WEEK -5

B)-#include <stdio.h>

int findLargestEven(int \*arr, int size) {

int \*ptr = arr;

int largestEven = 0;

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

if (\*ptr % 2 == 0 && \*ptr > largestEven) {

largestEven = \*ptr;

}

ptr++;

}

return largestEven;

}

int main() {

int numbers[] = {2, 7, 14, 9, 8, 22, 11, 6};

int size = sizeof(numbers) / sizeof(numbers[0]);

int result = findLargestEven(numbers, size);

if (result != 0) {

printf("The largest even number is: %d\n", result);

} else {

printf("No even number found in the array.\n");

}

 return 0;

}

C)-

#include <stdio.h>

int main() {

int rows = 4;

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

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

printf("%d ", j);

}

printf("\n");

}

return 0;

}

D)- #include <stdio.h>

int main() {

int rows = 4;

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

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

printf("%d ", i);

}

printf("\n");

}

    return 0;

}

J)-- #include <stdio.h>

int main() {

int rows = 5;

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

for (int j = rows; j >= i; j--) {

printf("%d ", j);

}

printf("\n");

}

    return 0;

}

K)-- #include <stdio.h>

int main() {

int rows = 5;

int columns = 5;

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

for (int j = 1; j <= columns; j++) {

if (i == 1 || i == rows || j == 1 || j == columns) {

printf("\* ");

} else {

printf(" ");

}

}

printf("\n");

}

    return 0;

}

L)- #include <stdio.h>

int main() {

int rows = 5;

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

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

printf(" ");

}

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

printf("\*");

}

printf("\n");

}

    return 0;

}

M)- #include <stdio.h>

int main() {

int rows = 5;

// Upper part of the pattern

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

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

printf(" ");

}

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

printf("\*");

}

printf("\n");

}

// Lower part of the pattern

for (int i = rows - 1; i >= 1; i--) {

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

printf(" ");

}

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

printf("\*");

}

printf("\n");

}

    return 0;

}

WEEK-6

1)- #include <stdio.h>

#define MAX\_SIZE 100

void insert(int arr[], int\* n, int k, int element) {

if (k < 0 || k > \*n) {

printf("Invalid position\n");

} else {

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

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

}

arr[k] = element;

(\*n)++;

}

}

void delete(int arr[], int\* n, int k) {

if (k < 0 || k >= \*n) {

printf("Invalid position\n");

} else {

for (int i = k; i < \*n; i++) {

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

}

(\*n)--;

}

}

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

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

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

}

printf("\n");

}

int main() {

int arr[MAX\_SIZE];

int n = 0;

int choice, k, element;

while (1) {

printf("1. Insert\n");

printf("2. Delete\n");

printf("3. Display\n");

printf("4. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

printf("Enter the position: ");

scanf("%d", &k);

printf("Enter the element: ");

scanf("%d", &element);

insert(arr, &n, k, element);

break;

case 2:

printf("Enter the position: ");

scanf("%d", &k);

delete(arr, &n, k);

break;

case 3:

display(arr, n);

break;

case 4:

return 0;

default:

printf("Invalid choice\n");

}

}

    return 0;

}

2)- #include <stdio.h>

#define MAX\_SIZE 100

void find\_min\_max(int arr[], int n) {

if (n == 0) {

printf("The array is empty\n");

return;

}

int min = arr[0];

int max = arr[0];

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

if (arr[i] < min) {

min = arr[i];

}

if (arr[i] > max) {

max = arr[i];

}

}

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

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

}

int main() {

int arr[MAX\_SIZE];

int n;

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

scanf("%d", &n);

printf("Enter the elements: ");

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

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

}

find\_min\_max(arr, n);

    return 0;

}

3)- #include <stdio.h>

int main() {

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

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

int sum = 0;

float average;

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

sum += arr[i];

}

average = (float)sum / n;

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

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

    return 0;

}

4)-- #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 arr[j] and arr[j+1]

int temp = arr[j];

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

arr[j+1] = temp;

}

}

}

}

void printArray(int arr[], int size) {

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

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

}

printf("\n");

}

int main() {

int arr[] = {64, 34, 25, 12, 22, 11, 90};

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

bubbleSort(arr, n);

printf("Sorted array: \n");

printArray(arr, n);

    return 0;

}

5)-- #include <stdio.h>

// Linear Search Function

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

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

if (arr[i] == x)

return i;

}

return -1;

}

// Binary Search Function

int binarySearch(int arr[], int l, int r, int x) {

while (l <= r) {

int m = l + (r - l) / 2;

// Check if x is present at mid

if (arr[m] == x)

return m;

// If x greater, ignore left half

if (arr[m] < x)

l = m + 1;

// If x is smaller, ignore right half

else

r = m - 1;

}

// If we reach here, then element was not present

return -1;

}

int main() {

int arr[] = {2, 3, 4, 10, 40};

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

int x = 10;

// Linear Search

int result = linearSearch(arr, n, x);

(result == -1) ? printf("Element is not present in array\n")

: printf("Element is present at index %d\n", result);

// Binary Search

result = binarySearch(arr, 0, n - 1, x);

(result == -1) ? printf("Element is not present in array\n")

: printf("Element is present at index %d\n", result);

    return 0;

}

6)-- #include <stdio.h>

int main() {

int arr[20];

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

printf("Enter 20 integers:\n");

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

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

}

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

if(arr[i] > 0)

positive++;

else if(arr[i] < 0)

negative++;

else

zero++;

if(arr[i] % 2 == 0)

even++;

else

odd++;

}

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

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

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

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

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

    return 0;

}

7)-- #include <stdio.h>

int main() {

int arr[10];

int arr1[5], arr2[5];

printf("Enter 10 integers:\n");

for(int i = 0; i < 10; i++) {

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

}

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

arr1[i] = arr[i];

arr2[i] = arr[i+5];

}

printf("After splitting:\n");

printf("First array: ");

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

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

}

printf("\n");

printf("Second array: ");

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

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

}

printf("\n");

    return 0;

}

WEEK-7

1)- #include <stdio.h>

#define ROWS 3

#define COLS 3

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

int sum = 0;

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

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

sum += matrix[i][j];

}

}

return sum;

}

int main() {

int matrix[ROWS][COLS] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

int sum = calculateSum(matrix);

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

    return 0;

}

2)- #include <stdio.h>

#define SIZE 3

void addMatrices(int a[SIZE][SIZE], int b[SIZE][SIZE], int result[SIZE][SIZE]) {

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

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

result[i][j] = a[i][j] + b[i][j];

}

}

}

void multiplyMatrices(int a[SIZE][SIZE], int b[SIZE][SIZE], int result[SIZE][SIZE]) {

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

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

result[i][j] = 0;

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

result[i][j] += a[i][k] \* b[k][j];

}

}

}

}

void printMatrix(int matrix[SIZE][SIZE]) {

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

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

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

}

printf("\n");

}

}

int main() {

int a[SIZE][SIZE] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

int b[SIZE][SIZE] = {{10, 11, 12}, {13, 14, 15}, {16, 17, 18}};

int result[SIZE][SIZE];

addMatrices(a, b, result);

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

printMatrix(result);

multiplyMatrices(a, b, result);

printf("Product of matrices:\n");

printMatrix(result);

    return 0;

}

3)- #include <stdio.h>

#define SIZE 3

int calculateDiagonalSum(int matrix[SIZE][SIZE]) {

int sum = 0;

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

sum += matrix[i][i];

}

return sum;

}

void printUpperTriangular(int matrix[SIZE][SIZE]) {

printf("Upper triangular matrix:\n");

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

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

if (i <= j) {

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

} else {

printf("0 ");

}

}

printf("\n");

}

}

void printLowerTriangular(int matrix[SIZE][SIZE]) {

printf("Lower triangular matrix:\n");

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

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

if (i >= j) {

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

} else {

printf("0 ");

}

}

printf("\n");

}

}

int main() {

int matrix[SIZE][SIZE] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

int sum = calculateDiagonalSum(matrix);

printf("Sum of all diagonal elements: %d\n", sum);

printUpperTriangular(matrix);

printLowerTriangular(matrix);

    return 0;

}

4)- #include <stdio.h>

#define ROWS 3

#define COLS 3

void calculateFrequency(int matrix[ROWS][COLS]) {

int odd = 0, even = 0;

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

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

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

even++;

else

odd++;

}

}

printf("Frequency of odd elements: %d\n", odd);

printf("Frequency of even elements: %d\n", even);

}

int main() {

int matrix[ROWS][COLS] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

calculateFrequency(matrix);

    return 0;

}

5)- #include <stdio.h>

#define ROWS 3

#define COLS 3

void calculateRowColumnSum(int matrix[ROWS][COLS]) {

int rowSum, colSum;

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

rowSum = 0;

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

rowSum += matrix[i][j];

}

printf("Sum of row %d: %d\n", i+1, rowSum);

}

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

colSum = 0;

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

colSum += matrix[j][i];

}

printf("Sum of column %d: %d\n", i+1, colSum);

}

}

int main() {

int matrix[ROWS][COLS] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

calculateRowColumnSum(matrix);

    return 0;

}

6)- #include <stdio.h>

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

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

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

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

}

printf("\n");

}

}

int main() {

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

printMatrix(matrix);

    return 0;

}

7)- #include <stdio.h>

int main() {

int matrix[3][3];

int i, j;

int isDiagonal = 1, isUpperTriangular = 1, isLowerTriangular = 1;

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

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

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

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

}

}

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

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

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

isDiagonal = 0;

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

isLowerTriangular = 0;

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

isUpperTriangular = 0;

}

}

if(isDiagonal)

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

else if(isUpperTriangular)

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

else if(isLowerTriangular)

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

else

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

    return 0;

}

8)- #include <stdio.h>

int main() {

int matrix[3][3];

int i, j;

int count = 0;

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

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

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

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

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

count++;

}

}

if(count > ((3\*3)/2))

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

else

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

    return 0;

}