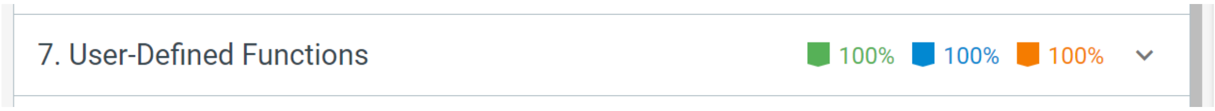
Assignment xx Algorithmic Design Document

Make a copy before you begin (File -> Make a copy). Add the Assignment # above and complete the sections below BEFORE you begin to code and submit with your Assignment to D2L (File -> Download -> PDF). The sections will expand as you type.

zyBooks

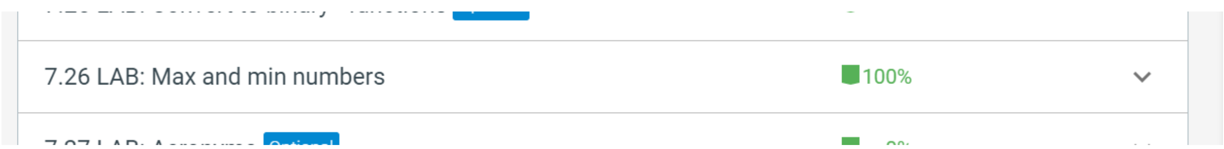
Add your zyBooks screenshots for the % and assigned zyLabs completions below. Required percentages: all assigned zyLabs, Challenge Activity with at least 70%, and Participation Activity with at least 80%.

zyLabs, Challenge, and Participation % Screenshot:



7. User-Defined Functions 🟩 100% 🟦 100% 🟧 100%

Assigned zyLabs completion Screenshot:



7.26 LAB: Max and min numbers 🟩100%

Assignment

Program description:

This program will take a list of numbers that you enter, and normalize them based on the largest number

Before you begin coding, **you must first plan out the logic** and think about what data you will use to test your program for correctness. All programmers plan before coding - this saves a lot of time and frustration! Use the steps below to identify the inputs and outputs, calculations, and steps needed to solve the problem.

Algorithmic design:

a. Identify all of the user input. What are the data types of the inputs? Define the input variables.

Integer numVals //Gets the total array size to check
Integer userInput //Repeatedly called to add data to array

b. Describe the program output. What is displayed to the user? What are the data types of the output? Define the output variables.

Basic text outputs:
Welcome!
Instruction to enter numVals
Instruction to enter userInput
List of all numbers after normalization (“5 0 9 2 etc…”)
Goodbye!

Error handling text (on trigger):
Error: Input can not be negative.
Error: Error: Number cannot be negative, zero, or greater than 20

c. What calculations do you need to do to transform inputs into outputs? List all formulas needed, if applicable. If there are no calculations needed, state there are no calculations for this algorithm.

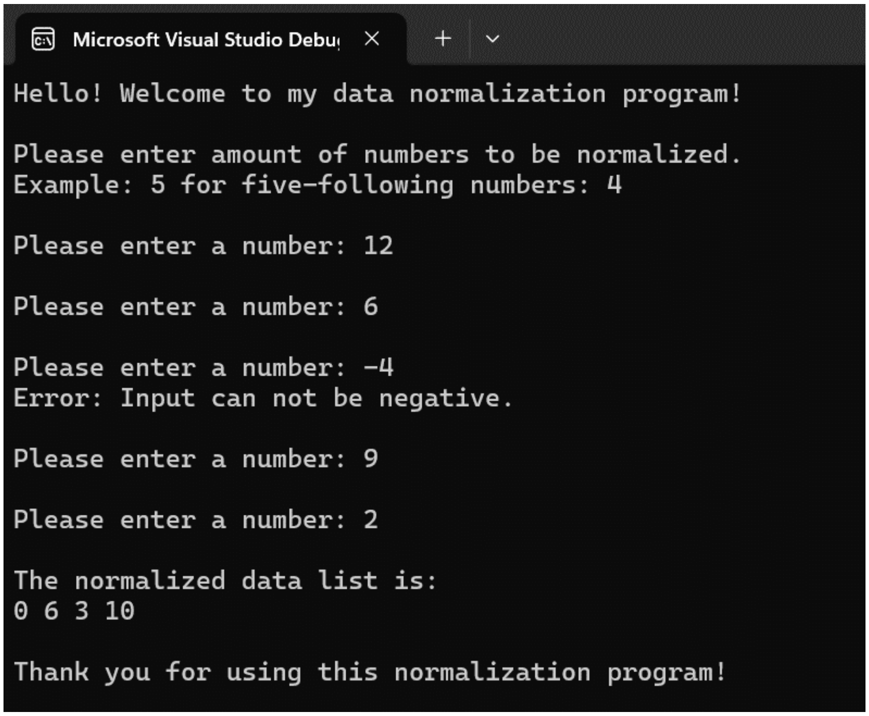
Basic arithmetic:
valueMax – userNumber //normalization calculation

d. Design the logic of your program using pseudocode or flowcharts. See pseudocode syntax at the bottom of this document. Here is where you would use conditionals, loops, functions or array constructs (if applicable) and list the steps in transforming inputs into outputs. Walk through your logic steps with the test data from the assignment document.

```
START

FUNCTION GetUserInput(Int numVals)
- - DECLARE Int i
- -
- - FOR i = 0 TO i == numVals
- - - - DISPLAY Enter number {i}
- - - - INPUT userValues[i]
- -
- - RETURN userValues


FUNCTION Int GetMaxInt(Int listInts[], Int numVals)
- - DECLARE Int valueMax
- - DECLARE Int i
- -
- - SET valueMax = listInts[0]
- - FOR i = 1 TO i <= listInts[0]
- - - - IF listInts[i] > valueMax
- - - - - - valueMax = listInts[i]
- -
- - RETURN valueMax


FUNCTION Int MakeNormals(Int userValues[], Int valueMax, Int numVals)
- - DECLARE Int i
- - DECLARE listNormals[numVals]
- -
- - FOR i = 0 TO i == numValues
- - - - SET listNormals[i] = valueMax - userValues[i]
- -
- - RETURN listNormals


FUNCTION Int OutputData(Int listNormals, Int numVals)
- - DECLARE Int i
- -
- - DISPLAY The normalized data values are:
- - FOR i = 0 TO i == numVals
- - - - DISPLAY {listNormals[i]}
- - DISPLAY


FUNCTION Int main (void)
- - DECLARE Int userValues
- - DECLARE Int valueMax
- - DECLARE Int numVals
- - DECLARE Int listNormals[20]
- -
- - DISPLAY "Hello! Welcome to my data normalization program!"
- -
- - DO
- - - - DISPLAY Please enter amount of numbers to be normalized (ex: 5 for five-following numbers)
- - - - INPUT numVals
- - - -
- - - - IF numVals > 12 || numVals <= 0
- - - - - - DISPLAY Error: Number cannot be less than zero or greater than 20
- - - -
- - WHILE numVals > 0 AND numVals < 20
- -
- - SET userValues = GetUserInput(numValues)
- -
- - SET valueMax = GetMaxInt(userValues, numVals)
- -
- - SET listNormals = MakeNormals(userValues, valueMax, numVals)
- -
- - OutputData(listNormals, numVals)
- -
- - DISPLAY Thank you for using this normalization program!
- -
- - RETURN 0

END
```

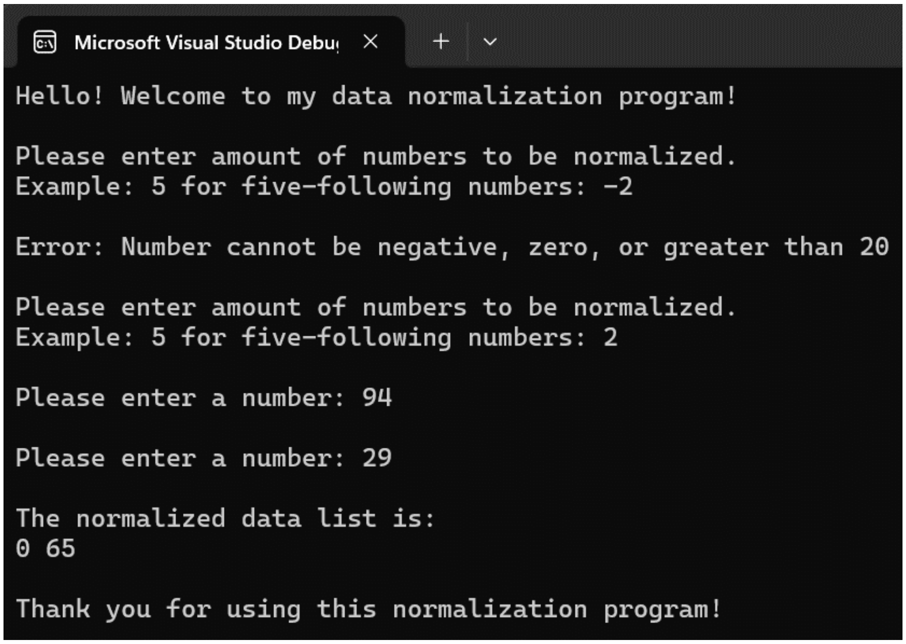
e. Include 2 Sample Program Runs for your program using your own set of data. This data set must be different from my Sample Runs in the Assignment document. This process is similar to Unit Testing and will help you test your program better.

Sample Program Run 1:



```
Hello! Welcome to my data normalization program!

Please enter amount of numbers to be normalized.
Example: 5 for five-following numbers: 4

Please enter a number: 12

Please enter a number: 6

Please enter a number: -4
Error: Input can not be negative.

Please enter a number: 9

Please enter a number: 2

The normalized data list is:
0 6 3 10

Thank you for using this normalization program!
```

Sample Program Run 2:



```
Hello! Welcome to my data normalization program!

Please enter amount of numbers to be normalized.
Example: 5 for five-following numbers: -2

Error: Number cannot be negative, zero, or greater than 20

Please enter amount of numbers to be normalized.
Example: 5 for five-following numbers: 2

Please enter a number: 94

Please enter a number: 29

The normalized data list is:
0 65

Thank you for using this normalization program!
```

Pseudocode Syntax

Think about each step in your algorithm as an action and use the verbs below:

To do this:	Use this verb:	Example:

Create a variable	DECLARE	DECLARE integer num_dogs
Print to the console window	DISPLAY	DISPLAY “Hello!”
Read input from the user into a variable	INPUT	INPUT num_dogs
Update the contents of a variable	SET	SET num_dogs = num_dogs + 1
Conditionals		

Use a single alternative conditional	IF *condition* THEN
	    statement
	    statement
	END IF
		IF num_dogs > 10 THEN
		     DISPLAY “That is a lot of dogs!”
		END IF

Use a dual alternative conditional	IF *condition* THEN
	    statement
	    statement
	ELSE
	    statement
	    statement
	END IF
		IF num_dogs > 10 THEN
		     DISPLAY “You have more than 10 dogs!”
		ELSE
		     DISPLAY “You have ten or fewer dogs!”
		END IF

Use a switch/case statement	SELECT *variable or expression*
	  CASE *value_1:*
	    statement
	    statement
	  CASE *value_2:*
	    statement
	    statement
	  CASE *value_2:*
	    statement
	    statement
	  DEFAULT:
	    statement
	    statement
	END SELECT
		SELECT num_dogs
		   CASE 0: DISPLAY “No dogs!”
		   CASE 1: DISPLAY “One dog..”
		   CASE 2: DISPLAY “Two dogs..”
		   CASE 3: DISPLAY “Three dogs..”
		   DEFAULT: DISPLAY “Lots of dogs!”
		END SELECT

Loops		

Loop while a condition is true - the loop body will execute 0 or more times.	WHILE *condition*
	  statement
	  statement
	END WHILE
		SET num_dogs = 1
		WHILE num_dogs < 10
		   DISPLAY num_dogs, “ dogs!”
		   SET num_dogs = num_dogs + 1
		END WHILE

Loop while a condition is true - the loop body will execute 1 or more times.	DO
	  statement
	  statement
	WHILE *condition*
		SET num_dogs = 1
		DO
		   DISPLAY num_dogs, “ dogs!”
		   SET num_dogs = num_dogs + 1
		WHILE num_dogs < 10

Loop a specific number of times.	FOR *counter* = *start* TO *end*
	  statement
	  statement
	END FOR
		FOR count = 1 TO 10
		   DISPLAY num_dogs, “ dogs!”
		END FOR

Functions		

Create a function	FUNCTION *return_type name (parameters)*
	   statement
	   statement
	END FUNCTION
		`FUNCTION Integer add(Integer num1, Integer num2)`
		`   DECLARE Integer sum`
		`   SET sum = num1 + num2`
		`   RETURN sum`
		`END FUNCTION`

Call a function	CALL *function_name*	`CALL add(2, 3)`
Return data from a function	RETURN *value*	`RETURN 2 + 3`