## 3.8 Exercises

Write a complete Python program to solve each of the following problems.

3.1. Write a program that inputs a number and says if it is "nice" or not. If the number is even it is nice if it is divisible by 100. If the number is odd it is nice if it is divisible by 25. Here is a typical run

> **Enter a number:** 125
> **That is a nice number!**

3.2. Now modify the program from (Problem 3.1) by wrapping a loop around it. Your program should quit when the user enters the number 0:

> **Enter a number:** 125
> **That is a nice number!**
> **Enter a number:** 350
> **That number is not nice.**
> **Enter a number:** 5400
> **That is a nice number!**
> **Enter a number:** 0
> **goodbye!**

3.3. Write a program that inputs a sequence of strings, terminated by the empty string. For each string the program should say whether it contains "bob" as a substring. You can use the **in** operator to check for substrings: "bob" **in** s is True if string s contains "bob" as a substring and False if it doesn't.

3.4. Write a program that reads numbers from the user, and then prints their average. As you know, the average is the sum of the numbers divided by how many there are. Be sure to make the sum be a float, not an integer; one way to do this is to start it off at 0.0.

> **Enter a number:** 4
> **Enter a number:** 12
> **Enter a number:** 8
> **Enter a number:** 0
> **The average of those numbers is 8.00**

3.5. Revise the program from Problem 3.4 so that in addition to the average it prints the largest and smallest of the values it has seen. For example:

> **Enter a number:** 4
> **Enter a number:** 12
> **Enter a number:** 8
> **Enter a number:** 3
> **Enter a number:** 13
> **Enter a number:** 11

**Enter a number:** `0`
**The average of those numbers is 8.50**
**The largest is 13 and the smallest is 3.**

3.6. Write a program that reads strings over and over until it sees the word "quit", at which point it prints "Goodbye" and stops. While it is reading strings, each time it sees the string "bob" it replies "BOB RULES!" It igores all other words except "quit".

**Enter a string:** `foobar`
**Enter a string:** `Oberlin`
**Enter a string:** `Bob`
**Enter a string:** `bob`
**BOB RULES!**
**Enter a string:** `Harry Potter`
**Enter a string:** `quit`
**Goodbye**

3.7. Here is a harder one. Write a program that inputs a string and says if it is a palindrome: a word that is the same forwards and backwards, like "bob" or "anna" or "amanaplanacanalpanama". One way to do this is to have a counter start at the front of the string and another start and the end. At each step you compare the letter at each counter; if these are the same you increment the front counter and decrement (subtract 1 from) the back counter. Continue on until you find a difference or the counters cross.

3.8. Write a program that finds all of the perfect numbers less than 100,000. A number is called *perfect* if its factors (including 1 but not the number itself), sum to the number. For example, 6 is perfect because its factors are 1,2, 3 and 6 and $1 + 2 + 3 = 6$.

3.9. Write a program that computes the factorial of integer `n`. This is the product of the numbers from 1 to `n`. For example, the factorial of 5 is $1 \times 2 \times 3 \times 4 \times 5 = 120$. Factorials quickly become very large; even 10 factorial, which is 3628800, has 7 digits. See what happens if you try to compute 100 factorial. Add a feature to your program to print the number of digits in the factorials in addition to their values.