CS 161A: Programming and Problem Solving I

Assignment A07 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.

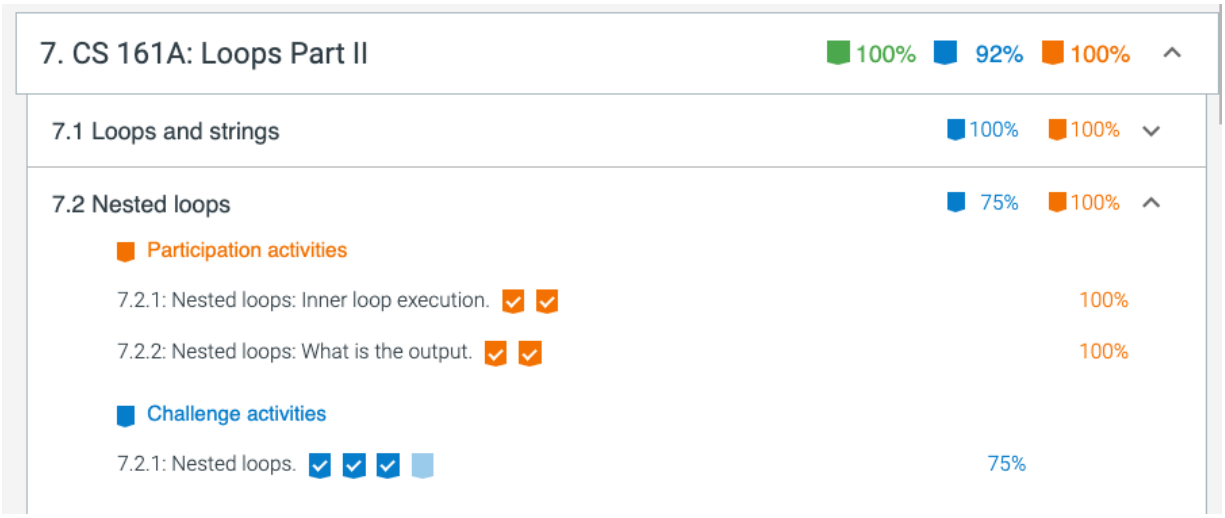
Planning your program before you start coding is part of the development process. In this document you will:

- ❏ 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%.

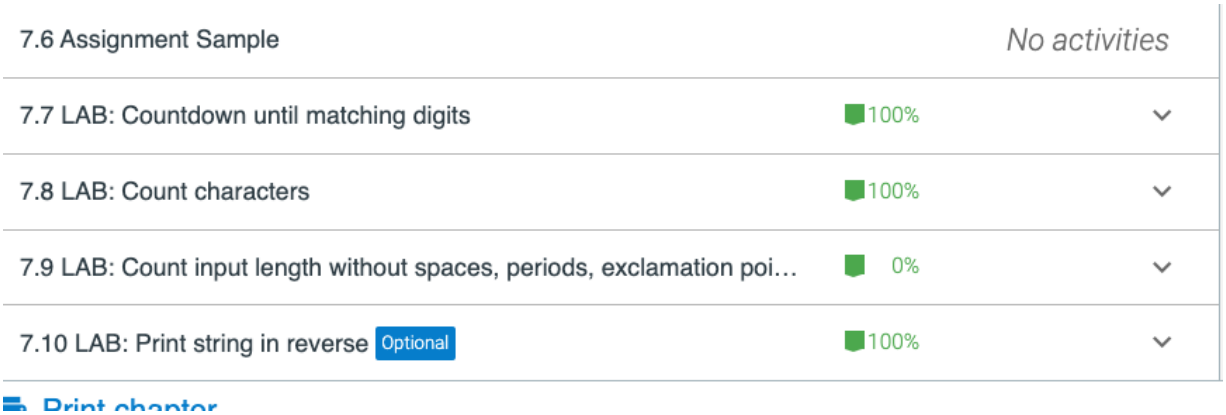
Challenge and Participation % screenshot:



I must have done that final part of 7.2.1 Challenge Activity 7 times over and over (and correctly) but nothing will mark the final level. I even started it before today and tried again today to see. Unsure if it is a bug but I know I have to mess with Zybooks a lot due to bugs sometimes, like re-taking lots of activities/labs but this one is stuck on “OFF” it seems.

Assigned zyLabs completion screenshot:

The 7.9 ZyLABs said hidden by the instructor by the way.



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 will be a vending machine where the user tells how they pay in coins, and then it will print a menu for dispensing coffee or tea to the user to choose from, and then a prompt for quantity. If the money deposited is enough, you will receive the item or it will tell you ask you for more cash.

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 Gina's Coffee/Tea Vending Machine!

Enter coins - 5, 10, or 25 only: 5
Enter coins - 5, 10, or 25 only: 25
Enter coins - 5, 10, or 25 only: 25
Enter coins - 5, 10, or 25 only: 5
Enter coins - 5, 10, or 25 only: 10
Enter coins - 5, 10, or 25 only: 10
Enter coins - 5, 10, or 25 only: 0

Your balance is $0.80

Please pick an option ($0.25 each):
    C/c: Coffee
    T/t: Tea
    Q/q: Quit
>> k
Invalid Option! Please choose a valid option!
>> 9
Invalid Option! Please choose a valid option!
>> c

How many would you like?
>> f
Invalid Option!
How many would you like?
>> 2

Your total: $0.50
Your balance: $0.30
```

```
Please pick an option ($0.25 each):
C/c: Coffee
T/t: Tea
Q/q: Quit
>> q

Your total is $0
Your balance is $0.30
Happy Snacking!
```

```
Welcome to Gina's Coffee/Tea Vending Machine!

Enter coins - 5, 10, or 25 only: 5
Enter coins - 5, 10, or 25 only: 25
Enter coins - 5, 10, or 25 only: 0

Your balance is $0.30

Please pick an option ($0.25 each):
    C/c: Coffee
    T/t: Tea
    Q/q: Quit
>> c

How many would you like?
>> 2

Your total is $0.50
Your balance is $0.30
Not enough change!! Please add more coins.

Enter coins - 5, 10, or 25 only: 5
Enter coins - 5, 10, or 25 only: 25
Enter coins - 5, 10, or 25 only: 0

Your balance: $0.60

Please pick an option ($0.25 each):
    C/c: Coffee
    T/t: Tea
    Q/q: Quit
>> T

How many would you like?
>> 1

Your total is $0.25
Your balance is $0.35
```

```
Happy Snacking!
```

```
Welcome to Gina's Coffee/Tea Vending Machine!

Enter coins - 5, 10, or 25 only: 5
Enter coins - 5, 10, or 25 only: 25
Enter coins - 5, 10, or 25 only: 0

Your balance is $0.30

Please pick an option ($0.25 each):
    C/c: Coffee
    T/t: Tea
    Q/q: Quit
>> q

Your total is $0
Your balance is $0.30

Happy Snacking!
```

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.
○ **coin** (integer) ○ **choice** (char) ○ **quantity** (integer)
b. Identify and list all of the user output and their data types.
○ **balance** (double) ○ **total** (double)
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.

- **balance** = **balance** + **coin** / 100 [to convert cents to dollar bills]
- **total** = **COST_PER_ITEM** * **quantity** [to display the user's total due]
- **balance** = **balance** - **total** [to reset the balance for display]

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.

1. **DISPLAY** Welcome Message and program information as shown in Sample Run
2. **DO** the following:
    a. **DISPLAY** coin message for user to input amount of coins
    b. **INPUT** **coin**
    c. **WHILE** **coin** is NOT 5 & 10 & 25 & 0
        i. **DISPLAY** invalid message
        ii. **INPUT** clear()
        iii. **INPUT ignore characters up to newline**
        iv. **INPUT** **coin**
    d. **SET** **balance** = **balance** + **coin**
3. **WHILE** **coin** is NOT 0 | end **DO WHILE** loop |
4. **DISPLAY** fixed << setprecision(2) to set decimal places for money/cents 0.00

    // menu & money status below

5. **DISPLAY** new line
6. **WHILE** = true:
    a. **DISPLAY** **balance** and menu below

```
Please pick an option ($0.25 each):
    1. C/c: Coffee
    2. T/t: Tea
    3. Q/q: Quit
      >>
```

    b. **INPUT** **choice**
    c. **IF** **choice** is Q or q
        i. **DISPLAY** your total message + **balance**
        ii. **BREAK** program
    d. END IF
    e. **IF** **choice** is NOT C,c,T,t,Q,q
        i. **DISPLAY** invalid error
        ii. **continue program**
    f. END IF
    g. **DISPLAY** prompt for quantity of selection chosen
    h. **INPUT** **quantity**
    i. **WHILE** input is NOT empty or **quantity** is less than 1

i. **DISPLAY** invalid error, re-prompt user
ii. cin.clear()
iii. **INPUT ignore characters up to new line**
iv. **INPUT quantity**

j. **END WHILE**
k. **SET total = COST_PER_ITEM * quantity**
l. **IF total > balance**
i. **DISPLAY** total message + **total** + **balance** + prompt to add more coins
ii. **SET balance = balance - total**
iii. **DISPLAY** total message + **total** + **balance**
iv. **BREAK** program
v. DO
1. DISPLAY enter coins prompt
2. INPUT coin
3. WHILE coin IS NOT 5, 10, 25, or 0
a. DISPLAY invalid coin message, prompt again
b. clear input stream
c. ignore input stream
d. INPUT coin
4. END WHILE
5. SET balance += coin / 100.0

vi. WHILE coin is not 0
1. **DISPLAY** total balance message + balance

m. ELSE
i. SET balance = balance - total
ii. DISPLAY total message + total + balance message + balance

n. **END IF**

7. **END WHILE**
8. **DISPLAY** “Happy Snacking!”
9. **END 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	INPUT	`INPUT num_dogs`

into a variable		
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`