Assignment xx Algorithmic Design Document

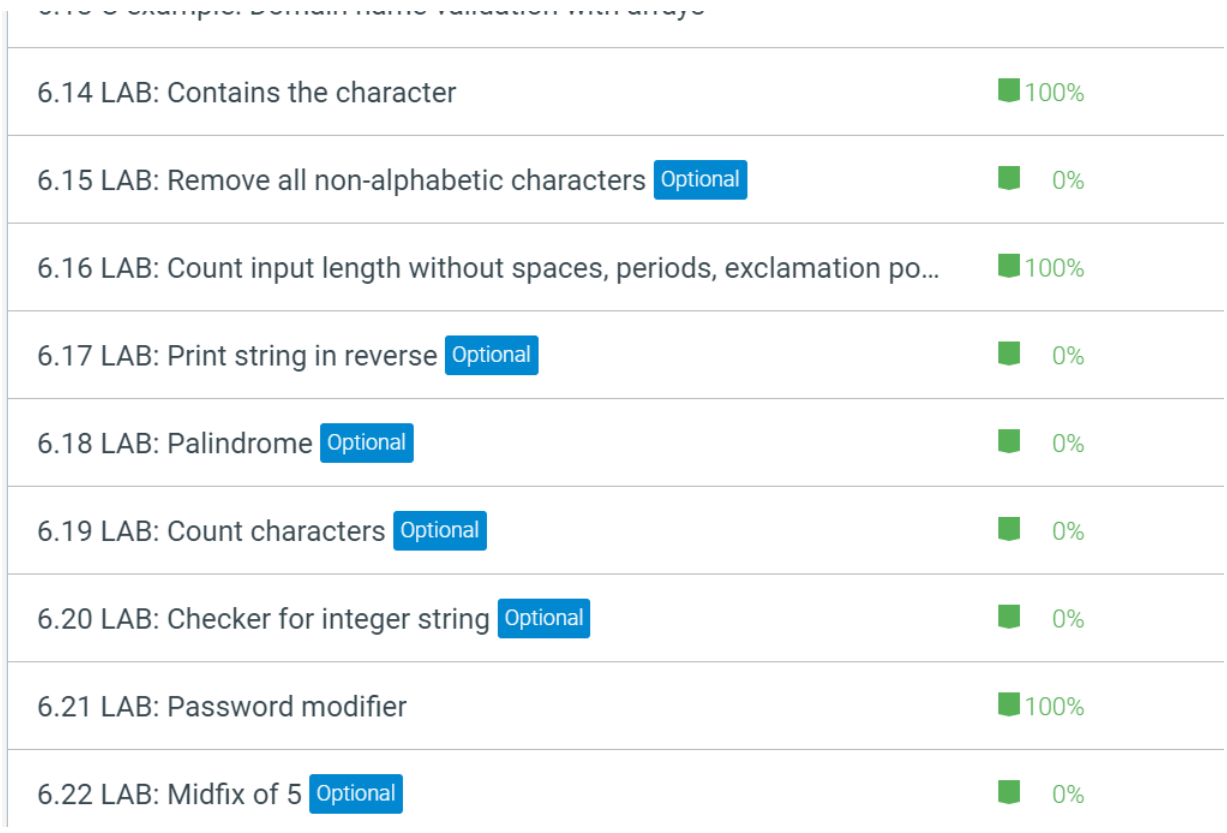
Make a copy before you begin (File -> Make a copy). Add the Assignment # above and complete the sections below BEFORE you begin to code and submit with your Assignment to D2L (File -> Download -> PDF). The sections will expand as you type.

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 is a wonderful tool to create a uniform file name! Simply give it your name, whether the assignment is late or not, the date of completion, and file name to create a professional looking file name.

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.

This program uses two variables to store user inputs.
Char userChoice
Char Array fileName

We prompt the user for an input that is written to specific parts of the array based on where in the array the last valid write was. We require input for the last name, first name, lateness, ID, timestamp, and file name functions. Input validation is done on a per-function basis that alter how the fileName input can be written to.

Finally, we prompt for a choice in whether the user wants to proceed further with the program or quit.

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

The program will output a series of messages to the user prompting for inputs. If an input is deemed invalid, an error will be displayed prompting a retry of the data entry.

If program has run correctly, the constructed file name will be displayed. If that file name exceeds 100 characters, then a warning message is thrown advising the user to re-run this tool. We still give them the file name in case it is 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.

Only rudimentary arithmetic is being calculated as bookmarking for where to operate alterations to the array

Set lengthPreCat = length of array before start of function;
Set lengthPostCat = length of array before after function input call;

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 Str userChoice = ""
DECLARE Str fileName[256] = ""
DECLARE Int lengthPreCat = 0
DECLARE Int lengthPostCat = 0
DECLARE Int keepGoing = 0

DISPLAY "Hello! Welcome to this file name encoder"

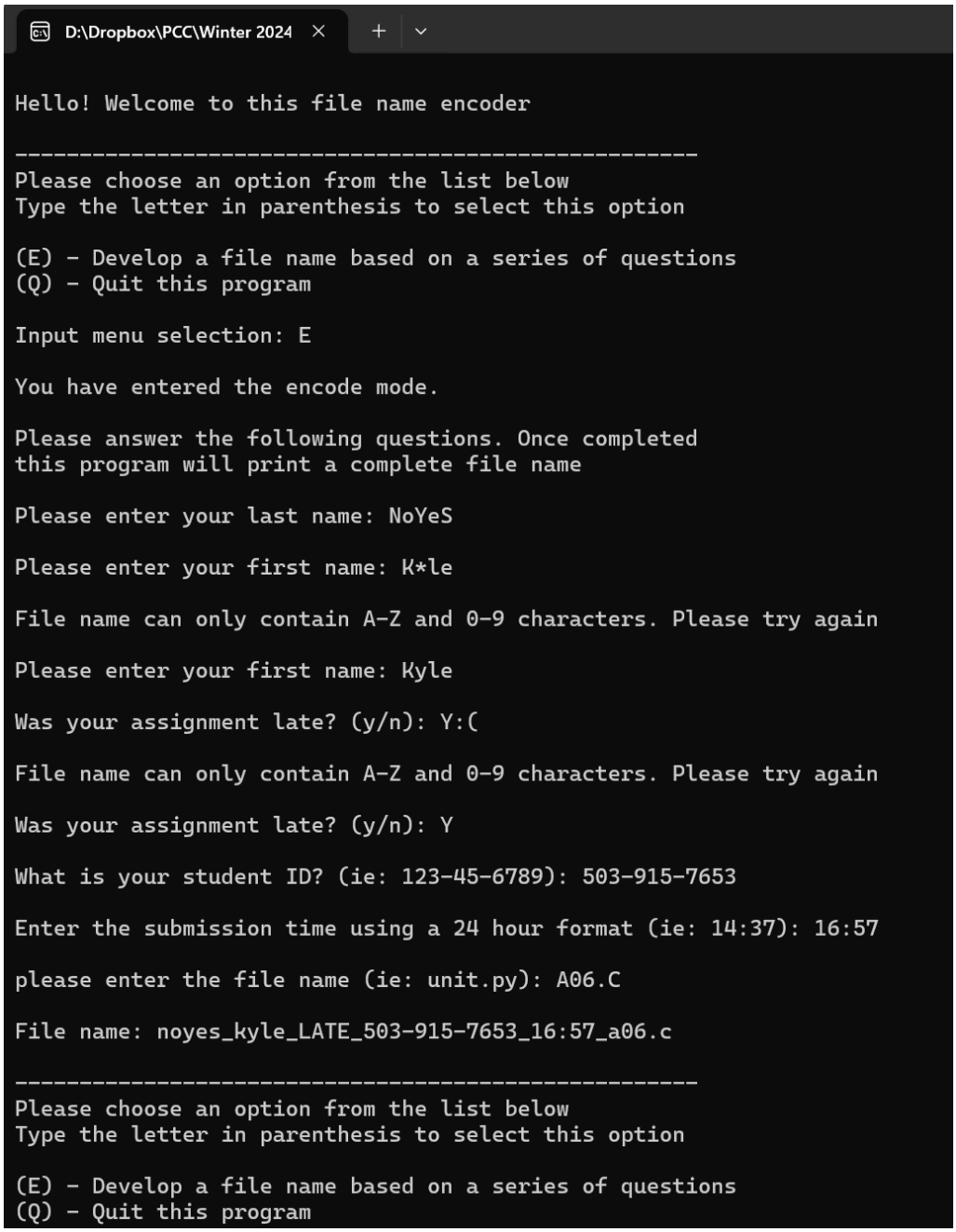
WHILE userChoice != "Q"
- - DISPLAY "Please choose an option from the list below"
- -
- - DISPLAY "Type the word in parenthesis to select this option"
- - DISPLAY "(Enocde) - Develop a file name based on a series of questions"
- - DISPLAY "(Help) - Summary of how the encoding system works"
- - DISPLAY "(Quit) - Quit this program"
- -
- - DISPLAY "Input menu selection: "
- - INPUT userChoice
- -
- - IF userChoice[0] == "e" OR userChoice[0] == "E"
- - - - DISPLAY "You have entered the encode mode"
- - - - DISPLAY "Please answer the following questions. Once completed,"
- - - - DISPLAY "this program will print a complete file name"
- - - -
- - - -
- - - - DO
- - - - - - SET lengthPreCat = length(fileName)
- - - - - - SET keepGoing = 1
- - - - - - DISPLAY "Please enter your last name: "
- - - - - - INPUT fileName
- - - - - - IF fileName == [A-Z] AND fileName == [0-9] AND fileName == [_]
- - - - - - - - SET fileName == fileName + "_"
- - - - - - - - keepGoing = 0
- - - - - - ELSE
- - - - - - - - DISPLAY "File name can only contain A-Z and 0-9 characters. Please try again"
- - - - - - - - FOR i = lengthPreCat TO lengthPostCat
- - - - - - - - - - fileName[i] = "\0"
- - - - WHILE keepGoing == 1
- - - -
- - - - DO
- - - - - - SET lengthPreCat = length(fileName)
- - - - - - SET keepGoing = 1

- - - - - - DISPLAY "Please enter your first name: "
- - - - - - INPUT fileName
- - - - - - IF fileName == [A-Z] AND fileName == [0-9] AND fileName == [_]
- - - - - - - - SET fileName == fileName + "_"
- - - - - - - - keepGoing = 0
- - - - - - ELSE
- - - - - - - - DISPLAY "File name can only contain A-Z and 0-9 characters. Please try again"
- - - - - - - - FOR i = lengthPreCat TO lengthPostCat
- - - - - - - - - - fileName[i] = "\0"
- - - - WHILE keepGoing == 1
- - - -
- - - - DO
- - - - - - SET lengthPreCat = length(fileName)
- - - - - - SET keepGoing = 1
- - - - - - DISPLAY "Was your assignment late? (y/n): "
- - - - - - INPUT fileName
- - - - - - IF fileName == [A-Z] AND fileName == [0-9] AND fileName == [_]
- - - - - - - - SET fileName == fileName + "_"
- - - - - - - - keepGoing = 0
- - - - - - ELSE
- - - - - - - - DISPLAY "File name can only contain A-Z and 0-9 characters. Please try again"
- - - - - - - - FOR i = lengthPreCat TO lengthPostCat
- - - - - - - - - - fileName[i] = "\0"
- - - - WHILE keepGoing == 1
- - - -
- - - - DO
- - - - - - SET lengthPreCat = length(fileName)
- - - - - - SET keepGoing = 1
- - - - - - DISPLAY "What is your student ID? (ie: 123-45-6789): "
- - - - - - INPUT fileName
- - - - - - IF fileName == [A-Z] AND fileName == [0-9] AND fileName == [_] AND fileName == [-
]
- - - - - - - - SET fileName == fileName + "_"
- - - - - - - - keepGoing = 0
- - - - - - ELSE
- - - - - - - - DISPLAY "Input can only contain numbers 0-9 and dashes. Please try again"
- - - - - - - - FOR i = lengthPreCat TO lengthPostCat
- - - - - - - - - - fileName[i] = "\0"
- - - - WHILE keepGoing == 1
- - - -
- - - - DO
- - - - - - SET lengthPreCat = length(fileName)
- - - - - - SET keepGoing = 1
- - - - - - DISPLAY "Enter the submission time using a 24 hour format (ie: 14:37): "
- - - - - - INPUT fileName
- - - - - - IF fileName == [A-Z] AND fileName == [0-9] AND fileName == [_]
- - - - - - - - SET fileName == fileName + "_"
- - - - - - - - keepGoing = 0

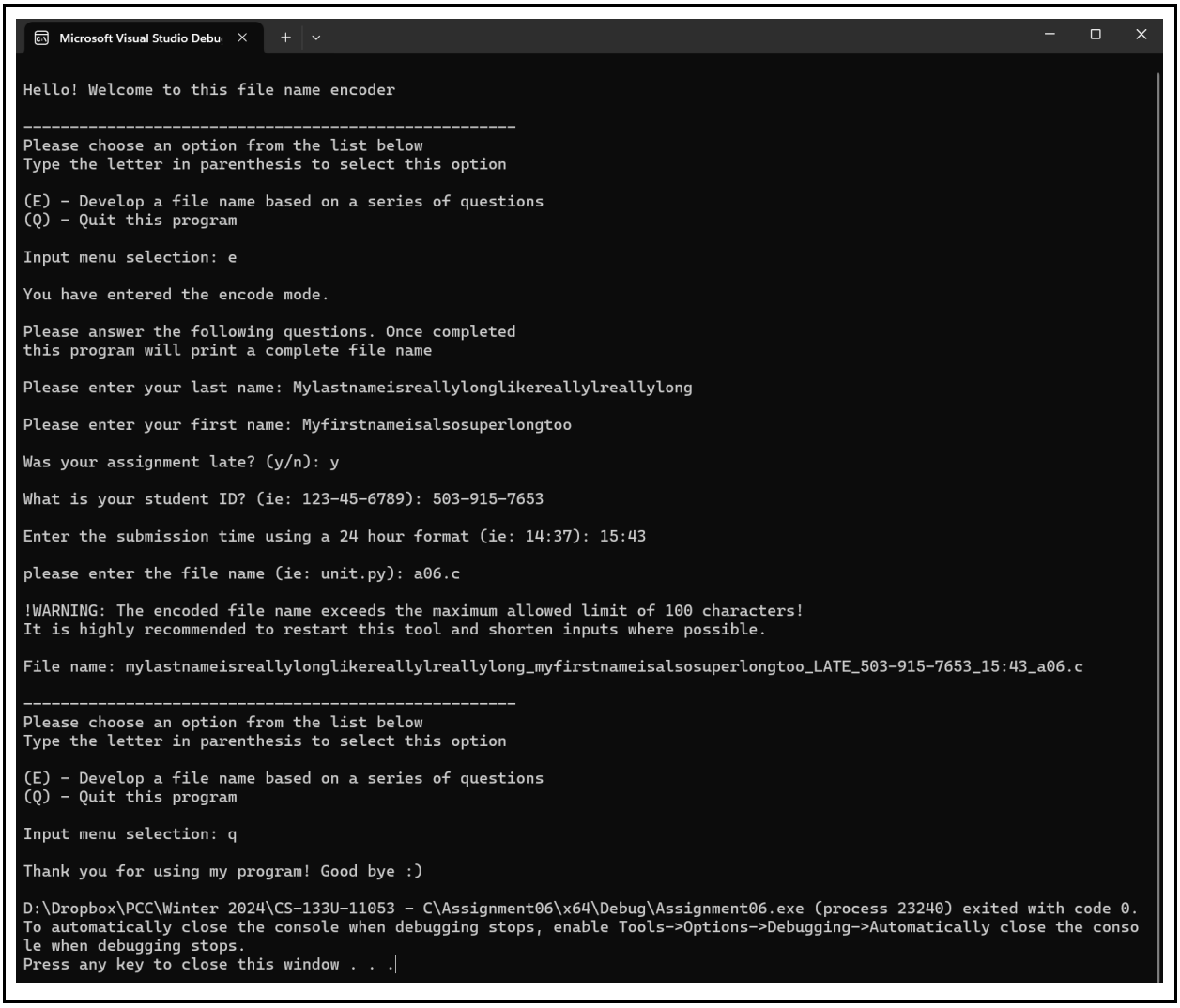
```
- - - - - - ELSE
- - - - - - - - DISPLAY "File name can only contain A-Z and 0-9 characters. Please try again"
- - - - - - - - FOR i = lengthPreCat TO lengthPostCat
- - - - - - - - - - fileName[i] = "\0"
- - - - WHILE keepGoing == 1
- - - -
- - - - DO
- - - - - - SET lengthPreCat = length(fileName)
- - - - - - SET keepGoing = 1
- - - - - - DISPLAY "please enter the file name (ie: unit.py): "
- - - - - - INPUT fileName
- - - - - - IF fileName == [A-Z] AND fileName == [0-9] AND fileName == [_]
- - - - - - - - keepGoing = 0
- - - - - - ELSE
- - - - - - - - DISPLAY "File name can only contain A-Z and 0-9 characters. Please try again"
- - - - - - - - FOR i = lengthPreCat TO lengthPostCat
- - - - - - - - - - fileName[i] = "\0"
- - - - WHILE keepGoing == 1
- - - -
- - - - IF lengthPostcat > 100
- - - - - - DISPLAY "WARNING: File name exceeds maximum allowed 100 characters.
Recommend redo!"
- - - -
- - - - FOR i = 0 TO i = lengthPostcat
- - - - - - IF fileName[i] is alpha
- - - - - - - - SET fileName[i] = lower(fileName[i])
- - - - DISPLAY "File name: {fileName}"
- - ELSE
- - - - DISPLAY "Given input was invalid. Please type the option name within the parentheses"
- -
DISPLAY "Thank you for using my program! Good bye :)"

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:



Sample Program Run 2:



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`