Cash

Implement a program that calculates the minimum number of coins required to give a user change.

```
$ python cash.py
Change owed: 0.41
4
```

Specification

Write, in a file called `cash.py` in `~/pset6/cash/`, a program that first asks the user how much change is owed and then spits out the minimum number of coins with which said change can be made, exactly as you did in Problem Set 1, except that your program this time should be written (a) in Python and (b) in CS50 IDE.

Use `get_float` from the CS50 Library to get the user's input and `print` to output your answer. Assume that the only coins available are quarters (25¢), dimes (10¢), nickels (5¢), and pennies (1¢).

> We ask that you use `get_float` so that you can handle dollars and cents, albeit sans dollar sign. In other words, if some customer is owed $9.75 (as in the case where a newspaper costs 25¢ but the customer pays with a $10 bill), assume that your program's input will be `9.75` and not `$9.75` or `975`. However, if some customer is owed $9 exactly, assume that your program's input will be `9.00` or just `9` but, again, not `$9` or `900`. Of course, by nature of floating-point values, your program will likely work with inputs like `9.0` and `9.000` as well; you need not worry about checking whether the user's input is "formatted" like money should be.

If the user fails to provide a non-negative value, your program should re-prompt the user for a valid amount again and again until the user complies.

Incidentally, so that we can automate some tests of your code, we ask that your program's last line of output be only the minimum number of coins possible: an integer followed by a newline.

Usage

Your program should behave per the example below.

```
$ python cash.py
Change owed: 0.41
4
```

Testing

No `check50` for this problem, but be sure to test your code for each of the following.

Run your program as `python cash.py` , and wait for a prompt for input. Type in `0.41` and press enter. Your program should output `4` .

Run your program as `python cash.py` , and wait for a prompt for input. Type in `0.01` and press enter. Your program should output `1` .

Run your program as `python cash.py` , and wait for a prompt for input. Type in `0.15` and press enter. Your program should output `2` .

Run your program as `python cash.py` , and wait for a prompt for input. Type in `1.60` and press enter. Your program should output `7` .

Run your program as `python cash.py` , and wait for a prompt for input. Type in `23` and press enter. Your program should output `92` .

Run your program as `python cash.py` , and wait for a prompt for input. Type in `4.2` and press enter. Your program should output `18` .

Run your program as `python cash.py` , and wait for a prompt for input. Type in `-1` and press enter. Your program should reject this input as invalid, as by re-prompting the user to type in another number.

Run your program as `python cash.py` , and wait for a prompt for input. Type in `foo` and press enter. Your program should reject this input as invalid, as by re-prompting the user to type in another number.

Run your program as `python cash.py` , and wait for a prompt for input. Do not type anything, and press enter. Your program should reject this input as invalid, as by re-prompting the user to type in another number.