PROGRAMMING FUNDAMENTALS (THEORY) [CS-102]

Behram Khan

Assignment # 05

* Try to solve them yourselves, if you try and get stuck and frustrated do another question and comeback to it another time.
* After trying to solve it but stuck then ask ChatGPT to give you hints on how to solve it rather than giving you complete code and it will help you develop problem solving skills.
* Try to do atleast 1-2 questions daily as they are difficult and you won’t be able to do them on night of duedate (many of you will be reading this on that day ☹)

1. **Reverse a String (Functions + Strings) (Hard Question)**

**Scenario: Create a program to reverse a string.  
Tasks:**

1. **It should ask user to input different inputs like "hello" and "world".**
2. **Write a function to reverse a given string.**
3. **The function should only reverse the string and we should display the reversed string in the main function. (use pass by reference method)**

**Try to solve it yourself and if you get stuck on it after trying for 2 days then refer to the video.**

**Try to develop your own problem-solving skills.**

[**https://www.youtube.com/watch?v=PtSHcou0WIs&ab\_channel=PortfolioCourses**](https://www.youtube.com/watch?v=PtSHcou0WIs&ab_channel=PortfolioCourses)

1. void reverseString(Parameter1) {
2. int len = strlen(str);
3. //Make Reverse logic here
5. }
6. int main() {
7. char str[100];
8. printf("Enter a string: ");
9. scanf("%99s", str);
10. reverseString(PassInput);
11. printf("Reversed String: %s\n", str);
12. return 0;
13. }

Why do we need **int len = strlen(str); ?** Explain

1. **Shopping Cart System (Functions + Arrays + Pointers) (Easy Question)**

**Scenario: Simulate a shopping cart system where users can add items and calculate the total cost.  
Tasks:**

1. **Use an array to store item prices.**
2. **Write functions to:**
   * **Add items to the cart.**
   * **Display the cart contents.**
   * **Calculate the total cost using pointers.**

**Sample Provided below:**

#include <stdio.h>

void addItem(float \*cart, int \*count, float price) {

//This function is used to add price of item to cart and keep track of number of

//items in cart

    //Make Logic here

}

float calculateTotal(Parameter1 ,Parameter 2) {

    float total = 0;

    for (int i = 0; i < ? ; i++) {

        total // Make logic ;

    }

    return total;

}

void displayCart(Parameter1 ,Parameter2 ) {

    printf("Cart Contents:\n");

    for (int i = 0; i < ?? ; i++) {

        printf("Item %d: $%.2f\n", i + 1, cart[i]);

    }

}

int main() {

    float cart[10]; // Storing item prices

    int count = 0; //keeping track of items in cart

    addItem( PassInput, PassInput, 10.99);

    addItem( PassInput, PassInput, 5.49);

    addItem( PassInput, PassInput, 20.75);

    displayCart(Input1,Input2);

    printf("Total: $%.2f\n", calculateTotal(cart, count));

    return 0;

}

1. **Matrix Operations (Functions + Arrays + Pointers) (Easy Question)**

**Scenario: Perform basic operations on matrices, such as addition.  
Tasks:**

1. **Use a 2D array to represent matrices.**
2. **Write function to:**
   * **Add two matrices.**
3. **Use pointers to pass the matrices to the functions.**
4. **If you don’t know how matrix addition works, refer to this video then** [**https://www.youtube.com/watch?v=QXUbFzEd3Ww&t=118s&ab\_channel=TheOrganicChemistryTutor**](https://www.youtube.com/watch?v=QXUbFzEd3Ww&t=118s&ab_channel=TheOrganicChemistryTutor)
5. #include <stdio.h>
6. int SIZE = 2; // Matrix size
7. void addMatrices(parameters to function) {
8. //Logic to sum the matrix
9. }
10. void displayMatrix(int mat[SIZE][SIZE]) {
11. for (int i = 0; i < SIZE; i++) {
12. for (int j = 0; j < SIZE; j++) {
13. //Logic to print 2d array
14. }
15. printf("\n");
16. }
17. }
18. int main() {
19. int mat1[2][2] = {{1, 2}, {3, 4}};
20. int mat2[2][2] = //Add 5,6,7,8 to this.
21. int result[SIZE][SIZE];
22. printf("Matrix Addition:\n");
23. addMatrices(Inputs to function, think how many inputs we need.);
24. displayMatrix(result);
25. return 0;
26. }

**Output:**

**Matrix Addition:**

**6 8**

**10 2**

1. **Count Vowels in a String (Functions + Strings + Pointers) (Intermediate Question)**

**Scenario: Write a program to count the number of vowels in a string.  
Tasks:**

1. **Write a function to check if a character is a vowel.**
2. **Write another function to count the vowels in a string using the first function.**

**Sample Input:**

**Enter a string: hi everyone**

**Number of vowels: 5**

#include <stdio.h>

#include <ctype.h>

int isVowel(char ch) {

    ch = tolower(ch); // Convert character to lowercase (from A to a)

    return ch == '?' || ch == '?' || ch == '?' || ch == '?' || ch == '?';

}

int countVowels(Parameter1) {

    int count = 0;

    while (\*str != /\*What will be here (what’s at end of a string array?)\*/ )

{ // Loop until the end of the string

//Make Logic

    }

    return count;

}

int main() {

    char str[100];

    printf("Enter a string: ");

    scanf("%99s", str); // Use %99s to prevent buffer overflow (google/chatgpt //about %99s)

    int vowels = countVowels(Input);

    printf("Number of vowels: %d\n", vowels);

    return 0;

}

**Explain why we use in this prorgam ‘ \0 ’ ?**

**Draw call stack for this program showing which method is being pushed and when is it being popped.**

1. **Implement a recursive function in C that prints the numbers 1 through 5, one number per recursion. The main function should call this recursive function to start the process. (Easy Question)**

#include <stdio.h>

void printRecursion(int count) {

//This function should print 1 in the first recursion, 2 in second, 3 in third ..

//Make BaseCase

//Make Recursive Call

}

int main() {

    printf("Printing recursion numbers up to 5:\n");

    printRecursion(1); // Start recursion with 1

    return 0;

}

**Output:**

**Printing recursion numbers up to 5:**

**Recursion number: 1**

**Recursion number: 2**

**Recursion number: 3**

**Recursion number: 4**

**Recursion number: 5**

**6. Write a program that uses recursion to print an equilateral triangle of a given size. (Hard Question)**

**Note this video is in Java but try to understand the concept/logic he is doing.**

**System.out.println() is equivalent to Printf() of C.**

[**https://www.youtube.com/watch?v=EeS4a0mpXRQ&ab\_channel=TomG**](https://www.youtube.com/watch?v=EeS4a0mpXRQ&ab_channel=TomG)

**This video solves a right triangle pattern but your task is to print an equilateral triangle.**

**Try to first solve this problem using for loops (you have done this in previous assignment) and see how the patterns are printing and what should be the base condition and what should we pass to the recursive call.**

#include <stdio.h>

void printTriangle(int rows, int current) {

//Write Base Case here

    // Print spaces for alignment

    for (int i = 0; i < rows - current; i++) {

        printf(" ");

    }

    // Print stars for the current row here

    printTriangle(Passing modified values to function); // Recursive call for the next row

}

int main() {

    int size;

    printf("Enter the size of the equilateral triangle: ");

    scanf("%d", &size);

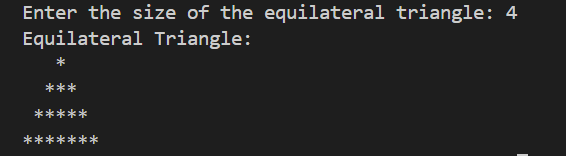
    printf("Equilateral Triangle:\n");

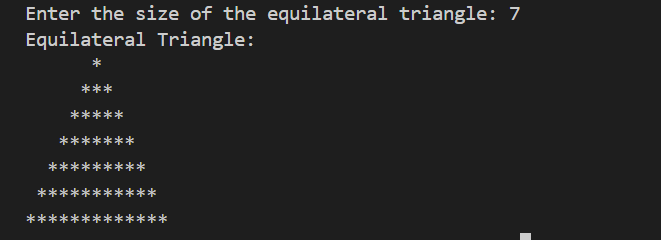
    printTriangle(size, 1); // Start printing from the first row

    return 0;

}

**Output:**





**Question 7. (Hard Question)**

**Watch this for sizeof(arr)/sizeof(arr[0]) explaination**

[**https://www.youtube.com/watch?v=iBFzKvCzXsw&ab\_channel=NesoAcademy**](https://www.youtube.com/watch?v=iBFzKvCzXsw&ab_channel=NesoAcademy)

**This question is hard, take your time doing it, if you get stuck and frustrated you should stop and come back to it after sometime or the next day.**

**Use these videos to understand the below problem:**

[**https://www.youtube.com/watch?v=e-lA6W7GKng&ab\_channel=edSlash**](https://www.youtube.com/watch?v=e-lA6W7GKng&ab_channel=edSlash)

[**https://www.youtube.com/watch?v=wVArsfX4WyQ&ab\_channel=AbhishekSensei**](https://www.youtube.com/watch?v=wVArsfX4WyQ&ab_channel=AbhishekSensei)

**Write a C function called removeElement that takes an integer array nums, its size numsSize, and an integer value val as input. The function should remove all occurrences of val from the array *in-place* (i.e., by modifying the original array). The remaining elements should be shifted to the beginning of the array, and the remaining space should be filled with zeros. The main function should demonstrate the usage of removeElement by initializing an array, calling the function, and then printing the modified array.**

**For example:**

**If nums = {3, 2, 2, 3, 4, 5, 2} and val = 2, after calling removeElement,**

**nums should become {3, 3, 4, 5, 0, 0, 0}. All Occurances of 2 are removed and rest of elements are moved to left and 2’s are replaced with 0’s.**

**Write a main function to test your removeElement function and print the modified array.**

#include<stdio.h>

void removeElement(int \*nums, int numsSize, int val) {

    int count = 0;

    for (int i = 0; i < numsSize; i++) {

        //Make Logic Here

}

int main() {

    int nums[] = {3, 2, 2, 3, 4, 5, 2};

    int numsSize = sizeof(nums) / sizeof(nums[0]);

    int val = 2;

    removeElement(nums, numsSize, val);

    printf("Updated array: ");

    for (int i = 0; i < numsSize; i++) {

        printf("%d ", nums[i]);

    }

    printf("\n");

    return 0;

}

**Question 8. (Intermediate Question)**

**We have learned the concept of modular programming and learned that functions are used to divide our program into small sub programs performing specific tasks meaning we divide our program into small logical parts all working together.**

**Below is Treasure hunter game code, divide this project into functions so that we separate it into logical parts (those part which are performing a specific task will be converted into a function).**

* **Remember to use pointers (pass by reference) as we want functions to perform tasks but also reflect back changes into main function.**
* **You should use global variable for Height and Width, so they can be accessed by all functions.**
* **if you get error that music file is not present, you can download it from project files uploaded in Google Classroom or just remove the code for music.**

**I have performed one for you:**

**They drawGrid function is used to display the grid, So I have separated its code. Its performing a specific task of printing the grid.**

**Do similar for the rest of the code.**

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#include <time.h>

void drawGrid(int playerX, int playerY, int coinX, int coinY);

int main() {

    system("start /min PrepareForBattle.wav");

    int score = 0;

    const int WIDTH = 16;

    const int HEIGHT = 7;

    int playerX = WIDTH / 2;

    int playerY = HEIGHT / 2;

    int flag = 1;

    srand(time(0));

    int coinX = rand() % WIDTH;

    int coinY = rand() % HEIGHT;

    time\_t startTime = time(0); // Start time

    char input;

    while (flag == 1) {

        system("cls");

        // Check if player collected the coin

        if (playerX == coinX && playerY == coinY) {

            printf("You collected the coin!\n");

            score++;

            coinX = rand() % WIDTH;

            coinY = rand() % HEIGHT;

        }

        // Draw the grid

        drawGrid(playerX, playerY, coinX, coinY);

        // Display elapsed time

        printf("Time elapsed: %ld seconds\n", time(0) - startTime);

        printf("Score: %d\n", score);

        input = \_getch();

       switch (input) {

            case 'w':

                if (playerY > 0)

                playerY--;

                break;

            case 's':

                if (playerY < HEIGHT - 1)

                playerY++;

                break;

            case 'a':

                if (playerX > 0)

                playerX--;

                break;

            case 'd':

                if (playerX < WIDTH - 1)

                playerX++;

                break;

            case 'q':

                flag = 0;

                system("taskkill /IM Microsoft.Media.Player.exe /F");

                break;

            default:

                printf("Invalid input. Please try again.\n");

                \_getch(); // Consume the invalid input

                break;

        }

    }

    return 0;

}

void drawGrid(int playerX, int playerY, int coinX, int coinY) {

    for (int i = 0; i < HEIGHT; i++) {

        for (int j = 0; j < WIDTH; j++) {

            if (i == playerY && j == playerX) {

                printf("@");

            } else if (i == coinY && j == coinX) {

                printf("O");

            } else {

                printf(".");

            }

        }

        printf("\n");

    }

}