C, 110

Student ID: [redacted]

Air University
Final Examination: Fall 2024

Section II: Subjective
(To be solved on Answer Books only)

Subject: Programming Fundamentals
Class: BS-CyS Fall 2024 (Eve)
Section(s): A & B
Course Code: CS111

Time Allowed: 3 Hours
Max Marks: 100
FM's Name: Hilmand Khan
FM's Signature: [illegible]

INSTRUCTIONS

- Attempt responses on the answer book only.
- Nothing is to be written on the question paper.
- Rough work or writing on question paper will be considered as use of unfair means.
- Tables / calculators are not allowed.

Q1.	CLO 1	**Briefly describe the following.** 2 a)Recursion 2 b)Default Arguments 2 c)Call by value & Call by reference d)Double Pointer e)Function Overloading	**Marks (10)**
Q2.	CLO 1	**Write a brief answer to the following questions.** a)How do you concatenate two strings in C++? b)How do you convert a string to uppercase or lowercase in C++? c)How do you find the length of a string in C++? d)How do you access individual characters in a string in C++? e)How do you count the number of a character appearance in a string in C++?	**Marks (10)**
Q3.	CLO 2	a)Create a function that counts the number of syllables a word has. Each syllable is separated with a dash -. numberSyllables("buf-fet") → 2 numberSyllables("beau-ti-ful") → 3 numberSyllables("mon-u-men-tal") → 4 numberSyllables("on-o-mat-o-poe-ia") → 6 **Don't forget to return the result**	**Marks (20)**

b) Sam and Frodo need to be close. If they are side by side in the array, your function should return true. If there is a name between them, return false.

middleEarth(["Frodo", "Sam", "Gandalf"]) → true
middleEarth(["Frodo", "Saruman", "Sam"]) → false
middleEarth(["Orc", "Sam", "Frodo", "Legolas"]) → true

c) A **set** is a collection of unique items. A **set** can be formed from an array by removing all duplicate items. Create a function that sorts an array and removes all duplicate items from it.

set([1, 3, 3, 5, 5]) → [1, 3, 5]
set([4, 4, 4, 4]) → [4]
set([5, 7, 8, 9, 10, 15]) → [5, 7, 8, 9, 10, 15]
set([3, 3, 3, 2, 1]) → [1, 2, 3]

d) Write a function that replaces all vowels in a string with a specified vowel.

vowReplace("apples and bananas", "u") → "upplus und bununus"
vowReplace("cheese casserole", "o") → "chooso cossorolo"
vowReplace("stuffed jalapeno poppers", "e") → "steffed jelepene peppers"

Q4. **CLO 3** **Solve any two of the following:** **Marks (30)**

a) Create a function that performs an even-odd transform to an array, n times. Each even-odd transformation:
1. Adds two (+2) to each odd integer.
2. Subtracts two (-2) from each even integer.

evenOddTransform([3, 4, 9], 3) → [9, -2, 15]
// Since [3, 4, 9] => [5, 2, 11] => [7, 0, 13] => [9, -2, 15]
evenOddTransform([0, 0, 0], 10) → [-20, -20, -20]
evenOddTransform([1, 2, 3], 1) → [3, 0, 5]

b) Given a list of directions to spin, "left" or "right", return an integer of how many full **360°** rotations were made. Note that each word in the array counts as a **90°** rotation in that direction.

spinAround(["left", "right", "left", "right"]) → 0
spinAround(["right", "right", "right", "right", "right", "right", "right", "right"]) → 2
spinAround(["left", "left", "left", "left"]) → 1

c) YouTube like and dislike button are set up in a way that you cannot like and dislike a video at the same time. There are two other interesting rules to be noted about the interface: Pressing

		a button, which is already active, will undo your press. If you press the like button after pressing the dislike button, the like button overwrites the previous "dislike" state. The same is true for the other way round. Create a function that takes in an array of button inputs and returns the final state. likeOrDislike(["Dislike"]) → "Dislike" likeOrDislike(["Like", "Like"]) → "Nothing" likeOrDislike(["Dislike", "Like"]) → "Like" likeOrDislike(["Like", "Dislike", "Dislike"]) → "Nothing"	

Q5.	**CLO 4**	**Solve one of the following:**
		
		a)Create a function that validates a password to conform to the following rules:
		Length between 6 and 24 characters.
		At least one uppercase letter (A-Z).
		At least one lowercase letter (a-z).
		At least one digit (0-9).
		Maximum of 2 repeated characters.
		"aa" is OK 👍
		"aaa" is NOT OK 👎
		Supported special characters:
		
		validatePassword("P1zz@") → false // Too short.
		validatePassword("iLoveYou") → false // Missing a number.
		validatePassword("Fhg93@") → true // OK!
		
		b)A consecutive-run is a list of adjacent, consecutive integers. This list can be either increasing or decreasing. Create a function that takes an array of numbers and returns the length of the longest consecutive-run.
		
		longestRun([1, 2, 3, 10, 11, 15]) → 3
		// Longest consecutive-run: [1, 2, 3].
		longestRun([5, 4, 2, 1]) → 2
		// Longest consecutive-run: [5, 4] and [2, 1].
		longestRun([3, 5, 7, 10, 15]) → 1
		// No consecutive runs, so we return 1.
			Marks (30)