CSI 333 – Programming at the Hardware-Software Interface – Fall 2011

Programming Assignment V

Date given: Nov. 15, 2011 **Due date:** Dec. 5, 2011
Weightage: 10%

The deadline for this assignment is **11 PM, Monday, December 5, 2011**. The assignment will *not* be accepted after that deadline. **There is no two-day grace period for this assignment.**

Very important: This assignment also has two parts and both parts must be done in MAL. Thus, you will be two MAL source files for this assignment by the deadline specified above. The names of the two files that you submit must be `p5a.mal` and `p5b.mal`.

The total grade for the assignment is 100 points, with 40 points for Part (a) and 60 points for Part (b). As in the case of the other projects, your program must be well documented. (Please consult the handouts on MAL to understand how MAL programs are documented.)

Description of Part (a):

For this part, your MAL program must be in a file named `p5a.mal`. In the following discussion, a **positive** integer is one that is *strictly* greater than zero. Also, we say that an integer x **evenly divides** an integer y if the remainder when y is divided by x is zero.

Recall that each integer is represented using 32 bits in MIPS. Assume that the bits are numbered *right to left* from 0 to 31. (Thus, the rightmost bit is numbered 0 and the leftmost bit is numbered 31.) We say that the bits 0 through 15 form the **right half** and the bits 16 through 31 form the **left half** of the integer.

You are required to write a MAL program that prompts a user for a positive decimal integer, reads the integer typed by the user and outputs the following values.

(a) The total number of 1's in the *right half* of the binary representation of the integer.

(b) The total number of 0's in the *left half* of the binary representation of the integer.

(c) The highest power of 2 that evenly divides the integer.

(d) The value of the largest digit in the decimal representation of the integer.

Example: Suppose the user types the decimal integer 1536. The 32-bit binary representation of 1536 is as follows:

```
0000 0000 0000 0000 0000 0110 0000 0000
```

For this example, the required answers are as follows.

(a) The number of 1's in the right half of the binary representation of the given integer = 2.

(b) The number of 0's in the left half of the binary representation of the given integer = 16.

(c) The largest power of 2 that evenly divides the given integer = 9. (See the note following this example.)

(d) The value of the largest digit in the decimal representation of the given integer is 6.

Note: In the above example, the largest power of 2 that evenly divides 1536 is 9. This is because $1536 = 3 \times 512 = 3 \times 2^9$. This highest power (namely, 9) is also the number of 0's *at the end* of the binary representation of 1536.

Program outline: The outline for your program for Part (a) must be the following.

```
1. Prompt the user for a positive integer.
2. Read the integer.
3. Compute the four quantities mentioned above and print the answers.
4. Stop.
```

In particular, note that each time your program for Part (a) is executed, it should handle *just one* integer.

You may assume that the value typed by the user is a positive decimal integer. Thus, there is no need to do any error checking in this part.

Description of Part (b):

For this part, your MAL program must be in a file named `p5b.mal`. *It must have at least one function in addition to the main program.* For the purposes of Part (b), you may assume the following.

1. Any line of text typed by a user has at most 80 characters *including* the newline character.
2. A **whitespace** character refers to a space, a tab or the newline character.
3. A **word** is any sequence of characters that does not contain a whitespace character.

In this part, you are required to write a MAL program that prompts the user for a line of text and reads the line typed by the user. If the line contains just white space characters your program should simply output the message "`Line contains only white space characters.`" and stop. Otherwise, your program should compute and output the following.

(a) The number of non-whitespace characters in the line.

(b) The number of words in the line.

(c) The maximum length of a word in the line.

(d) The minimum length of a word in the line.

(e) The word of maximum length in the line. (If there are two or more words of maximum length in the line, then the program should print the word of maximum length that appears *first* in the line.)

(f) The word of minimum length in the line. (If there are two or more words of minimum length in the line, then the program should print the word of minimum length that appears *first* in the line.)

Example: Suppose the line typed by the user is the following:

```
It was  the  best   of times  and it  was the  worst of times.
```

The answers for the above line are:

```
No. of non-whitespace characters: 43
No. of words: 13
Maximum length of a word: 6
Minimum length of a word: 2
Word of maximum length: times.
Word of minimum length: It
```

Note that the word of maximum length (6) is `"times."` (without the quotes), which includes the punctuation mark at the end. (Recall that a word is any sequence of characters that does *not* include a whitespace character.) There are several words of minimum length (2) in the above text. The first such word is `"It"` (again, without the quotes).

Program outline: The outline for your program for Part (b) must be the following.

```
1. Prompt the user for a line of text.
2. Read the line of text.
3. If the line has only whitespace characters
       Print the message "Line contains only white space characters"
   else
       Compute the quantities mentioned above and print the answers.
4. Stop.
```

In particular, note that each time your program for Part (b) is executed, it should handle *just one* line of text. Note also that except for checking whether the input line consists of just whitespace characters, no other error checks are needed.

Information about `README` file: The `README` file for this assignment will be available by 10 PM on Sunday, November 20, 2011. The name of the file will be `prog5.README` and it will be in the directory `~csi333/public/prog5` on `itsunix.albany.edu`.