CSC241 - Assignment 1

Reading: Chapters 1 and 2

Instructions:

1. **Read the HW guidelines** posted in d2l.depaul.edu > contents > admin.
2. The file containing your solutions MUST be named **hw1.py**. In a comment at the top of your file, include the name(s) of any collaborators.
3. IDLE exercises:
   - Perform these in IDLE
   - Copy your solutions from IDLE to hw1.py and comment out
4. Programming problems:
   - Write a function definition (def) for each problem and include in hw1.py.
   - Run your module (F5)
   - Test the function in IDLE, run the tests specified in the descriptions below.
   - To receive full credit, the **names** of files, functions and the **output** must be **exactly** as indicated here.
5. **Test your code** by downloading the file **hw1TEST.p**y in the same working folder. You can then add the following at the bottom of hw1.py and then run your module:

   ```
   if __name__=='__main__':
       import doctest
       print( doctest.testfile( 'hw1TEST.py') )
   ```

   Then run your module (F5). IF everything works, you will get something like this:

   ```
   >>>
   TestResults(failed=0, attempted=18)
   >>>
   ```

   If not, you will get error messages. Fix your errors, rerun the test, and submit to d2l > dropbox when you finish.

IDLE exercises

Start by assigning `s1 = '&'` and `s2 = '#'` in the Python shell. Then write string expressions involving only s1 and s2 and the string operators + and * that evaluate to the following. Make your string expressions as succinct (as short) as you can.

a. '&&#'
b. '##&'
c. '&&#&&#'
d. '&&&&&&&&&&'
e. '&&##&&##&&##'
f. '&&&&&#####&&&&&#####&&&&&#####'

Programming problems

1. Write a function total that request a decimal value representing a price and an integer representing a quantity from the user. If both numbers are positive (not negative and not zero), your program should print the price times the quantity using a dollar sign appropriately. If either value is negative or zero, the program should print an error message indicating that both numbers must be greater than zero. Here are sample runs:

```
>>> total()
Enter a price: 13.25
Enter a quantity: 6
The total price is: $ 79.5
>>> total()
Enter a price: 13.25
Enter a quantity: -6
Error: both numbers must be greater than zero.
>>> total()
Enter a price: 0.0
Enter a quantity: 4
Error: both numbers must be greater than zero.
>>> total()
Enter a price: -23.55
Enter a quantity: 3
Error: both numbers must be greater than zero.
>>> total()
Enter a price: -3.50
Enter a quantity: -4
Error: both numbers must be greater than zero.
>>> total()
Enter a price: 0
Enter a quantity: 0
Error: both numbers must be greater than zero.
>>>
```

2. Implement a program that requests a number from the user and indicates whether that number is positive (strictly greater than 0), negative (strictly less than 0), or equal to 0. You may assume that the user enters a numeric value. Sample runs:

   ```
   >>> posNegZero()
   Enter a number: -2.3
   You entered a negative number.
   >>> posNegZero()
   Enter a number: 0.0
   You entered zero.
   >>> posNegZero()
   Enter a number: 23.55
   You entered a positive number.
   >>>
   ```

3. Write a program that requests a number from the user and determines whether that number is a perfect square (the square of an integer) or not. You may assume that the user enters a numeric value. Sample runs:

   ```
   >>> square()
   Enter an integer: 9
   9 is 3 squared.
   >>> square()
   Enter an integer: 12
   12 is not a perfect square.
   >>> square()
   Enter an integer: 0
   0 is 0 squared.
   >>> square()
   Enter an integer: 7
   7 is not a perfect square.
   >>>
   ```