CISC 1600—Computer Science I
Spring, 2012

Programming Project # 1: Counting Coins
Original Due Date: Monday 2 February 2015
Revised Due Date: Wednesday 4 February 2015

Write a program that prompts the user to enter some number of

- pennies (1-cent coins)
- nickels (5-cent coins)
- dimes (10-cent coins)
- quarters (25-cent coins)
- half dollars (50-cent coins)

Query the user separately for the number of each size coin. Your program should then print out the number of each kind of coin, followed by the value of all the coins.

Here are a couple of sample runs of the program:

```
erdos@dsm:proj1$ proj1
Enter the number of coins you have for each denomination.
  Pennies? 23
  Nickels? 17
  Dimes? 14
  Quarters? 7
  Half dollars? 3
You have 23 pennies.
You have 17 nickels.
You have 14 dimes.
You have 7 quarters.
You have 3 half dollars.
The value of all your coins is 573 cents.
erdos@dsm:proj1$ proj1
Enter the number of coins you have for each denomination.
  Pennies? 1
  Nickels? 2
  Dimes? 0
  Quarters? 3
  Half dollars? 2
You have 1 penny.
You have 2 nickels.
You have 0 dimes.
You have 3 quarters.
You have 2 half dollars.
```

```
The value of all your coins is 186 cents.
erdos@dsm:proj1$ proj1
Enter the number of coins you have for each denomination.
  Pennies? 5
  Nickels? 1
  Dimes? 3
  Quarters? 2
  Half dollars? 1
You have 5 pennies.
You have 1 nickel.
You have 3 dimes.
You have 2 quarters.
You have 1 half dollar.
The value of all your coins is 140 cents.
erdos@dsm:proj1$ proj1
Enter the number of coins you have for each denomination.
  Pennies? 5
  Nickels? 0
  Dimes? 0
  Quarters? 0
  Half dollars? 0
You have 5 pennies.
You have 0 nickels.
You have 0 dimes.
You have 0 quarters.
You have 0 half dollars.
The value of all your coins is 5 cents.
erdos@dsm:proj1$ proj1
Enter the number of coins you have for each denomination.
  Pennies? 5
  Nickels? 0
  Dimes? 0
  Quarters? 0
  Half dollars? 2
You have 5 pennies.
You have 0 nickels.
You have 0 dimes.
You have 0 quarters.
You have 2 half dollars.
The value of all your coins is 105 cents.
```

A few considerations and hints:

1. You are to do this using only the techniques found in Chapters 1 through 3 of the text. In other words, you don't need to read ahead to figure out how to do this. There *are* less verbose ways to solve this problem using a loop and vectors, but we're not ready for that yet.

2. Unless you decide to get a head start, I expect that most of you will work on this assignment during your lab period (I won't be surprised if many of you finish this assignment in lab.) However, to get maximum utility out of your lab session, you *must* design and code your solution before you set foot into the lab. I will not let you approach a terminal until you show me that you have written out the program in advance.

3. I have made an executable version of this program for you to try out. The executable is available as the file

   `~agw/class/cs1/share/proj1/proj1`

   on the Departmental Linux machines. This means you can execute it by logging in on one of these machines, and doing the shell commands

   ```
   cd ~agw/class/cs1/share/proj1
   proj1
   ```

4. Use (appropriately-named) variables to store the number of half-dollars, quarters, dimes, nickels, and pennies.

5. Note that the program's output is grammatically correct, using singular and plural names as appropriate; this is slightly complicated by the fact that the plural of "penny" is "pennies", rather than "pennys". You can handle this via `if-else` statements, which are briefly seen on page 80 of the text.

6. You should do this project, in a subdirectory `proj1` of your `~/private/cs1` directory. You can create this via the shell command

   `mkdir␣~/private/cs1/proj1`

   (You shouldn't need the `-p` flag, since you previously created `~/private/cs1` when you did Project 0.) To guarantee that any files you create for this project (e.g., the C++ source code file `proj1.cc` and the executable program `proj1`) live in this directory, you will need to issue the shell command

   `cd␣~/private/cs1/proj1`

   before you start creating the file `proj1.cc`, i.e., before you run "`emacs proj1.cc`".

7. Remember to include the standard libraries by putting

   ```
   #include <bjarne/std_lib_facilities.h>
   ```

   near the beginning of your program. *Unless stated otherwise, this will be standard operating procedure from now on.*

8. You should test your program with the five sets of input values shown above. If you get the same answers as seen above, your program is good enough. For example, you needn't worry about erroneous input data (such as a negative number of coins).

9. As with Project 0, use the `photo` program so you can have something to turn in. *Don't do this until the program is working and you're ready to submit your solution!* The commands you should issue are as follows:

```
photo
cat proj1.cc
g++ -std=c++11 -o proj1 proj1.cc
proj1
proj1
proj1
proj1
proj1
exit
```

The input values you should use when running `proj1` are the same as were shown above. Note that you need to run `proj1` five times, once for each data set. *Do **not** use any other sets of input values.*

This will create a file, named `typescript`, in the current directory, containing a listing of `proj1.cc`, evidence that it compiles with no errors or warnings, and a sample execution. This `typescript` file may have "funny" characters, some of which may be removed by issuing the shell command

```
photo-clean typescript > proj1.out
```

You should make sure that your file `proj1.out` really contains what you think it contains. (It might be empty, or it might contain out-of-date material.) You can do this by looking at the file (via the shell command "`more proj1.out`") or by using the shell command "`ls -lt`" to ensure that `proj1.out` is nonempty and that it is newer than the `typescript` file before proceeding further. You may need to delete a pre-existing `proj1.out` file should this not be the case.

You can now turn in `proj1.out` in one of two ways:

(a) You can simply send me an email, letting me know that you're done with the project, and that I can get the clean typescript file myself. For this to work, your file must be named[1]

```
~/private/cs1/proj1/proj1.out
```

or I won't be able to find it. If you follow the instructions given above (i.e., make sure that you're in the `~/private/cs1/proj1` directory before you start your work and that the clean typescript is named `proj1.out` within that directory), then this will happen automatically.

(b) You can email it to me, by issuing the shell command

```
mail -s "Project 1" -r bovik@cs.cmu.edu agw < proj1.out
```

The `-r` flag indicates a "return address". In other words, this is where you want the acknowledgment message to be sent. **Of course, you should replace `bovik@cs.cmu.edu` by your own email address.**

[1] As always, the tilde denotes your home directory.

Extra credit (10 points): In addition to reporting the total number of cents, see if you figure out how to print the total in the usual "dollars and cents" format, e.g.:

```
erdos@dsm:proj1$ proj1
Enter the number of coins you have for each denomination.
  Pennies? 23
  Nickels? 17
  Dimes? 17
  Quarters? 7
  Half dollars? 3
You have 23 pennies.
You have 17 nickels.
You have 17 dimes.
You have 7 quarters.
You have 3 half dollars.
The value of all your coins is 603 cents, which is $6.03.
```

If you choose to implement this feature, you must still use the same five test data sets given at the beginning of this handout.

Good luck!