CS 161A: Programming and Problem Solving I

Discussion 7

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. My Links

Challenge and Participation % 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 is a palindrome detector. Take an input and check whether it is the same backwards and forwards (ignoring spaces and capitalization). It will then output a response as either yes a palindrome, or not a palindrome.

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:

```
Type in a word or phrase: Corinne Enniroc

Corinne Enniroc is a palindrome!

-

Type in a word or phrase: Corinne Corinne

Corinne Corinne is not a palindrome!
```

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.

userString (string): the user input of word or phrase

Copy (string): copy of the user input, to be displayed exactly as entered

b. Identify and list all of the user output and their data types.

isPal (bool): determination of whether (true) or not (false) the userString is a palindrome

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.

1) Remove any spaces, by finding and replacing " " with "":

    While- statement to detect whether any spaces present

        Find " ", and assign spaceIndex = userString.find(" ")

        At index value, userString.replace(spaceIndex, 1, "")

2) Make everything lowercase

    Length = userString.size()

    For-statement to cycle through every index of userString

        At index i, tolower function at i: userString.at(i) = tolower(userString.at(i));

3) Detect if bool variable is true or false, given the input

    For-statement to cycle through every index

        If- statement to determine if char at first character exactly equals char at last index, using i from start to length - i

            isPal = true

    Else

        isPal = false

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.

INCLUDE iostream, iomanip, string, cctype

DECLARE string userString

DECLARE string copy

DECLARE bool isPal, initialize as false

DECLARE integer spaceIndex

DECLARE integer length

```
DISPLAY "Type in a word or phrase:"

INPUT userString

SET copy = userString


WHILE userString contains " " (space)

    SET spaceIndex = userString.find(" ");

    SET userString.replace(spaceIndex, 1, "");

END WHILE


SET length = userString.size()

FOR counter: 0 to length

    Make character at i is assigned to a lowercase

END FOR


FOR counter: 1 to and including length

    IF userString char at i - 1 exactly equals length - 1 THEN

        SET isPal = true

    ELSE

        SET isPal = false

    END IF

END FOR


IF isPal is true THEN

    DECLARE input is a palindrome!

ELSE
```

> DECLARE input is NOT a palindrome!
>
> END IF

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 ```