Typing game LC-3 Project Proposal

Hyomin Cha

Author Note:

I am current students at Utah Valley University Research is being done without the need for additional funding. All work will be done on the time and discretion of the teammates. This research was supported and approved by our great professor.

**Introduction**

I'm trying to create a program that scores words as they are typed. The score will be accumulated by the number of letters in a game-like format.

**Brief project & system description**

I will give some word examples and if you are filled in without typos, the score will be recorded. After the specified number of words and no typos, the score will be accumulated.

**Explanation and background**

To be honest, I had no idea what I wanted to do. At some point, I realized that a simple game would be a good idea, so I started the project. I haven't come up with anything concrete yet, but I think it will fulfill the requirements of the assignment.

**Project Discussion**

There are many uses for this project. It can be used to build a system for accumulating points and looping through familiar games.

**Problem Statement (what will the project solve)**

I'll use Lc3 to write my machine code.

**Perceived Needs (why is this a good project)**

First, it meets the requirements of the assignment. And secondly, such a system is a fundamental system used in various applications. Writing down letters, checking for equality, and giving a score involves a variety of mathematical operations.

**Applications (when and how can the project result be used)**

The output is used in its complete form at the time of the final submission. May be used in mathematical programs.

**Project Goals (project accomplishment desired)**

1. There is a space for the user to write words.
2. The word presented will change each time.
3. Scores are accumulated until they are incorrect.
4. You want to write at least 200 lines of code.

**Deployment Strategy (How will users get and use program)**

1. Download or run the program.
2. Use as directed by the application.

**Constraints (conditions bearing on the project operation)**

1. The word presented cannot be longer than 8 characters.
2. The word is case sensitive. All words will be in lowercase letters.
3. The words presented may be limited.

**Time line**

1. 3.4 complete proposal
2. 3.25 design
3. Mid of April complete project
4. End of semester submit project

**Introduction**

This project is a typing practice game using assembly language. You just have to guess the word.

**Description**

The user enters a word of 10 characters or less. They can then use each letter to build the word.

**Program Architecture**

Inside the loop, the program stores each character in a string called "word" and displays an asterisk on the console to mask the character. The program also stores a blank underscore in another string called "guess" to represent the hidden characters. The loop continues until the user enters a new line character. After the user has inputted a word, the second part of the program begins. It starts by displaying a second prompt and initializing the used and guess strings to null. Then, it displays the length of the word and the current state of the guess string on the console.

The program then enters a loop that reads characters entered by the user and checks if they are present in the word string. If the character is present, it updates the guess string with the character in the correct position. If the character is not present, the program displays a message indicating that the character is not in the word. The loop continues until the user has correctly guessed the entire word or has made too many incorrect guesses.

The program uses several registers to keep track of the state of the game. R0 is used for input/output, R1 is a temporary variable, R2 holds the address of the word string, R3 holds the address of the guess string, R4 holds the length of the word, and R5 holds the address of the used string.

**User Interface**

The user interface of this typing game is intuitive. The program uses a loop to check for matches between you and the words you provide. When you type a word and press Enter, the program checks your input and outputs a match/mismatch. A mismatch resets the accumulated number of correct answers. The match score is incremented.

**Procedural Design**

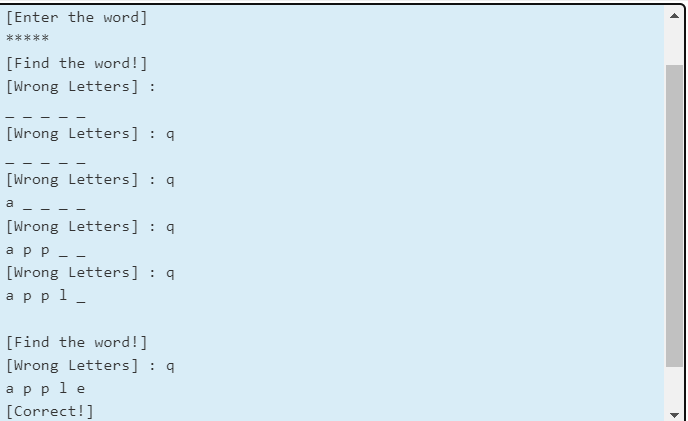
1. Display the first prompt by loading the message address to R0 and calling PUTS.
2. Load the address of the word and guess to R2 and R3, respectively.
3. Load the character '\_' into R4 for later use.
4. Set the word counter, R5, to 0.
5. Read input characters until a new line is entered:
   1. Read the input characters
   2. Check and store the input character

**Process Flow Diagrams**

Diagram

Description automatically generated

**INPUT/OUTPUT SCREEN SHOT**

****