CS 161A: Programming and Problem Solving I

Assignment A05 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. 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:

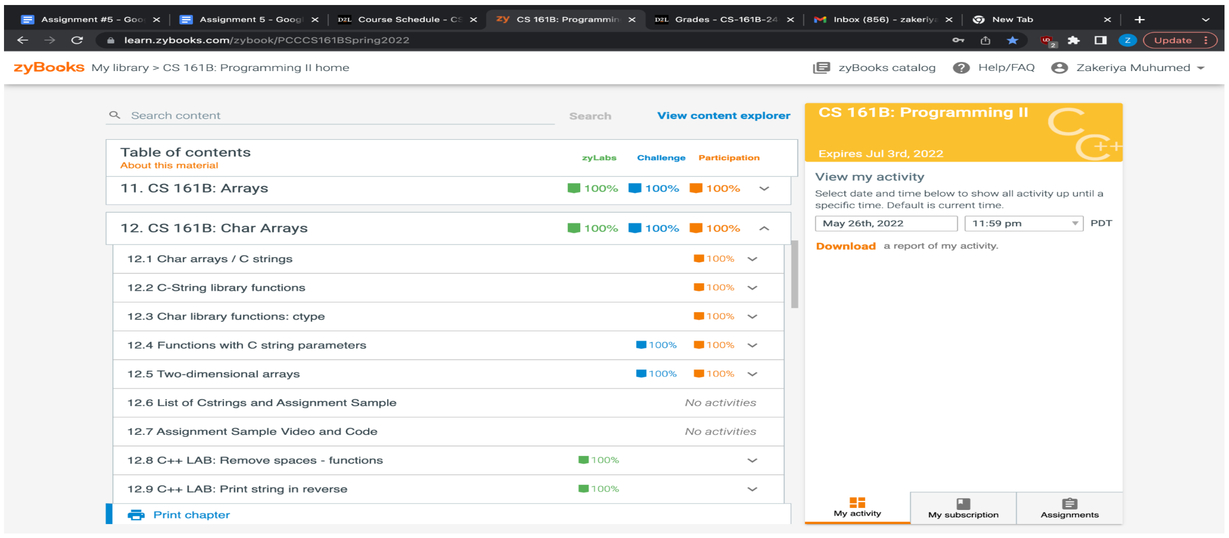

Table of contents	zyLabs	Challenge	Participation
About this material			
11. CS 161B: Arrays	100%	100%	100%
12. CS 161B: Char Arrays	100%	100%	100%
12.1 Char arrays / C strings			100%
12.2 C-String library functions			100%
12.3 Char library functions: ctype			100%
12.4 Functions with C string parameters		100%	100%
12.5 Two-dimensional arrays		100%	100%
12.6 List of Cstrings and Assignment Sample	No activities		
12.7 Assignment Sample Video and Code	No activities		
12.8 C++ LAB: Remove spaces - functions	100%		
12.9 C++ LAB: Print string in reverse	100%		

Print chapter

CS 161B: Programming II

Expires Jul 3rd, 2022

View my activity

Select date and time below to show all activity up until a specific time. Default is current time.

May 26th, 2022 | 11:59 pm | PDT

Download a report of my activity.

Assigned zyLabs completion screenshot:

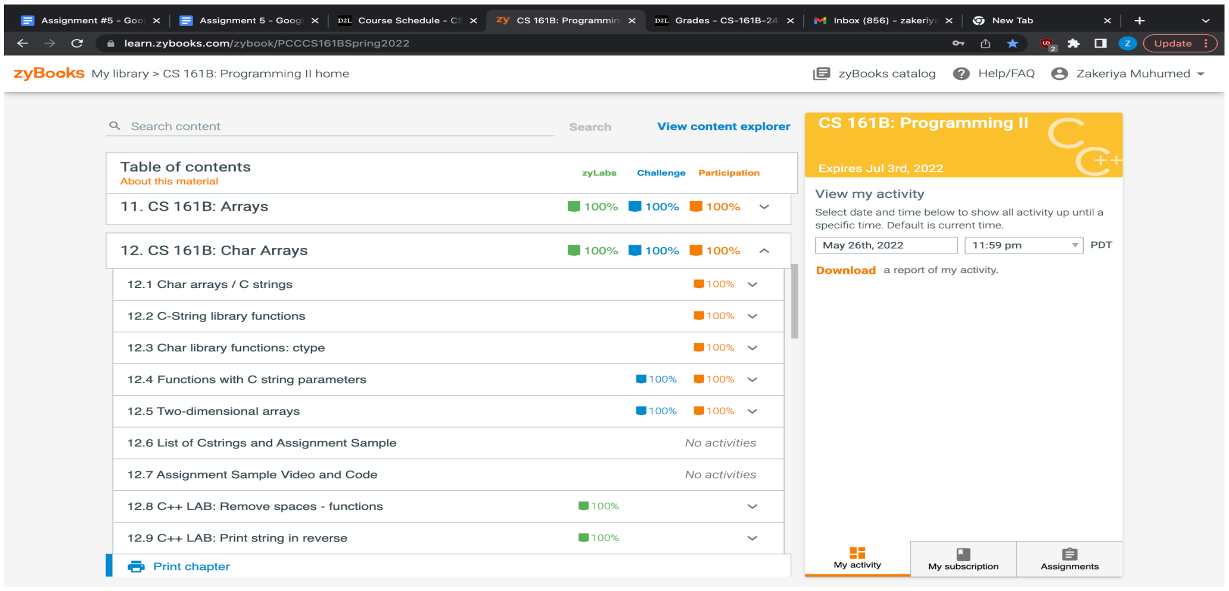
zyBooks My library > CS 161B: Programming II home

Table of contents	zyLabs	Challenge	Participation
About this material			
11. CS 161B: Arrays	100%	100%	100%
12. CS 161B: Char Arrays	100%	100%	100%
12.1 Char arrays / C strings			100%
12.2 C-String library functions			100%
12.3 Char library functions: ctype			100%
12.4 Functions with C string parameters		100%	100%
12.5 Two-dimensional arrays		100%	100%
12.6 List of Cstrings and Assignment Sample	No activities		
12.7 Assignment Sample Video and Code	No activities		
12.8 C++ LAB: Remove spaces - functions	100%		
12.9 C++ LAB: Print string in reverse	100%		

Print chapter

CS 161B: Programming II

Expires Jul 3rd, 2022

View my activity

Select date and time below to show all activity up until a specific time. Default is current time.

May 26th, 2022 | 11:59 pm | PDT

Download a report of my activity.

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:

In this program, you will be writing a function to count the occurrence of each word in a string of text, and keep only unique words in the array.

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 Word Frequency Counter!!

This frequency will count the number of occurrences of
each word. The number of words in your list must be
entered first followed by the list of words separated by
space. These are the rules of this frequency counter!

Enter the count of words first (as a whole number) and the list of
words separated by space:

8 Hey Hi Hey Priya How are you Priya

Your list before deletes and counts:
Hey
Hi
Hey
Priya
How
Are
You
Priya

The frequency counts and list with unique words are as below:

Hey 2
Hi 1
Priya 2
How 1
are 1
```

```
you 1

Thank you for using my frequency counter!
```

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.

Algorithmic design:
● Identify and list all of the user input and their data types.
○ Char array userString to store users list of words ○ wordCount as integer for number of words
● Identify and list all of the user output and their data types.
○ numList as int array to store word frequency ○ userString as Char array to store users list of words
● 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.
○ Just addition to count the number of words
● 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.
1. **FUNCTION** menu()   a. **DISPLAY** welcome message and the instructions   **END FUNCTION menu** 2. **FUNCTION** readString(array of Cstring, array for numList, int wordCount)

```
        a. DISPLAY  prompt “ Enter number message”
        b. INPUT wordCount
        c. WHILE words less than 0 and word is greater than 10
             i.   DISPLAY error message
             ii.  INPUT wordCount
        d. END WHILE
        e. FOR LOOP i = 0, until word, increase i by 1
             i.   INPUT userString of i
        f. END FOR

    END FUNCTION readString()

3.  FUNCTION countWords(array of userString, list for numList, int wordCount)
        a. FOR int i until wordCount, increase i
        b. FOR int j until wordCount, increase i
                        1. IF strcmp function, userString of i, and userString of j equal 0
                               a. SET numList of i ++
                        2. END IF
             ii.  END FOR for j
        c. END FOR for i

    END FUNCTION countWords()


4.  FUNCTION printWordCount(array of strings, list of wordCount, int count)
        a. FOR LOOP from i = 0 , till count, increase i
             i.   IF numList of i == 0
                    1. DISPLAY userString of i
             ii.  ELSE
                    1. DISPLAY userString and numlist arrays
        b. END FOR LOOP

    END FUNCTION printWordCount()

5.  FUNCTION  remove(int row, int wordCount, char userString, int numList)
        a. FOR int i = row, until i = wordCount , increase i
             i.   strcpy of userString of i and userString of i+1
             ii.  numList of i = numList +1
        b. END FOR LOOP
             i.   wordCount decrease

        END OF remove()

6.  FUNCTION main()
        a. DECLARE an char array for the userString
        b. DECLARE an array of ints for the numList
    c. DECLARE wordCount as int
    d. CALL readString function with 2 array and wordCount as para
    e. Display "Your list before deletes and counts"
    f. CALL printWordCount with 2 array and wordCount as para
    g. CALL countWords and the numList array will get updated automatically
    h. FOR int i = 0 until wordCount, increase i
        i. IF num of i less than 1
            1. FOR int j = i + 1, until j is wordCount, increase j
                a. IF FUNCTION strcmp userString of i and j == 0
                    i. CALL remove function
                    ii. decrease i
    i. DISPLAY "The frequency counts and list with unique words are as below:"
    j. CALL printWordCount and DISPLAY the info to the user
    k. DISPLAY Thank you message

END FUNCTION main()
```

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
		`IF num_dogs > 10 THEN`
		`     DISPLAY "You have more than 10 dogs!"`
		`ELSE`
		`     DISPLAY "You have ten or fewer dogs!"`

	END IF	`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`