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, March. 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)

The following table shows the credit card number for the major credit cards. The number consists of digits only. The prefix is the left-most digits in the card number. The length is the number of digits in the card number including the prefix.

CARD TYPE	Prefix	Length
MASTERCARD	51-55	16
VISA	4	13, 16
AMEX	34 37	15

Write a program that reads from the standard input a sequence of credit card numbers (the sequence may or may not contain non-digits) each in a separate line, until EOF. Then display one of the following 2 messages for every card number read.

- If the number is valid, then display VALID followed by (VISA, MC, or AMEX) followed by a new line. There is exactly one space between valid and the card name.
- If the number is not valid, display INVALID NUMBER followed by a new line

Submit as 2031 LT2A1 CC.c

Question 2 (10 points)

Read from the standard input, till EOF, and display the number of lines and the number of white spaces (space or tab) as follows. The result is displayed once after you finish reading the input.

Lines: xx ("Lines: %d\n")
White spaces: xx ("White spaces: %d\n");

Consider the number of lines to be the number of '\n' character in the file.

Submit as 2031 LT2A2 line_count.c

Question 3 (10 points)

Read and validate a database (Read from the standard input until EOF).

- Each record is up to 40 characters and ended by a new line.
- The 40 or less characters are all alphabet characters or digits. No special characters, punctuation characters, or white spaces.
- Each record starts with either a digit or non-digit
    - If the left-most (first) character is a digit, the rest of the line should NOT contain any digits, only alphabet letters.
    - If the first (left-most) character is a non-digit (cap or small letter), the rest of the line can contains any combination of small cap alphabet letters and digits.

For every line read, your program should display VALID or INVALID followed by a new line.

Examples: input in red, displayed output in black

3abcdegkknheftyAnm
VALID
7asdfgh5hg6hjkj
INVALID
thhjdshfhkjjkdfgkjdhfhfgjhsdgfds
VALID
Thjdhgfhdgf4hgd34fhgf
VALID
ThjgeyrnhjfdewAghgh
INVALID

Submit as LT2A3 check.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.

Also do man pages on the following

`isupper, isalpha, isdigit, isalpha`. …