IS187: Java Programming I

Perfect Change Calculator (50 points) 2022

In this program (closely related to the MakeChange calculator done previously) you will make change for a dollar from a specific number of available coins. To do this, a method that runs at the beginning will ask the user for a specific set of coins to use. Note that in part B, you will incorporate the MakeChange algorithm by making the choice of using a coinset an option.

A sample run for **Part A** would be as follows:

Welcome to the Perfect Change calculator!

No. of Quarters: 3

No. of Dimes: 1

No. of Nickels: 2

No. of Pennies: 5

What value would you like change for? (1-100 cents, 101=New Coin set, or 0=quit): **23**

For 23 cent(s) I give:

0 quarters, 1 dimes, 2 nickels, 3 pennies leaving

3 quarters, 0 dime, 0 nickels, 2 pennies

What value would you like change for? (1-100 cents, 101=New Coin set, or 0=quit): **100**

For: 100 cent(s) I give:

3 quarters, 1 dimes, 2 nickels, 5 pennies leaving:

0 quarters, 0 dime, 0 nickels, 0 pennies

What value would you like change for? (1-100 cents, 101=New Coin set, or 0=quit): **101**

No. of Quarters: 3

No. of Dimes: 1

No. of Nickels: 2

No. of Pennies: 2

What value would you like change for? (1-100 cents, 101=New Coin set, or 0=quit): **4**

I could not make change for: 4 cents (I am short: 2 cents)

What value would you like change for? (1-100 cents, 101=New Coin set, or 0=quit): **0**

Thanks for using the Perfect Change calculator!

**Part B: Incorporating MakeChange**

For this version of the program you will make the use of a coinset an option. In other words, you will ask at the beginning if the user has a coinset, and if so you accept the coins and proceed as in part A. But if not, then the program will provide change using the same algorithm as coded in MakeChange (i.e., most efficient coins given without regard to limits on any particular coin). This also means that the user may go back and forth between using a coin set and not; by using option 101 but answering ‘N’ to the new ‘Do you have a specific set of coins’ the user effectively turns off the PerfectChange feature and returns to making change in the most efficient way possible. A run will now look as follows:

Welcome to the Perfect Change calculator!

Do you have a set of coins for making change? (Y/N): **Y**

No. of Quarters: 3

No. of Dimes: 1

No. of Nickels: 2

No. of Pennies: 5

What value would you like change for? (1-100 cents, 101=New Coin set, or 0=quit): **23**

For 23 cent(s) I give:

0 quarters, 1 dimes, 2 nickels, 3 pennies leaving

3 quarters, 0 dime, 0 nickels, 2 pennies

What value would you like change for? (1-100 cents, 101=New Coin set, or 0=quit): **101**

Do you have a set of coins for making change? (Y/N): **N**

What value would you like change for? (1-100 cents, 101=New Coin set, or 0=quit): **23**

For 23 cent(s) I give: 0 quarters, 2 dimes, 0 nickels, and 3 pennies

What value would you like change for? (1-100 cents, 101=New Coin set, or 0=quit): **101**

Do you have a set of coins for making change? (Y/N): **Y**

No. of Quarters: 3

No. of Dimes: 1

No. of Nickels: 2

No. of Pennies: 2

What value would you like change for? (1-100 cents, 101=New Coin set, or 0=quit): **4**

I could not make change for: 4 cents (I am short: 2 cents)

What value would you like change for? (1-100 cents, 101=New Coin set, or 0=quit): **0**

Thanks for using the Perfect Change calculator!

Finally, add another option to the ‘how much change’ question (i.e., **add option 102**) that produces an ‘all’ output result. Thus, if the user enters 102, the program should loop to produce results for all values 1 through 100 (using the coin counts if entered or the ‘most efficient’ method – whichever is currently selected in the program). This option should not fundamentally modify any of the existing methods in the program, but merely add a new method to loop through all possible values from 1 – 100 (making use of the other program methods, as needed).

Call the program **PerfectChange.java** and zip the finished netbeans project from the top-level folder. Submit it through blackboard as a separate zip file in the designated assignment bucket.