins will be dispense and print it like this:



In this example the user received 2.5 pounds change, so he got 2 one pound and 1 50p

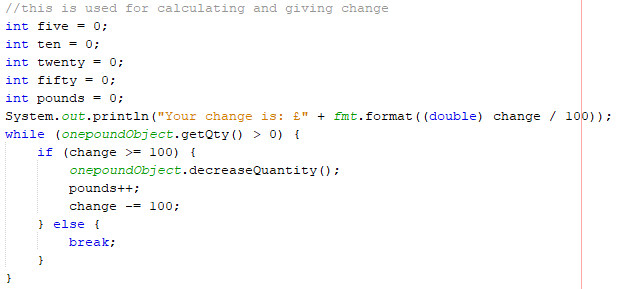
The user will get this message if he will not add enough money to buy:



Here is the code for printing the change for each coin



I had created a loop for each coin where is checking if the coins available in machine are grater then 0 will run the loop and if change is bigger than the value of the coin will decrease the quantity of the coins object once and add one to the variable of the printing change after that it will subtract the value of the coin from the change. The loop will run until the conditions are not meet and it will jump to the next one



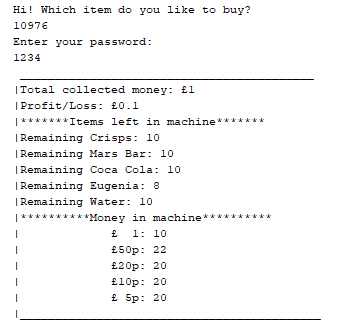
Here is the rest of the code for the change



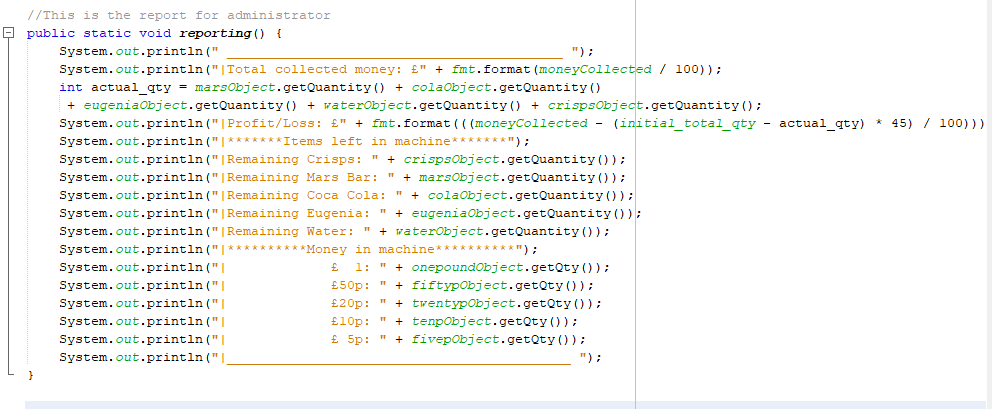
Here you can see the administrator reporting.

When the administrator will input his code 10976 he will see a screen asking for the password after he will insert the password set to 1234 now he will see the reporting witch contain: total money collected by the machine, profit of the machine calculated with the buy price of each item of 0.45 pounds in our example we sold 2 Eugenia for 0.50 pounds each so this is 1 pound considering the buy price of each item of 0.45 pounds the profit will be 0.1 pounds.

Next, we have the stock left into machine and money currently into machine sorted by each coin

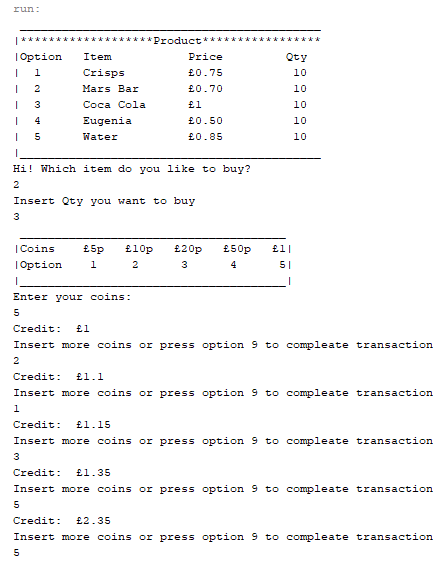


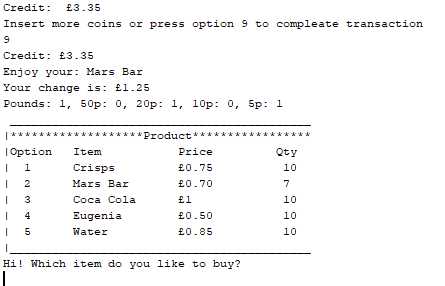
And this is the code:



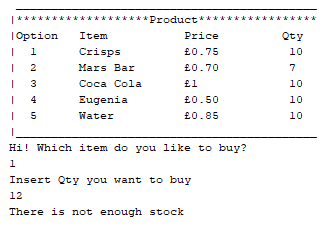
Screenshots to test the program:

* Dispense a snack if it is in stock and the price has been paid.

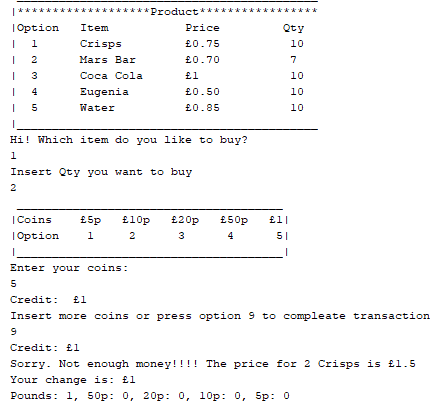




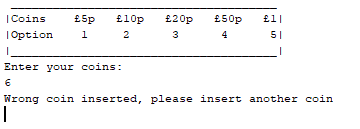
* Decline the transaction if the snack is out of stock.



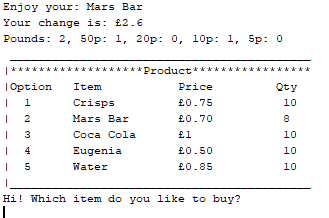
* Decline a transaction if the money for the snack is not paid.



* Reject a denomination not listed above.



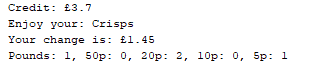
* Display the new quantity of the snacks after each purchase.



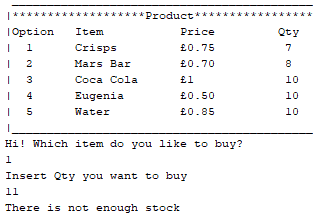
* Coins should be inserted one at a time for example if the price is £2.75, coins in the stated denominations above should be entered one at a time until either the price is met or exceeded.



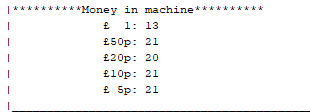
* Dispense the change if the user has entered more money than the price of the snack.



* Display an \*\*\*out of stock\*\*\* message if a snack is unavailable.



* Change should be dispensed using the current stock of change available in the machine.



* A power user feature which can be entered when the power user enters from the main menu the option 10976 and then prompting for a password taking him/her to a submenu where they can view the total amount of money in the machine and the total profit or losses.

