CIS11 Course Project Part 1: Documenting the Project

Fill in the following areas (purple).

Introduction

1.1 Purpose

The purpose of this program is to automatically perform basic processing and analysis of students' grades. The intention is to improve the workflow of schools by automating more tedious and time consuming tasks.

1.2 Intended Audience and Users

The primary audience is our CIS-11 professor Dr. Kasey Nguyen. The program would be intended for teachers or students who are calculating their final grade by inputting five test scores.

1.3 Product Scope

The project is going to be kept fairly simple, with room for improvement. For now, it will provide basic mathematical functions specific for grading, while also setting data structures that could be used for more complex programs.

1.4 Reference

Source Documents for the Program Requirements and Specification

CIS 11 Course Project Part 1 FINAL.docx"

- **Option B: Test Score Calculator**

1) Contain appropriate addresses: origination, fill, array, input and output.
2) Display minimum, max, average values/grades in the console.
3) Use appropriate labels and comments.
4) Contain appropriate instructions for arithmetic, data movement and conditional operations.
5) Compromise 2 or more subroutines and implement subroutine calls.
6) Use branching for control: conditional and iterative.
7) Manage overflow and storage allocation.
8) Manage stack: include PUSH-POP operation on stack.
9) Include save-restore operations.
10) Include a pointer.
11) Implement ASCII conversion operations.
12) Use appropriate system call directives.

Companion Application Requirements Documents (If applicable)

1) Project Documentation
2) E-portfolio
3) LC-3 Editor and simulator

2. Overall Description

2.1 Product Perspective

Primary program objectives

- Compute minimum, maximum, and averages for grades.
- Take inputs for individual assignments, and display results to the user.
- Hold grades in a dynamic data collection, (potentially more than 5)

2.2 Product Functions

The overall description of functionality:

- The program will have to take in ASCII inputs from a keyboard. It then must process this input into data that can do arithmetic.
- The program must be able to store the assignments in a stack so they can be used to do further calculations.
- The program must take grades from stack, then be able to perform comparisons between them to find the minimum and maximum.
- Perform addition and division to compute the average.
- Convert data to ASCII symbols to display on screen.

Technical functionality

- Multiplication
- Division
- Find minimum, find maximum (branch operators)
- Compute average (looping addition, division and multiplication, branching for rounding)
- Grade stack management (Push, Pop)
- ASCII conversion
- Digit stack management (Push, Pop)
- Error messages
- Menus (loops, branching)
  Use of LC-3 editor and simulator.

2.3 User Classes and Characteristics

Jason Atalig: Responsible for designing and coding mathematical computations

Nadja Hernandez: Responsible for coding the stack to store and give access to grades.

Luis Rodriguez: Responsible for creating a menu for the user to input the scores.

2.4 Operating Environment

The program is being developed in LC3. It is intended to be used on a Windows 10 operating system using the LC3 simulator.

2.5 Design and Implementation Constraints

Access to the LC-3 application. The LC-3 application can only run on windows so someone on an apple computer may not be able to access it. It will be easy to enter your test scores and get the information they want.

Program constraints include the inability to input decimal value grades. The amount of grades that can be analyzed is limited to the size of the stack. Depending on time constraints, the console display may not be robust.

LC3 simulator also limits how we can enter input, making the user experience somewhat tedious and confusing.

2.6 Assumptions and Dependencies

It is assumed that the user knows how to operate a mouse and keyboard and the LC3 simulator program. They must understand the limitation of how many grades the program can hold at once and have the grades prepared to input.

Since the program runs on LC3, it is assumed the user knows how to input the numbers on the console.

3. External Interface Requirements

3.1 User Interfaces

The member will access the program using a console, which is provided by the LC3 simulator. The user will access a menu and enter key inputs to navigate.

3.2 Hardware Interfaces

The code is fairly light by modern standards. Any machine that can run the LC3 simulator, has a screen, and can take key input will do fine.

3.3 Software Interfaces

The application will require windows and the LC-3 editor and simulator.

3.4 Communications Interface

Since the application runs on a program you download onto your computer, the user does not need to have access to the web, internet, or have network connectivity.

4. Detailed Description of Functional requirements

4.1 Type of Requirement (summarize from Section 2.2)

Menu Display:

Purpose: Tells user to input grades and display results.

Processing: Take ASCII inputs and convert them to integer values.

Input: Keyboard input

Output: Minimum Grade, Maximum Grade, Average Grade, Error messages

Data: Assignment Grades, stored in stack

Digit Stack:

Purpose: Organize user input to perform ascii conversion

Processing: Take ASCII input to decimal value, or vice versa

Input: Keyboard

Output: ASCII or integer, depending on input

Data: digits

Minimum and Maximum Finder:

Purpose: Find the assignment with highest grade, and the assignment with the lowest grade

Processing: Access grades from stack, and using branching to compare values

Input: N/A, data from program

Output: Minimum and Maximums, stored in unique memory location

Stack Requirement

Purpose: Stores all the test scores.
Inputs: Inputs are taken from the menu function and stored in the stack.
Processing: The input will be stored in the stack to be accessed by other functions.
Outputs: Assignment Grades
Data: Test scores

Average Finder

Purpose: Will calculate the average of the test scores inputted.
Inputs: Inputs are taken from the stack.
Processing: The assignment grades will be accessed from stack. The grades will then be added together in a loop and stored in an accumulator. The accumulator will then go under integer division by the number of grades. To get a decimal value (2 significant figures), the remainder will be multiplied by 100 and divided again by the number of grades, then branching will be used to determine when to round.
Outputs: The average grade will be outputted.
Data: individual grades (stored in stack), average grade (unique address)

4.2 Performance requirements

The program should be able to perform calculations and display input fairly quickly so that it feels responsive to users. Error handling should also be implemented to make sure the user understands why their input is incorrect.The functionality of the stack and mathematical computations should be scalable to allow for upgrading the amount of grades we can analyze at once.