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:

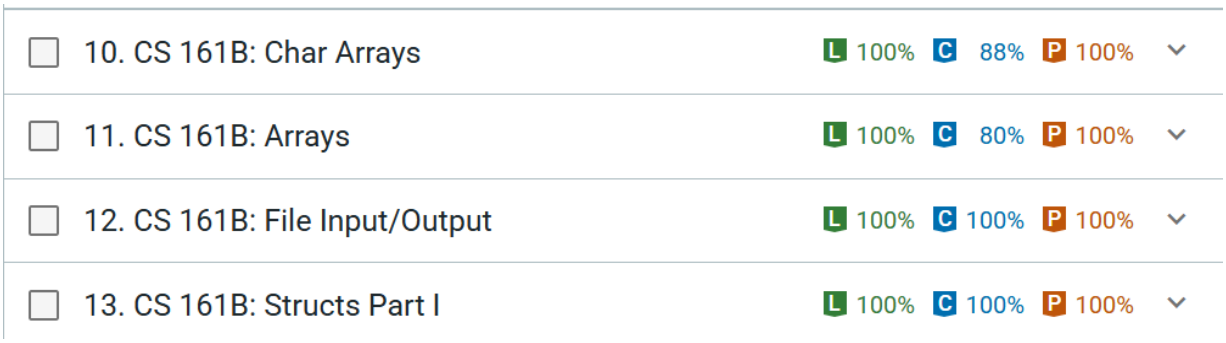
- ❏ Paste a screenshot of your zyBooks Challenge and Participation %
- ❏ Paste a screenshot of your assigned zyLabs completion
- ❏ Write a detailed description of your program, at least two complete sentences
- ❏ If applicable, design a sample run with test input and output
- ❏ Identify the program inputs 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:



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 analyzes video game sales and provides insight into popular game genres by console. This will read data from an attached file and populate it into an array that can be updated to reflect changing numbers, and run functions to see most and least popular genres as well as which consoles they were most popular on.

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!
This program analyzes video game sales and provides insight into
popular game genres by console. You can Add, Remove, and Edit
entries as well as Print the dataset and see which genre was the
most popular and on what console, or quit the program.

Pick an option from below:
```

```
(A)dd entry
(R)emove entry
(E)dit entry
(P)rint data
(S)how most popular
(Q)uit

A
Please enter the name of the game (30 characters or less): Elden
Ring
Enter the genre : RPG
Enter the NA_Sales (in millions) : 30.0

Pick an option from below:

(A)dd entry
(R)emove entry
(E)dit entry
(P)rint data
(S)how most popular
(Q)uit

p

Name;Console;Genre;NA Sales
1.) Wii sports;Wii;Sports;41.49
2.) Super Mario Bros.;NES;Platform;29.08
3.) Mario Kart Wii;Wii;Racing;15.85
4.) Wii Sports Resort;Wii;Sports;15.75
5.) Elden Ring;PC;RPG,30.00


Pick an option from below:

(A)dd entry
(R)emove entry
(E)dit entry
(P)rint data
(S)how most popular
(Q)uit

S
```

```
The most popular genre was Sports on the Wii.
Total Sales: 57.24

The least popular genre was Racing on the Wii.
Total Sales: 15.85

Pick an option from below:

(A)dd entry
(R)emove entry
(E)dit entry
(P)rint data
(S)how most popular
(Q)uit


g
Invalid option!
Pick an option from below:

(A)dd entry
(R)emove entry
(E)dit entry
(P)rint data
(S)how most popular
(Q)uit

R

Which entry would you like to remove?

Name;Console;Genre;NA Sales
1.) Wii sports;Wii;Sports;41.49
2.) Super Mario Bros.;NES;Platform;29.08
3.) Mario Kart Wii;Wii;Racing;15.85
4.) Wii Sports Resort;Wii;Sports;15.75
5.) Elden Ring;PC;RPG,30.00

3

Entry removed. Here is your new list:

Name;Console;Genre;NA Sales
1.) Wii sports;Wii;Sports;41.49
```

```
2.) Super Mario Bros.;NES;Platform;29.08
3.) Wii Sports Resort;Wii;Sports;15.75
4.) Elden Ring;PC;RPG,30.00

Pick an option from below:

(A)dd entry
(R)emove entry
(E)dit entry
(P)rint data
(S)how most popular
(Q)uit

e

Which entry would you like to edit?

Name;Console;Genre;NA Sales
1.) Wii sports;Wii;Sports;41.49
2.) Super Mario Bros.;NES;Platform;29.08
3.) Wii Sports Resort;Wii;Sports;15.75
4.) Elden Ring;PC;RPG,30.00

1

Which field would you like to edit?

Name;Console;Genre;NA Sales
1.) Name
2.) Console
3.) Genre
4.) NA Sales

4

Please enter the new NA_Sales (in millions): 43.3

Entry edited. Here is your new list:

Name;Console;Genre;NA Sales
1.) Wii sports;Wii;Sports;43.30
2.) Super Mario Bros.;NES;Platform;29.08
3.) Wii Sports Resort;Wii;Sports;15.75
4.) Elden Ring;PC;RPG,30.00
```

```
Pick an option from below:

(A)dd entry
(R)emove entry
(E)dit entry
(P)rint data
(S)how most popular
(Q)uit

Q

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.

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 option; Char name[]; Char console[]; Char genre[]; Double naSales;

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).

Array of Games

```
Struct Game {
    Char name[];
    Char console[];
    Char genre[];
    Double naSales;
}
```

Double mostPopularSales;

Double leastPopularSales;

Char mostPopularGenre[];

Char leastPopularGenre[];

Char mostPopularConsole[];

Char leastPopularConsole[];

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.

Double mostPopularSales = sum of all naSales of games with genre = mostPopularGenre;

Double leastPopularSales = sum of all naSales of games with genre = leastPopularGenre;

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.

```
Int main(){

DECLARE Game games[];

DISPLAY "Welcome!"
DISPLAY "This program analyzes video game sales and provides
insight into popular game genres by console. You can Add, Remove,
and Edit entries as well as Print the dataset and see which genre
was the most popular and on what console, or quit the program."

do{
    displayMenu();
    switch(option){
    }
} while (option != 'Q' && option != 'q');

DISPLAY "Thank you for using my program!";


FUNCTION displayMenu
    DISPLAY "Pick an option from below:"

    DISPLAY "(A)dd entry"
    DISPLAY "(R)emove entry"
    DISPLAY "(E)dit entry"
    DISPLAY "(P)rint data"
    DISPLAY "(S)how most popular"
    DISPLAY "(Q)uit"


FUNCTION addEntry
    DISPLAY "Please enter the name of the game (30 characters or
less):"
    SET games[count].Name = INPUT;
    DISPLAY "Enter the genre :"
    SET games[count].Genre = INPUT;
    DISPLAY "Enter the NA_Sales (in millions) :"
    SET games[count].naSales = INPUT
    count++;

FUNCTION removeEntry
```

```
    DECLARE num
    DISPLAY “Which entry would you like to remove?”
    printData(games)
    INPUT num
    SET num -= 1;
    for(int i = num; i < count << i++){
        games[i] = games[i + 1];
    }
    count --;

FUNCTION editEntry
    DECLARE int num, index
    DISPLAY “Which entry would you like to edit?”
    printData(games)
    INPUT index
    SET num -=1;
    DISPLAY “Which field would you like to edit?”

    DISPLAY “Name;Console;Genre;NA Sales”
    DISPLAY “1.) Name”
    DISPLAY “2.) Console”
    DISPLAY “3.) Genre”
    DISPLAY “4.) NA Sales”

    INPUT num;

    DISPLAY “Please enter the new {ValueToBeUpdated}”
    SET games[index].SelectedValueToBeUpdated = cin

    DISPLAY “Entry edited. Here is your new list:”

    printData(games);

FUNCTION printData
    DISPLAY “Name;Console;Genre;NA Sales”;
    for(int i = 0; i < count; i++){
        DISPLAY games[i].Name + “;”
        DISPLAY games[i].Console + “;”
        DISPLAY games[i].Genre + “;”
        DISPLAY games[i].naSales  + “\n”
    }

FUNCTION analyzeData
    DECLARE Double mostPopularSales, leastPopularSales
```

```
    DECLARE char mostPopularGenre[], leastPopularGenre[],
mostPopularConsole[], leastPopularConsole[]
    Char genres[];
    Int count[];
    For (int i = 0; i < count; i++)
        For (int j = 0; j < sizeof(genres); j++)
            if(games[i].Genre = games[j].Genre)
                Count[j] += 1;
            Else
                Count[j] = 1;
    For (int i = 0; i < sizeof(genres); i++){
        For (int j = i + 1; j < sizeof(genres); j++)
            if(count[j] > count[i])
                mostPopularGenre = genre[j]
    SET Double mostPopularSales = sum of all naSales of games with
genre = mostPopularGenre;

    SET Double leastPopularSales = sum of all naSales of games with
genre = leastPopularGenre;


    DISPLAY "The most popular genre was " + mostPopularGenre + " on
the " + mostPopularConsole + "."
    DISPLAY "Total Sales: " + mostPopularSales;

    DISPLAY "The least popular genre was " + leastPopularGenre + "
on the " + leastPopularConsole + "."
    DISPLAY "Total Sales: " + leastPopularSales;
```

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	SET	`SET num_dogs = num_dogs + 1`

variable		
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* = *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`