CS 161A/B: Programming and Problem Solving I

Algorithm 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. The sections will expand as you type. When you are finished, download this document as a PDF (File -> Download -> PDF) and submit to D2L.

This document contains an interactive checklist. To mark an item as complete, click on the box (the entire list will be highlighted), then right click (the clicked box will only be highlighted), and choose the checkmark.

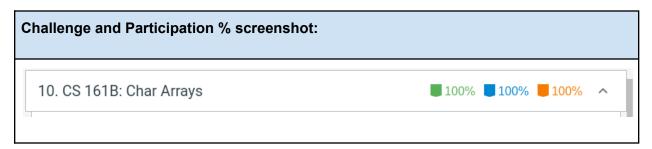
Planning your program before you start coding is part of the development process. In this document you will:

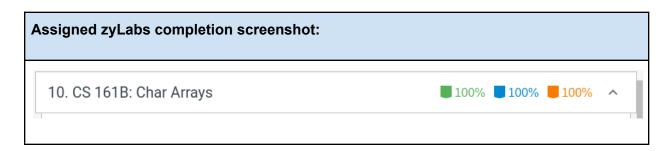
Paste a	screenshot of y	our zv	Books	Challenge	and Partic	cipation	%
i doto d	ool collollot of	, Ou: 2 y	DOONG	Cilancinge	and i and	oipation	/ 0

- ☐ Paste a screenshot of your assigned zyLabs completion
- ☐ Write a detailed description of your program, at least two complete sentences
- ☐ If applicable, design a sample run with test input and output
- ☐ Identify the program inputs and their data types
- ☐ Identify the program outputs and their data types
- Identify any calculations or formulas needed
- ☐ Write the algorithmic steps as pseudocode or a flowchart
- ☐ Tools for flowchart Draw.io Diagrams.net

1. 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%.





2. Program Description

In the box below, describe the purpose of the program. You must include a detailed description with at least two complete sentences.

Program description:

This program will allow for PCC to encode and build the file name for D2L. You should ask the user at the beginning if they are building the name of the file (encoding it) or if the program wants to quit. Then, the user will need to enter in information to allow the program to build the encoded file name or quit.

3. Sample Run

If you are designing your own program, you will start with a sample run. Imagine a user is running your program - what will they see? What inputs do you expect, and what will be the outputs from the given inputs? Choose test data you will use to test your program. Calculate and show the expected outputs. Use the sample run to test your program.

Sample run:

```
Welcome to my fileName encoding program!!

Please pick an option below:
(e) Encode a file name
(q) quit
>>e
This program will ask you a few questions and generate an encoded fileName based on your answers.

Enter your last name: Iyer

Enter your first name: GD

Was your assignment Late (y/n)? Y

Enter your Student-ID (format: 222-22-2222): 234-05-4556

Enter the file name: a05.cpp

Enter the time submitted (military time - ex: 18:24 for 6:24pm):
13:45
```

```
Your encoded file name is: iyer_gd_LATE_4556_1345_a05.cpp

Please pick an option below:
(e)Encode a file name
(q)quit
>>b

Invalid option! Please try again!!

Please pick an option below:
(e)Encode a file name
(q)quit
>>q

Thank you for using my fileName generator!
```

4. Algorithmic Design

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.

Use the pseudocode syntax shown in the document, supplemented with English phrases if necessary. **Do not include any implementation details (e.g. source code file names, class or struct definitions, or language syntax)**. Do not include any C++ specific syntax or data types.

Algorithmic design:

a. Identify and list all of the user input and their data types. Include a variable name, data type, and description. Data types include string, integer, floating point, (single) character, and boolean. Data structures should be referenced by name, e.g. "array of integer" or "array of string (for CS161B and up).

char IName, fName, parsedID, fileName, strTime

bool lateFlag

b. Identify and list all of the user output and their data types. Include a variable name, data type, and description. Data types include string, integer, floating point, (single) character, and boolean. Data structures should be referenced by name, e.g. "array of integer" or "array of string" (for CS161B and up).

char encodedFileName

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. Formulae should reference the variable names from step a and step b as applicable.

stdID + 7

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

Use the syntax shown at the bottom of this document and plain English phrases. Do not include any implementation details (e.g. file names) or C++ specific syntax.

FUNCTION void welcome()

DISPLAY "Welcome to my fileName encoding program!!"

END FUNCTION

FUNCTION char displayMenu()

DECLARE char option

DISPLAY "Please pick an option below:"

DISPLAY "(e)Encode a file name"

DISPLAY "(q)quit"

DISPLAY ">>"

INPUT option

WHILE option != 'e' or option != 'q'

DISPLAY "Invalid option! Please try again!!"

DISPLAY "Please pick an option below:"

DISPLAY "(e)Encode a file name"

```
DISPLAY "(q)quit"
   DISPLAY ">>"
   INPUT option
 END WHILE
 IF option == 'e'
   DISPLAY encode()
 ELSE
   DISPLAY "Thank you for using my fileName generator"
 END IF
 RETURN option
END FUNCTION
FUNCTION void readInput(char fName[], char IName[], bool &lateFlag)
 DECLARE int i
 DECLARE char lateOrNot
 DISPLAY "Enter your last name: "
 INPUT IName
 DISPLAY "Enter your first name: "
 INPUT fName
 DISPLAY "Was your assignment Late (y/n)? "
 INPUT lateOrNot
 WHILE lateOrNot != 'y' and lateOrNot != 'Y' and lateOrNot != 'n' and lateOrNot != 'N'
   DISPLAY "Invalid input! Please try again!!"
   DISPLAY "Was your assignment Late (y/n)? "
   INPUT lateOrNot
```

```
END WHILE
 IF lateOrNot == 'y' or lateOrNot == 'Y'
  SET lateFlag = true
 ELSE IF lateOrNot == 'n' or lateOrNot == 'N'
   SET lateFlag = false
 END IF
 FOR (i = 0; i < strlen(IName); i++)
   SET IName[i] = tolower(IName[i]);
 END FOR
 FOR (i = 0; i < strlen(fName); i++)
   SET fName[i] = tolower(fName[i]);
 END FOR
END FUNCTION
FUNCTION void readInput(char parsedID[], char fileName[])
 DECLARE char stdID[51]
 DISPLAY "Enter your Student-ID (format: 222-22-222):"
 INPUT stdID
 DISPLAY "Enter the file name: "
 INPUT fileName
 SET strncpy(parsedID, stdID + 7, 4)
END FUNCTION
FUNCTION void readTime(char strTime[])
 DECLARE int hour = 0, min = 0;
```

```
DECLARE char discard
 DISPLAY "Enter the time submitted (military time - ex: 18:24 for 6:24pm): "
 INPUT hour, discard, min
 WHILE (!cin || discard != ':')
   DISPLAY "Invalid input! Please try again!!"
   CLEAR
   DISPLAY "Enter the time submitted (military time - ex: 18:24 for 6:24pm): "
   INPUT hour, discard, min
 END WHILE
 IGNORE
 SET strncpy(strTime, to_string(hour).c_str(),10)
 SET strcat(strTime, to_string(min).c_str())
END FUNCTION
FUNCTION void encode()
 DECLARE char IName[51], fName[51], lateOrNot, parsedID[51], fileName[51],
strTime[51], encodeFileName[51]
 DECLARE bool lateFlag
 DISPLAY "This program will ask you a few questions and generate an encoded fileName
based on your answers."
 DISPLAY readInput(fName, IName, lateFlag)
 DISPLAY readInput(parsedID, fileName)
 DISPLAY readTime(strTime)
 IF lateFlag == true
   SET strcat(IName, "_")
   SET strcpy(encodeFileName, IName)
```

```
SET strcat(encodeFileName, fName)
   SET strcat(encodeFileName, "_")
   SET strcat(encodeFileName, "LATE")
   SET strcat(encodeFileName, "_")
   SET strcat(encodeFileName, parsedID)
   SET strcat(encodeFileName, "_")
   SET strcat(encodeFileName, strTime)
   SET strcat(encodeFileName, "_")
   SET strcat(encodeFileName, fileName)
   DISPLAY "Your encoded file name is: ", encodeFileName
 ELSE IF lateFlag == false
   SET strcat(IName, "_")
   SET strcpy(encodeFileName, IName)
   SET strcat(encodeFileName, fName)
   SET strcat(encodeFileName, "_")
   SET strcat(encodeFileName, parsedID)
   SET strcat(encodeFileName, "_")
   SET strcat(encodeFileName, strTime)
   SET strcat(encodeFileName, "_")
   SET strcat(encodeFileName, fileName)
   DISPLAY "Your encoded file name is: ", encodeFileName
 END IF
END FUNCTION
FUNCTION int main()
```

```
DISPLAY welcome();

SET char option = displayMenu();

WHILE option != 'q'

SET option = displayMenu();

END WHILE

END FUNCTION
```

5. 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	<pre>IF num_dogs > 10 THEN DISPLAY "That is a lot of dogs!" END IF</pre>	
Use a dual alternative conditional	IF condition THEN statement statement ELSE statement statement END IF	<pre>IF num_dogs > 10 THEN</pre>	
Use a switch/case statement	SELECT variable or expression CASE value_1: statement statement CASE value_2:	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	

	statement statement CASE value_2: statement statement DEFAULT: statement statement statement	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	<pre>SET num_dogs = 1 WHILE num_dogs < 10 DISPLAY num_dogs, " dogs!" SET num_dogs = num_dogs + 1 END WHILE</pre>			
Loop while a condition is true - the loop body will execute 1 or more times.	DO statement statement WHILE condition	<pre>SET num_dogs = 1 DO DISPLAY num_dogs, " dogs!" SET num_dogs = num_dogs + 1 WHILE num_dogs < 10</pre>			
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			