CS 161A: Programming and Problem Solving I

Assignment A05 Sample 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.

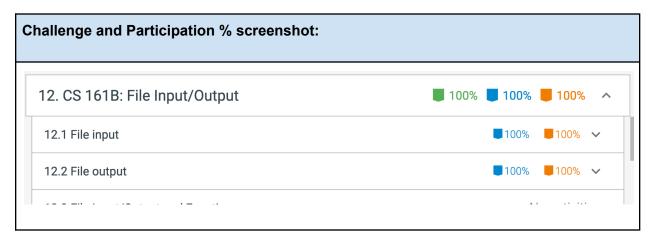
Planning your program before you start coding is part of the development process. In this document you will:

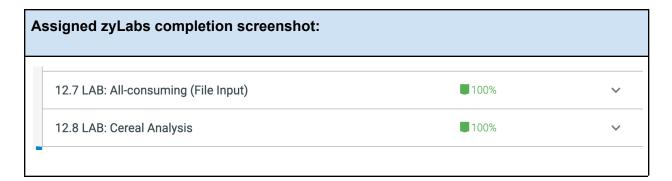
7	Paste a screenshot of	/OUR 7\	Pooks	Challongo	and	Darticipation	0/
_	Taste a soreenshoror	your z	y DOOKS	Chancinge	anu	r articipation	70

- ☑ 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 reads data from a file containing job titles, employment types, salaries, and remote work ratios. It analyzes the data to identify the position with the highest salary and provides a summary of the total number of positions and the average salary.

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.

```
Welcome to Gina's Data Analysis Program!
This program reads data from a file, performs analysis, and displays results.

Position with the highest salary:
Job Title: Research Engineer
Position: FT
Salary: $275000
Remote Ratio: 0%

Summary Analysis:
Total Positions: 10
Average Salary: $149664.00
```

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.
 - File items.txt (contains information about various job positions, including job titles, employment types (FT or CT), salaries, and remote work ratios)
- Identify and list all of the user output and their data types.
 - maxSalaryIndex (integer): stores the index of the job position with the highest salary (in the for loop)
 - totalPositions (integer): stores the total number of positions listed in the data file items txt
 - averageSalary (double): stores the avg salary calculated from all positions listed in the data file items.txt
 - o FUNCTIONS:
 - 1. analyzeData (void): function responsible for identifying the job position with the highest salary and printing out the details
 - 2. sumAnalysis (void): function that calculates total number of positions and average salary and prints out summary
- 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.
 - maxSalaryIndex: if (salaries[i] > salaries[maxSalaryIndex]) then maxSalaryIndex = i
 - averageSalary = totalSalary / count
- 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.

// Define constants

```
DECLARE ITEMS AS 10
DECLARE MAXCHAR AS 51
// Function prototypes
FUNCTION welcome()
FUNCTION openFile(inFile)
FUNCTION loadData(inFile, items[][], positions[][], salaries[], ratios[])
FUNCTION analyzeData(items[][], positions[][], salaries[], ratios[], count)
FUNCTION sumAnalysis(items[][], positions[][], salaries[], ratios[], count)
// Main function
FUNCTION main()
  DECLARE in File AS if stream
  DECLARE count AS Integer
  DECLARE items[ITEMS][MAXCHAR]
  DECLARE positions[ITEMS][MAXCHAR]
  DECLARE salaries[ITEMS]
  DECLARE ratios[ITEMS]
  // Display welcome message
  welcome()
  // Open file and check if it opens
  IF NOT openFile(inFile) THEN
    OUTPUT "File did not open! Program terminating!!"
    EXIT
  END IF
  // Load data from file
  count = loadData(inFile, items, positions, salaries, ratios)
  CLOSE inFile
  // Analyze the data
  analyzeData(items, positions, salaries, ratios, count)
  sumAnalysis(items, positions, salaries, ratios, count)
END FUNCTION
// Function to display welcome message
FUNCTION welcome()
  OUTPUT "Welcome to the Data Analysis Program!"
  OUTPUT "This program reads data from a file, performs analysis, and displays
results."
END FUNCTION
// Function to open file and check if it opens
FUNCTION openFile(inFile)
  OPEN "items.txt" AS inFile
  RETURN inFile.is open()
END FUNCTION
```

```
// Function to load data from file into arrays
FUNCTION loadData(inFile, items[][], positions[][], salaries[], ratios[])
  DECLARE count AS Integer
  SET count = 0
  WHILE count < ITEMS AND NOT inFile.end of file()
    READ items[count], positions[count], salaries[count], ratios[count] FROM inFile
     INCREMENT count
  END WHILE
  RETURN count
END FUNCTION
// Function to analyze the data
FUNCTION analyzeData(items[][], positions[][], salaries[], ratios[], count)
  DECLARE maxSalaryIndex AS Integer
  // Find the position with the highest salary
  SET maxSalaryIndex = 0
  FOR i FROM 1 TO count - 1
     IF salaries[i] > salaries[maxSalaryIndex] THEN
       SET maxSalaryIndex = i
     END IF
  END FOR
  // Output the position with the highest salary
  OUTPUT "Position with the highest salary:"
  OUTPUT "Job Title: " + items[maxSalaryIndex]
  OUTPUT "Position: " + positions[maxSalaryIndex]
  OUTPUT "Salary: $" + salaries[maxSalaryIndex]
END FUNCTION
// Function to perform summary analysis
FUNCTION sumAnalysis(items[][], positions[][], salaries[], ratios[], count)
  DECLARE totalPositions AS Integer
  DECLARE totalSalary AS Double
  DECLARE averageSalary AS Double
  // Calculate total positions and total salary
  SET totalPositions = count
  FOR i FROM 0 TO count - 1
     INCREMENT totalSalary BY salaries[i]
  END FOR
  // Calculate average salary
  SET averageSalary = totalSalary / count
  // Output summary analysis
  OUTPUT "Summary Analysis:"
```

OUTPUT "Total Positions: " + totalPositions
OUTPUT "Average Salary: \$" + AVERAGE_SALARY (formatted to two decimal places)
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 statement END IF		<pre>IF num_dogs > 10 THEN</pre>			
SELECT variable or expression CASE value_1: statement CASE value_2: statement statement CASE value_2: statement CASE value_2: statement Statement DEFAULT: statement statement statement Statement Statement Statement Statement Statement 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!" 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	<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		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 RETURN value function		RETURN 2 + 3						