CS 161B: Programming and Problem Solving I

Assignment a02 Algorithm Design Document

by Josiah Appert

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.

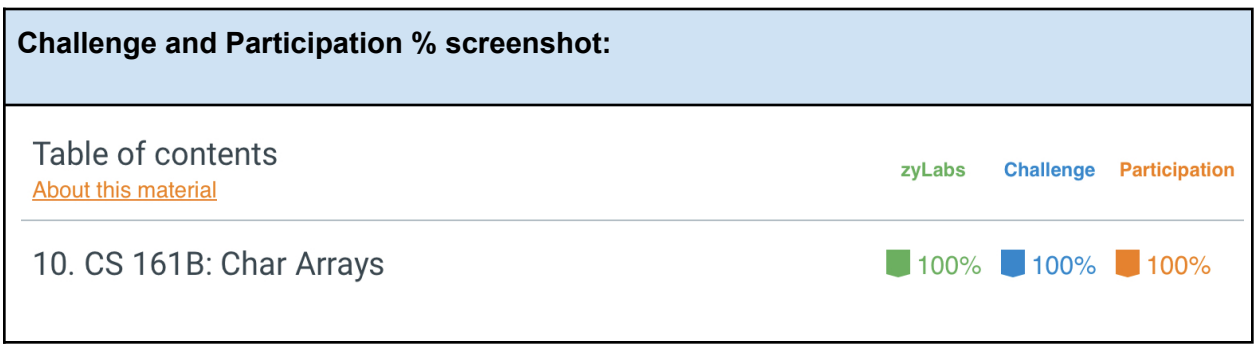
Planning your program before you start coding is part of the development process. In this document you will:

- ☑ ~~Paste a screenshot of your zyBooks Challenge and Participation %~~
- ☑ ~~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%.

Challenge and Participation % screenshot:



Assigned zyLabs completion screenshot:

Table of contents

zyLabs

About this material

10.9 C++ LAB: Remove spaces - functions 100%

Lab activity

User score: 10 / 10 points

10.10 C++ LAB: Print string in reverse 100%

Lab activity

User score: 10 / 10 points

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 creates an appropriate filename based upon input taken from the user. To do so, it demonstrates use of string functions from the cstring library, character arrays, overloaded functions, and functions with value and reference parameters.

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!
```

```
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: Appert

Enter your first name: Josiah

Was your assignment Late (y/n)? 1

Invalid Option! Please try again!!

Was your assignment Late (y/n)? n

Enter your Student-ID (format: 222-22-2222): 434-45-2425

Enter the file name: d35.doc

Enter the time submitted (military time - ex: 18:24 for 6:24pm): 34:23

Invalid Option! Try Again!!

>>23:24

Your encoded file name is: appert_josiah_2425_2324_d35.doc

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).

- menuChoice as character
- lName as character array
- fName as character array
- isLate as character
- stdID as character array
- fileName as character array
- hourVar as integer
- discardColon as character
- minVar as integer

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).

- encodeFileName as characters array

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.

- SET constant character MAX_STRING = 20
- WHILE CALL displayMenu() == 'e'
    - CALL encode(encodeFileName)
    - DISPLAY "Your encoded file name is: ", encodeFileName
- END WHILE
- DO
    - DISPLAY "Please pick an option below:"
    - DISPLAY "(e)Encode a file name"
    - DISPLAY "(q)quit"
    - INPUT into menuChoice
    - SET menuChoice = tolower(menuChoice)
    - IF !cin OR menuChoice != 'q' AND menuChoice != 'e' THEN
        - DISPLAY "Invalid option! Please try again!!"
        - clear input stream
    - END IF
- WHILE !cin OR menuChoice != 'q' && menuChoice != 'e'
- IF lateFlag == true THEN

        - CALL strcat(encodeFileName, "LATE_")
- END IF
- DO
    - DISPLAY "Was your assignment Late (y/n)? "
    - INPUT into isLate
    - SET isLate = CALL tolower(isLate)
    - IF isLate != 'y' AND isLate != 'n' THEN
        - DISPLAY "Invalid option! Please try again!!"
    - END IF
- WHILE isLate != 'y' AND isLate != 'n'
- FOR i = 0 TO strlen(strLower)
    - SET strLower[i] = CALL tolower(strLower[i])
- END FOR
- WHILE invalid input OR discardColon != ':'
    - DISPLAY "Invalid option! Please try again!!"
    - clear input stream
    - INPUT into hourVar, discardColon, minVar
- END WHILE

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.

1. SET constant character MAX_STRING = 20
2. FUNCTION integer main()
    a. DECLARE character encodeFileName[50]
    b. CALL welcome()
    c. WHILE CALL displayMenu() == 'e'
        i. CALL encode(encodeFileName)
        ii. DISPLAY "Your encoded file name is: ", encodeFileName
    d. END WHILE
    e. DISPLAY "Thank you for using my fileName generator!"
    f. RETURN 0;

    END FUNCTION main()

3. FUNCTION void welcome()
    a. DISPLAY "Welcome to my fileName encoding program!!"

    END FUNCTION welcome()

4. FUNCTION character displayMenu()
    a. DECLARE character menuChoice
    b. DO
        i. DISPLAY "Please pick an option below:"
        ii. DISPLAY "(e)Encode a file name"
        iii. DISPLAY "(q)quit"
        iv. INPUT into menuChoice
        v. SET menuChoice = tolower(menuChoice)
        vi. IF !cin OR menuChoice != 'q' AND menuChoice != 'e' THEN
            1. DISPLAY "Invalid option! Please try again!!"
            2. clear input stream
        vii. END IF
    c. WHILE !cin OR menuChoice != 'q' && menuChoice != 'e'
    d. RETURN menuChoice

    END FUNCTION displayMenu()

5. FUNCTION void encode(character array encodeFileName[])
    a. DECLARE character array fName[MAX_STRING]
    b. DECLARE character array lName[MAX_STRING]
    c. DECLARE boolean lateFlag
    d. DECLARE character array parsedID[MAX_STRING]
    e. DECLARE character array fileName[MAX_STRING]
    f. DECLARE character array strTime[MAX_STRING]
    g. DISPLAY "This program will ask you a few questions and generate an"
    h. DISPLAY "encoded fileName based on your answers."
    i. CALL readInput(fName, lName, lateFlag)
    j. CALL readInput(parsedID, fileName)
    k. CALL readTime(strTime)
    l. CALL strncpy(encodeFileName, lName, 10)
    m. CALL strcat(encodeFileName, "_")
    n. CALL strcat(encodeFileName, fName)
    o. CALL strcat(encodeFileName, "_")
    p. IF lateFlag == true THEN
        i. CALL strcat(encodeFileName, "LATE_")
    q. END IF
    r. CALL strcat(encodeFileName, parsedID)
    s. CALL strcat(encodeFileName, "_")
    t. CALL strcat(encodeFileName, strTime)
    u. CALL strcat(encodeFileName, "_")
    v. CALL strcat(encodeFileName, fileName)
    w. CALL strcat(encodeFileName, "\0")

END FUNCTION encode()

6. FUNCTION void readInput(character array fName[], character array lName[], boolean &lateFlag)
    a. DECLARE character isLate
    b. DISPLAY "Enter your last name: "
    c. INPUT into lName
    d. CALL strToLower(lName)
    e. DISPLAY "Enter your first name: "
    f. INPUT into fName
    g. CALL strToLower(fName)
    h. DO
        i. DISPLAY "Was your assignment Late (y/n)? "
        ii. INPUT into isLate
        iii. SET isLate = CALL tolower(isLate)
        iv. IF isLate != 'y' AND isLate != 'n' THEN
            1. DISPLAY "Invalid option! Please try again!!"
        v. END IF
    i. WHILE isLate != 'y' AND isLate != 'n'

END FUNCTION readInput()

7. FUNCTION void strToLower(character array strLower[])
    a. FOR i = 0 TO strlen(strLower)
        i. SET strLower[i] = CALL tolower(strLower[i])
    b. END FOR

END FUNCTION strToLower()

8. FUNCTION void readInput(character array parsedID[], character array fileName[])
    a. DECLARE character array stdID[MAX_STRING]
    b. DISPLAY "Enter your Student-ID (format: 222-22-2222): "
    c. INPUT into stdID
    d. CALL strncpy(parsedID, stdID + 7, 4)
    e. DISPLAY "Enter the file name: "
    f. INPUT into fileName

END FUNCTION readInput()

9. FUNCTION void readTime(character array strTime[])
    a. DECLARE integer hourVar = 0
    b. DECLARE integer minVar = 0
    c. DECLARE character discardColon
    d. DISPLAY "Enter the time submitted (military time - ex: 18:24 for 6:24pm): "
    e. INPUT into hourVar, discardColon, minVar

```
    f. WHILE invalid input OR discardColon != ':'
        i.   DISPLAY "Invalid option! Please try again!!"
        ii.  clear input stream
        iii. INPUT into hourVar, discardColon, minVar
    g. END WHILE
    h. CALL strncpy(strTime, CALL to_string(hourVar).c_str(), 10)
    i. CALL strcat(strTime, CALL to_string(minVar).c_str())

END FUNCTION readTime()
```

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
		`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
		`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!"`

	statement CASE *value_2:* *statement* *statement* DEFAULT: *statement* *statement* END SELECT	`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`