# *CIS-11 Test Score Calculator*

**Team: JTTP**
**Members: Isaiah Lopez**
**Nicolas Aborita**

**Project Name: Test Score Calculator Program**
**Date: 06-12-2025**

**Advisor: Kasey Nguyen, PhD**

## Part I – Application Overview

### Objectives

Create an LC-3 program to calculate test scores based on user-inputted scores.  The user will input 5 test scores and the program will calculate a minimum, maximum, average, and letter equivalence of each score and output the results thereafter.

#### Why are we doing this?

- **The purpose of the project is to allow ease of access for users trying to see certain statistics of their grades.**
- **Using the program allows users to simply input scores as numbers and see a letter grade equivalence that many are used to rather than calculating by hand. This allows for better efficiency when calculating the letter grade, average score, minimum score and maximum scores of the students in the class.**
- **The test score calculator will benefit users from teachers, students, and parents. Users will be able to compare the students test scores with others in the class with the calculated average, minimum, and maximum scores of the class. The letter grade also helps determine how well the students did in equivalence to their test scores.**

### Business Process

Before using the given application, users would have to manually input letter grades to each individual test score. To calculate the average, minimum, and maximum, users would also gather all the test scores and calculate them manually too.

Using the given application, grades are gathered via inputted scores based on a maximum score. The inputted numbers will be gathered from users and then displayed to outside sources for other users to see in a grade equivalent letter that makes it easier for people like students and parents to understand. A = Good, F = Bad. The outside source will also display the average, minimum, and maximum of the scores inputted. This creates more efficiency in allowing the user to calculate the letter grade, average, minimum, and maximum for the inputted test scores.

### User Roles and Responsibilities

Different types of users can use this application from teachers, students, and parents. The user in general would be inputting the test scores into the program for other parents, teachers, and students to see. The user will input a score from 0-100 only.

### Production Rollout Considerations

At this time, the only consideration is made for a single user with very limited data to be inputted and outputted. No large-scale considerations are needed to be made.

### Terminology

**Stack** - Data structure to hold data in a last-in-first-out manner.

## Part II – Functional Requirements

The application will prompt the user to input 5 valid test score numbers into the display terminal. Each score is put into the stack to calculate the minimum, maximum, and average of the test scores. Once the calculation is done each of the scores will be outputted along with the maximum, minimum, and average in respective order.

## Statement of Functionality

The program is to prompt the user to input a number 0-100. If the user inputs an incorrect character/number, the program prompts the user of their mistake and asks them to input the number correctly until they do input a number correctly. Once a number is input valid, the program stores the number into the stack. After 5 numbers are stored in the stack, the program calculates the maximum, minimum, average, and letter equivalent grades to all the scores and statistics listed. This information is stored and the stack data is deleted. Then the stored data is placed in the stack for output. This will be in the order of average, minimum, maximum, and inputted scores. Since the data is in the stack, the order of outputs will be scores, maximum, minimum, and average alongside their letter grade equivalence next to them. Once all data in the stack is displayed the program will end.

### Scope

The user will first input 5 valid test scores into the program. The program then has to validate each of the scores to see if they are numbers. Using the test scores, the program will calculate the minimum, maximum, and the average of the scores given. After that, with each test score, there will also be a corresponding letter grade.

### Performance

Due to little data being used, program calculation and functionality should be nearly instant.

### Usability

The user will input the test scores to be calculated on a display screen one at a time so that the scores can be saved onto the stack. The program will display all of its following functions after 5 total test scores have been inputted.

The user may input a wrong character in the program. If so, the program should correct the user and allow for another chance to input a correct character.

## Documenting Requests for Enhancements

| Date | Enhancement | Requested by | Notes | Priority | Release No/ Status |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |

# Part III – Appendices

### Flow chart or pseudo-code.

```
Load Display Prompt to R0
Display Prompt at R0 with PUTS - "Enter 5 test scores (numbered 0-100): "
*This portion of the program loops 5 times
        Use register to get the inputted score
        Use the Validate subroutine to check for correct input
                If validated return value
                Else, prompt the user "Invalid Input. Try again." and attempt input again
Push input onto the stack
*This portion uses the pointer at the top of the stack for calculation
Add input to the average variable and save the average to the top of the stack
      If it is the first user input, set the minimum and maximum variables to said input and
      save the minimum and maximum to the top of the stack
      Else, compare new input to maximum and minimum
*After 5 inputs loop ends

*This portion starts at the top of the stack until the stack is empty
Get the value at the pointer location (starting from the top)
Use the value in the grade-letter subroutine to convert
Display score and grade letter to output
```

**Output Example**

```
Scores:
 75
 C

 100
 A

 9
 F

82
B

68
D

Maximum: 100
Minimum: 9
Average: 66
```

Once displayed program will end.

**Subroutines**

Various subroutines will be used to obtain some information such as the average as well as some being used for conversion. These are subroutines that will be used once or more times in the program:

DIVIDE:

Obtain dividend from R2 and divisor from R3. Copy dividend to R4, check if it can be subtracted from by divisor (R3). If so then proceed to with subtraction and add one to the quotient (R4), else return function and R2 returns the remainder.

CONVERT:

Obtain value from a register and copy onto another. Subtract copied register by number related to grade. For example, if there is some value and we subtract 90 if the value returned is positive or zero then the value should correlate to an A. If so then store the ASCII character "A" onto a register and put onto the stack.

INPUT

Will display a prompt to user to request a number 0-100 to be entered. User is to input said value via their keyboard and the value will then be stored into a register to be stored into memory or the stack. Once stored and input validated, the user will be asked once more to input a value until there are 5 values stored.

*These subroutines are subject to change based on the needs of the program and more subroutines may be added.