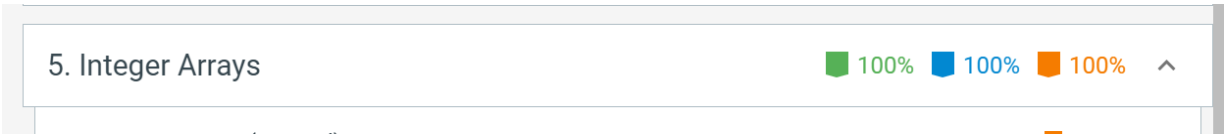
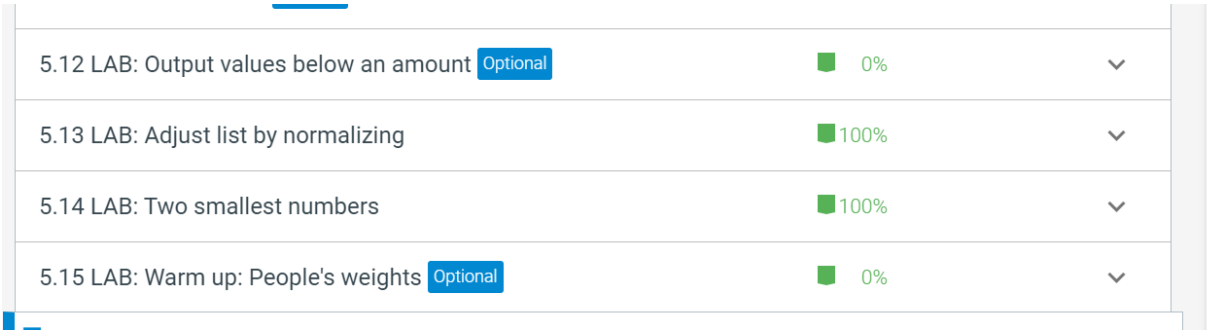
Assignment xx Algorithmic Design Document

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:


Assigned zyLabs completion Screenshot:


Assignment

Program description:
This program will allow you to build a stock tracking portfolio with up to 100 entries. This will record, update, delete, and display data with our without a filter

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.

INPUT String userInputStr – Used for character based menu navigation
INPUT Integer userInputInt – used for simple num based selection
INPUT Double userInputDub – used for decimal balues

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

OUTPUT Strings
All data output is in a text format with variables integrated. The output is uniformized in how it displays to ensure that program behavior isn't ever surprising to the end user.

No data is outputted to a file, but data is stored in a manner that allows for future save operations if needed.

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.

No complex calculations are performed. Here is the primary simple operations:

List inputs: IS input >= 0 AND < 100
List inputs: IS input == 100

Price filter: IS list[index] >= input

Delete Data: SET stockID[index] = 0
Delete Data: SET stockName[index] = 0
Delete Data: SET stockPriceNow[index] = 0
Delete Data: SET stockPriceStart[index] = 0

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

DECLARE Double userInputDub = 0.0
DECLARE Integer userInputInt = 0
DECLARE String userInputStr = ""
DECLARE Integer stockID[99]
DECLARE Integer stockName[99]
DECLARE Double stockPriceNow[99]
DECLARE Double stockPriceStart[99]
DECLARE Integer parseID = 0
DECLARE Double parsePriceNow = 0.0
DECLARE Double parsePriceStart = 0.0
DECLARE Integer keepGoing = 0
DECLARE Integer didRun = 0

```
DECLARE Integer i = 0


- - DISPLAY "Welcome to this stock tracking program!"

WHILE userInputStr[255] != "q"
- - SET keepGoing = 0
- - SET userInputDub = 0
- - SET didRun = 0

- - DISPLAY "Main menu"
- - DISPLAY "Please select one of the options below by entering the option name or first
letter"

- - DISPLAY "Create - Create new stock data for any ID"
- - DISPLAY "Delete - Remove stock data for a given ID"
- - DISPLAY "Update - Update a currently existing stock price"
- - DISPLAY "Price Filter - Display all stock data above a given price"
- - DISPLAY "Show All - Display all recorded stock data"
- - DISPLAY "Quit - Terminate the program"
- -
- - DISPLAY "Please Enter selection: "
- - INPUT userInputStr
- -
- - IF userInputStr[0] == "c" OR userInputStr[0] == "C" AND didRun != 1
- - - - SET didRun = 1
- - - - DISPLAY "You have entered the Stock Creation menu."
- - - -
- - - - DISPLAY "Please enter an ID greater-than or equal to 0 and less than 100"
- - - - INPUT userInputInt
- - - -
- - - - IF userInputInt >= 0 and < 100
- - - - - - IF stockID[userInputInt] != 1 AND keepGoing != 1
- - - - - - - - SET parseID = userInputInt
- - - - - - - - DISPLAY "Selected ID: {userInputInt}"
- - - - - - - -
- - - - - - - - IF userInputStr != ""
- - - - - - - - - - SET parseName = userInputStr
- - - - - - - - - - DISPLAY "Chosen name {parseName}"
- - - - - - - - - -
- - - - - - - - - - DISPLAY "Please enter the Stock price"
- - - - - - - - - - INPUT userIntDub
- - - - - - - - - -
- - - - - - - - - - IF userIntDub >= 0
- - - - - - - - - - - - SET parsePriceNow = userIntDub
- - - - - - - - - - - - SET parsePriceStart = userIntDub
- - - - - - - - - - - - DISPLAY "Stock price successfully set to {userIntDub}"
- - - - - - - - - - - -
```

```
- - - - - - - - - - - - DISPLAY "Created data: "
- - - - - - - - - - - - DISPLAY DISPLAY "Stock ID: {parseID} - - Current price: {parsePriceNow}"
- - - - - - - - - - - - SET stockID[parseID] = 1
- - - - - - - - - - - - SET stockName[parseID] = parseName
- - - - - - - - - - - - SET stockPriceNow[parseID] = parsePriceNow
- - - - - - - - - - - - SET stockPriceStart[parseID] = parsePriceNow
- - - - - - - - - - -
- - - - - - - - ELSE
- - - - - - - - - - DISPLAY "Input can not be less than zero
- - - - - - - - - - SET keepGoing = 1
- - - - - -
- - - - - - ELSE
- - - - - - - - DISPLAY "Stock name can not be blank"
- - - - - - - - SET keepGoing = 1
- - - - - -
- - - - ELSE
- - - - - - DISPLAY "Stock data already exist at this ID. Please remove or edit this data."
- - - - - - SET keepGoing = 1

- - IF userInputStr[0] == "u" OR userInputStr[0] == "U" AND didRun != 1
- - - - SET didRun = 1
- - - - DISPLAY "You have entered the Stock Update menu."
- - - - DISPLAY "Please enter the Stock ID you would like to update: "
- - - - INPUT userInputInt
- - - -
- - - - IF userInputInt <= 0 AND > 100
- - - - - - SET parseID = userInputInt
- - - - - - SET parseName = stockName[userInputInt]
- - - - - - SET parsePriceNow = stockPriceNow[userInputInt
- - - - - - SET parsePriceStart = stockPriceStart[userInputInt]
- - - - - -
- - - - - - IF parseName != ""
- - - - - - - - DISPLAY "Stock ID: {parseID} - - Stock Name: {parseName} - - Current price:
{parsePriceNow} - - Starting price: {parsePriceStart}"
- - - - - - - - DISPLAY "Please enter the new stock price."
- - - - - - - - INPUT userInputDub
- - - - - - - -
- - - - - - - -
- - - - - - - - IF userInputDub >= 0
- - - - - - - - - - SET stockPriceNow[indexPoint] = userInputDub
- - - - - - - - - - DISPLAY "Stock ID: {parseID} new price set as {userInputDub}"
- - - - - - - -
- - - - - - ELSE
- - - - - - DISPLAY "No data for Stock ID {i}. Please use creation menu or choose a new ID"
- - - -
- - - - ELSE
- - - - - - DISPLAY "Input must be a num between 0 and 100"
```

```
- - IF userInputStr[0] == "d" OR userInputStr[0] == "D" AND didRun != 1
- - - - SET didRun = 1
- - - - DISPLAY "You have entered the Stock removal menu."
- - - -
- - - - DISPLAY "Please enter an ID greater-than or equal to 0 and less than 100"
- - - - INPUT userInputInt
- - - -
- - - - IF userInputInt >= 0 AND userInputInt < 100 AND keepGoing != 0
- - - - - - IF stockID[userInputInt] = 1
- - - - - - - - SET parseID = userInputInt
- - - - - - - - DISPLAY "Stock data found."
- - - - - - - - DISPLAY "Stock ID: {parseID} - - Stock Name: {parseName} - - Current price:
{parsePriceNow} - - Starting price: {parsePriceStart}"
- - - - - - - - DISPLAY "Type Yes to delete this data, or No to keep this data"
- - - - - - - - INPUT userInputStr
- - - - - - - - IF userInputStr[0] == "y" OR userInputStr[0] == "Y"
- - - - - - - - - - SET stockID[userInputInt] = 0
- - - - - - - - - - SET stockName[userInputInt] = ""
- - - - - - - - - - SET stockPriceNow[userInputInt] = 0
- - - - - - - - - - SET stockPriceStart[userInputInt] = 0
- - - - - - - - - - DISPLAY "All data deleted for this stock ID"
- - - - - - ELSE
- - - - - - - - DISPLAY "No stock data recorded for this ID"


- - IF userInputStr[0] == "p" OR userInputStr[0] == "P" AND didRun != 1
- - - - DISPLAY "You have entered the price filter menu."
- - - - DISPLAY "Please enter the minimum price filter: "
- - - - INPUT userInputDub
- - - - FOR i = 0 TO 99
- - - - - - IF stockID[i] == 1
- - - - - - - - SET didRun = 1
- - - - - - - - IF stockPriceNow[i] >= userInputDub
- - - - - - - - - - DISPLAY "Stock ID: {stockID[i]} - - Stock Name: {stockName[i]} - - Current price:
{stockPriceNow[i]} - - Starting price: {stockPriceStart[i]}"
- - - - IF didRun == 1
- - - - - - DISPLAY "Data parse complete."
- - - - ELSE
- - - - - - DISPLAY "No stock data found at or above filter price of {userInputDub}. Please run
again."
- - - - SET didRun = 1

- - IF userInputStr[0] == "d" OR userInputStr[0] == "D" AND didRun != 1
- - - - FOR i = 0 TO 99
- - - - - - IF stockID[i] == 1
- - - - - - - - SET didRun = 1
- - - - - - - - DISPLAY "Stock ID: {stockID[i]} - - Stock Name: {stockName[i]} - - Current price:
{stockPriceNow[i]} - - Starting price: {stockPriceStart[i]}"
```

```
- - - - IF didRun == 1
- - - - - - DISPLAY "Data parse complete."
- - - - ELSE
- - - - - - DISPLAY "No stock data found. Please create data then run again."
- - - - SET didRun = 1

- - IF userInputStr[0] == "q" OR userInputStr[0] == "Q" AND didRun != 1
- - - - SET userInputStr[0] = ""
- - - - DISPLAY "Please retype Quit to confirm."
- - - - INPUT userInputStr
- - - - IF userInputStr[0] == "q" OR userInputStr[0] == "Q"
- - - - - - userInputStr[255] = "q"
- - - - ELSE
- - - - - - SET userInputStr[0] = ""
- - - - - - DISPLAY "Aborted program end sequence."


- - IF keepGoing != 0
- - - - DISPLAY "Changes successfully discarded. Returning to the main menu."

- - IF didRun == 0
- - - - DISPLAY "{userInputStr} is an invalid option. Returning to the main menu."

DISPLAY Program is now ending. Thank you for using my tracking tool!
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.


Microsoft Visual Studio Debu
Main menu
Please select one of the options below by entering the option name or first letter
Create - Create new stock data for any ID
Delete - Remove stock data for a given ID
Update - Update a currently existing stock price
PriceFilter - Display all stock data above a given price
ShowAll - Display all recorded stock data
Quit - Terminate the program
Please enter selection: create
You have entered the Stock Creation menu.
Please enter an ID greater-than or equal to 0 and less than 100: 2
Selected ID: 2
Please enter the stock price: 55
Stock price successfully set to $55.00
Created data:
Stock Name: 2 -- Current Price: 55.00
- -
Main menu
Please select one of the options below by entering the option name or first letter
Create - Create new stock data for any ID
Delete - Remove stock data for a given ID
Update - Update a currently existing stock price
PriceFilter - Display all stock data above a given price
ShowAll - Display all recorded stock data
Quit - Terminate the program
Please enter selection: delete
You have entered the Stock deletion menu.
Please enter the 2 digit Stock ID you would like to delete
or enter 100 to leave this menu: 02
All data for Stock 2 successfully deleted.
- -
Main menu
Please select one of the options below by entering the option name or first letter
Create - Create new stock data for any ID
Delete - Remove stock data for a given ID
Update - Update a currently existing stock price
PriceFilter - Display all stock data above a given price
ShowAll - Display all recorded stock data
Quit - Terminate the program
Please enter selection: Quit
Please retype Quit to confirm: q
Program is now ending. Thank you for using my tracking tool!
D:\Dropbox\PCC\Winter 2024\CS-133U-11053 - C\Assignment05\x64\Debug\Assignment05.exe (process 14012) ex
ited with code 0.

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`