ECE 36800 Assignment #1

Original Due: 1:00 PM, Tuesday, September 3
Extended: 1:00 PM, Thursday, September 5

Description

Brute-force is a straightforward approach to solving a problem, usually directly based on the problem statement and definitions of the concepts involved. For this warm-up assignment, you will write a program that reads a number of cents from standard input, and prints out to standard output all combinations of quarters, dimes, nickels, and pennies that add up to that number of cents. Your program should first print out the minimum number of coins needed by using the largest coin values (i.e., quarter, dime, nickel, and penny in order) to make the amount before showing additional coins.

For clarity regarding the statement about using the largest coin values first, consider the example of 90 cents below. Without this prioritization, the third output line would appear before the second one. However, by prioritizing the use of the largest coin values, the program first uses the maximum number of dimes, and then prints the minimum number of coins needed based on that.

For example, given the following input:

```
10
```

your program should output:

```
0 quarter(s), 1 dime(s), 0 nickel(s), 0 pennie(s)
0 quarter(s), 0 dime(s), 2 nickel(s), 0 pennie(s)
0 quarter(s), 0 dime(s), 1 nickel(s), 5 pennie(s)
0 quarter(s), 0 dime(s), 0 nickel(s), 10 pennie(s)
```

Example with a higher number, given the following input:

```
90
```

your program should output:

```
3 quarter(s), 1 dime(s), 1 nickel(s), 0 pennie(s)
3 quarter(s), 1 dime(s), 0 nickel(s), 5 pennie(s)
3 quarter(s), 0 dime(s), 3 nickel(s), 0 pennie(s)
3 quarter(s), 0 dime(s), 2 nickel(s), 5 pennie(s)
...
...
0 quarter(s), 0 dime(s), 1 nickel(s), 85 pennie(s)
0 quarter(s), 0 dime(s), 0 nickel(s), 90 pennie(s)
```

Grading

This assignment will be graded based on both the correctness and memory safety of your implementation. Each test case will be worth the same amount of points, and will be all-or-nothing (i.e. you either have the correct output or you don't). Note that even minor differences in output will cause you to fail grading test cases. You will receive a 0 if your submission fails to terminate within a 10-minute time limit.

You may submit to Gradescope as often as you'd like before the deadline. Only your active submission (by default the most recent) will be counted. While the score given to you by the autograder is likely a good indicator of your final grade on the assignment, we reserve the right to add additional test cases after the submission deadline.

Submission

Submit any source/header files with your implementation, as well as a Makefile that builds a target called `a1`, to Gradescope. Note that to receive points, your submission must work on eceprog.