|  |
| --- |
| C PROGRAMMING WEEKLY QUESTIONS |
| NAME – UDAY VERMA  CLASS AL  ROLL NO. – 72  GLA UNIVERSITY |

***Programming Questions week 6:-***

Q. 1 Write a menu driven program to insert and delete elements of kth position to an array of size N. Q. 2 Write the program to print the biggest and smallest element in an array.

Q. 3 Write the program to print the sum and average of an array.

Q. 4 Write the program to sort an array using bubble sort.

Q. 5 Write the program to search an element using linear search as well as binary search.

Q. 6 Take an array of 20 integer inputs from user and print the following:

a. number of positive numbers

b. number of negative numbers

c. number of odd numbers

d. number of even numbers

e. number of 0.

Q. 7 Take an array of 10 elements. Split it into middle and store the elements in two different arrays. E.g.- INITIAL array: 58, 24, 13, 15, 63, 9, 8, 81, 1, 78 After splitting: 58, 24, 13, 15, 63 9, 8, 81, 1, 78

Q8. Write the program to count frequency of each element in an array.

**ANSWERS**

Q. 1: Menu Driven Program for Insertion and Deletion c

#include <stdio.h>

#define MAX\_SIZE 100

void insertElement(int arr[], int \*n, int position, int element) {

if (\*n >= MAX\_SIZE) {

printf("Array is full, cannot insert.\n");

return;

}

for (int i = \*n; i >= position; i--) {

arr[i] = arr[i - 1];

}

arr[position - 1] = element;

(\*n)++;

}

void deleteElement(int arr[], int \*n, int position) { // Delete element at specified position

if (\*n == 0) {

printf("Array is empty, cannot delete.\n");

return;

}

for (int i = position - 1; i < \*n - 1; i++) {

arr[i] = arr[i + 1];

}

(\*n)--;

}

int main() {

int arr[MAX\_SIZE], n, choice, position, element;

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

scanf("%d", &n);

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

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

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

}

printf("Menu:\n1. Insert Element\n2. Delete Element\nEnter choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

printf("Enter position and element to insert: ");

scanf("%d %d", &position, &element);

insertElement(arr, &n, position, element);

break;

case 2:

printf("Enter position to delete: ");

scanf("%d", &position);

deleteElement(arr, &n, position);

break;

default:

printf("Invalid choice.\n");

}

printf("Updated Array:\n");

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

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

}

return 0;

}

Q. 2: Program to Print Biggest and Smallest Element

#include <stdio.h>

int main() {

int arr[] = {12, 45, 23, 67, 89, 34, 56, 21, 9, 72};

int n = sizeof(arr) / sizeof(arr[0]);

int max = arr[0], min = arr[0];

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

if (arr[i] > max) {

max = arr[i];

}

if (arr[i] < min) {

min = arr[i];

}

}

printf("The largest element is: %d\n", max);

printf("The smallest element is: %d\n", min);

return 0;

}

Q. 3: Program to Print Sum and Average

#include <stdio.h>

int main() {

int arr[] = {5, 10, 15, 20, 25};

int n = sizeof(arr) / sizeof(arr[0]);

int sum = 0;

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

sum += arr[i];

}

float average = (float)sum / n;

printf("Sum: %d\n", sum);

printf("Average: %.2f\n", average);

return 0;

}

Q. 4: Program to Sort an Array using Bubble Sort

#include <stdio.h>

void bubbleSort(int arr[], int n) {

for (int i = 0; i < n - 1; i++) {

for (int j = 0; j < n - i - 1; j++) {

if (arr[j] > arr[j + 1]) {

// Swap the elements if they are in the wrong order

int temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

int main() {

int arr[] = {64, 25, 12, 22, 11};

int n = sizeof(arr) / sizeof(arr[0]);

bubbleSort(arr, n);

printf("Sorted array: ");

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

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

}

return 0;

}

Q. 5: Program to Search Element using Linear and Binary Search

#include <stdio.h>

int linearSearch(int arr[], int n, int key) {

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

if (arr[i] == key) {

return i; // Return the index if found

}

}

return -1; // Return -1 if not found

}

int binarySearch(int arr[], int low, int high, int key) {

while (low <= high) {

int mid = low + (high - low) / 2;

if (arr[mid] == key) {

return mid; // Return the index if found

}

if (arr[mid] < key) {

low = mid + 1;

} else {

high = mid - 1;

}

}

return -1; // Return -1 if not found

}

int main() {

int arr[] = {1, 3, 5, 7, 9, 11, 13, 15, 17, 19};

int n = sizeof(arr) / sizeof(arr[0]);

int key;

printf("Enter the element to search: ");

scanf("%d", &key);

int linearIndex = linearSearch(arr, n, key);

if (linearIndex != -1) {

printf("Linear Search: Element found at index %d\n", linearIndex);

} else {

printf("Linear Search: Element not found\n");

}

int binaryIndex = binarySearch(arr, 0, n - 1, key);

if (binaryIndex != -1) {

printf("Binary Search: Element found at index %d\n", binaryIndex);

} else {

printf("Binary Search: Element not found\n");

}

return 0;

}

Q. 6: Program to Count Positive, Negative, Odd, Even, and Zero

#include <stdio.h>

int main() {

int arr[20];

int positive = 0, negative = 0, odd = 0, even = 0, zero = 0;

printf("Enter 20 integer inputs:\n");

for (int i = 0; i < 20; i++) {

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

if (arr[i] > 0) {

Q. 7: Program to Split an Array into Two

#include <stdio.h>

int main() {

int initialArray[] = {58, 24, 13, 15, 63, 9, 8, 81, 1, 78};

int n = sizeof(initialArray) / sizeof(initialArray[0]);

int middle = n / 2;

int firstArray[middle];

int secondArray[n - middle];

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

if (i < middle) {

firstArray[i] = initialArray[i];

} else {

secondArray[i - middle] = initialArray[i];

}

}

printf("Initial Array: ");

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

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

}

printf("\nAfter Splitting:\n");

printf("First Array: ");

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

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

}

printf("\nSecond Array: ");

for (int i = 0; i < n - middle; i++) {

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

}

return 0;

}

Q. 8: Program to Count Frequency of Each Element

#include <stdio.h>

#define MAX\_SIZE 100

int main() {

int arr[MAX\_SIZE];

int n;

// Input the size of the array

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

scanf("%d", &n);

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

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

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

}

int frequency[MAX\_SIZE] = {0};

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

frequency[arr[i]]++;

}

printf("Element\tFrequency\n");

for (int i = 0; i < MAX\_SIZE; i++) {

if (frequency[i] > 0) {

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

}

}

return 0;

}

**Week – 7 Programming Questions**

Q. 1 Write the program to print row major and column major matrix.

Q. 2 Write the program to print sum of a whole matrix.

Q. 3 Write a program to add and multiply two 3x3 matrices. You can use 2D array to create a matrix.

Q. 4 Write the program to print sum of all diagonal elements, upper triangular matrix and lower triangular matrix.

Q. 5 Write the program to find the frequency of odd and even elements in matrix.

Q. 6 Write the program to find sum of each row and sum of each column of matrix.

Q. 7 Initialize a 2D array of 3\*3 matrix. E.g.- 1 2 3 2 3 4 3 4 5

Q. 8 A square matrix, one having the same number of rows and columns, is called a diagonal matrix if it’s only non-zero elements are on the diagonal from upper left to lower right. It is called upper triangular matrix if all elements bellow the diagonal are zeroes, and lower triangular matrix, if all the elements above the diagonal are zeroes. Write a program that reads a matrix and determines if it is one of the above mentioned three special matrices.

Q. 9 Write the program to check whether the matrix is sparse matrix or not.

**ANSWERS**

Q. 1: Program to Print Row Major and Column Major Matrix

#include <stdio.h>

void printRowMajor(int matrix[3][3]) {

printf("Row Major Order:\n");

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

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

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

}

printf("\n");

}

}

Void printColumnMajor(int matrix[3][3]) {

printf("Column Major Order:\n");

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

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

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

}

printf("\n");

}

}

int main() {

int matrix[3][3] = {{1, 2, 3},

{4, 5, 6},

{7, 8, 9}};

printRowMajor(matrix);

printColumnMajor(matrix);

return 0;

}

Q. 2: Program to Print Sum of a Whole Matrix

#include <stdio.h>

int main() {

int matrix[3][3] = {{1, 2, 3},

{4, 5, 6},

{7, 8, 9}};

int sum = 0;

// Calculate sum of the matrix

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

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

sum += matrix[i][j];

}

}

printf("Sum of the whole matrix: %d\n", sum);

return 0;

}

Q. 3: Program to Add and Multiply Two 3x3 Matrices

#include <stdio.h>

void addMatrices(int A[3][3], int B[3][3], int result[3][3]) {

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

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

result[i][j] = A[i][j] + B[i][j];

}

}

}

void multiplyMatrices(int A[3][3], int B[3][3], int result[3][3]) {

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

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

result[i][j] = 0;

for (int k = 0; k < 3; k++) {

result[i][j] += A[i][k] \* B[k][j];

}

}

}

}

int main() {

int matrix1[3][3] = {{1, 2, 3},

{4, 5, 6},

{7, 8, 9}};

int matrix2[3][3] = {{9, 8, 7},

{6, 5, 4},

{3, 2, 1}};

int sumResult[3][3];

int productResult[3][3];

addMatrices(matrix1, matrix2, sumResult);

multiplyMatrices(matrix1, matrix2, productResult);

printf("Sum of Matrices:\n");

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

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

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

}

printf("\n");

}

printf("\nProduct of Matrices:\n");

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

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

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

}

printf("\n");

}

return 0;

}

Q. 4: Program to Print Sum of All Diagonal Elements, Upper Triangular Matrix, and Lower Triangular Matrix

#include <stdio.h>

int main() {

int matrix[3][3] = {{1, 2, 3},

{4, 5, 6},

{7, 8, 9}};

int diagonalSum = 0, upperTriangularSum = 0, lowerTriangularSum = 0

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

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

if (i == j) {

diagonalSum += matrix[i][j];

}

if (i < j) {

upperTriangularSum += matrix[i][j];

}

if (i > j) {

lowerTriangularSum += matrix[i][j];

}

}

}

printf("Sum of Diagonal Elements: %d\n", diagonalSum);

printf("Sum of Upper Triangular Elements: %d\n", upperTriangularSum);

printf("Sum of Lower Triangular Elements: %d\n", lowerTriangularSum);

return 0;

}

Q. 5: Program to Find the Frequency of Odd and Even Elements in a Matrix

#include <stdio.h>

int main() {

int matrix[3][3] = {{1, 2, 3},

{4, 5, 6},

{7, 8, 9}};

int evenFrequency = 0, oddFrequency = 0;

// Count frequency of odd and even elements

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

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

if (matrix[i][j] % 2 == 0) {

evenFrequency++;

} else {

oddFrequency++;

}

}

}

printf("Frequency of Even Elements: %d\n", evenFrequency);

printf("Frequency of Odd Elements: %d\n", oddFrequency);

return 0;

}

Q. 6: Program to Find Sum of Each Row and Sum of Each Column of a Matrix

#include <stdio.h>

int main() {

int matrix[3][3] = {{1, 2, 3},

{2, 3, 4},

{3, 4, 5}};

int rowSum[3] = {0};

int columnSum[3] = {0};

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

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

rowSum[i] += matrix[i][j];

columnSum[j] += matrix[i][j];

}

}

printf("Sum of each row:\n");

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

printf("Row %d: %d\n", i + 1, rowSum[i]);

}

printf("Sum of each column:\n");

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

printf("Column %d: %d\n", j + 1, columnSum[j]);

}

return 0;

}

Q. 7: Initialize a 2D array of 3x3 matrix

#include <stdio.h>

int main() {

int matrix[3][3] = {{1, 2, 3},

{2, 3, 4},

printf("Initialized 3x3 Matrix:\n");

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

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

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

}

printf("\n");

}

return 0;

}

Q. 8: Program to Check for Diagonal, Upper Triangular, or Lower Triangular Matrix

#include <stdio.h>

int isDiagonalMatrix(int matrix[3][3]) {

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

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

if (i != j && matrix[i][j] != 0) {

return 0; // Not a diagonal matrix

}

}

}

return 1; // Diagonal matrix

}

int isUpperTriangularMatrix(int matrix[3][3]) {

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

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

if (matrix[i][j] != 0) {

return 0; // Not an upper triangular matrix

}

}

}

return 1; // Upper triangular matrix

}

int isLowerTriangularMatrix(int matrix[3][3]) {

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

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

if (matrix[i][j] != 0) {

return 0; // Not a lower triangular matrix

}

}

}

return 1; // Lower triangular matrix

}

int main() {

int matrix[3][3] = {{1, 2, 3},

{0, 4, 5},

{0, 0, 6}};

if (isDiagonalMatrix(matrix)) {

printf("It is a diagonal matrix.\n");

} else if (isUpperTriangularMatrix(matrix)) {

printf("It is an upper triangular matrix.\n");

} else if (isLowerTriangularMatrix(matrix)) {

printf("It is a lower triangular matrix.\n");

} else {

printf("It is not a diagonal, upper triangular, or lower triangular matrix.\n");

}

return 0;

}

Q. 9: Program to Check Whether the Matrix is Sparse Matrix or Not

#include <stdio.h>

#define ROWS 3

#define COLS 3

int isSparseMatrix(int matrix[ROWS][COLS]) {

int zeroCount = 0;

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

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

if (matrix[i][j] == 0) {

zeroCount++;

}

}

}

return zeroCount > (ROWS \* COLS) / 2; // If more than half of the elements are zero, it's considered sparse

}

int main() {

int matrix[ROWS][COLS] = {{1, 0, 0},

{0, 0, 0},

{0, 0, 0}};

if (isSparseMatrix(matrix)) {

printf("It is a sparse matrix.\n");

} else {

printf("It is not a sparse matrix.\n");

}

return 0;

}

**C- Programming Language Week – 8**

**Programming Questions**

Q. 1 Write a C program to create, initialize and use pointers.

Q. 2 Write a C program to add two numbers using pointers.

Q. 3 Write a C program to swap two numbers using pointers.

Q. 4 Write a C program to input and print array elements using pointer.

Q. 5 Write a C program to copy one array to another using pointer.

Q. 6 Write a C program to swap two arrays using pointers.

Q. 7 Write a C program to reverse an array using pointers.

Q. 8 Write a C program to add two matrix using pointers.

Q. 9 Write a C program to multiply two matrix using pointers.

ANSWERS

Q. 1: C Program to Create, Initialize, and Use Pointers

#include <stdio.h>

int main() {

int num = 5;

int \*ptr;

ptr = &num;

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

return 0;

}

Q. 2: C Program to Add Two Numbers Using Pointers

#include <stdio.h>

int main() {

int num1, num2, sum;

int \*ptr1, \*ptr2;

Input two numbers

printf("Enter first number: ");

scanf("%d", &num1);

printf("Enter second number: ");

scanf("%d", &num2);

Create pointers and assign addresses

ptr1 = &num1;

ptr2 = &num2;

sum = \*ptr1 + \*ptr2;

printf("Sum: %d\n", sum);

return 0;

}

Q. 3: C Program to Swap Two Numbers Using Pointers

#include <stdio.h>

void swap(int \*a, int \*b) {

int temp = \*a;

\*a = \*b;

\*b = temp;

}

int main() {

int num1, num2;

printf("Enter first number: ");

scanf("%d", &num1);

printf("Enter second number: ");

scanf("%d", &num2);

printf("Before swapping: num1 = %d, num2 = %d\n", num1, num2);

swap(&num1, &num2);

printf("After swapping: num1 = %d, num2 = %d\n", num1, num2);

return 0;

}

Q. 4: C Program to Input and Print Array Elements Using Pointer

#include <stdio.h>

int main() {

int arr[5];

int \*ptr;

printf("Enter 5 array elements:\n");

for (ptr = arr; ptr < arr + 5; ptr++) {

scanf("%d", ptr);

}

printf("Array elements are: ");

for (ptr = arr; ptr < arr + 5; ptr++) {

printf("%d ", \*ptr);

}

return 0;

}

Q. 5: C Program to Copy One Array to Another Using Pointer

#include <stdio.h>

void copyArray(int \*source, int \*destination, int size) {

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

\*(destination + i) = \*(source + i);

}

}

int main() {

int sourceArray[5] = {1, 2, 3, 4, 5};

int destinationArray[5];

copyArray(sourceArray, destinationArray, 5);

printf("Copied Array: ");

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

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

}

return 0;

}

Q. 6: C Program to Swap Two Arrays Using Pointers

#include <stdio.h>

void swapArrays(int \*arr1, int \*arr2, int size) {

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

int temp = \*(arr1 + i);

\*(arr1 + i) = \*(arr2 + i);

\*(arr2 + i) = temp;

}

}

int main() {

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

int arr2[5] = {6, 7, 8, 9, 10};

// Swap arrays using pointer

swapArrays(arr1, arr2, 5);

// Display the swapped arrays

printf("Array 1 after swapping: ");

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

printf("%d ", arr1[i]);

}

printf("\nArray 2 after swapping: ");

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

printf("%d ", arr2[i]);

}

return 0;

}

Q. 7: C Program to Reverse an Array Using Pointers

#include <stdio.h>

void reverseArray(int \*arr, int size) {

int \*start = arr;

int \*end = arr + size - 1;

while (start < end) {

// Swap elements pointed by start and end

int temp = \*start;

\*start = \*end;

\*end = temp;

// Move the pointers toward each other

start++;

end--;

}

}

int main() {

int arr[5] = {1, 2, 3, 4, 5};

reverseArray(arr, 5);

printf("Reversed Array: ");

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

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

}

return 0;

}

Q. 8: C Program to Add Two Matrices Using Pointers

#include <stdio.h>

void addMatrices(int (\*mat1)[3], int (\*mat2)[3], int (\*result)[3], int rows, int cols) {

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

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

\*(\*(result + i) + j) = \*(\*(mat1 + i) + j) + \*(\*(mat2 + i) + j);

}

}

}

void displayMatrix(int (\*mat)[3], int rows, int cols) {

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

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

printf("%d ", \*(\*(mat + i) + j));

}

printf("\n");

}

}

int main() {

int mat1[3][3] = {{1, 2, 3},

{4, 5, 6},

{7, 8, 9}};

int mat2[3][3] = {{9, 8, 7},

{6, 5, 4},

{3, 2, 1}};

int result[3][3];

// Add matrices using pointers

addMatrices(mat1, mat2, result, 3, 3);