**91.203.201 Homework 3**

**Due Date: 9/24 (W) 11:59 PM**

Write a MIPS program which repeatedly read one line at a time from the keyboard. For each line that you read, echo print of the input line and types of individual characters on the line as defined by

Characters Type Comment

0 1 .. 9 1 Digits

A B .. Z a b .. z 2 Letters

\* + - / 3 Operators

. ( ) , : 4 Delimiters

b 5 blank

# 6 End of the Line

For example, when a line of the input below

THISLOOP: LWU R2, 3 #

the output from the program will be like

THISLOOP: LWU R2, 3 #

222222224 222 2141 6

The blank type of 5 can be left out.

And this process is repeated by inputting another line, etc.

**Approaches:**

1. Reading input lines

Each input line can be read by calling a procedure, called getline(), by allocating the maximum of 80 character space and calling a syscall for reading a string. Make it sure that your main program calls getline repeatedly until a special symbol (‘#’) is entered to terminate the loop. A procedure call to getline is made by the following MIPS statement:

jal getline

An example of getline procedure to read an input string:

getline:

la $a0, st\_prompt # Prompt to enter a new line

li $v0, 4

syscall

la $a0, inBuf # read a new line

li $a1, 80

li $v0, 8

syscall

jr $ra

.data

inBuf: .space 80

st\_prompt: .asciiz "Enter a new input line. \n"

2. Linear search of a character

A simple approach to find a type of a character is to arrange all characters and their types into an array. This make a change to the character set simply. Use the following table and write a linear search program to perform the search for an input character from the input string. Be careful that a character from the input string is stored in a byte where as the search table below is organized in units of words (4-byte each). The linear search part in your program HAS TO BE coded as a function. When you compare a letter from the input string to characters in Tabchar (listed below), make it sure that you use ‘**lb**’ instruction to move only a byte out of the input string to one of registers.

**What to hand in:**

* MIPS Source code with sufficient comments (comments will be graded).

Email to [airingzhang@yahoo.com](mailto:airingzhang@yahoo.com).

Notes:

1. Comments in an assembly program is very important, especially it is difficult to follow which registers contain what values. Too much of comments clutters the program and makes it harder to read. Be judicious about comments so that one can follow the flow of the program from comments. Grades will be affected by how well you include comments.
2. From the input keyboard, you can enter the maximum of 80 characters. You can reserve a space of 80 characters by

inBuf: .byte 80

1. The end of the input string can be detected by the first Null character (0x00).
2. The reshuffled string is stored at a different location.
3. For a word in a string, you need to be able to access individual bytes in the word, and use ‘lb’ (‘sb’) instruction to access (or store) a byte.
4. After a successful run of your program, take a screen shot of the memory map, and indicate by arrows where the input string and the reshuffled string are stored.

.data

Tabchar: .word ' ', 5

.word '#', 6

.word '(', 4

.word ')', 4

.word '\*', 3

.word '+', 3

.word ',', 4

.word '-', 3

.word '.', 4

.word '/', 3

.word '0', 1

.word '1', 1

.word '2', 1

.word '3', 1

.word '4', 1

.word '5', 1

.word '6', 1

.word '7', 1

.word '8', 1

.word '9', 1

.word ':', 4

.word 'A', 2

.word 'B', 2

.word 'C', 2

.word 'D', 2

.word 'E', 2

.word 'F', 2

.word 'G', 2

.word 'H', 2

.word 'I', 2

.word 'J', 2

.word 'K', 2

.word 'L', 2

.word 'M', 2

.word 'N', 2

.word 'O', 2

.word 'P', 2

.word 'Q', 2

.word 'R', 2

.word 'S', 2

.word 'T', 2

.word 'U', 2

.word 'V', 2

.word 'W', 2

.word 'X', 2

.word 'Y', 2

.word 'Z', 2

.word 'a', 2

.word 'b', 2

.word 'c', 2

.word 'd', 2

.word 'e', 2

.word 'f', 2

.word 'g', 2

.word 'h', 2

.word 'i', 2

.word 'j', 2

.word 'k', 2

.word 'l', 2

.word 'm', 2

.word 'n', 2

.word 'o', 2

.word 'p', 2

.word 'q', 2

.word 'r', 2

.word 's', 2

.word 't', 2

.word 'u', 2

.word 'v', 2

.word 'w', 2

.word 'x', 2

.word 'y', 2

.word 'z', 2