CS 161A: Programming and Problem Solving I

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. 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.

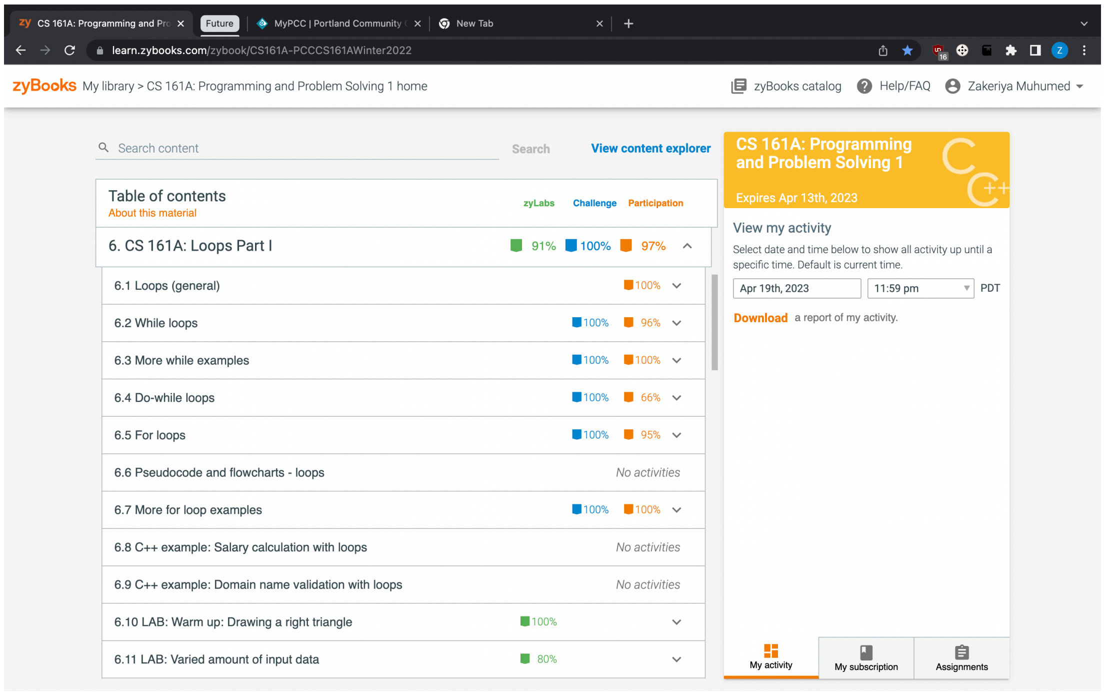
Planning your program before you start coding is part of the development process. In this document you will:

❏ 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:



CS 161A: Programming and Pro | Future | MyPCC | Portland Community | New Tab

learn.zybooks.com/zybook/CS161A-PCCCS161AWinter2022

zyBooks My library > CS 161A: Programming and Problem Solving 1 home — zyBooks catalog — Help/FAQ — Zakeriya Muhumed

Search content — Search — View content explorer

Table of contents
About this material

Section	zyLabs	Challenge	Participation
6. CS 161A: Loops Part I	🟩 91%	🟦 100%	🟧 97%
6.1 Loops (general)			🟧 100%
6.2 While loops		🟦 100%	🟧 96%
6.3 More while examples		🟦 100%	🟧 100%
6.4 Do-while loops		🟦 100%	🟧 66%
6.5 For loops		🟦 100%	🟧 95%
6.6 Pseudocode and flowcharts - loops			*No activities*
6.7 More for loop examples		🟦 100%	🟧 100%
6.8 C++ example: Salary calculation with loops			*No activities*
6.9 C++ example: Domain name validation with loops			*No activities*
6.10 LAB: Warm up: Drawing a right triangle	🟩 100%		
6.11 LAB: Varied amount of input data	🟩 80%		

CS 161A: Programming and Problem Solving 1
Expires Apr 13th, 2023

View my activity
Select date and time below to show all activity up until a specific time. Default is current time.
Apr 19th, 2023 | 11:59 pm | PDT
Download a report of my activity.

My activity | My subscription | Assignments

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:
you will be entering in jar and gumball dimensions like you did for Assignment 2, and you will continue asking the user for jar and gumball dimensions until they are done. While the user wants to add more input, you will add the data to **accumulators** (an accumulator is a variable the program uses to calculate a sum or product of a series of values) to perform statistical analysis after they are done.

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 Gumball Guesser program!

Enter the radius of a gumball (cm)
and the volume of a jar (mL) separated by a space: 1 500
Estimate of gumballs in the jar: 76

Do you want to enter more (y/n): y

Enter the radius of a gumball (cm)
and the volume of a jar (mL) separated by a space: .75 450
Estimate of gumballs in the jar: 162

Do you want to enter more (y/n): y

Enter the radius of a gumball (cm)
and the volume of a jar (mL) separated by a space: .5 123
Estimate of gumballs in the jar: 150

Do you want to enter more (y/n): n

Number of entries: 3
Average number of gumballs: 129.33
Largest gumball: 4.19 cm^3
Jar size for largest gumball estimate: 450 mL

Thank you for using my program!
```

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:
a. Identify and list all of the user input and their data types.
Prompt the user to enter the gumball radius and jar size. Use the same code you used in Assignment 2.
b. Identify and list all of the user output and their data types.

Welcome message

- ❏ When the user is finished entering gumball and jar information:
    - ❏ Display the number of entries entered.
    - ❏ Calculate and display the average number of gumballs for the number of jars entered.
    - ❏ Display the largest gumball size entered.
    - ❏ Display the jar size that contains the largest number of gumballs (careful - this is not the largest jar, but the size of the jar that had the largest estimate!).

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.

- ❏ Calculate and print the estimated number of gumballs that can fit in the jar. Use the same code you used in Assignment 2.

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.

Rewrite all the code from assignment 2. Declare avgnum, numenteries, numballs string 'n',char fill and largest

Do statement with the first excuse of original code assignment 2. Set num entires =+1

Print Do you want to enter more (y/n): with user input fill

Then write if statement for avg and largest.

End do statement

While statement fill is not 'n'.

Print Number of entries, Average number of gumballs, avgNum Largest gumball, Jar size for largest gumball estimate

Print "Thank you for using my program

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
	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* = *star*t 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`