York University
Lassonde School of Engineering
Dept. of Electrical Engineering and Computer Science
EECS 2031
Software Tools
Winter 2017

EECS2031	Lab Test 1	Software Tools

Thursday, Feb. 9th, 2017   6:00– 8:00pm

Last page contains some I/O function prototypes; you may want to look at it first

Question 1 (10 points)

Write a program that reads a sequence of integers (positive, negative, or zero) and calculate the average of the positive integers only. **The result should be displayed as floating point number in 6 places with 2 digits after the decimal point (right justified, so use %6.2f) followed by a new line.** If there are no positive numbers, you should display the following statement followed by a new line
No positive numbers!
For example (what you type is red, system types in black, CTRL-D on a separate line after -7)

```
EECS% a.out
3
6
-5
12
9
0
-7
  7.50
```

Submit as 2031 LT1A1 av.c

Question 2 (10 points)

Write a program that reads a *special* string from the standard input and display its length. The *special* string does not contain any consecutively repeated characters. For example a C string that contains the phrase LIKE is represented as 'L' 'I' 'K' 'E' '\0'.
Since our string doesn't contain any consecutively repeated characters, we repeat the last character to terminate the string, so the above string is represented as 'L' 'I' 'K' 'E' 'E', where the second 'E' terminates the string (and is not a part of the string).
Here are few strings to represent few phrases.
The maximum length of the string is 50 characters

String	Representation	Output
Hi there	Hi theree	8
This test is easy	This test is easyy	17
I am free today	Could not be represented because there are two consecutive 'e's in the string	
I am busy Friday morning	I am busy Friday morningg	24 (space is a part of the string)

Your program should read one line of text and display its length as a left justified integer (just %d) followed by a new line character

Submit as 2031 LT1A2 string.c

Question 3 (10 points)

Write a program that reads from the standard input an inventory file using the following format:

Product ID   **price per unit**   **Number of units available**

The file starts with an integer to indicate the number of records, followed by records each record in a separate line.

The first two fields are in every record. The third field if 0 it is not there at all, for example.

```
3
AC1234          132.25          12
KO1234          412.67
BB9876          305.00          18
```

That means the second product has 0 units.

Your program calculates the total inventory value and display it as a left justified floating point number with two decimal digits (%.2f note there is a decimal point before the 2) followed by a new line character.

Constraints:
- ID is a 6-character string
- The maximum number of records in the file is 30

Submit as LT1A3 inventory.c

Some Functions you might need

`int fgetc(FILS *stream);`

Reads the next character from stream and returns it as unsigned int, or returns EOF on end of file or error

`char * fgets(char *s, int size, FILE *stream);`

Reads in at most one less than *size* characters from *stream* and stores them into the buffer pointed to by *s*. Reading stops after an **EOF** or a newline. If a newline is read, it is stored into the buffer. A terminating null byte is stored after the last character in the buffer.
Returns s on success or NULL on error

In both cases, you may read from the standard input using stdin

For example,

`fgets(s, 10, stdin)`

Reads from the standard input maximum of 9 characters and stores them in the buffer pointed to by s. No need to *open stdin* or *stdout*, they are automatically opened for you.

`int scanf(const char *format, ...);`

Returns the number of input items successfully matched or EOF if end of file.