Project 1 – Change Calculator

Objective: To gain experience with input/output statements, modulo and integer division, and if-statements.

Description: You will be implementing a change calculator.  Your program will prompt the user for

1) The customer name

2) The item description

3) The amount tendered by the customer (in cents).

The program should randomly generate the price of the item that was purchased. This price should be in the range of 1 to 2000 cents. Output the customer name, description, cost (convert to dollars with decimal point), amount tendered (with decimal point) and change in one dollar bills, quarters, nickels, dimes and pennies. Your program output should appear formatted like the example below:

## Change making program by Mike Canniff

## What is the customer's first name? Bob

## Enter an item description: Books

## The price of Books (in cents): 1459

## Please enter the amount tendered in cents: 2000

## Bob bought Books for 14.59 and tendered 20.00

## Bob's change is:

## 5 one-dollar bills

## 1 quarter

## 1 dime

## 1 nickel

1 penny

## Hints

Note that the example execution shows input in pennies (1459 and 2000). You should use an appropriate data type to store all values in pennies. You will need to use quite a few variables and the modulo operator (%) to solve this problem. You will need to format some output to a width of 6 spaces.

## Implementation Details

* Start with a comment block at the beginning of your source code and add comments to the right of your variable declarations. See lab samples for details.
* Note that the input amount is in pennies to avoid handling decimal points
* You can assume that the user will always enter valid numbers when requested.
* When the resulting number of coins of a particular type is one, then your code should print out the singular version of that coin. Otherwise, use the plural version. E.g.
  + 1 Penny
  + 3 Pennies
* Your program should use constant definition for the cent value of each coin (const int NICKEL = 5;).

Helpful Code: The computer needs to generate random numbers in this project. In order to be able to generate random numbers, you must #include the ctime library at the top of your program. The code for creating random numbers should look like this:

// Place this ONCE at the beginning of your program

srand((unsigned)time(0)); // Initialize random number generator (ONCE)

int compNum;

// Use this line as often as needed to get a new random number between 1 and 5

// compNum is a variable name and you can use a different one if you prefer.

compNum = rand()%5 + 1;

Project Requirements:

You **must** follow all of the good programming practices discussed in class:

* Comment your code thoroughly.
* Indent your code appropriately.
* Use meaningful variable names.
* No global variables.
* Provide the user with understandable prompts and instructions.
* Make sure your name is included in comments at the top of your code.
* You are NOT allowed to use goto statements in this or any other COMP 51 projects.
* …

If this is not done, points will be deducted from your program and it will be impossible to earn an ‘A’.

## Extra Credit:

Use the built in cout formatting options to print out currency values. You can also experiment with DOS based color coding for input and output displays. Use your favorite search engine on how to do this.

Getting Started: If you sit down and try to implement the entire project in one-shot, it may seem over-whelming. It is important to break down the implementation into small manageable pieces and get each piece working correctly before starting on the next one. As you get more comfortable with programming, you’ll naturally start to see how to break apart the projects into small pieces. Below, I’ve suggested a possible sequence of steps you could take to implement your project. Feel free to use your own sequence if you prefer:

1. Write the code the code to prompt the users for input value.
2. Write the code to generate the item amount.
3. Write the code to calculate each dollar and coin amount.
4. Write the code to display the results.
5. Lastly, do extra credit (if you have time).

You should run and thoroughly test your code after each step. It is MUCH easier to find 1 error in 5 new lines of code than it is to find 10 errors in 50 lines of code.

Submission: Using the Sakai assignment feature, you should submit the source code (.cpp file). **Make sure you click to submit your assignment after uploading the file attachment!**