**4. Arrays and Pointers and Functions:**

**a. Write a C program to find the minimum, maximum and average in an array of integers.**

#include <stdio.h>

#define MAX\_SIZE 100 // Maximum array size

int main()

{

int arr[MAX\_SIZE];

int i, max, min, size,avg=0,sum=0;

/\* Input size of the array \*/

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

scanf("%d", &size);

/\* Input array elements \*/

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

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

{

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

}

/\* Assume first element as maximum and minimum \*/

max = arr[0];

min = arr[0];

// Find maximum and minimum in all array elements.

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

{

/\* If current element is greater than max \*/

if(arr[i] > max)

{

max = arr[i];

}

/\* If current element is smaller than min \*/

if(arr[i] < min)

{

min = arr[i];

}

sum+=arr[i];

avg=sum/size;

}

/\* Print maximum and minimum element \*/

printf("\nMaximum element = %d\n", max);

printf("Minimum element = %d\n", min);

printf("Average = %d", avg);

return 0;

}

**Output**

**b. Write a functions to compute mean, variance, Standard Deviation, sorting of n elements in single dimension array.**

#include <stdio.h>

#include <math.h>

int cal(int [10],int);

#define MAXSIZE 10

void main()

{

float x[MAXSIZE];

int i, n;

printf("Enter the value of N \n");

scanf("%d", &n);

printf("Enter %d real numbers \n", n);

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

{

scanf("%f", &x[i]);

}

cal(x,n);

}

int cal(int x[10],int n)

{

int i, sum=0, sum1 = 0;

float average, variance, std\_deviation;

/\* Compute the sum of all elements \*/

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

{

sum = sum + x[i];

}

average = sum / (float)n;

/\* Compute variance and standard deviation \*/

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

{

sum1 = sum1 + pow((x[i] - average), 2);

}

variance = sum1 / (float)n;

std\_deviation = sqrt(variance);

printf("Average of all elements = %.2f\n", average);

printf("variance of all elements = %.2f\n", variance);

printf("Standard deviation = %.2f\n", std\_deviation);

}

**Output:**

**c. Write a C program that uses functions to perform the following:**

**i. Addition of Two Matrices**

#include <stdio.h>

void matrixAddition(int a[10][10], int b[10][10], int sum[10][10]);

int rows, columns;

int main()

{

int a[10][10], b[10][10], sum[10][10], i, j;

/\* get the number of rows and columns from user \*/

printf("Enter the no of rows and columns:");

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

/\* input first matrix \*/

printf("Enter the input for first matrix:\n");

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

{

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

{

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

}

}

/\* input second matrix \*/

printf("Enter the input for second matrix:\n");

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

{

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

{

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

}

}

/\* matrix addtion \*/

matrixAddition(a, b, sum);

/\* print the results \*/

printf("\nResult of Matrix Addition:\n");

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

{

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

{

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

printf("\t");

}

printf("\n");

}

return 0;

}

/\* adds two matrices and stores the output in third matrix \*/

void matrixAddition(int a[10][10], int b[10][10], int sum[10][10])

{

int i, j;

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

{

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

{

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

}

}

return;

}

**Output:**

**ii. Multiplication of Two Matrices**

#include <stdio.h>

int mulmatrix(int [10][10],int [10][10],int,int,int,int);

int main() {

int m, n, p, q, c, d, k,first[10][10], second[10][10];

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

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

printf("Enter elements of first matrix\n");

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

for (d = 0; d < n; d++)

scanf("%d", &first[c][d]);

}

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

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

if (n != p)

printf("The matrices can't be multiplied with each other.\n");

else{

printf("Enter elements of second matrix\n");

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

for (d = 0; d < q; d++)

scanf("%d", &second[c][d]);

}

mulmatrix(first,second,m,n,p,q);

}

int mulmatrix(int first[10][10],int second[10][10],int m,int n,int p,int q)

{

int c,d,k,sum=0,multiply[10][10];

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

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

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

sum = sum + first[c][k]\*second[k][d];

}

multiply[c][d] = sum;

sum = 0;

}

}

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

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

for (d = 0; d < q; d++)

printf("%d\t", multiply[c][d]);

printf("\n");

}

}

return 0;

}

**Output:**

**d. Write C programs that use both recursive and non-recursive functions.**

**i. To find the factorial of a given integer.**

#include <stdio.h>

int recfactorial(int x);

int nonrecfactorial(int x);

void main()

{

int n, a, b;

printf("Enter any number\n");

scanf("%d", &n);

a = recfactorial(n);

printf("The factorial of a given number using recursion is %d \n", a);

b = nonrecfactorial(n);

printf("The factorial of a given number using nonrecursion is %d ", b);

}

int recfactorial(int x){

int f;

if(x == 0){

return(1);

}

else{

f = x \* recfactorial(x - 1);

return(f);

}

}

int nonrecfactorial(int x){

int i, f = 1;

for(i = 1;i <= x; i++)

{

f = f \* i;

}

return(f);

}

**ii. To find the GCD (greatest common divisor) of two given integers.**

#include <stdio.h>

int recgcd(int x, int y);

int nonrecgcd(int x, int y);

void main()

{

int a, b, c, d;

printf("Enter two numbers a and b");

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

c = recgcd(a, b);

printf("The gcd of two numbers using recursion is %d\n", c);

d = nonrecgcd(a, b);

printf("The gcd of two numbers using nonrecursion is %d", d);

}

int recgcd(int x, int y){

if(y == 0){

return(x);

}

else{

return(recgcd(y, x % y));

}

}

int nonrecgcd(int x, int y){

int z;

while(x % y != 0){

z = x % y;

x = y;

y = z;

}

return(y);

}

**Output**

**iii. To find x^n**

#include <stdio.h>

long power (int, int);

long nonrec (int x, int n);

int main(){

int n, x;

long result;

printf("Enter a number: ");

scanf("%d", &x);

printf("Enter it's power: ");

scanf("%d", &n);

result = power (x, n);

printf("%d^%d is %ld", x, n, result);

result = nonrec (x,n);

printf("%d^%d is %ld", x, n, result);

return 0;

}

long power (int x, int n){

if (n!=0){

return (x \* power(x, n - 1));

}

return 1;

}

long nonrec (int x, int n)**{**

int i,y=1;

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

y=y\*x;

**}**

printf(“%d”,y);

**}**

**Output**

**e. Write a program for reading elements using pointer into array and display the values using array.**

#include <stdio.h>

int main(){

int arr[10]; //declare integer array

int \*pa; //declare an integer pointer

int i;

pa=&arr[0]; //assign base address of array

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

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

{

scanf("%d",pa+i); //reading through pointer

}

printf("\nEntered array elements are:\n");

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

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

}

return 0;

}

**Output:**

**f. Write a program for display values reverse order from array using pointer.**

#include<stdio.h>

#define MAX 30

void main()

{

   int size, i, arr[MAX];

   int \*ptr;

   ptr = &arr[0];

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

   scanf("%d", &size);

   printf("\nEnter %d integers into array: ", size);

   for (i = 0; i < size; i++)

{

      scanf("%d", ptr);

      ptr++;

   }

ptr = &arr[size - 1];

   printf("\nElements of array in reverse order are :");

   for (i = size - 1; i >= 0; i--)

{

      printf("\nElement%d is: %d ", i, \*ptr);

      ptr--;

   }

}

**Output:**

**g. Write a program through pointer variable to sum of n elements from array.**

#include<stdio.h>

#include<conio.h>

void main()

{

int numArray[10];

int i, sum = 0;

int \*ptr;

printf("\nEnter 10 elements : ");

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

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

ptr = numArray; /\* a=&a[0] \*/

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

{

sum = sum + \*ptr;

ptr++;

}

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

}

**Output:**