

LPCC ASSIGNMENT

WITH THE HELP OF CHATGPT



QUESTION 1. C Program to take 5 values from the users and store them in an array.

#include <stdio.h>

int main() {

int values[5]; // declare an integer array of size 5

int i;

printf("Enter 5 integer values:\n");

// loop to read values from user input and store them in the array

for(i = 0; i < 5; i++) {

printf("Value %d: ", i+1);

scanf("%d", &values[i]);

}

// print the values stored in the array

printf("Values stored in the array are:\n");

for(i = 0; i < 5; i++) {

printf("%d\n", values[i]);

}

return 0;

}

QUESTION 2. C Program to print array elements.

#include <stdio.h>

int main() {

int arr[] = {10, 20, 30, 40, 50}; // declare an integer array

int i;

printf("Elements of the array are:\n");

// loop to print the elements of the array

for(i = 0; i < 5; i++) {

printf("%d\n", arr[i]);

}

return 0;

}

QUESTION 3. C Program to storing elements in a matrix and printing it.

#include <stdio.h>

int main() {

int rows, columns, i, j;

int matrix[100][100]; // declare a 2D integer array

// prompt the user to enter the number of rows and columns

printf("Enter the number of rows and columns of the matrix: ");

scanf("%d %d", &rows, &columns);

// prompt the user to enter the elements of the matrix

printf("Enter the elements of the matrix:\n");

for(i = 0; i < rows; i++) {

for(j = 0; j < columns; j++) {

printf("Element (%d,%d): ", i+1, j+1);

scanf("%d", &matrix[i][j]);

}

}

// print the elements of the matrix

printf("The matrix is:\n");

for(i = 0; i < rows; i++) {

for(j = 0; j < columns; j++) {

printf("%d ", matrix[i][j]);

}

printf("\n");

}

return 0;

}

OUTPUT

Enter the number of rows and columns of the matrix: 2 3

Enter the elements of the matrix:

Element (1,1): 1

Element (1,2): 2

Element (1,3): 3

Element (2,1): 4

Element (2,2): 5

Element (2,3): 6

The matrix is:

1 2 3

4 5 6

QUESTION 4. C Program to multiplication of two matrix.

#include <stdio.h>

int main() {

int m, n, p, q, i, j, k;

int matrix1[100][100], matrix2[100][100], result[100][100];

// prompt the user to enter the dimensions of the matrices

printf("Enter the number of rows and columns of matrix 1: ");

scanf("%d %d", &m, &n);

printf("Enter the number of rows and columns of matrix 2: ");

scanf("%d %d", &p, &q);

// check if the matrices can be multiplied

if(n != p) {

printf("Matrices cannot be multiplied\n");

return 0;

}

// prompt the user to enter the elements of matrix 1

printf("Enter the elements of matrix 1:\n");

for(i = 0; i < m; i++) {

for(j = 0; j < n; j++) {

printf("Element (%d,%d): ", i+1, j+1);

scanf("%d", &matrix1[i][j]);

}

}

// prompt the user to enter the elements of matrix 2

printf("Enter the elements of matrix 2:\n");

for(i = 0; i < p; i++) {

for(j = 0; j < q; j++) {

printf("Element (%d,%d): ", i+1, j+1);

scanf("%d", &matrix2[i][j]);

}

}

// multiply the matrices

for(i = 0; i < m; i++) {

for(j = 0; j < q; j++) {

result[i][j] = 0;

for(k = 0; k < n; k++) {

result[i][j] += matrix1[i][k] \* matrix2[k][j];

}

}

}

// print the result matrix

printf("Result matrix is:\n");

for(i = 0; i < m; i++) {

for(j = 0; j < q; j++) {

printf("%d ", result[i][j]);

}

printf("\n");

}

return 0;

}

QUESTION 5. C Program to perform sum of two matrix.

#include <stdio.h>

int main() {

int m, n, i, j;

int matrix1[100][100], matrix2[100][100], sum[100][100];

// prompt the user to enter the dimensions of the matrices

printf("Enter the number of rows and columns of the matrices: ");

scanf("%d %d", &m, &n);

// prompt the user to enter the elements of matrix 1

printf("Enter the elements of matrix 1:\n");

for(i = 0; i < m; i++) {

for(j = 0; j < n; j++) {

printf("Element (%d,%d): ", i+1, j+1);

scanf("%d", &matrix1[i][j]);

}

}

// prompt the user to enter the elements of matrix 2

printf("Enter the elements of matrix 2:\n");

for(i = 0; i < m; i++) {

for(j = 0; j < n; j++) {

printf("Element (%d,%d): ", i+1, j+1);

scanf("%d", &matrix2[i][j]);

}

}

// add the matrices

for(i = 0; i < m; i++) {

for(j = 0; j < n; j++) {

sum[i][j] = matrix1[i][j] + matrix2[i][j];

}

}

// print the sum matrix

printf("Sum of the matrices is:\n");

for(i = 0; i < m; i++) {

for(j = 0; j < n; j++) {

printf("%d ", sum[i][j]);

}

printf("\n");

}

return 0;

}

QUESTION 6. C Program to perform repetition of Array

#include <stdio.h>

int main() {

int arr[100], n, i, j;

// prompt the user to enter the number of elements in the array

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

// prompt the user to enter the elements of the array

printf("Enter the elements of the array:\n");

for(i = 0; i < n; i++) {

printf("Element %d: ", i+1);

scanf("%d", &arr[i]);

}

// prompt the user to enter the repetition factor

printf("Enter the repetition factor: ");

scanf("%d", &j);

// print the original array

printf("Original array:\n");

for(i = 0; i < n; i++) {

printf("%d ", arr[i]);

}

printf("\n");

// repeat the array j times

printf("Repeated array:\n");

for(i = 0; i < n\*j; i++) {

printf("%d ", arr[i%n]);

}

printf("\n");

return 0;

}

QUESTION 7. C Program to perform program to find the row with maximum number of one’s.

#include <stdio.h>

#define MAX\_ROWS 10

#define MAX\_COLS 10

int main() {

int matrix[MAX\_ROWS][MAX\_COLS];

int rows, cols, i, j;

int max\_row = -1, max\_count = -1;

// prompt the user to enter the size of the matrix

printf("Enter the number of rows and columns of the matrix: ");

scanf("%d %d", &rows, &cols);

// prompt the user to enter the elements of the matrix

printf("Enter the elements of the matrix:\n");

for(i = 0; i < rows; i++) {

for(j = 0; j < cols; j++) {

scanf("%d", &matrix[i][j]);

}

}

// find the row with the maximum number of ones

for(i = 0; i < rows; i++) {

int count = 0;

for(j = 0; j < cols; j++) {

if(matrix[i][j] == 1) {

count++;

}

}

if(count > max\_count) {

max\_count = count;

max\_row = i;

}

}

// print the row with the maximum number of ones

printf("Row %d has the maximum number of ones: %d\n", max\_row+1, max\_count);

return 0;

}

QUESTION 8. C Program to rotate a matrix by 90 degree clockwise.

#include <stdio.h>

#define MAX\_ROWS 10

#define MAX\_COLS 10

int main() {

int matrix[MAX\_ROWS][MAX\_COLS];

int rows, cols, i, j;

// prompt the user to enter the size of the matrix

printf("Enter the number of rows and columns of the matrix: ");

scanf("%d %d", &rows, &cols);

// prompt the user to enter the elements of the matrix

printf("Enter the elements of the matrix:\n");

for(i = 0; i < rows; i++) {

for(j = 0; j < cols; j++) {

scanf("%d", &matrix[i][j]);

}

}

// rotate the matrix by 90 degrees clockwise

int rotated\_matrix[MAX\_COLS][MAX\_ROWS];

for(i = 0; i < cols; i++) {

for(j = 0; j < rows; j++) {

rotated\_matrix[i][j] = matrix[rows-1-j][i];

}

}

// print the rotated matrix

printf("Rotated matrix:\n");

for(i = 0; i < cols; i++) {

for(j = 0; j < rows; j++) {

printf("%d ", rotated\_matrix[i][j]);

}

printf("\n");

}

return 0;

}

OUTPUT

Enter the number of rows and columns of the matrix: 3 4

Enter the elements of the matrix:

1 2 3 4

5 6 7 8

9 10 11 12

Rotated matrix:

9 5 1

10 6 2

11 7 3

12 8 4

QUESTION 9. Write a C Program of 1D Array and 2D array.Combine them in one separate matrix.

#include <stdio.h>

int main() {

int arr1[5] = {1, 2, 3, 4, 5}; // 1D array

int arr2[2][3] = {{1, 2, 3}, {4, 5, 6}}; // 2D array

int matrix[3][5]; // Combined matrix

int i, j, k = 0;

// Copy 1D array to matrix

for(i = 0; i < 1; i++) {

for(j = 0; j < 5; j++) {

matrix[i][j] = arr1[k];

k++;

}

}

// Copy 2D array to matrix

for(i = 1; i < 3; i++) {

for(j = 0; j < 5; j++) {

if(j < 3) {

matrix[i][j] = arr2[i-1][j];

}

else {

matrix[i][j] = 0;

}

}

}

// Display the combined matrix

printf("Combined matrix:\n");

for(i = 0; i < 3; i++) {

for(j = 0; j < 5; j++) {

printf("%d ", matrix[i][j]);

}

printf("\n");

}

return 0;

}

QUESTION 10. WAP of 3D array using c programming.

#include <stdio.h>

int main() {

int arr[2][3][4] = {

{{1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11, 12}},

{{13, 14, 15, 16}, {17, 18, 19, 20}, {21, 22, 23, 24}}

}; // 3D array

int i, j, k;

// Display the 3D array

printf("3D array:\n");

for (i = 0; i < 2; i++) {

for (j = 0; j < 3; j++) {

for (k = 0; k < 4; k++) {

printf("%d ", arr[i][j][k]);

}

printf("\n");

}

printf("\n");

}

return 0;

}

QUESTION 11. WAP of function to show name and age of student using C programming.

#include <stdio.h>

void showStudentInfo(char \*name, int age);

int main() {

char name[50];

int age;

printf("Enter your name: ");

scanf("%s", name);

printf("Enter your age: ");

scanf("%d", &age);

showStudentInfo(name, age);

return 0;

}

void showStudentInfo(char \*name, int age) {

printf("Name: %s\n", name);

printf("Age: %d\n", age);

}

QUESTION 12. WAP of function to show sum of n numbers.

#include <stdio.h>

int sumOfN(int n);

int main() {

int n;

printf("Enter a positive integer: ");

scanf("%d", &n);

printf("The sum of the first %d numbers is %d.\n", n, sumOfN(n));

return 0;

}

int sumOfN(int n) {

int sum = 0;

for (int i = 1; i <= n; i++) {

sum += i;

}

return sum;

}

QUESTION 13. WAP of recursion function which show factorial of n numbers.

#include <stdio.h>

int factorial(int n);

int main() {

int n;

printf("Enter a positive integer: ");

scanf("%d", &n);

printf("The factorial of %d is %d.\n", n, factorial(n));

return 0;

}

int factorial(int n) {

if (n == 0) {

return 1;

}

else {

return n \* factorial(n - 1);

}

}

QUESTION 14. WAP of recursion function which represents factorial of n numbers.

#include <stdio.h>

void printFactorial(int n, int result);

int main() {

int n;

printf("Enter a positive integer: ");

scanf("%d", &n);

printf("%d! = ", n);

printFactorial(n, 1);

return 0;

}

void printFactorial(int n, int result) {

if (n == 0) {

printf("%d", result);

}

else {

printf("%d x ", n);

printFactorial(n - 1, result \* n);

}

}

QUESTION 15. WAP of tower of Hanoi using recursion function in c programming.

#include <stdio.h>

void towerOfHanoi(int n, char source, char destination, char auxiliary);

int main() {

int n;

printf("Enter the number of disks: ");

scanf("%d", &n);

towerOfHanoi(n, 'A', 'C', 'B');

return 0;

}

void towerOfHanoi(int n, char source, char destination, char auxiliary) {

if (n == 1) {

printf("Move disk 1 from %c to %c\n", source, destination);

return;

}

towerOfHanoi(n - 1, source, auxiliary, destination);

printf("Move disk %d from %c to %c\n", n, source, destination);

towerOfHanoi(n - 1, auxiliary, destination, source);

}

**QUESTION 16.** WAP of math function (1)sqrt 2.power() 3.sin() 4.cos() 5.tan() Write separate program for each function.

1. Program to calculate square root using **sqrt** function:

#include <stdio.h>

#include <math.h>

int main() {

double number, result;

printf("Enter a number: ");

scanf("%lf", &number);

result = sqrt(number);

printf("The square root of %lf is %lf.\n", number, result);

return 0;

}

1. Program to calculate power using **pow** function:

#include <stdio.h>

#include <math.h>

int main() {

double base, exponent, result;

printf("Enter the base: ");

scanf("%lf", &base);

printf("Enter the exponent: ");

scanf("%lf", &exponent);

result = pow(base, exponent);

printf("%lf raised to %lf is %lf.\n", base, exponent, result);

return 0;

}

1. Program to calculate sine using **sin** function:

#include <stdio.h>

#include <math.h>

int main() {

double angle, result;

printf("Enter the angle in degrees: ");

scanf("%lf", &angle);

result = sin(angle \* M\_PI / 180);

printf("The sine of %lf degrees is %lf.\n", angle, result);

return 0;

}

1. Program to calculate cosine using **cos** function:

#include <stdio.h>

#include <math.h>

int main() {

double angle, result;

printf("Enter the angle in degrees: ");

scanf("%lf", &angle);

result = cos(angle \* M\_PI / 180);

printf("The cosine of %lf degrees is %lf.\n", angle, result);

return 0;

}

1. Program to calculate tangent using **tan** function:

#include <stdio.h>

#include <math.h>

int main() {

double angle, result;

printf("Enter the angle in degrees: ");

scanf("%lf", &angle);

result = tan(angle \* M\_PI / 180);

printf("The tangent of %lf degrees is %lf.\n", angle, result);

return 0;

}

QUESTION 17. WRITE A C PROGRAM OF LIST OF FOLLOWING:

1. Int strlen

2. Strcpy

3. Strcat

4. Strrev

5. Strupr

6. Strlwr

1. **strlen**: Calculates the length of a string.
2. **strcpy**: Copies one string to another.
3. **strcat**: Concatenates two strings.
4. **strrev**: Reverses a string.
5. **strupr**: Converts a string to uppercase.
6. **strlwr**: Converts a string to lowercase.

c

#include <stdio.h>

#include <string.h>

int main() {

char str1[100], str2[100], str3[100];

// strlen function

printf("Enter a string: ");

scanf("%s", str1);

printf("The length of the string is %d.\n", strlen(str1));

// strcpy function

printf("Enter another string: ");

scanf("%s", str2);

strcpy(str3, str2);

printf("The copied string is %s.\n", str3);

// strcat function

printf("Enter a string to concatenate: ");

scanf("%s", str2);

strcat(str1, str2);

printf("The concatenated string is %s.\n", str1);

// strrev function

printf("Enter a string to reverse: ");

scanf("%s", str1);

strrev(str1);

printf("The reversed string is %s.\n", str1);

// strupr function

printf("Enter a string to convert to uppercase: ");

scanf("%s", str1);

strupr(str1);

printf("The uppercase string is %s.\n", str1);

// strlwr function

printf("Enter a string to convert to lowercase: ");

scanf("%s", str1);

strlwr(str1);

printf("The lowercase string is %s.\n", str1);

return 0;

}

QUESTION 18. WRITE A C PROGRAM TO STORES THE ADDRESS OF NUMBER VARIABLE USING POINTER.

#include <stdio.h>

int main() {

int num = 10;

int \*p;

p = &num; // storing the address of num variable using pointer

printf("The value of num is %d.\n", num);

printf("The address of num is %p.\n", &num);

printf("The value of p is %p.\n", p);

printf("The value stored at the address pointed by p is %d.\n", \*p);

return 0;

}

QUESTION 19. WRITE A C PROGRAM TO SWAP TWO NUMBERS WITHOUT USING THE THIRD VARIABLE USING POINTER.

#include <stdio.h>

int main() {

int a, b;

int \*p1, \*p2;

printf("Enter the values of a and b:\n");

scanf("%d %d", &a, &b);

printf("Before swapping: a = %d, b = %d\n", a, b);

p1 = &a; // storing the address of a variable in p1

p2 = &b; // storing the address of b variable in p2

\*p1 = \*p1 + \*p2; // adding the values stored at p1 and p2

\*p2 = \*p1 - \*p2; // subtracting the value of p2 from p1 and storing it in p2

\*p1 = \*p1 - \*p2; // subtracting the value of p2 from p1 and storing it in p1

printf("After swapping: a = %d, b = %d\n", a, b);

return 0;

}

QUESTION 20. WRITE A C PROGRAM OF THE FOLLOWING POINTER :

a. Increment

b.Decrement

c. Addition

d. Subtraction

#include <stdio.h>

int main() {

int arr[5] = {10, 20, 30, 40, 50};

int \*p = &arr[2]; // storing the address of the third element in p

printf("Value at p = %d\n", \*p);

printf("Address of p = %p\n", p);

// increment operation

p++;

printf("After increment, value at p = %d\n", \*p);

printf("After increment, address of p = %p\n", p);

// decrement operation

p--;

printf("After decrement, value at p = %d\n", \*p);

printf("After decrement, address of p = %p\n", p);

// addition operation

p = p + 2;

printf("After addition, value at p = %d\n", \*p);

printf("After addition, address of p = %p\n", p);

// subtraction operation

p = p - 1;

printf("After subtraction, value at p = %d\n", \*p);

printf("After subtraction, address of p = %p\n", p);

return 0;

}

QUESTION 21. WRITE A C PROGRAM OF DYNAMIC MEMORY ALLOCATION OF POINTER

a. Malloc()

b.Calloc()

c. Realloc()

d. Free()

Malloc

#include <stdio.h>

#include <stdlib.h>

int main() {

int size;

int \*arr;

printf("Enter size of array: ");

scanf("%d", &size);

// Allocate memory dynamically using malloc()

arr = (int\*) malloc(size \* sizeof(int));

// Check if memory allocation was successful

if(arr == NULL) {

printf("Memory allocation failed.");

exit(1);

}

// Input values into the dynamically allocated array

printf("Enter %d integers:\n", size);

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

scanf("%d", &arr[i]);

}

// Display the elements of the dynamically allocated array

printf("Elements of array are:\n");

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

printf("%d ", arr[i]);

}

// Deallocate memory allocated by malloc()

free(arr);

return 0;

}

Calloc

#include <stdio.h>

#include <stdlib.h>

int main() {

int size;

int \*arr;

printf("Enter size of array: ");

scanf("%d", &size);

// Allocate memory dynamically using calloc()

arr = (int\*) calloc(size, sizeof(int));

// Check if memory allocation was successful

if(arr == NULL) {

printf("Memory allocation failed.");

exit(1);

}

// Input values into the dynamically allocated array

printf("Enter %d integers:\n", size);

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

scanf("%d", &arr[i]);

}

// Display the elements of the dynamically allocated array

printf("Elements of array are:\n");

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

printf("%d ", arr[i]);

}

// Deallocate memory allocated by calloc()

free(arr);

return 0;

}

Realloc

#include <stdio.h>

#include <stdlib.h>

int main() {

int size1, size2;

int \*arr;

printf("Enter initial size of array: ");

scanf("%d", &size1);

// Allocate memory dynamically using malloc()

arr = (int\*) malloc(size1 \* sizeof(int));

// Check if memory allocation was successful

if(arr == NULL) {

printf("Memory allocation failed.");

exit(1);

}

// Input values into the dynamically allocated array

printf("Enter %d integers:\n", size1);

for(int i = 0; i < size1; i++) {

scanf("%d", &arr[i]);

}

// Display the elements of the dynamically allocated array

printf("Elements of array are:\n");

for(int i = 0; i < size1; i++) {

printf("%d ", arr[i]);

}

// Ask user for new size of array

printf("\nEnter new size of array: ");

scanf("%d", &size2);

// Reallocate memory using realloc()

arr = (int\*) realloc(arr, size2 \* sizeof(int));

// Check if memory reallocation was successful

if(arr == NULL) {

printf("Memory reallocation failed.");

exit(1);

}

// Input additional values into the dynamically allocated array

printf("Enter %d more integers:\n", size2-size1);

for(int i = size1; i < size2; i++) {

scanf("%d", &arr[i]);

}

// Display the updated elements of the dynamically allocated array

printf("Updated elements of array are:\n");

for(int i = 0; i < size2; i++) {

printf("%d ", arr[i]);

}

// Deallocate memory allocated by realloc()

free(arr);

return 0;

}

Free()

#include <stdio.h>

#include <stdlib.h>

int main() {

int \*ptr1, \*ptr2, \*ptr3, \*ptr4;

int n = 5;

// allocate memory using malloc()

ptr1 = (int\*)malloc(n \* sizeof(int));

if (ptr1 == NULL) {

printf("Memory allocation failed");

exit(1);

}

printf("Memory allocated using malloc(): %p\n", ptr1);

// allocate memory using calloc()

ptr2 = (int\*)calloc(n, sizeof(int));

if (ptr2 == NULL) {

printf("Memory allocation failed");

exit(1);

}

printf("Memory allocated using calloc(): %p\n", ptr2);

// reallocate memory using realloc()

n = 10;

ptr3 = (int\*)realloc(ptr1, n \* sizeof(int));

if (ptr3 == NULL) {

printf("Memory reallocation failed");

exit(1);

}

printf("Memory reallocated using realloc(): %p\n", ptr3);

// free memory using free()

free(ptr2);

printf("Memory freed using free()\n");

// set pointer to NULL after freeing memory

ptr2 = NULL;

return 0;

}

QUESTION 22. WRITE A C PROGRAM TO SHOW TYPES OF POINTER IN ONE PROGRAM.

#include <stdio.h>

int main() {

int num = 10;

int \*ptr\_int;

float \*ptr\_float;

char \*ptr\_char;

ptr\_int = &num;

ptr\_float = (float\*) &num;

ptr\_char = (char\*) &num;

printf("Integer Pointer: %p, Value: %d\n", ptr\_int, \*ptr\_int);

printf("Float Pointer: %p, Value: %f\n", ptr\_float, \*ptr\_float);

printf("Character Pointer: %p, Value: %c\n", ptr\_char, \*ptr\_char);

return 0;

}

QUESTION 23. WRITE A C PROGRAM OF DOUBLE POINTER

#include <stdio.h>

#include <stdlib.h>

int main() {

int num = 10;

int \*ptr = &num;

int \*\*dptr = &ptr;

printf("Value of num: %d\n", num);

printf("Address of num: %p\n", &num);

printf("Value of ptr: %p\n", ptr);

printf("Address of ptr: %p\n", &ptr);

printf("Value of dptr: %p\n", dptr);

printf("Address of dptr: %p\n", &dptr);

printf("Value pointed to by ptr: %d\n", \*ptr);

printf("Value pointed to by dptr: %d\n", \*\*dptr);

return 0;

}

QUESTION 24. WRITE A C PROGRAM OF DANGLING POINTER.

#include <stdio.h>

#include <stdlib.h>

int main() {

int \*ptr = (int\*) malloc(sizeof(int));

\*ptr = 42;

free(ptr);

printf("The value pointed to by the pointer is %d\n", \*ptr);

return 0;

}