CSCI 201 Computer Organization I

MIPS Programming Project 3

Due December 6th, 4PM ET

Description:

Assume your Howard ID as a decimal integer is X. Let N = 26 + (X % 11) where % is the modulo operation, and M = N – 10.

You will write a MIPS program that reads a string of *up to* 1000 characters from user input.

- With single comma as the delimiter, split the input string into multiple substrings (with the comma removed). If there is no comma in the input, the whole input string is considered a substring referred to below.
- For each substring, remove the leading and trailing blank spaces and tab characters if any. After that
  - If the substring has zero characters or more than 4 characters or has at least one illegal character (a character outside the set described below), the program prints a question mark "?" as an error message.
  - If the substring has only the characters from '0' to '9' and from 'a' to β and from 'A' to Δ, the program prints out the number of digits and a slash '/', followed by the unsigned decimal integer corresponding to the base-N number represented by the substring. β stands for the M-th lower case letter and Δ stands for the M-th upper case letter in the English alphabet. In a base-N number, both 'a' and 'A' correspond to the decimal integer of 10, both 'b' and 'B' to 11, and so on, and both β and Δ correspond to N – 1.
  - If there are multiple substrings, the numbers and the error message should be separated by a single comma.
- The program must exit after processing one single user input.
- The processing of the whole input string must be done in a subprogram (Subprogram A, *must be labelled as sub_a*). The main program must call Subprogram A and pass the string address into it **via the stack**. Subprogram A parses the string and prints out the integers and error messages one by one, with them separated by single comma. No return value is necessary from Subprogram A.
- When processing each substring, Subprogram A must call another subprogram (subprogram B, *must be labelled as sub_b*), where the substring address is passed into Subprogram B **via the stack**, and the decimal number or error message is returned to Subprogram A also **via the stack**.

Sample test cases (assuming the Howard ID is 12345678):

12345678 % 11 = 4, therefore the base is 26 + 4 = 30, β is 't' and Δ is 'T'.

- Input: 0, 1, 2
  Output: 1/0,1/1,1/2

- Input: A,B,C,D
  Output: 1/10,1/11,1/12,1/13
- Input: 012
  Output: 3/32
- Input: 100,200
  Output: 3/900,3/1800
- Input: 1023
  Output: 4/27063
- Input: A12t,,12345,axyz,a b,T00a
  Output: 4/270989,?,?,?,?,4/783010
- Input: a@t
  Output: ?

Requirements:

- The program must be able to run correctly under QtSpim.
- Commits must not be made after the submission deadline.
- The program must be named as project3.s.
- The program must be in **a new Github repo** different than that used for Project 1 & 2.
- The output must have the exact format as specified above. **No other messages should be printed except the numbers and the error messages.**
- The program must use one or more loops to process the characters in the user input, instead of producing multiple segments of similar code with each segment processing one single character.
- The subprograms must be labelled properly as required.
- All development must be done with Github. Specifically, there MUST BE ONE COMMIT for EVERY (or fewer for the last) five lines of new or modified MIPS code (excluding empty lines, and lines with only comment/directives/labels). More frequent commit is fine. With each commit, the commit message must explain the purpose of the added/changed code. **The commits must be done while the program is developed, NOT after**. Commits after the development completion are NOT ACCETPABLE.
- **Submissions not fully meeting the above requirements will lose significant portion (≥ 70%) of the credits.**
- Syntax error or program terminating abnormally without output will result in **zero credits**. Therefore, make sure to test your program completely before submission.

Submission:

- Add 'csdrli' (the instructor) as collaborators to your Github repository.
- Create a plain text file named readme.txt. Such a file can be created with notepad on Windows and -o on MacOS or Linux. The file should include ONLY the link to your Github repository that can be used to clone your repo.
- Submit the readme.txt file to http://hucs.dynu.net/lij/courses/submit_hw.html under "CSCI 201 MIPS Programming Project 3". Anything else must not be submitted.