Sonoma State University
Computer Science Department
CS 460 – Fall 2023 – Watts

Exercise 1
(Due dates: 23 August 2023 and 29 August 2023)

For **Exercise 1 Part A,** you are to familiarize yourself with our course website:
http://watts.cs.sonoma.edu/cs460f23/ and read the course syllabus.

Date Due: Wedesday, 23 August 2023, 6:59 am.

For **Exercise 1 Part B,** you will be refreshing your C++ programming skills and using
the course submission system.

1. Using an IDE or editor of your choice, write a well documented and formatted
   C++ program to solve the problem (Change Back) described in the specification
   at the end of this document.

2. If necessary, upload your program to your blue.cs.sonoma.edu account. Name the
   single file containing your code *lastname*E1b.cpp.

3. Compile your program using the g++ compiler. Test your program using input
   files of your choice.

4. Copy your .cpp file to the course dropbox: ~tiawatts/cs460drop and check the
   submissions page to make sure that your file was properly submitted. Within 10
   minutes, you should receive an email from me containing the results of the
   submission script's actions.

Practice submission: Monday 28 August 2023, 6:59 am.

Date Due: Wednesday, 30 August 2023, 6:59 am.

Problem 1b: Change Back

The Problem

Modern grocery stores now often have a ``U-Scan'' checkout lane - allowing the customer to scan and check out his/her own groceries, without the need of a human checker. These lanes require that change be provided automatically, after the customer enters his/her cash. You are to write a program that computes the bills and coins to be dispensed, minimizing the total number of bills and coins. (That is, for change totaling $5.50, you should not dispense 5 ones and 50 pennies, but a $5 bill and a 50-cent piece.) The bills and coins available for you to dispense are as follows: $50 bill, $20 bill, $10 bill, $5 bill, $1 bill, 50-cent coin, 25-cent coin, 10-cent coin, 5-cent coin, 1-cent coin.

Input

The input file will consist of two numbers per line, each constituting a transaction. The first number is the amount of the purchase, and the second one is the amount tendered by the customer. You may assume that the amount tendered is greater than or equal to the amount of purchase. Input will be terminated by a line with both numbers being 0.0. The name of the input file will be provided as a command line argument when the program is executed.

Output

Output for each transaction will be a series of lines showing the amount of change returned and detailing the number of bills and coins that will be dispensed as change, in descending order of monetary amount, one unit per line. If a bill/coin is not needed in the change returned, no output is produced for that bill/coin. (In other words, do not display `0 $1 bills'.) Proper use of plurals is required, as shown below. Each output set should start with the word TRANSACTION and the number of the data set preceded by a # and followed by a :. Separate transactions by a blank line. All output should be sent to standard output (cout).

Sample Input

```
42.15 50.00
2.77 5.00
99.99 100.00
0.0 0.0
```

Corresponding Output

```
TRANSACTION #1:
$7.85
1 $5 bill
2 $1 bills
1 50-cent coin
1 25-cent coin
1 10-cent coin

TRANSACTION #2:
$2.23
2 $1 bills
2 10-cent coins
3 1-cent coins

TRANSACTION #3:
$0.01
1 1-cent coin
```