Practice S03P03: Count input values

http://www.comp.nus.edu.sg/~cs1010/4_misc/practice.html

Reference: Week 4

Week of release: Week 4

Objectives: Repetition statement, more about **scanf()**

Task statement:

When a program needs to read in a number of values, there are a few approaches:

1. The number of input values is known before-hand, in which case the user can provide this number, and you may use a loop to read in exactly the number of inputs.

    A sample run is shown below:

```
How many input values to read? 3
Enter the values below:
4
5
2
Number of values entered = 3
```

2. The number of input values is unknown before-hand. You may choose a special value to indicate the end of inputs. For instance, if the input data are non-negative integers, you may enter a negative value to terminate the input.

    A sample run is shown below:

```
Enter integers, terminate with a negative integer:
4
5
2
-1
Number of values entered = 3
```

3. The number of input values is unknown before-hand, and you do not want (or it is impossible) to choose a special value to denote the end of inputs. Here, the user may press the 'end-of-input' key to terminate the inputs.

The 'end-of-input' key for interactive inputs on UNIX is **ctrl-d**.

A sample run is shown below:

```
Enter integers, terminate with ctrl-d
5
-2
3
12
(user pressed ctrl-d here)
Number of values entered = 4
```

We have done the first two approaches in class. In this exercise, you are to implement the third approach. To do this, you may use a certain feature of the **scanf()** statement which we have not been using so far.

The **scanf()** statement does more than just reading input. It also returns the number of inputs it has read.

For instance, the following code checks whether the **scanf()** statement has read in one input:

```
if (scanf("%d", &value) == 1)
    printf("One input was read and it is %d\n", value);
```

Sample runs:

```
Enter integers, terminate with ctrl-d
100
(user pressed ctrl-d here)
Number of values entered = 1
```

```
Enter integers, terminate with ctrl-d
5
-2
3
12
(user pressed ctrl-d here)
Number of values entered = 4
```