**C-PROGRAMMING PROJECT**

**NAME-ISHIKA RATHORE**

**CLASS ROLL NO.-33**

**UNIVERSITY ROLL NO.-2315000996**

**WEEK-5**

Q. 1 Write a program to print the following patterns:

a. \*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\* Sol. #include

<stdio.h>

int main() { int i, j;

for(i = 0; i < 4; i++) { for(j = 0; j < 5; j++) {

printf("\*");

}

printf("\n");

}

return 0;

}



b. 12345 12345

12345

12345

12345

Sol. #include

<stdio.h>

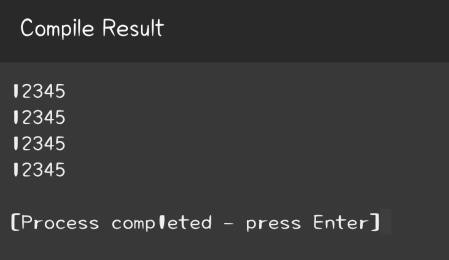
int main() { int i, j;

for(i = 1; i <= 4; i++) { for(j = 1; j <= 5; j++)

{ printf("%d", j); } printf("\n");

}

return 0;

}

1. 1 12 123

1234

Sol. #include

<stdio.h>

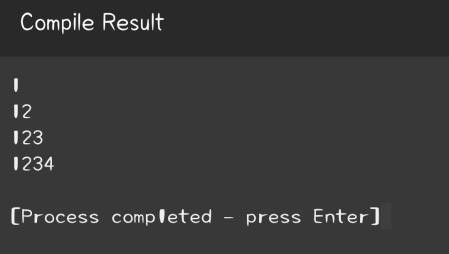
int main() { int i, j;

for(i = 1; i <= 4; i++) { for(j = 1; j <= i; j++)

{ printf("%d", j); } printf("\n");

}

return 0;

}

1. 1 22 333

4444

Sol. #include

<stdio.h>

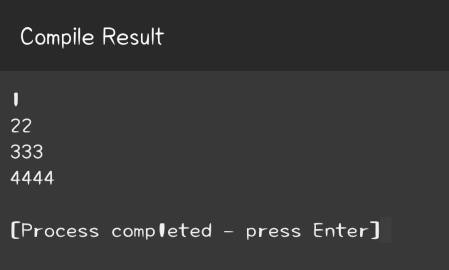
int main() { int i, j;

for(i = 1; i <= 4; i++) { for(j = 1; j <= i; j++)

{ printf("%d", i); } printf("\n");

}

return 0;

}

1. \*

\*\*

\*\*\*

\*\*\*\* Sol. #include

<stdio.h>

int main() { int i, j;

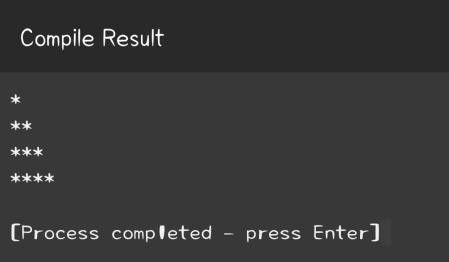
for(i = 1; i <= 4; i++) { for(j = 1; j <= i; j++)

{ printf("\*"); }

printf("\n");

}

return 0;

}

1. A

AB ABC ABCD

Sol. #include

<stdio.h>

int main() { int i, j, k;

for(i = 1; i <= 4; i++) { for(j = 4; j > i; j--) {

printf(" ");

}

for(k = 1; k <= i; k++) { printf("%c", 'A' + k - 1);

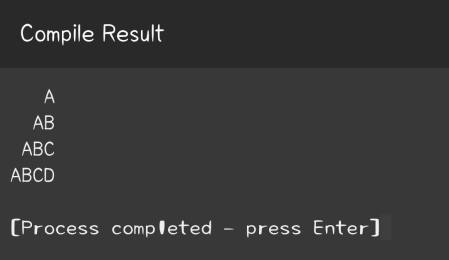
}

printf("\n");

}

return 0;

}



1. 1 2 3

4 5 6

7 8 9 10

Sol.

#include <stdio.h> int main() {

int i, j, num = 1;

for(i = 1; i <= 4; i++) { for(j = 1; j <= i; j++) {

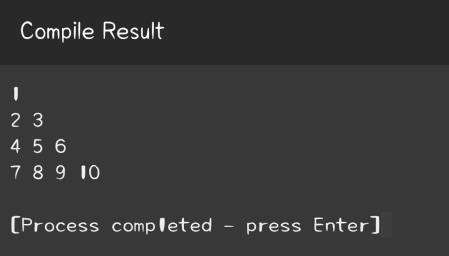
printf("%d ", num); num++;

}

printf("\n");

}

return 0;

}

1. 1 10 101

1010

10101

Sol. #include

<stdio.h>

int main() { int i, j;

for(i = 1; i <= 5; i++) { for(j = 1; j <= i; j++) { if(j % 2 != 0) {

printf("1");

} else {

printf("0");

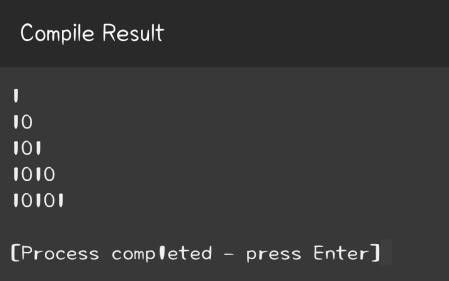
} }

printf("\n")

;

}

return 0;

}

1. 5

5 4

5 4 3

5 4 3 2

5 4 3 2 1

Sol. #include

<stdio.h>

int main() { int i, j;

for(i = 5; i >= 1; i--) {

for(j = 5; j >= i; j--) {

printf("%d ", j);

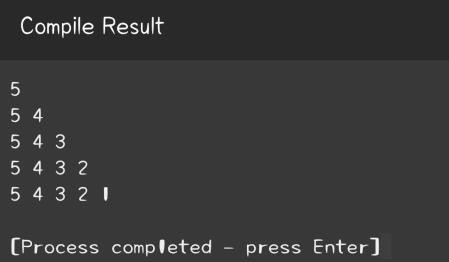
}

printf("\n");

}

return 0;

}



j. 5 4 3 2 1

5 4 3 2

5 4 3

5 4

5

Sol.

#include <stdio.h>

int main() { int i, j;

for(i = 5; i >= 1; i--) { for(j

= 5; j >= 6 - i; j--) {

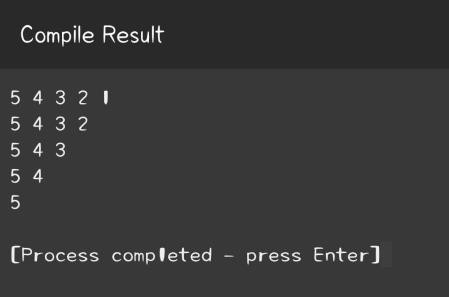
printf("%d ", j);

}

printf("\n");

}

return 0;

}

k. \*\*\*\*\*

\* \*

\* \*

\* \*

\*\*\*\*\* Sol. #include

<stdio.h>

int main() { int i, j;

for(i = 1; i <= 5; i++) { for(j = 1; j <= 5; j++) {

if(i == 1 || i == 5 || j == 1 || j == 5) { printf("\*");

} else {

printf(" ");

} }

printf("\n")

;

}

return 0;

}

1. . \*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

Sol.#include <stdio.h>

int main() {

int rows = 5;

for (int i = 1; i <= rows; i++) { // Spaces for (int space = 1; space <= rows - i; space++) { printf(" ");

}

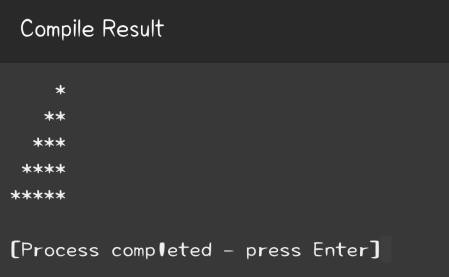
// Stars for (int j = 1; j

<= i; j++) { printf("\*"); }

printf("\n");

}

return 0;

}

1. \*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

\*\*\*\*

\*\*\*

\*\*

\*

Sol.#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 <= i; j++)

{ printf("\*"); } printf("\n");

}

// Lower part of the pattern for (int i = rows - 1; i >= 1; i--) {

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

{ printf("\*"); } printf("\n");

}

return 0;

}



n. 6 7 8

9 3 4

5

1 2

0

Sol. #include

<stdio.h>

int main() {

int i, j, k = 6;

for(i = 1; i <= 4; i++) { for(j = 1; j < i; j++)

{ printf(" ");

}

for(j = i; j <= 4; j++) { printf("%d ", k); k++;

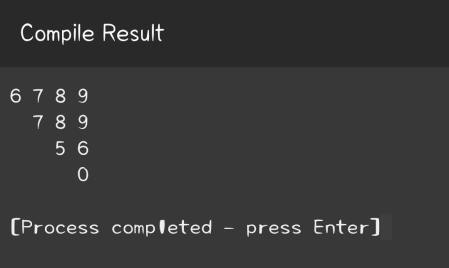
}

k = k - 2\*(i+1) + 1;

printf("\n");

}

return 0;

}

**WEEK-6**

Q. 1 Write a menu driven program to insert and delete elements of kth position to an array of size N.

Sol.-

#include <stdio.h>

void insertElement(int arr[], int \*n, int k, int element) { if (k < 1 || k > (\*n) + 1) {

printf("Invalid position for insertion.\n");

} else { (\*n)++; for (int i =

\*n; i > k; i--) { arr[i - 1]

= arr[i - 2];

}

arr[k - 1] = element; printf("Element %d inserted at position

%d.\n", element, k);

}

}

void deleteElement(int arr[], int \*n, int k) { if (k < 1 || k > \*n) {

printf("Invalid position for deletion.\n");

} else { int deletedElement = arr[k

- 1]; for (int i = k - 1; i < \*n - 1;

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

}

(\*n)--;

printf("Element %d deleted from position %d.\n", deletedElement, k);

}

}

void printArray(int arr[], int n)

{ printf("Current array: "); for (int i = 0; i < n; i++) { printf("%d ", arr[i]);

}

printf("\n");

}

int main() {

int N; printf("Enter the size of the array: "); scanf("%d", &N);

int array[N]; printf("Enter the elements of the array separated by space: "); for (int i = 0; i < N; i++) {

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

}

while (1) {

printf("\nMenu:\n"); printf("1. Insert element at kth position\n"); printf("2. Delete element at kth position\n"); printf("3. Print array\n"); printf("4. Exit\n");

int choice;

printf("Enter your choice (1-4): "); scanf("%d", &choice);

int k, element; switch (choice) {

case 1:

printf("Enter the position to insert: "); scanf("%d", &k); printf("Enter the element to insert: "); scanf("%d", &element); insertElement(array, &N, k, element); break;

case 2:

printf("Enter the position to delete:

"); scanf("%d", &k); deleteElement(array, &N, k); break;

case 3:

printArray(array, N); break;

case 4:

printf("Exiting program.\n"); return 0;

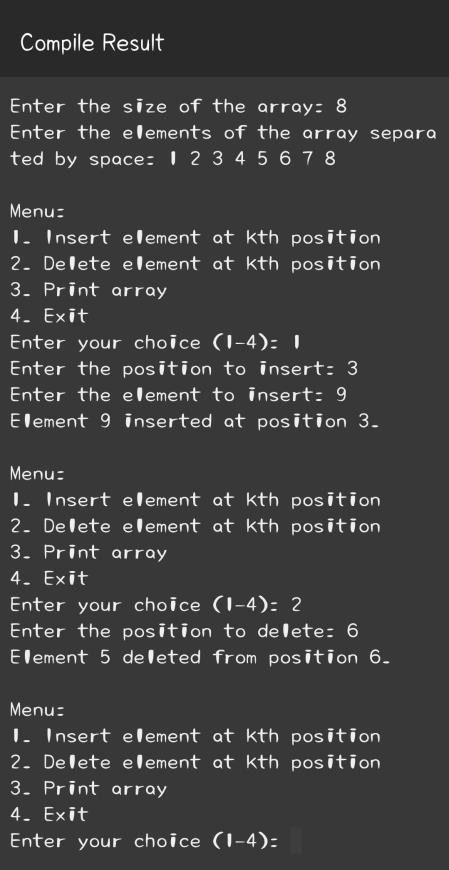
default:

printf("Invalid choice. Please enter a number between 1 and 4.\n");

}

}

return 0;

}

Q. 2 Write the program to print the biggest and smallest element in an array. Sol.- #include <stdio.h>

int main() {

int array[100], n, i, smallest, largest;

printf("Enter the number of elements in array\n"); scanf("%d", &n); printf("Enter %d integers\n", n);

for (i = 0; i < n; i++) scanf("%d", &array[i]);

smallest = largest = array[0];

for (i = 1; i < n; i++) { if (array[i] > largest)

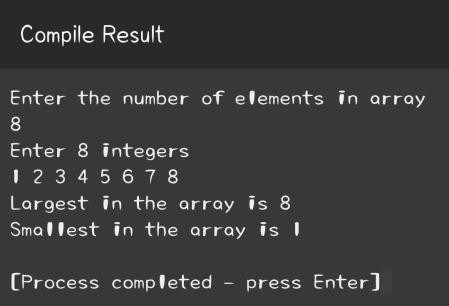
largest = array[i];

else if (array[i] < smallest) smallest = array[i];

}

printf("Largest in the array is %d\n", largest); printf("Smallest in the array is %d\n", smallest);

return 0;

}

Q. 3 Write the program to print the sum and average of an array. Sol.-#include <stdio.h>

int main() {

int n, i, sum = 0; float average;

printf("Enter the number of elements in array\n"); scanf("%d", &n);

int array[n]; printf("Enter %d integers\n", n);

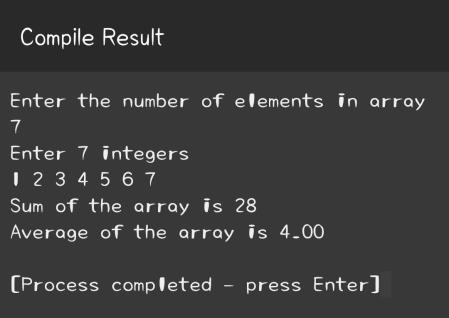
for (i = 0; i < n; i++) { scanf("%d", &array[i]); sum += array[i];

}

average = (float)sum/n; printf("Sum of the array is %d\n", sum); printf("Average of the array is %.2f\n", average);

return 0;

}



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

#include <stdio.h>

void swap(int \*xp, int \*yp) { int temp = \*xp;

\*xp = \*yp;

\*yp = temp;

}

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], &arr[j+1]);

}

}

}

void printArray(int arr[], int size) { for (int i=0; i < size; i++)

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

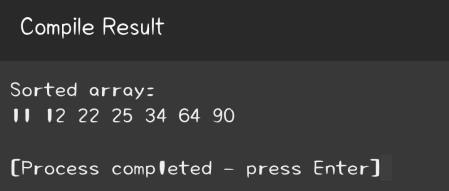
}

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;

}



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

// Linear Search #include <stdio.h>

int linearSearch(int array[], int n, int x) { for(int i = 0; i < n; i++)

if(array[i] == x) return i;

return -1;

}

// Binary Search

int binarySearch(int array[], int low, int high, int x) { if (high >= low) {

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

if (array[mid] == x) return mid;

if (array[mid] > x)

return binarySearch(array, low, mid - 1, x);

return binarySearch(array, mid + 1, high, x);

}

return -1;

}

int main() { int array[] = {2, 3, 4, 10, 40}; int x = 10;

// Using Linear Search

int result = linearSearch(array, 5, x);

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

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

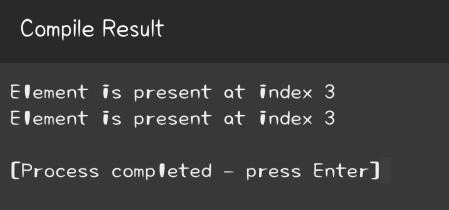
// Using Binary Search

int result2 = binarySearch(array, 0, 4, x);

(result2 == -1) ? printf("Element is not present in array\n")

: printf("Element is present at index %d\n", result2);

return 0;

}

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.

Sol.-

#include <stdio.h>

int main() {

int array[20];

int pos = 0, neg = 0, odd = 0, even = 0, zero = 0;

printf("Enter 20 integers:\n"); for(int i = 0; i < 20; i++) {

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

// Check positive/negative/zero if (array[i] > 0) pos++; else if (array[i] < 0) neg++; else zero++;

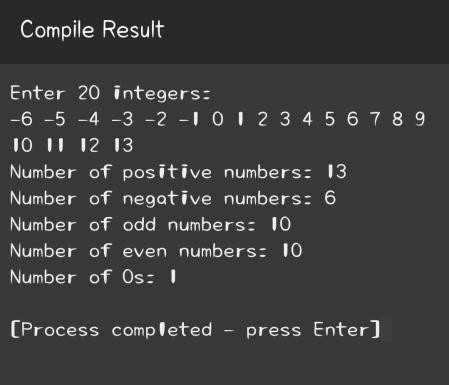
// Check odd/even if (array[i]

% 2 == 0) even++; else odd++;

}

printf("Number of positive numbers: %d\n", pos); printf("Number of negative numbers: %d\n", neg); printf("Number of odd numbers: %d\n", odd); printf("Number of even numbers: %d\n", even); printf("Number of 0s: %d\n", zero);

return 0;

}

1. 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

Sol.-

#include <stdio.h>

int main() {

int array[10] = {58, 24, 13, 15, 63, 9, 8, 81, 1, 78};

int array1[5], array2[5];

// Split the array for(int i

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

array1[i] = array[i]; array2[i] = array[i+5];

}

// Print the split arrays printf("First array after splitting: \n"); for(int i = 0; i < 5; i++) {

printf("%d ", array1[i]);

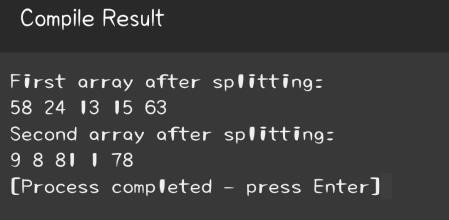
}

printf("\nSecond array after splitting: \n"); for(int i = 0; i < 5; i++) {

printf("%d ", array2[i]);

}

return 0;

}

1. Write the program to count frequency of each element in an array. Sol.-

#include <stdio.h>

int main() {

int array[100], freq[100]; int size, i, j, count;

printf("Enter size of the array: "); scanf("%d", &size);

printf("Enter elements in array: "); for(i = 0; i < size; i++) {

scanf("%d", &array[i]); freq[i] = -1;

}

for(i = 0; i < size; i++){ count = 1; for(j = i + 1; j

< size; j++){ if(array[i] ==

array[j]){

count++; freq[j] = 0;

}

}

if(freq[i] != 0){ freq[i] = count;

}

}

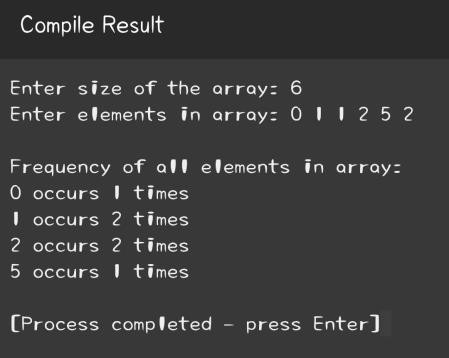
printf("\nFrequency of all elements in array: \n"); for(i = 0; i < size; i++){ if(freq[i] != 0){

printf("%d occurs %d times\n", array[i], freq[i]);

}

}

return 0;

}

**WEEK-7**

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

#include <stdio.h>

int main() {

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

{4, 5, 6},

{7, 8, 9}};

int i, j;

// Print in row-major order printf("Row-major order: \n"); for (i = 0; i < 3; i++) {

for (j = 0; j < 3; j++) { printf("%d ", array[i][j]);

} }

printf("\n");

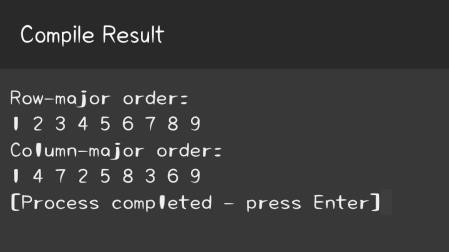
// Print in column-major order printf("Column-major order: \n"); for (i = 0; i < 3; i++) {

for (j = 0; j < 3; j++) { printf("%d ", array[j][i]);

}

}

return 0;

}

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

#include <stdio.h>

int main(){

int i, j, rows, columns, sum = 0; int matrix[10][10];

printf("Enter the number of rows and columns of the matrix: "); scanf("%d%d", &rows, &columns); printf("\nEnter elements of the matrix: \n");

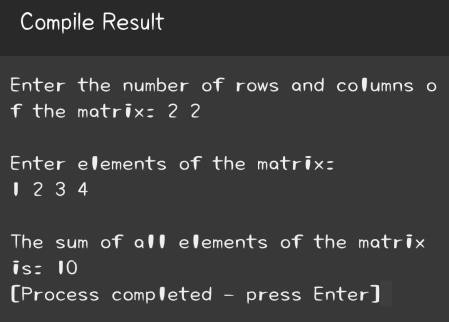
for (i = 0; i < rows; i++){

for (j = 0; j < columns; j++){ scanf("%d", &matrix[i][j]); sum = sum + matrix[i][j];

}

}

printf("\nThe sum of all elements of the matrix is: %d", sum); return 0;

}

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

Sol.-

#include <stdio.h> int main() {

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

{4, 5, 6},

{7, 8, 9}}; int

b[3][3] = {{10, 11, 12},

{13, 14, 15},

{16, 17, 18}}; int

sum[3][3], product[3][3]; int i, j, k;

// Add matrices for (i=0; i<3; i++) {

for (j=0; j<3; j++) { sum[i][j] = a[i][j] + b[i][j];

}

}

// Multiply matrices for (i=0; i<3; i++) { for (j=0;

j<3; j++) { product[i][j] =

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

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

}

}

}

// Print sum matrix printf("Sum of matrices: \n"); for (i=0; i<3; i++) {

for (j=0; j<3; j++) { printf("%d ", sum[i][j]);

}

printf("\n");

}

// Print product matrix printf("Product of matrices: \n"); for (i=0; i<3; i++) {

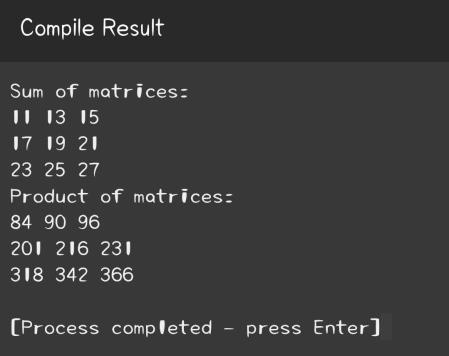
for (j=0; j<3; j++) { printf("%d ", product[i][j]);

}

printf("\n");

}

return 0;

}

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

Sol.-

#include <stdio.h>

int main() { int matrix[3][3] =

{{1, 2, 3},

{4, 5, 6},

{7, 8, 9}};

int i, j, sum = 0;

// Sum of diagonal elements for(i=0; i<3; i++) { for(j=0; j<3; j++) { if(i == j) {

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

}

}

}

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

// Print upper triangular matrix printf("Upper triangular matrix: \n"); for(i=0; i<3; i++) { for(j=0; j<3; j++)

{ if(i <= j) { printf("%d ", matrix[i][j]);

} else {

printf("0 ");

}

}

printf("\n");

}

// Print lower triangular matrix printf("Lower triangular matrix: \n"); for(i=0; i<3; i++) {

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

if(i >= j) { printf("%d ",

matrix[i][j]);

} else {

printf("0 ");

}

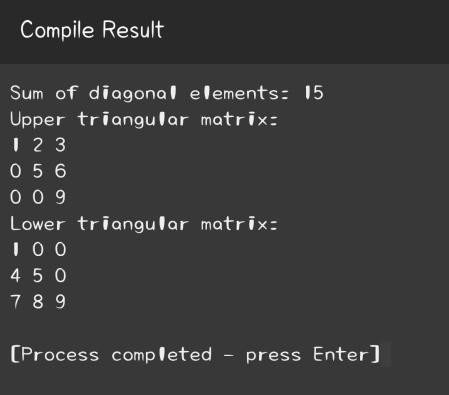
}

printf("\n");

}

return 0;

}



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

#include <stdio.h>

int main() {

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

{4, 5, 6},

{7, 8, 9}}; int i, j,

oddCount = 0, evenCount = 0;

// Count odd and even numbers for(i=0; i<3; i++) {

for(j=0; j<3; j++) { if(matrix[i][j] % 2 == 0) {

evenCount++;

} else {

oddCount++;

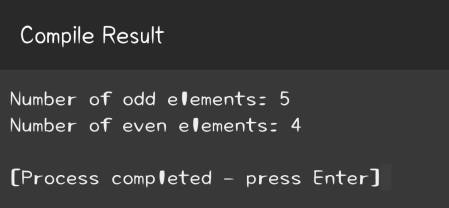
}

}

}

printf("Number of odd elements: %d\n", oddCount); printf("Number of even elements: %d\n", evenCount);

return 0;

}

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

Sol.-

#include <stdio.h>

int main() {

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

{4, 5, 6},

{7, 8, 9}};

int i, j, rowSum, colSum;

// Sum of each row for(i=0; i<3; i++) {

rowSum = 0;

for(j=0; j<3; j++) { rowSum += matrix[i][j];

}

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

}

// Sum of each column for(i=0; i<3; i++) {

colSum = 0;

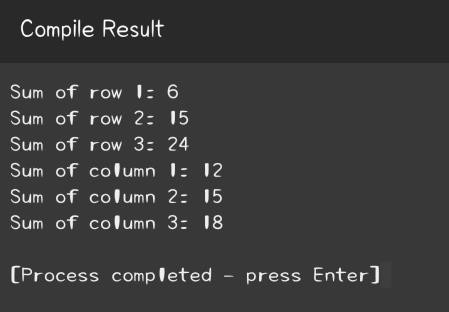
for(j=0; j<3; j++) { colSum += matrix[j][i];

}

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

}

return 0;

}

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

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 2 | 3 | 4 |
| 3 | 4 | 5 |

Sol.-

#include <stdio.h>

int main() {

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

{2, 3, 4},

{3, 4, 5}};

// Just to print the matrix for(int i=0; i<3; i++) {

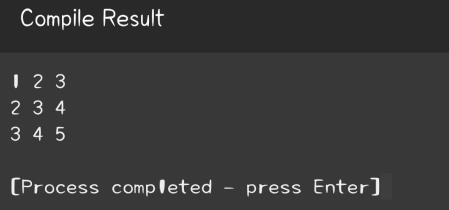
for(int j=0; j<3; j++) { printf("%d ", matrix[i][j]);

}

printf("\n");

}

return 0;

}

Q. 8 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.

Sol.- #include<stdio.h>

int main(){

int matrix[3][3] = {{1,0,0}, {0,2,0}, {0,0,3}};

int i, j, diagonal = 1, upper = 1, lower = 1;

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

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

}

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

}

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

}

}

}

if(diagonal == 1){

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

}

else if(upper == 1){

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

}

else if(lower == 1){

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

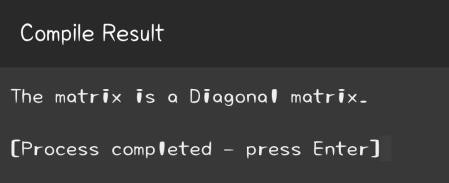
}

else{

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

}

return 0;

}

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

#include<stdio.h>

int main(){

int matrix[3][3] = {{1,0,0}, {0,2,0}, {0,0,3}};

int i, j, zeroCount = 0;

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

if(matrix[i][j] == 0){ zeroCount++;

}

}

}

if(zeroCount > ((3\*3)/2)){

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

}

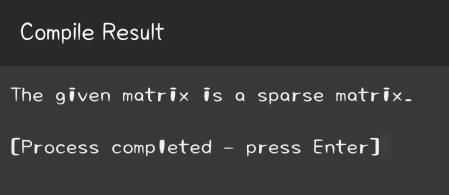
else{

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

}

return 0;

}



**WEEK-8**

Q. 1 Write a C program to create, initialize and use pointers. Sol.#include<stdio.h>

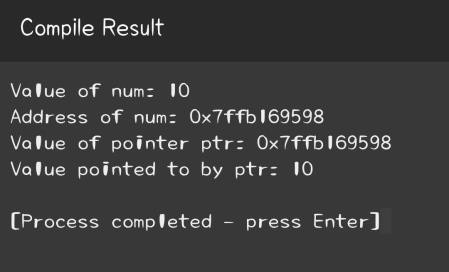
int main() {

int num = 10; // Declare and initialize an integer int

\*ptr; // Declare an integer pointer ptr = &num; // Initialize pointer with address of num printf("Value of num: %d\n", num); printf("Address of num:

%p\n", &num); printf("Value of pointer ptr: %p\n", ptr); printf("Value pointed to by ptr: %d\n", \*ptr);

return 0;

}

Q. 2 Write a C program to add two numbers using pointers. Sol.#include<stdio.h>

int main() {

int num1 = 5, num2 = 15, sum; int \*ptr1, \*ptr2;

ptr1 = &num1; // Pointer to num1 ptr2

= &num2; // Pointer to num2 sum =

\*ptr1 + \*ptr2; // Add two numbers printf("Sum = %d", sum);

return 0;

}

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

#include <stdio.h>

void swap(int\* n1, int\* n2) { int temp;

temp = \*n1;

\*n1 = \*n2;

\*n2 = temp;

}

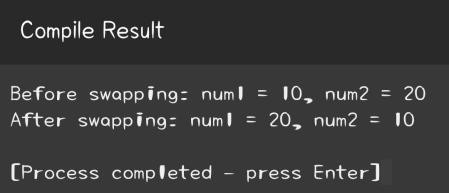
int main() {

int num1 = 10, num2 = 20;

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 Write a C program to input and print array elements using pointer. Sol.-

#include <stdio.h>

int main() {

int arr[5]; int \*ptr = arr; // Pointer to the array

int i;

printf("Enter array elements: \n"); for(i = 0; i < 5; i++) {

scanf("%d", ptr); ptr++;

}

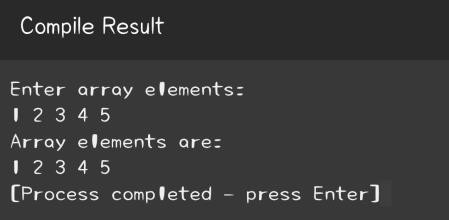
ptr = arr; // Reset pointer to start of array printf("Array elements are: \n");

for(i = 0; i < 5; i++) { printf("%d

", \*ptr); ptr++;

}

return 0;

}

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

#include <stdio.h>

int main() { int arr1[5] = {1, 2, 3, 4, 5}; int arr2[5]; int

\*ptr1 = arr1; int \*ptr2 = arr2;

int i;

// Copy arr1 to arr2 for(i = 0; i < 5; i++) {

\*(ptr2 + i) = \*(ptr1 + i);

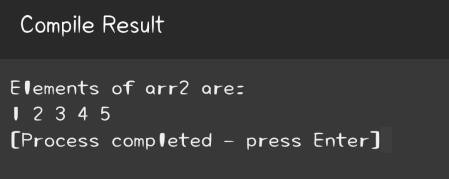
}

// Print arr2 elements printf("Elements of arr2 are: \n"); for(i = 0; i < 5; i++) {

printf("%d ", \*(ptr2 + i));

}

return 0;

}

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

#include <stdio.h>

void swap\_arrays(int \*arr1, int \*arr2, int n) { int i, temp;

for (i = 0; i < n; i++) { temp = \*(arr1 + i);

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

\*(arr2 + i) = temp;

}

}

int main() {

int arr1[] = {1, 2, 3, 4, 5}; int arr2[]

= {6, 7, 8, 9, 10}; int n =

sizeof(arr1) / sizeof(arr1[0]); int i;

printf("Original arrays:\n"); for (i = 0; i < n; i++) printf("%d ", arr1[i]);

printf("\n");

for (i = 0; i < n; i++) printf("%d ", arr2[i]); printf("\n");

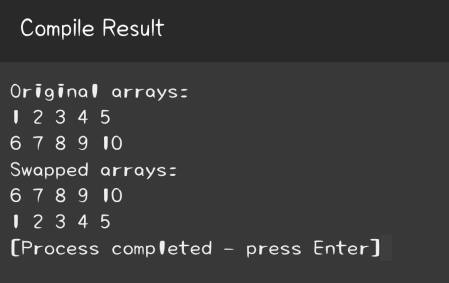
swap\_arrays(arr1, arr2, n); printf("Swapped arrays:\n"); for (i = 0; i

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

printf("\n");

for (i = 0; i < n; i++) printf("%d ", arr2[i]);

return 0;

}

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

#include <stdio.h>

void reverse\_array(int \*arr, int n) { int \*start\_ptr = arr; int

\*end\_ptr = arr + n - 1; int

temp;

while (end\_ptr > start\_ptr) { temp = \*start\_ptr;

\*start\_ptr = \*end\_ptr;

\*end\_ptr = temp; start\_ptr++; end\_ptr--;

}

}

int main() {

int arr[] = {1, 2, 3, 4, 5}; int n = sizeof(arr) / sizeof(arr[0]); int i;

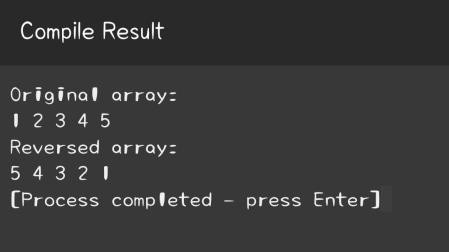
printf("Original array:\n"); for (i = 0; i < n; i++) { printf("%d ", arr[i]);

}

reverse\_array(arr, n); printf("\nReversed array:\n"); for (i = 0; i < n; i++) { printf("%d ", arr[i]);

}

return 0;

}

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

#include <stdio.h> #define SIZE 3

// Size of the matrix

void add\_matrices(int \*m1, int \*m2, int \*result, int size) { int i;

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

\*(result + i) = \*(m1 + i) + \*(m2 + i);

}

}

int main() {

int m1[SIZE][SIZE] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}; int

m2[SIZE][SIZE] = {{10, 11, 12}, {13, 14, 15}, {16, 17, 18}}; int

result[SIZE][SIZE]; int i, j; add\_matrices((int \*)m1, (int \*)m2, (int \*)result, SIZE);

printf("Result of addition:\n"); for (i = 0; i < SIZE; i++) { for (j = 0; j < SIZE; j++) {

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

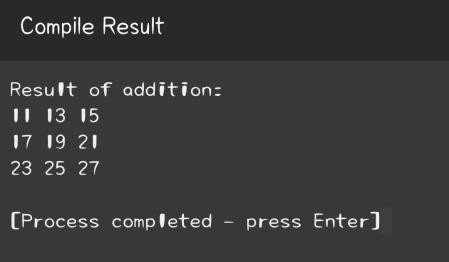
}

printf("\n");

}

return 0;

}



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

#include <stdio.h>

#define SIZE 3 // Size of the matrices

void multiply\_matrices(int \*m1, int \*m2, int \*result, int size) { int i, j, k; for (i = 0; i <

size; i++) {

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

\*(result + i\*size + j) = 0; for (k = 0; k < size; k++) {

\*(result + i\*size + j) += \*(m1 + i\*size + k) \* \*(m2 + k\*size + j);

}

}

}

}

int main() {

int m1[SIZE][SIZE] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}; int

m2[SIZE][SIZE] = {{10, 11, 12}, {13, 14, 15}, {16, 17, 18}}; int

result[SIZE][SIZE]; int i, j; multiply\_matrices((int \*)m1, (int

\*)m2, (int \*)result, SIZE);

printf("Result of multiplication:\n"); for (i = 0; i < SIZE; i++) {

for (j = 0; j < SIZE; j++) { printf("%d ", result[i][j]);

}

printf("\n");

}

return 0;

}



**WEEK-9**

Q. 1 Write a C program to Search string. Sol.-

#include <stdio.h> #include <string.h>

int main() { char string[100], search[50]; int position;

printf("Enter a string:\n"); gets(string);

printf("Enter the string to search:\n"); gets(search); char\* ptr = strstr(string, search);

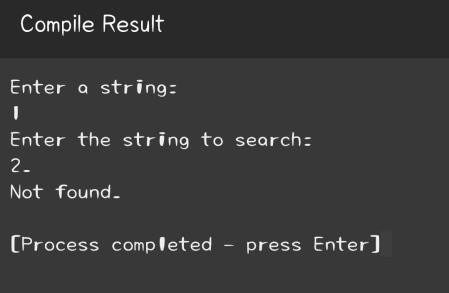
if(ptr) {

position = ptr - string; printf("Found at position: %d\n", position + 1);

} else { printf("Not found.\n");

}

return 0;

}

Q. 2 Write a C program to Reverse words in string. Sol.-

#include <stdio.h> #include <string.h>

void reverse(char \*begin, char \*end) { char temp;

while (begin < end) { temp = \*begin;

\*begin++ = \*end;

\*end-- = temp;

}

}

void reverseWords(char \*sentence) { char \*word\_begin = sentence; char \*temp = sentence;

while (\*temp) { temp++; if (\*temp == '\0') {

reverse(word\_begin, temp - 1);

} else if (\*temp == ' ') { reverse(word\_begin, temp - 1); word\_begin = temp + 1;

}

}

reverse(sentence, temp - 1);

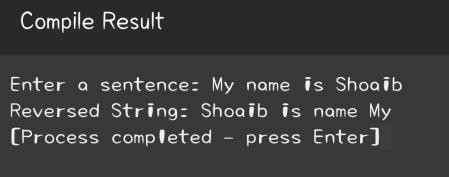
}

int main() {

char sentence[100];

printf("Enter a sentence: "); gets(sentence); reverseWords(sentence);

printf("Reversed String: %s", sentence); return 0;

}

Q. 3 Write a C program to count vowels, consonants, etc. Sol.-

#include <stdio.h>

int main() { char str[100];

int vowels = 0, consonants = 0, digits = 0, spaces = 0; int i = 0;

printf("Enter a string:\n"); gets(str);

while(str[i] != '\0') { if((str[i] >= 'a' && str[i] <= 'z') || (str[i] >= 'A' && str[i] <= 'Z')) { if(str[i] == 'a' || str[i] == 'e' || str[i] == 'i' || str[i] == 'o' || str[i] == 'u' || str[i] == 'A' || str[i] == 'E'

|| str[i] == 'I' || str[i] == 'O' || str[i] == 'U') { vowels++;

} else {

consonants++;

}

} else if(str[i] >= '0' && str[i] <= '9') { digits++;

} else if(str[i] == ' ') { spaces++;

}

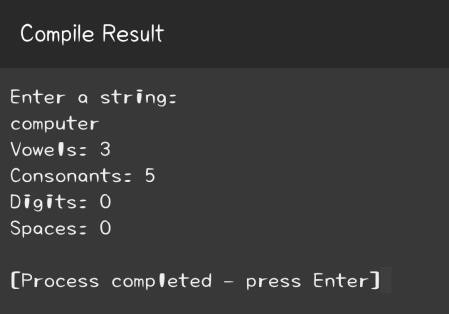
i++;

}

printf("Vowels: %d\n", vowels); printf("Consonants: %d\n", consonants);

printf("Digits: %d\n", digits); printf("Spaces: %d\n", spaces);

return 0;

}

Q. 4 Create a program to separate characters in a given string? Sol.-

#include <stdio.h> #include <string.h>

int main() { char str[100]; int i;

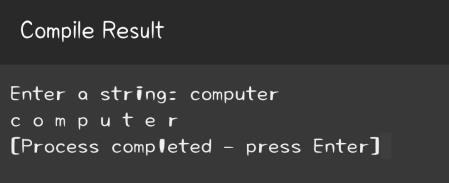
printf("Enter a string: "); gets(str);

for(i = 0; str[i] != '\0'; i++) {

printf("%c ", str[i]);

}

return 0;

}

Q. 5 Write a program to take two strings from user and concatenate them also add a space between them using strcat() function.

**Sample input: JAI GLA**

**Sample output: JAI GLA**

Sol.-

#include <stdio.h> #include <string.h> int main() {

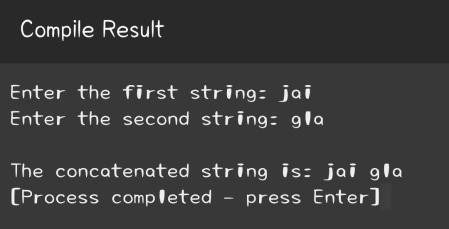
char str1[50], str2[50];

printf("Enter the first string: "); gets(str1);

printf("Enter the second string: "); gets(str2);

strcat(str1, " "); strcat(str1, str2); printf("\nThe concatenated string is: %s", str1);

return 0;

}

Q. 6 Write a C program to take a string from user and make it toggle its case i.e. lower case to upper case and upper case to lower case.

**Sample Input: HElLo wOrlD**

**Sample output: heLlO WoRLd**

Sol.-

#include <stdio.h>

int main() { char str[100]; int i;

printf("Enter a string: "); gets(str);

for(i = 0; str[i] != '\0'; i++) {

if(str[i] >= 'A' && str[i] <= 'Z') { str[i] = str[i] + 32;

}

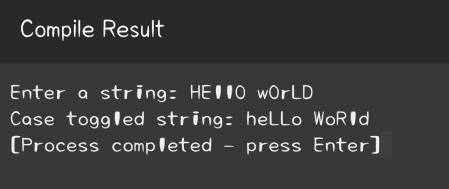
else if(str[i] >= 'a' && str[i] <= 'z') { str[i] = str[i] - 32;

}

}

printf("Case toggled string: %s", str);

return 0;

}

Q. 7 Write a C program to take two strings as input from user and check they are identical or not without using string functions.

**Sample input: Jai Gla Jai Gla**

**Sample output: Identical**

Sol.-

#include <stdio.h>

int main() { char str1[100], str2[100]; int i, flag = 0;

printf("Enter the first string: "); gets(str1);

printf("Enter the second string: "); gets(str2);

for(i = 0; str1[i] != '\0' || str2[i] != '\0'; i++) { if(str1[i] != str2[i]) { printf("Not Identical\n"); flag = 1; break;

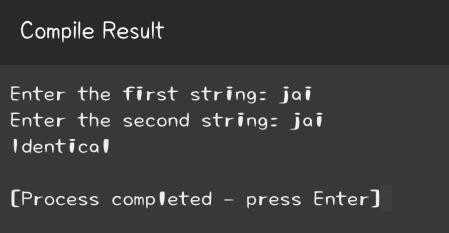
}

}

if(flag == 0) { printf("Identical\n");

}

return 0;

}

Q. 8 Write a C program to take a list of a student’s names from user by asking number of students and sort them alphabetical order.

**Sample Input: Bhisham Jayant**

**Abhishek Dhruv**

**Sample Output: Abhishek**

**Bhisham**

**Dhruv Jayant**

Sol.-

#include <stdio.h>

#include <string.h>

int main() { int i, j, n; char str[25][50], temp[50];

printf("How many students? "); scanf("%d", &n);

printf("Enter names of the students: "); for(i=0; i<n; i++) {

scanf("%s", str[i]);

}

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

if(strcmp(str[i], str[j]) > 0) { strcpy(temp, str[i]); strcpy(str[i], str[j]); strcpy(str[j], temp);

}

}

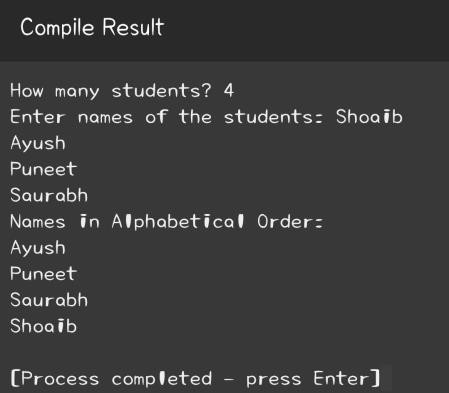
}

printf("Names in Alphabetical Order: \n"); for(i=0; i<n; i++) {

printf("%s\n", str[i]);

}

return 0;

}

# WEEK-10

Q. 1 Write a C program to find length of string using pointers. Sol.#include<stdio.h>

int string\_length(char\* ptr) { int length = 0;

while(\*ptr != '\0') {

length++; ptr++;

}

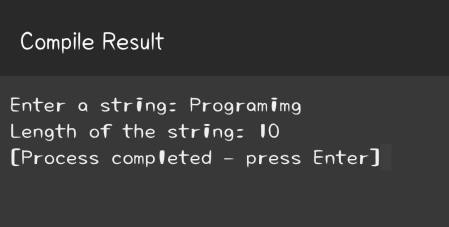
return length;

}

int main() { char str[50];

printf("Enter a string: "); gets(str); printf("Length of the string: %d", string\_length(str));

return 0;

}

Q. 2 Write a C program to copy one string to another using pointer. Sol.-

#include <stdio.h> void copy\_string(char

\*target, char \*source) { while(\*source) {

\*target = \*source; source++; target++;

}

\*target = '\0';

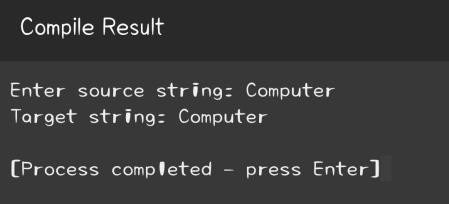
}

int main() {

char source[100], target[100];

printf("Enter source string: "); fgets(source, sizeof(source), stdin); copy\_string(target, source); printf("Target string: %s", target);

return 0;

}

Q. 3 Write a C program to concatenate two strings using pointers. Sol.#include<stdio.h>

void concatenate(char\* target, char\* source) { while(\*target) {

target++;

}

while(\*source) {

\*target = \*source; target++; source++;

}

\*target = '\0';

}

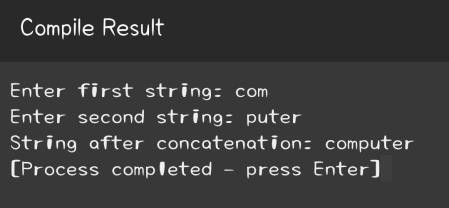
int main() {

char source[100], target[100];

printf("Enter first string: "); gets(target);

printf("Enter second string: "); gets(source); concatenate(target, source); printf("String after concatenation: %s", target);

return 0;

}

Q. 4 Write a C program to compare two strings using pointers. Sol.-

#include <stdio.h>

int compare\_strings(char \*str1, char \*str2) { while(\*str1 && (\*str1 == \*str2))

{

str1++; str2++;

}

return \*str1 - \*str2;

}

int main() {

char str1[100], str2[100];

printf("Enter first string: "); gets(str1);

printf("Enter second string: "); gets(str2); int result = compare\_strings(str1, str2); if(result

== 0) {

printf("Strings are equal.");

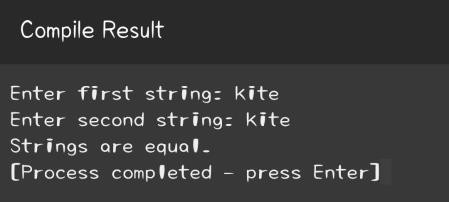
}

else {

printf("Strings are not equal.");

}

return 0;

}

Q. 5 WAP to find largest among three numbers using pointer Sol.#include

<stdio.h>

void find\_largest(int \*n1, int \*n2, int \*n3) { if(\*n1 > \*n2) {

if(\*n1 > \*n3) {

printf("The largest number is: %d", \*n1);

} else { printf("The largest number is: %d",

\*n3);

}

} else { if(\*n2 >

\*n3) {

printf("The largest number is: %d", \*n2);

} else { printf("The largest number is: %d",

\*n3);

}

}

}

int main() {

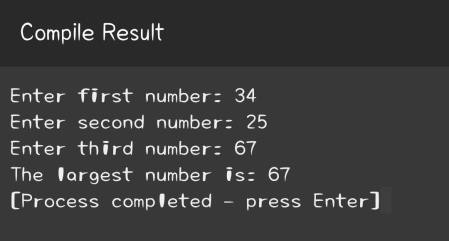
int n1, n2, n3;

printf("Enter first number: "); scanf("%d", &n1);

printf("Enter second number: "); scanf("%d", &n2);

printf("Enter third number: "); scanf("%d", &n3); find\_largest(&n1, &n2, &n3);

return 0;

}

Q. 6 WAP to find largest among three numbers using pointer. Sol.-

#include <stdio.h>

void find\_largest(int \*n1, int \*n2, int \*n3) { if(\*n1 > \*n2) {

if(\*n1 > \*n3) {

printf("The largest number is: %d", \*n1);

} else { printf("The largest number is: %d",

\*n3);

}

} else { if(\*n2 >

\*n3) {

printf("The largest number is: %d", \*n2);

} else { printf("The largest number is: %d",

\*n3);

}

}

}

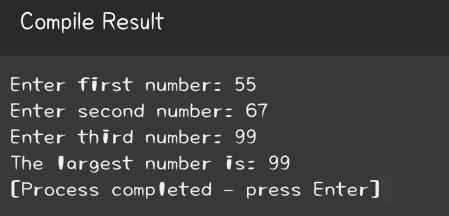
int main() {

int n1, n2, n3;

printf("Enter first number: "); scanf("%d", &n1);

printf("Enter second number: "); scanf("%d", &n2); printf("Enter third number: "); scanf("%d", &n3); find\_largest(&n1, &n2, &n3);

return 0;

}

Q. 7 WAP to find factorial of a number using pointer. Sol.-

#include <stdio.h>

void factorial(int \*num, int \*fact) {

\*fact = 1; for(int i = 1; i <=

\*num; i++) { \*fact \*= i;

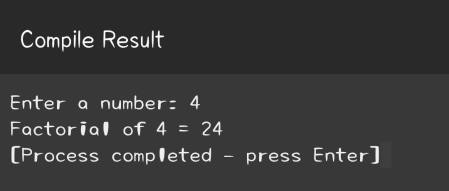
}

}

int main() { int num; int fact = 1;

printf("Enter a number: "); scanf("%d", &num); factorial(&num, &fact); printf("Factorial of %d = %d", num, fact);

return 0;

}

Q. 8 Write a program to print largest even number present in an array using pointer to an array. Sol.-

#include <stdio.h>

void largest\_even(int \*arr, int n) {

int largest = -1; for(int i = 0; i < n; i++) { if (\*(arr+i) % 2 == 0 && \*(arr+i) > largest) { largest = \*(arr+i);

}

}

if (largest != -1)

printf("The largest even number is: %d", largest); else

printf("No even number found");

}

int main() { int arr[100], n, i;

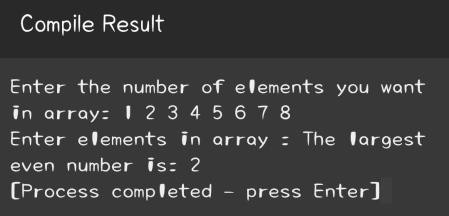
printf("Enter the number of elements you want in array: "); scanf("%d", &n);

printf("Enter elements in array : "); for(i = 0; i < n; i++) { scanf("%d", &arr[i]);

}

largest\_even(arr, n);

return 0;

}

Q. 9 WAP to find sum of elements of an array using array of pointer. Sol.-

#include <stdio.h>

int main() { int arr[5] = {1,

2, 3, 4, 5}; int \*ptr[5];

int sum = 0, i;

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

ptr[i] = &arr[i]; // Assign the address of each of array element.

}

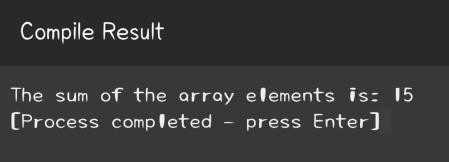
for(i = 0; i < 5; i++){

sum += \*ptr[i]; // Add the value at address stored in pointer.

}

printf("The sum of the array elements is: %d", sum);

return 0;

}

Q. 10 WAP to compute simple interest using pointers. Sol.-

#include <stdio.h>

void calculate\_simple\_interest(float \*p, float \*r, float \*t, float \*si) {

\*si = (\*p \* \*r \* \*t) / 100;

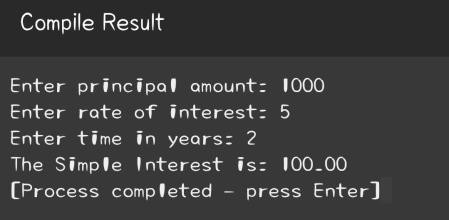
}

int main() {

float p, r, t, si;

printf("Enter principal amount: "); scanf("%f", &p); printf("Enter rate of interest: "); scanf("%f", &r); printf("Enter time in years: "); scanf("%f", &t); calculate\_simple\_interest(&p, &r, &t, &si); printf("The Simple Interest is: %.2f", si); return 0;

}



Q. 11 Write a program to print largest even number present in an array using pointer to an array. Sol.-

#include <stdio.h>

int find\_largest\_even(int \*arr, int n) {

int max\_even = -1; for(int i = 0; i < n; i++) { if(arr[i] % 2 == 0 && arr[i] > max\_even) {

max\_even = arr[i];

}

}

return max\_even;

}

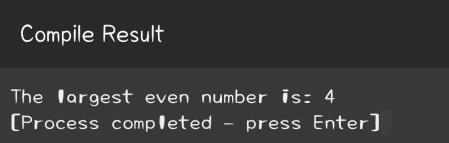
int main() { int arr[5] = {2, 4, 1, 3, 5}; int max\_even = find\_largest\_even(arr, 5); if(max\_even != -1) {

printf("The largest even number is: %d", max\_even);

} else { printf("No even number found in the array.");

}

return 0;

}

**WEEK-11**

Q. 1 Write a C function to return the maximum of three integers.

Sol.-

#include <stdio.h>

int max\_of\_three(int a, int b, int c) { int max = a;

if (b > max) {

max = b;

}

if (c > max) { max = c;

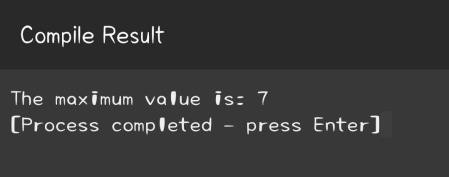
}

return max;

}

int main() {

int a = 3, b = 5, c = 7; int max = max\_of\_three(a, b, c); printf("The maximum value is: %d", max); return 0;

}

Q. 2 Write a C function to check if a given number is prime or not. Sol.-

#include <stdio.h>

int is\_prime(int num) { if(num <= 1)

return 0;

if(num <= 3)

return 1; if(num

% 2 == 0 ||

num % 3 == 0)

return 0;

for(int i = 5; i \* i <= num; i = i + 6) if(num % i == 0 || num % (i + 2) == 0)

return 0;

return 1;

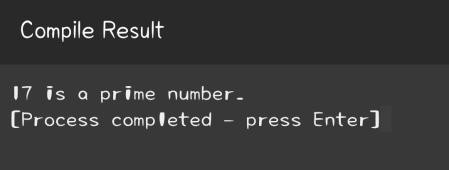
}

int main() {

int num = 17; if(is\_prime(num))

printf("%d is a prime number.", num); else

printf("%d is not a prime number.", num); return 0;

}

Q. 3 Write a C function to compute the factorial of a non-negative integer. Sol.-

#include <stdio.h>

int factorial(int n) { if(n == 0)

return 1; else

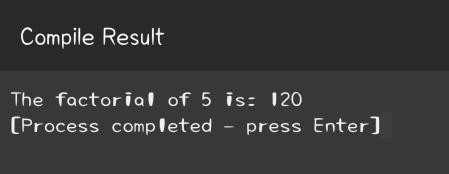
return n \* factorial(n-1);

}

int main() {

int num = 5;

printf("The factorial of %d is: %d", num, factorial(num)); return 0;

}

Q. 4 Write a C function to swap the values of two integers in actual arguments. Sol.-

#include <stdio.h>

void swap(int\* a, int\* b) { int temp = \*a;

\*a = \*b;

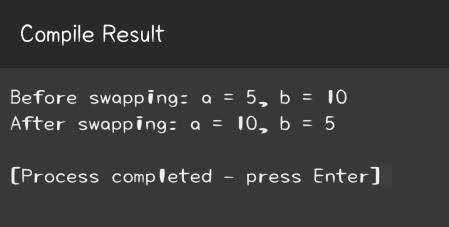
\*b = temp;

}

int main() {

int a = 5, b = 10; printf("Before swapping: a =

%d, b = %d\n", a, b); swap(&a, &b); printf("After swapping: a = %d, b = %d\n", a, b); return 0;

}

Q. 5 Write a C function to compute the sum and average of an array of integers. Sol.-

#include <stdio.h>

void sum\_and\_average(int arr[], int n, int\* sum, float\* avg) {

\*sum = 0; for(int i = 0; i

< n; i++) { \*sum += arr[i];

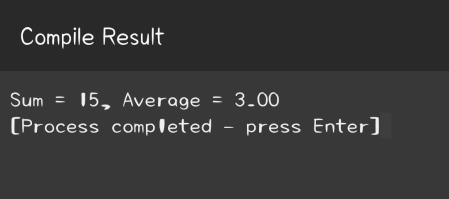
}

\*avg = (float)(\*sum) / n;

}

int main() {

int arr[] = {1, 2, 3, 4, 5}; int n = sizeof(arr) / sizeof(arr[0]); int sum = 0; float avg = 0.0f; sum\_and\_average(arr, n, &sum, &avg); printf("Sum = %d, Average = %.2f", sum, avg); return 0;

}

Q. 6 Write a C function to find the GCD (Greatest Common Divisor) of two nonnegative integers using Euclid's algorithm.

Sol.-

#include <stdio.h>

int gcd(int a, int b) { if(b == 0)

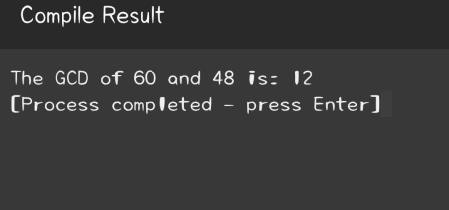
return a; else

return gcd(b, a % b);

}

int main() {

int num1 = 60, num2 = 48; printf("The GCD of %d and %d is: %d", num1, num2, gcd(num1, num2)); return 0;

}

Q. 7 Write a C function to check if a given string is a valid palindrome, considering only alphanumeric characters and ignoring cases.

Sol.-

#include <stdio.h> #include <string.h> #include <ctype.h>

int isPalindrome(char\* str) {

int start = 0, end = strlen(str) - 1;

while (start < end) {

if (!isalnum(str[start])) { start++;

} else if (!isalnum(str[end])) { end-

-;

} else if (tolower(str[start]) != tolower(str[end])) { return 0;

} else {

start++; end--;

}

}

return 1;

}

int main() {

char str[] = "A man, a plan, a canal: Panama";

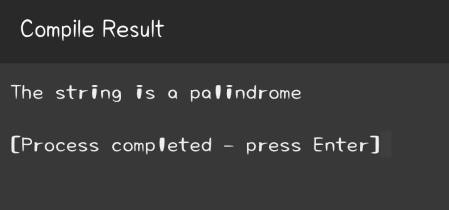
if(isPalindrome(str)) {

printf("The string is a palindrome\n");

} else { printf("The string is not a palindrome\n");

}

return 0;

}

Q. 8 Write a C function to calculate the sum and difference of two complex numbers. Sol.-

#include <stdio.h>

typedef struct complex

{ float real; float imag; } complex;

complex addComplex(complex n1, complex n2) { complex temp; temp.real =

n1.real + n2.real; temp.imag =

n1.imag + n2.imag; return temp;

}

complex subtractComplex(complex n1, complex n2) { complex temp; temp.real =

n1.real - n2.real; temp.imag =

n1.imag - n2.imag; return temp;

}

int main() {

complex n1 = {1.0, 2.0}, n2 = {3.0, 4.0}, result;

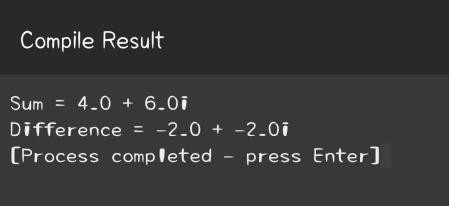
result = addComplex(n1, n2); printf("Sum = %.1f +

%.1fi\n", result.real, result.imag);

result = subtractComplex(n1, n2); printf("Difference =

%.1f + %.1fi", result.real, result.imag);

return 0;

}

H.O.T.S Questions

Q. 9 Write a C function to find the second largest and second smallest elements in an array of integers.

Sol.-

#include <stdio.h> #define SIZE 10

#define MAX 10000

void findSecondLargestSmallest(int arr[], int arrSize) { int i, first, second;

if (arrSize < 2) { printf(" Invalid Input "); return;

}

first = second = MAX;

for (i = 0; i < arrSize; i++) { if (arr[i] < first) { second

= first; first = arr[i];

}

else if (arr[i] < second && arr[i] != first) second = arr[i];

}

printf("The smallest element is %d and second smallest element is %d\n", first, second);

first = second = -MAX; for (i = 0; i < arrSize; i++) { if (arr[i] > first) { second = first; first = arr[i];

}

else if (arr[i] > second && arr[i] != first) second = arr[i];

}

printf("The largest element is %d and second largest element is %d", first, second);

}

int main() {

int numbers[SIZE], i;

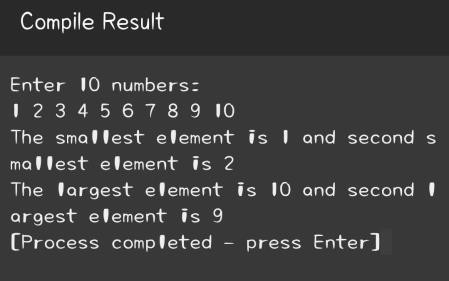
printf("Enter 10 numbers:\n"); for(i = 0; i < SIZE; i++) {

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

}

findSecondLargestSmallest(numbers, SIZE);

return 0;

}

Q. 10 Write a C function to find the number of occurrences of each unique element in an array. Sol.-

#include <stdio.h> #define MAX\_SIZE 100

void findElementCount(int arr[], int len) { int count[MAX\_SIZE] = {0};

for(int i = 0; i < len; i++) { count[arr[i]]++;

}

for(int i = 0; i < len; i++) { if(count[arr[i]] != 0) { printf("%d occurs %d times\n", arr[i], count[arr[i]]); count[arr[i]] = 0;

}

}

}

int main() {

int numbers[MAX\_SIZE], num, i;

printf("Enter number of elements to be stored in the array: "); scanf("%d", &num);

printf("Enter elements in array : \n"); for(i = 0; i < num; i++) { scanf("%d", &numbers[i]);

}

findElementCount(numbers, num);

return 0;

}

