**LAB TASK - 1**

1.Write a C program to find the sum of elements in an array

using pointers.

#include <stdio.h>

int main() {

int arr[10];

int i, n, sum = 0;

int \*pt;

printf("\n Enter the number of elements to store in the array: ");

scanf("%d", &n);

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

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

printf(" your element - %d : ", i + 1);

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

}

pt = arr;

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

sum = sum + \*pt;

pt++;

}

printf(" The sum of array is : %d\n\n", sum);

return 0;

}

2.Write a C program to swap the values of two integers

using pointers.

#include <stdio.h>

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

int temp;

temp = \*a;

\*a = \*b;

\*b = temp;

}

int main() {

int num1, num2;

int \*pt1, \*pt2;

printf("\n Enter the first number: ");

scanf("%d", &num1);

printf("\n Enter the second number: ");

scanf("%d", &num2);

pt1 = &num1;

pt2 = &num2;

printf("\n Before swapping: a = %d, b = %d", \*pt1, \*pt2);

swap(pt1, pt2);

printf("\n After swapping: a = %d, b = %d", \*pt1, \*pt2);

return 0;

}

3. Write a C program to reverse a string using pointers.

#include <stdio.h>

#include <string.h>

void reverse(char \*str) {

char \*start = str;

char \*end = str + strlen(str) - 1;

char temp;

while (end > start) {

temp = \*start;

\*start = \*end;

\*end = temp;

start++;

end--;

}

}

int main() {

char str[] = "IAM VINAY";

printf("Original string: %s\n", str);

reverse(str);

printf("Reversed string: %s\n", str);

return 0;

}

4. Write a C program to calculate the power of a number

using pointers to functions.

#include <stdio.h>

double power(double base, int exponent, double (\*power\_func)(double, int));

double power\_recursive(double base, int exponent) {

if (exponent == 0) {

return 1;

}

return base \* power\_recursive(base, exponent - 1);

}

double power\_iterative(double base, int exponent) {

double result = 1;

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

result \*= base;

}

return result;

}

int main() {

double base;

int exponent;

double result;

printf("Enter base number: ");

scanf("%lf", &base);

printf("Enter exponent: ");

scanf("%d", &exponent);

double (\*power\_recursive\_ptr)(double, int) = power\_recursive;

result = power(base, exponent, power\_recursive\_ptr);

printf("Power (recursive): %.2lf\n", result);

double (\*power\_iterative\_ptr)(double, int) = power\_iterative;

result = power(base, exponent, power\_iterative\_ptr);

printf("Power (iterative): %.2lf\n", result);

return 0;

}

double power(double base, int exponent, double (\*power\_func)(double, int))

{

return power\_func(base, exponent);

}

5.Write a C program that dynamically allocates memory for

a 2D array based on user input.

#include <stdio.h>

#include <stdlib.h>

int\*\* allocate\_2D\_array(int rows, int cols) {

int\*\* array = (int\*\*) malloc(rows \* sizeof(int\*));

if (array == NULL) {

printf("Memory allocation failed.\n");

exit(1);

}

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

array[i] = (int\*) malloc(cols \* sizeof(int));

if (array[i] == NULL) {

printf("Memory allocation failed.\n");

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

free(array[j]);

}

free(array);

exit(1);

}

}

return array;

}

void free\_2D\_array(int\*\* array, int rows) {

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

free(array[i]);

}

free(array);

}

int main() {

int rows, cols;

printf("Enter number of rows: ");

scanf("%d", &rows);

printf("Enter number of columns: ");

scanf("%d", &cols);

int\*\* array = allocate\_2D\_array(rows, cols);

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

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

printf("Enter element [%d][%d]: ", i, j);

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

}

}

printf("2D array:\n");

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

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

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

}

printf("\n");

}

free\_2D\_array(array, rows);

return 0;

}