Class: CIS11 (22953)
Name: Bowie Lam
Student Id: 2617067

CIS11 Course Project Part 1: Documenting the Project

Fill in the following areas (purple).

Introduction

1.1 Purpose

The objective of this project is to develop a program in LC-3 assembly language that computes and presents statistical analyses of a dataset comprising five examination scores. The analyses will encompass calculating the minimum score, the maximum score, the arithmetic mean/average of the scores, and the conversion of the mean score into its corresponding letter grade.

This aims to demonstrate the application of assembly language in processing and interpreting educational data, thereby providing insights into the performance metrics of a given student cohort..

1.2 Intended Audience and Users

The primary audience for this program is students and instructors who need a quick and efficient way to analyze test results. The users will be individuals familiar with using LC-3's console as interface to input data and view output.

1.3 Product Scope

The program is designed to be a straightforward yet powerful tool for educational environments, focusing on calculating and displaying test score statistics and letter grades using assembly language.

1.4 Reference

Source Documents for the Program Requirements and Specification

Option B: Test Score Calculator of provided "CIS11 Course Project Part 1 FINAL-1-1.docx" document.

Input and output are as seen below:

Input	Output

Test Score 1 Test Score 2 Test Score 3 Test Score 4 Test Score 5	Display maximum, minimum, average scores and letter grade equivalence. e.g. for equivalencies: 0 - 50 = F, 60 - 69 = D, 70 - 79 = C, 80 - 89 = B, 90 - 100 = A

2. Overall Description

2.1 Product Perspective

Primary program objectives and goals:

- Efficiently collect five test scores from the user.
- Calculate and display the minimum, maximum, and average scores.
- Determine and display the letter grade corresponding to the average score

Data type:

- Demonstrate the use of basic assembly language constructs such as loops, conditionals, subroutines, and stack operations.
- The program will handle integer data types for test scores and their derived values (minimum, maximum, average).

o 16-bit integers (for scores) and ASCII characters (for letter grades).

2.2Product Functions

The overall description of functionality:

Input Handling	Prompting and reading five test scores from the user. e.g. Test Score 1 -> Test Score 2 -> Test Score 3 -> Test Score 4 -> Test Score 5 or separate by space or enter
Data Processing	Calculating the minimum score. Calculating the maximum score. Calculating the average score. Determining the letter grade for each score.
Output	Displaying the results to the console.

Technical functionality

• Input Handling: Responsible for acquiring input data from the user in a structured and efficient manner. • Arithmetic Calculations (Min, Max, Average): Performs mathematical operations, such as finding the minimum, maximum, and average values of a set of numbers. • Letter Grade Determination: Converts numeric scores into corresponding letter grades based on a pre-defined grading system. • Output Display: Formats and presents the results of the program's calculations in a clear and user-friendly manner within LC-3 console.	Subroutines: • GET_INPUT • CALCULATE_MIN_MAX • CALCULATE_AVERAGE • DETERMINE_GRADE • DISPLAY_RESULTS Operations: • Arithmetic operations (addition, division) • Comparison operations (finding minimum and maximum) • Data movement (loading, storing values) • Branching (conditional and iterative) • Stack operations (PUSH, POP) • ASCII conversion

2.3 User Classes and Characteristics

Students: Primarily interested in seeing their results and understanding their performance.

Instructors: Use the program to quickly assess class performance and identify areas where students may need additional support.

2.4 Operating Environment

The application will run on an LC-3 simulator or any assembly .obj viewer.
No specific operating system requirements. As long as you can load up the .obj file in assembler, you're good to go.

2.5 Design and Implementation Constraints

- Limited to the LC-3 instruction set and its capabilities.
- Must handle potential overflow issues during calculations.

In addition, in my test runs I noticed test scores occupy 6-bits in my current implementation which does incur errors in offsetting 5-bit in instructions like ADD

2.6 Assumptions and Dependencies

The user will input valid integers, range of 5 test scores between 0 and 100.

The LC-3 simulator is correctly installed and functioning.

Goal is to run on a LC-3 console. I'd love to challenge myself and see if I can generate a web viewable file (i.e. .html ?!?) to run this on web browser.

3. External Interface Requirements

3.1 User Interfaces

- The user will interact with the program through the console.
- Prompts will guide the user to input scores.
- Results will be displayed directly on the console once all inputs are allocated

3.2 Hardware Interfaces

- Monitor to display LC-3 console.
- Given operating system utilized, a mouse might be needed to hover and activate console window.
- A keyboard to interact with console

3.3 Software Interfaces

Ideally, the LC-3 console would be the preferred option. Nevertheless, .obj web viewers are also acceptable. Using the x3000 initial memory location eliminates the need for adjustments, making it a more convenient plug-and-play solution

3.4 Communications Interface

The program does not require any web, internet, or network connectivity if the LC-3 console is used.

However, for devices that are not compatible with the LC-3 console or do not support running the LC-3's *.obj file, it would be necessary to have network connectivity to utilize local or off-device machines or to utilize the internet to access the provided online simulators used throughout the course.

4. Detailed Description of Functional requirements

4.1 Type of Requirement (summarize from Section 2.2)

Functions	Input Handling: • Prompt the user to enter five test scores. Minimum Calculation: • Identify the lowest score among the five inputs. Maximum Calculation: • Identify the highest score among the five inputs. Average Calculation: • Compute the average score by summing up all five scores and dividing by five.

	Grade Determination: • Assign grade letters based on the following criteria: o 90-100: A o 80-89: B o 70-79: C o 60-69: D o Below 60: F Output Results: • Display the following information: o The five test scores entered by the user. o The minimum score. o The maximum score. o The average score. o The letter grade for each score.
Purpose	Process and analyze test scores, providing summary statistics and letter grades.
Inputs	Five test scores entered by the user. Separated by enter or space
Output	Minimum, maximum, average scores, and corresponding letter grades
Data	Scores and computed values stored internally in memory

4.2 Performance requirements

The program needs to be efficient and provide real-time feedback. It should process input

scores, display the results, and handle all operations quickly without any noticeable delays.

i.e. No delays and no waiting.

4.3 Flow Chart OR Pseudocode.

```
.ORIG x3000 ; Default memory
; Labels and memory allocation
START ; Program start
ARRAY .BLKW 5 ; Array to store the five test scores
MIN_SCORE .FILL x7FFF ; Initialize to maximum possible value
MAX_SCORE .FILL x8000 ; Initialize to minimum possible value
AVG_SCORE .BLKW 1
; Main Program
```

```
MAIN
JSR READ_SCORES ; Read scores from the user
JSR CALC_MIN ; Calculate minimum score
JSR CALC_MAX ; Calculate maximum score
JSR CALC_AVG ; Calculate average score
JSR DISPLAY_RESULTS ; Display results and letter grades
HALT
; Subroutines
READ_SCORES
; Code to read 5 scores into ARRAY
RET
CALC_MIN
; Code to find the minimum score in ARRAY
RET
CALC_MAX
; Code to find the maximum score in ARRAY
RET
CALC_AVG
; Code to calculate the average score
RET
DISPLAY_RESULTS
; Code to display minimum, maximum, average, and letter grades
RET
; Utility Subroutines (e.g., for ASCII conversion)
CONVERT_TO_ASCII
; Code for ASCII conversion
RET
.END
```