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.

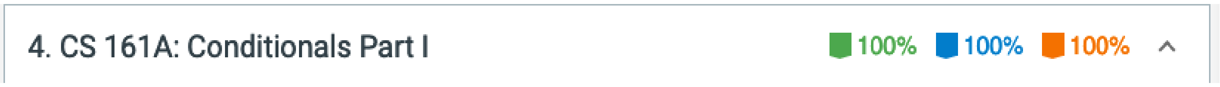
This document contains an interactive checklist. To mark an item as complete, click on the box (the entire list will be highlighted), then right click (the clicked box will only be highlighted), and choose the checkmark.

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

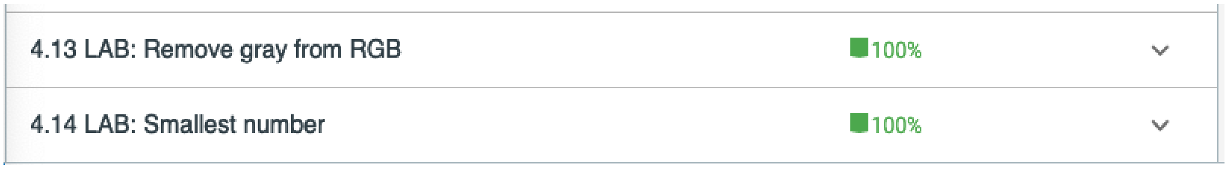
- ❏ Paste a screenshot of your zyBooks Challenge and Participation %
- ❏ Paste a screenshot of your assigned zyLabs completion
- ❏ Write a detailed description of your program, at least two complete sentences
- ❏ If applicable, design a sample run with test input and output
- ❏ Identify the program inputs 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:
 4. CS 161A: Conditionals Part I 🟩 100% 🟦 100% 🟧 100%

Assigned zyLabs completion screenshot:



4.13 LAB: Remove gray from RGB 🟩100%

4.14 LAB: Smallest number 🟩100%

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 presents age and cost information for riding trimet, then accepts input from the user to select a type of ticket. Based on the previous number of tickets purchased this month, the program then either presents a free to ride ticket message, or a remaining balance required to earn a free to ride ticket.

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 TriMet Hop Fastpass!

Fastpass Choices                        Ticket $
--------------------------------------  --------
A. Adult (ages 18-64)                    $2.50
H. Honored Citizen (65+, disabilities)   $1.25
Y. Youth (ages 7-17)                     $1.25

Note: Ride for 2 hours and 30 minutes with each ticket.

Ride free for the rest of the month after spending $100
with an Adult pass, or $28 with an Honored Citizen or
Youth pass!

Enter Fastpass (A, H, Y): h
Enter the number tickets purchased this month: 1

You have purchased 1 ticket!
You have paid: $ 1.25
Spend $ 26.75 more to earn free rides for the rest of the month!

Thank you for riding TriMet!
```

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.
passType, char: A-a,H-h,Y-y ticketCount: int, >= 0
b. Identify and list all of the user output and their data types.
freeRides: bool totalSpend: double, equation, positive amountToFree: double, equation, >= 0
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.

totalSpend = ticketCount * ticketPrice
amountToFree = (28.00 - totalSpend) || (100.00 - totalSpend)

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.

```
DECLARE double youthPrice = 1.25, adultPrice = 2.50, seniorSpend = 1.25, ticketPrice,
totalSpend, amountToFree

DECLARE int ticketCount

DECLARE bool freeRides

DECLARE char passType



OUTPUT “Welcome to TriMet Hop Fastpass!


Fastpass Choices                        Ticket $
--------------------------------------  --------
A. Adult (ages 18-64)                    $2.50
H. Honored Citizen (65+, disabilities)   $1.25
Y. Youth (ages 7-17)                     $1.25

Note: Ride for 2 hours and 30 minutes with each ticket.

Ride free for the rest of the month after spending $100
with an Adult pass, or $28 with an Honored Citizen or
Youth pass!

Enter Fastpass (A, H, Y):”

SELECT passType
  CASE A: CASE a:
    ticketPrice = adultPrice
  CASE H: CASE h:
    ticketPrice = seniorPrice
  CASE Y: CASE y:
    ticketPrice = youthPrice
  DEFAULT:
    OUTPUT “Invalid Fastpass. Try again”
    exit(1)
END SELECT

OUTPUT “Enter the number tickets purchased this month: “

INPUT  ticketCount
```

```
SET totalSpend = ticketCount * ticketPrice

IF ticketPrice < 2.50 && totalSpend >= 28.00 THEN
  freeRides = true
ELSE IF ticketPrice >= 2.50 && totalSpend >= 100.00
  freeRides = true
ELSE
  freeRides = false
END IF

IF ticketCount == 1 THEN
  OUTPUT “You have purchased <ticketCount> ticket”
ELSE
  OUTPUT “You have purchased <ticketCount> tickets”
END IF

OUTPUT “YOU HAVE PAID $ <totalSpend / 1.00>”

IF freeRides == false && ticketPrice < 2
  amountToFree = 28.00 - totalSpend
  OUTPUT “Spend $<amountToFree> more to earn free rides for the rest of the month!”
ELSE IF freeRides == false && ticketPrice >=2.50
  amountToFree = 100.00 - totalSpend
  OUTPUT “Spend $<amountToFree> more to earn free rides for the rest of the month!”
ELSE IF freeRides == true
  OUTPUT “You have earned free rides for the rest of the month!”
ELSE
  OUTPUT “Something went wrong. Please contact the system admin”
END IF

OUTPUT “Thank you for riding Trimet”
```

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* = *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`