Exercise 1:
1. Write a C program to print a block F using hash (#), where the F has a height of six characters and width of five and four characters.

Source Code:

```
#include <stdio.h>
int main()
{
printf("######\n");
printf("#\n");
printf("#\n");
printf("#####\n");
printf("#\n");
printf("#\n");
printf("#\n");
return(0);
}
```

Output:

```
######
#
#
#####
#
#
#
```

Exercise 1:
2. Write a C program to compute the perimeter and area of a rectangle with a height of 7 inches and a width of 5 inches.

Source Code:

```
#include <stdio.h>
/* height and width of a rectangle in inches */
int width;
int height;
int area;
int perimeter;
int main() {
height = 7;
```

```
width = 5;
perimeter = 2*(height + width);
printf("\nPerimeter of the rectangle = %d inches\n", perimeter);
area = height * width;
printf("\nArea of the rectangle = %d square inches\n", area);
return(0);
}
```

Output:
Perimeter of the rectangle = 24 inches
Area of the rectangle = 35 square inches

Exercise 1:
3. Write a C program to display multiple variables and sizes

```
/*
Write a C program to display multiple variables and sizes.
*/
#include <stdio.h>
int main()
{
//variables declarartion & initializattion
char ch = 'A';
char title[20] = "C Program";
short s = 32767;
int i = 2147483647;
int num = 15;
long li = 2147483647;
long long lli = 9223372036854775807;
float f = 3.123456;
double df = 1.123456789012345678;
long double ldf = 1.123456;
unsigned int ui = 4294967295;

//Display of values of variables
printf("Variables, Formats, and Values: \n");
printf("Character              = %c\n", ch);
printf("String                 = %s\n", title);
printf("Short integer          = %d\n", s);
printf("integer                = %d\n", i);
```

```
printf("integer binary octal hexdecimal = %d %o %x\n", num,num,num);
printf("long integer          = %ld\n", li);
printf("long long integer     = %lld\n", lli);
printf("float                 = %f\n", f);
printf("float with exponent   = %e\n", f);
printf("double                = %lf\n", df);
printf("long double           = %Lf\n", ldf);
printf("unsigned integer      = %u \n\n", ui);

//Display of sizeof variables
printf("Variable Sizes: \n");
printf("character size        = %d\n", sizeof(ch));
printf("string size           = %d\n", sizeof(title));
printf("integer size          = %d\n", sizeof(i));
printf("long integer size     = %d\n", sizeof(li));
printf("long long int size    = %d\n", sizeof(lli));
printf("float size            = %d\n", sizeof(f));
printf("double size           = %d\n", sizeof(df));
printf("size of long double   = %d\n", sizeof(ldf));
return 0;
}
```

Output:

```
Variables, Formats, and Values:
Character             = A
String                = C Program
Short integer         = 32767
integer               = 2147483647
integer binary octal hexdecimal = 15 17 f
long integer          = 2147483647
long long integer     = 9223372036854775807
float                 = 3.123456
float with exponent   = 3.123456e+000
double                = 1.123457
long double           = -0.000000
unsigned integer      = 4294967295

Variable Sizes:
character size        = 1
```

```
string size           = 20
integer size          = 4
long integer size     = 4
long long int size    = 8
float size            = 4
double size           = 8
size of long double   = 12
```

OR

3. Write a C program to display multiple variables

```
/*
Variable declaration:
int a = 123, b = 12345;
long ax = 1234567890;
short s = 2023;
float x = 3.23459;
double dx = 1.3415927;
char c = 'A';
unsigned long ux = 1234567890;
*/
#include <stdio.h>
int main()
{
int a = 123, b = 12345;
long ax = 1234567890;
short s = 2023;
float x = 3.23459;
double dx = 1.3415927;
char c = 'A';
unsigned long ux = 1234567890;

printf("a + c = %d\n", a + c);
printf("x + c = %f\n", x + c);
printf("dx + x = %f\n", dx + x);
printf("((int) dx) + ax = %ld\n", ((int) dx) + ax);
printf("a + x = %f\n", a + x);
printf("s + b = %d\n", s + b);
printf("ax + b = %ld\n", ax + b);
printf("s + c = %hd\n", s + c);
```

```
printf("ax + c = %ld\n", ax + c);
printf("ax + ux = %lu\n", ax + ux);
return 0;
}
```

Output:
a + c = 188
x + c = 68.234590
dx + x = 4.576183
((int) dx) + ax = 1234567891
a + x = 126.234590
s + b = 14368
ax + b = 1234580235
s + c = 2088
ax + c = 1234567955
ax + ux = 2469135780

Exercise 2:
1. Write a C program to calculate the distance between the two points.

```
/*
Formula: square root of ((x2-x1)*(x2-x1) - (y2-y1)*(y2-y1))
*/
#include <stdio.h>
#include <math.h>
int main() {
float x1, y1, x2, y2, gdistance;
printf("Input x1: ");
scanf("%f", &x1);
printf("Input y1: ");
scanf("%f", &y1);
printf("Input x2: ");
scanf("%f", &x2);
printf("Input y2: ");
scanf("%f", &y2);
gdistance = ((x2-x1)*(x2-x1))+((y2-y1)*(y2-y1));
printf("Distance between the given points: %.4f", sqrt(gdistance));
printf("\n");
return 0;
}
```

Output:
Input x1: 2
Input y1: 2
Input x2: 6
Input y2: 4
Distance between the given points: 4.4721

Exercise 2:
2. Write a C program that accepts 4 integers p, q, r, s from the user where r and s are positive and p is even. If q is greater than r and s is greater than p and if the sum of r and s is greater than the sum of p and q print "Correct values", otherwise print "Wrong values".

```
#include <stdio.h>
int main()
{
int p, q, r, s;
printf("\nInput [even] integer for p: ");
scanf("%d", &p);
printf("\nInput any integer for q: ");
scanf("%d", &q);
printf("\nInput [+ve] integer for r: ");
scanf("%d", &r);
printf("\nInput [+ve] integer for s: ");
scanf("%d", &s);

if((q > r) && (s > p) && ((r+s) > (p+q)) && (r > 0) && (s > 0) &&
(p%2==0))
{
    printf("\nCorrect values\n");
}
else {
    printf("\nWrong values\n");
}

return 0;
}
```

Output:

Input [even] integer for p: 4

Input any integer for q: 7

Input [+ve] integer for r: 6

Input [+ve] integer for s: 5

Wrong values

Exercise 3:

1. Write a c program to make a single calculator to add, subtract, multiply or divide using switch case

```
/*
Write a c program to make a single calculator to add, subtract,
multiply or divide using switch case
*/
#include <stdio.h>
int main()
{
 char op;
 float num1, num2, result=0.0f;
 /* Print welcome message */
 printf("Welcome to Simple Calculator\n");
 printf("------------------------------------\n");
 printf("Enter [number 1] [+ - * /] [number 2]\n");
 /* Input two number and operator from user */
 scanf("%f %c %f", &num1, &op, &num2);
 /* Switch the value and perform action based on operator*/
 switch(op)
 {
    case '+':
        result = num1 + num2;
        break;
    case '-':
        result = num1 - num2;
        break;
```

```
    case '*':
        result = num1 * num2;
        break;
    case '/':
        result = num1 / num2;
        break;
    default:
        printf("Invalid operator");
 }
 /* Prints the result */
 printf("%.2f %c %.2f = %.2f", num1, op, num2, result);
 return 0;
}
```

Output:
Welcome to Simple Calculator

Enter [number 1] [+ - * /] [number 2]
10 + 5
10.00 + 5.00 = 15.00

Exercise 3:
2. Write a program in C which is a Menu-Driven Program to compute the area of the various geometrical shape.

```
#include <stdio.h>
#include<string.h>
void main ()
{
int choice,r,l,w,b,h;
float area;
char shape_name[30];
printf("Menu\n");
printf("1 : Area of Circle\n");
printf("2 : Area of Rectangle\n");
printf("3 : Area of Triangle\n");
printf("Select your choice : ");
scanf("%d",&choice);
switch(choice)
```

```
{    case 1:
         printf("Enter radious of the circle : ");
         scanf("%d",&r);
         area=3.14*r*r;
         strcpy(shape_name,"Circle");
         break;
     case 2:
        // strcpy("Rectangle",shape_name);
         printf("Enter length and width of the rectangle : ");
         scanf("%d%d",&l,&w);
         area=l*w;
         strcpy(shape_name,"Rectangle");
         break;
     case 3:
        // strcpy("Triangle",shape_name);
         printf("Enter the base and hight of the triangle :");
         scanf("%d%d",&b,&h);
         area=.5*b*h;
         strcpy(shape_name,"Triangle");
         break;
}
printf("The area of %s is : %5.2f\n",shape_name,area);
}
```

Output:
Menu
1 : Area of Circle
2 : Area of Rectangle
3 : Area of Triangle
Select your choice : 2

Enter length and width of the rectangle : 4 6
The area of Rectangle is : 24.00

Exercise 3:

3. Write a C program to calculate the factorial of a given number.

```
#include <stdio.h>
void main(){
    int i,f=1,num;
    printf("Input the number : ");
    scanf("%d",&num);
    for(i=1;i<=num;i++)
        f=f*i;
    printf("The Factorial of %d is: %d\n",num,f);
}
```

Output:
Input the number : 7
The Factorial of 7 is: 5040

Exercise 4:
1. Write a program in C to display the n terms of even natural number and their sum.

```
#include <stdio.h>
void main()
{
int i,n,sum=0;
printf("Enter number of terms : ");
scanf("%d",&n);
printf("\nThe even numbers are :");
for(i=1;i<=n;i++)
{
    printf("%d ",2*i);
    sum+=2*i;
}

printf("\nSum of Natural Even Numbers upto %d terms : %d \n",n,sum);
}
```

Output:
Enter number of terms: 5

The even numbers are: 2 4 6 8 10
Sum of Natural Even Numbers upto 5 terms: 30

Exercise 4:
2. Write a program in C to display the n terms of harmonic series and their sum. 1 + 1/2 + 1/3 + 1/4 + 1/5 ... 1/n terms.

```
#include <stdio.h>
void main()
{
int i,n;
float s=0.0;
printf("Enter the number of terms: ");
scanf("%d",&n);
printf("\n\n");
for(i=1;i<=n;i++)
{
if(i<n)
{
printf("1/%d + ",i);
s+=1/(float)i;
}
if(i==n)
{
printf("1/%d ",i);
s+=1/(float)i;
}
}
printf("\nSum of Series upto %d terms: %f \n",n,s);
}
```

Output:
Enter the number of terms: 3
1/1 + 1/2 + 1/3
Sum of Series upto 3 terms: 1.833333

Exercise 4:
3. Write a C program to check whether a given number is an Armstrong number or not.

```
/*
Armstrong numbers are 0, 1, 153, 370, 371, 407, etc.
Examples
If the entered number is 371, then 371 = 3*3*3 + 7*7*7 + 1*1*1 = 27 + 343
+ 1 = 371.
```

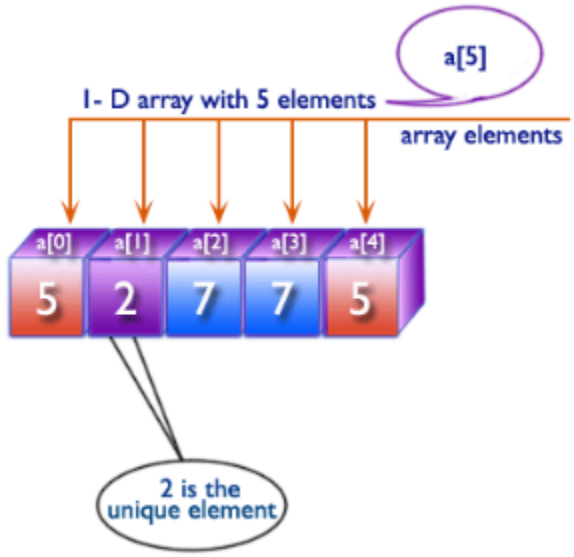
```
Since the sum of the cubes of every digit is equal to 371, the output will
Be "The given number is an Armstrong number."
If the entered number is 150, then 150 = 1*1*1 + 5*5*5 + 0*0*0 = 1 + 125 +
0 = 126.
Since the sum of the cubes of every digit is not equal to 150, the output
will be "The given number is not an Armstrong number"
*/

#include <stdio.h>
void main(){
int num,r,sum=0,temp;
printf("Enter a number: ");
scanf("%d",&num);
for(temp=num;num!=0;num=num/10){
    r=num % 10;
    sum=sum+(r*r*r);
}
if(sum==temp)
    printf("%d is an Armstrong number.\n",temp);
else
    printf("%d is not an Armstrong number.\n",temp);
}
```

Output:
Input a number: 221
221 is not an Armstrong number.

Input a number: 407
407 is an Armstrong number.

Exercise 5:

1. Write a program in C to print all unique elements in an array.




```
#include <stdio.h>
int main()
{
    int arr1[100], n,ctr=0;
    int i, j, k;
       printf("\n\nPrints all unique elements of your array:\n");
       printf("-----------------------------------------\n");
       printf("Enter the number of elements to be stored in the array: ");
       scanf("%d",&n);
       printf("___ Enter %d elements in the array ___\n",n);
       for(i=0;i<n;i++)
            {
          printf("element - %d : ",i);
          scanf("%d",&arr1[i]);
        }
    printf("\nThe unique elements found in the array are: \n");
    for(i=0; i<n; i++)
    {   //printf("\ni: %d",i);
        ctr=0;
        for(j=0,k=n; j<k; j++)
        {    //printf("\n j: %d",j);
             //printf(" arr[i]: %d", arr1[i]);
             //printf(" arr[j]: %d", arr1[j]);
            /*Increment the counter when the search value is duplicate.*/
```

```
            if (i!=j)
            {
               if(arr1[i]==arr1[j])
              {
                 ctr++;
              }
             }
         }
        if(ctr==0)
         {
           printf("\n\n%d ",arr1[i]);
         }
    }
       printf("\n\n");
}
```

Output:
The unique elements found in the array are:
7

i: 0
j: 0 arr[i]: 4 arr[j]: 4
j: 1 arr[i]: 4 arr[j]: 6
j: 2 arr[i]: 4 arr[j]: 4
j: 3 arr[i]: 4 arr[j]: 6
j: 4 arr[i]: 4 arr[j]: 7
i: 1
j: 0 arr[i]: 6 arr[j]: 4
j: 1 arr[i]: 6 arr[j]: 6
j: 2 arr[i]: 6 arr[j]: 4
j: 3 arr[i]: 6 arr[j]: 6
j: 4 arr[i]: 6 arr[j]: 7
i: 2
j: 0 arr[i]: 4 arr[j]: 4
j: 1 arr[i]: 4 arr[j]: 6
j: 2 arr[i]: 4 arr[j]: 4
j: 3 arr[i]: 4 arr[j]: 6
j: 4 arr[i]: 4 arr[j]: 7
i: 3
j: 0 arr[i]: 6 arr[j]: 4
j: 1 arr[i]: 6 arr[j]: 6
j: 2 arr[i]: 6 arr[j]: 4
j: 3 arr[i]: 6 arr[j]: 6

```
 j: 4 arr[i]: 6 arr[j]: 7
i: 4
 j: 0 arr[i]: 7 arr[j]: 4
 j: 1 arr[i]: 7 arr[j]: 6
 j: 2 arr[i]: 7 arr[j]: 4
 j: 3 arr[i]: 7 arr[j]: 6
 j: 4 arr[i]: 7 arr[j]: 7
```

Exercise 5:
2. Write a program in C to separate odd and even integers in separate arrays.

```
/*
arr1 - to input all numbers
arr2 - to copy even numbers
arr3 - to copy odd numbers
*/

#include <stdio.h>
void main()
{
int arr1[10], arr2[10], arr3[10];
int i,j=0,k=0,n;
printf("\n\nSeparate odd and even integers into separate arrays:\n");
printf(" \n");
printf("Enter the number of elements to be stored in Arr1:");
scanf("%d",&n);
printf("Enter %d elements in the array :\n",n);
for(i=0;i<n;i++)
{
    printf("Element - %d : ",i);
    scanf("%d",&arr1[i]);
}
for(i=0;i<n;i++)
{
    if (arr1[i]%2 == 0)
    {
        arr2[j] = arr1[i];
        j++;
    }
```

```
    else
    {
        arr3[k] = arr1[i];
        k++;
    }
}
printf("\nThe Even elements copied to Arr2 are : \n");
for(i=0;i<j;i++)
{
printf("%d ",arr2[i]);
}
printf("\nThe Odd elements copied to Arr3 are :\n");
for(i=0;i<k;i++)
{
printf("%d ", arr3[i]);
}
printf("\n\n");
}
```

Output:
Separate odd and even integers into separate arrays:

Enter the number of elements to be stored in Arr1:5
Enter 5 elements in the array :
Element - 0 : 100
Element - 1 : 105
Element - 2 : 107
Element - 3 : 110
Element - 4 : 120

The Even elements copied to Arr2 are :
100 110 120
The Odd elements copied to Arr3 are :
105 107

Exercise 5:
3. Write a program in C to sort elements of array in ascending order.

```
#include <stdio.h>
void main()
{
```

```
int arr1[100];
int n, i, j, tmp;
printf("\n\nSort elements of array in ascending order :\n ");
printf(" \n");
printf("What is the size of array : ");
scanf("%d", &n);
printf("Enter %d elements in the array :\n",n);

for(i=0;i<n;i++)
{
    printf("Element - %d : ",i);
    scanf("%d",&arr1[i]);
}

for(i=0; i<n; i++)
{
    for(j=i+1; j<n; j++)
    {
        if(arr1[j] <arr1[i])
        {
            tmp = arr1[i];
            arr1[i] = arr1[j];
            arr1[j] = tmp;
        }
    }
}

printf("\nElements of array in sorted ascending order:\n");

for(i=0; i<n; i++)
{
    printf("%d ", arr1[i]);
}

printf("\n\n");
}
```

Output:
Sort elements of array in ascending order :

What is the size of array : 5
Enter 5 elements in the array :
Element - 0 : 5
Element - 1 : 9
Element - 2 : 3
Element - 3 : 6
Element - 4 : 1

Elements of array in sorted ascending order:
1 3 5 6 9

Exercise 6:
1. Write a program in C for the multiplication of two square Matrices.

```
#include <stdio.h>
void main()
{
int arr1[50][50],brr1[50][50],crr1[50][50],i,j,k,arows,acols,brows,bcols,sum=0;
printf("\n\nMultiplication of two Matrices :\n");
printf(" \n");
printf("\nEnter the rows and columns of first matrix : ");
scanf("%d %d",&arows,&acols);
printf("\nEnter the rows and columns of second matrix : ");
scanf("%d %d",&brows,&bcols);

if(acols!=brows){
    printf("Mutiplication of Matrix is not possible.");
    printf("\nColumns of 1st matrix and Rows of 2nd matrix must be
same.");
}
else
{
    printf("Enter elements in the first matrix :\n");
    for(i=0;i<arows;i++)
    {
        for(j=0;j<acols;j++)
        {
            printf("Element - [%d],[%d] : ",i,j);
            scanf("%d",&arr1[i][j]);
        }
    }
```

```
printf("Enter elements in the second matrix :\n");
for(i=0;i<brows;i++)
{
    for(j=0;j<bcols;j++)
    {
        printf("element - [%d],[%d] : ",i,j);
        scanf("%d",&brr1[i][j]);
    }
}

printf("\n\nThe First matrix is :");
for(i=0;i<arows;i++)
{
    printf("\n");
    for(j=0;j<acols;j++)
        printf("%d\t",arr1[i][j]);
}

printf("\n\nThe Second matrix is :");
for(i=0;i<brows;i++)
{
    printf("\n");
    for(j=0;j<bcols;j++)
        printf("%d\t",brr1[i][j]);
}

//multiplication of matrix
for(i=0;i<arows;i++)
    for(j=0;j<bcols;j++)
        crr1[i][j]=0;
for(i=0;i<arows;i++) //row of first matrix
{
    for(j=0;j<bcols;j++) //column of second matrix
    {
        sum=0;
        for(k=0;k<acols;k++)
        {
            sum=sum+arr1[i][k]*brr1[k][j];
        }
```

```
                crr1[i][j]=sum;
            }
        }
        printf("\n\nThe multiplication of two matrices is : ");
        for(i=0;i<arows;i++)
        {    printf("\n");
            for(j=0;j<bcols;j++)
            {
               printf("%d\t",crr1[i][j]);
            }
        }
}  //closing brace for else block
printf("\n\n");
}
```

Output:

```
Multiplication of two Matrices :
Enter the rows and columns of first matrix : 2 2
Enter the rows and columns of second matrix : 2 2
Enter elements in the first matrix :
Element - [0],[0] : 5
Element - [0],[1] : 0
Element - [1],[0] : 2
Element - [1],[1] : 3
Enter elements in the second matrix :
element - [0],[0] : 4
element - [0],[1] : 9
element - [1],[0] : 2
element - [1],[1] : 1

The First matrix is :
5       0
2       3

The Second matrix is :
4       9
2       1

The multiplication of two matrices is :
20      45
14      21
```

Exercise 6:
2. Write a program in C to find the transpose of a given matrix.

```
#include <stdio.h>
void main()
{
int arr1[50][50],brr1[50][50],i,j,k=0,r,c;
printf("\n\nTranspose of a Matrix :\n");
printf(" \n");
printf("\nInput the rows and columns of the matrix : ");
scanf("%d %d",&r,&c);
printf("Input elements in the first matrix :\n");

for(i=0;i<r;i++)
{
    for(j=0;j<c;j++)
    {
    printf("element - [%d],[%d] : ",i,j);
    scanf("%d",&arr1[i][j]);
    }
}

printf("\nThe matrix is :\n");
for(i=0;i<r;i++)
{
    printf("\n");
    for(j=0;j<c;j++)
    printf("%d\t",arr1[i][j]);
}

for(i=0;i<r;i++)
{
    for(j=0;j<c;j++)
    {
    brr1[j][i]=arr1[i][j];
    }
}

printf("\n\nThe transpose of a matrix is : ");
for(i=0;i<c;i++){
```

```
    printf("\n");
    for(j=0;j<r;j++){
        printf("%d\t",brr1[i][j]);
    }
}
printf("\n\n");
}
```

Output:
Transpose of a Matrix :

Input the rows and columns of the matrix : 2 2
Input elements in the first matrix :
element - [0],[0] : 5
element - [0],[1] : 10
element - [1],[0] : 7
element - [1],[1] : 15

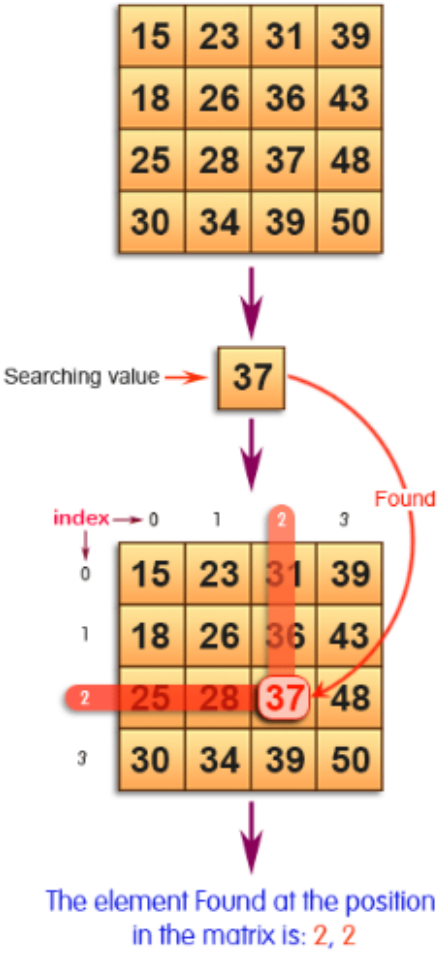
The matrix is :

5 10
7 15

The transpose of a matrix is :
5 7
10 15

Exercise 7:

1. Write a program in C to search an element in a row wise and column wise sorted matrix.




```
#include <stdio.h>
void main()    {
int a[10][10],n,m,i,j,x;
printf("Enter size of matrix Rows Cols: ");
scanf("%d %d",&n,&m);
printf("\nEnter the elements of matrix:\n");
for(i=0;i<n;i++)
{
    for(j=0;j<m;j++)
    {   printf("Element [%d],[%d] : ",i,j);
        scanf("%d",&a[i][j]);
    }
}
```

```
printf("\nEnter the search element: ");
scanf("%d",&x);

printf("\nGiven matrix is:\n");
for(i=0;i<n;i++)
{
    for(j=0;j<m;j++)
    {
        printf("%d\t",a[i][j]);
    }
    printf("\n");
}

i=0;
j=m-1;
while(i<n&&j>=0)
{
    if(a[i][j]==x)
    {
        printf("\nElement %d is found at %d, %d\n",x,i+1,j+1);
        return;
    }

if(a[i][j]<x)
    i++;
else
    j--;
}

printf("\nElement %d is not found\n",x);
}
```

Output:

Enter size of matrix Rows Cols: 2 2

Enter the elements of matrix:
Element [0],[0] : 4
Element [0],[1] : 6
Element [1],[0] : 9

Element [1],[1] : 10

Enter the search element: 9

Given matrix is:
4 6
9 10

Element 9 is found at 2, 1

Exercise 7:
2. Write a program in C to print individual characters of string in reverse order.

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
void main()
{
char str[100]; /* Declares a string of size 100 */
int len,i;
printf("\n\nPrint individual characters of string in reverse order :\n");
printf(" \n");
printf("Input the string : ");
fgets(str, sizeof str, stdin);
len=strlen(str);
printf("The characters of the string in reverse are : \n");
for(i=len;i>=0;i--)
{
    printf("%c ", str[i]);
}
printf("\n");
}
```

Output:
Print individual characters of string in reverse order :

Input the string : California
The characters of the string in reverse are :

a i n r o f i l a C

Exercise 8:

1. Write a program in C to compare two strings without using string library functions.

```
/* Write a program in C to compare two strings without using string
library functions.
*/
#include<stdio.h>
int main() {
char str1[30], str2[30];
int i;
printf("\nEnter string 1 :");
gets(str1);
printf("\nEnter string 2 :");
gets(str2);
i = 0;
while (str1[i] == str2[i] && str1[i] != '\0')
    i++; //Increments counter i until a char in strings is unequal &
string1 reaches its end

//ASCII value for '\0' is indeed 0
if (str1[i] > str2[i]) //true if str2[i] has reached \0
   { printf("str1 > str2");
    //printf("\nstr1[i] is %c & str2[i] is %c",str1[i],str2[i]);
   }
else if (str1[i] < str2[i])  //true if str1[i] has reached \0
 {   printf("str1 < str2");
    //printf("\nstr1[i] is %c & str2[i] is %c",str1[i],str2[i]);
}
else
    {printf("str1 = str2"); //true if both str1[1] & str2[i] have reached \0
    //printf("\nstr1[i] is %c & str2[i] is %c",str1[i],str2[i]);
    }
return (0);
}
```

Output 1:

Enter string 1 :C Program
Enter string 2 :C Programming
str1 < str2

Output 2:
Enter string 1 :C Programming
Enter string 2 :C Program
str1 > str2

Output 3:
Enter string 1 :Problem Solving
Enter string 2 :Problem Solving
str1 = str2

Exercise 8:
2. Write a program in C to copy one string to another string.

```
/*
Write a program in C to copy one string to another string.
*/

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
void main()
{
char str1[100], str2[100];
int i;
printf("\n\nCopy one string into another string :\n");
printf(" \n");
printf("Input the string : ");
fgets(str1, sizeof str1, stdin);
/* Copies string1 to string2 character by character */
i=0;
while(str1[i]!='\0')
{
    str2[i] = str1[i];
    i++;
}
//Makes sure that the string is NULL terminated
str2[i] = '\0';
printf("\nThe First string is : %s\n", str1);
printf("The Second string is : %s\n", str2);
printf("Number of characters copied : %d\n\n", i);
}
```

Output:
Copy one string into another string :

Input the string : Block Chain Technology

The First string is : Block Chain Technology

The Second string is : Block Chain Technology

Number of characters copied : 23

Exercise 9:
1. Write a C Program to Store Information Using Structures with Dynamic Memory Allocation

```
/*Write a C Program to Store Information Using Structures with Dynamic
Memory Allocation
*/

#include <stdio.h>
#include<stdlib.h>
struct grocery
{
int price;
char item[30];
};
int main()
{
struct grocery *ptr;
int i, totalPurchItems;
printf("Enter total items to purchase: ");
scanf("%d", &totalPurchItems);
/*
Dynamically allocates memory for totalPurchItems structures
with pointer ptr pointing to the 1st address.
*/
ptr = (struct grocery*) malloc (totalPurchItems * sizeof(struct grocery));
for(i = 0; i < totalPurchItems; ++i)
{
    printf("\nEnter name of the item: ");
```

```
    scanf("%s", &(ptr+i)->item);
    printf("Enter price of the item: ");
    scanf("%d", &(ptr+i)->price);
}
printf("\nPurchased Information:\n");
for(i = 0; i < totalPurchItems ; ++i)
    printf("%s\t%d\n", (ptr+i)->item, (ptr+i)->price);

return 0;
}
```

Output:

Enter total items to purchase: 2

Enter name of the item: Bread
Enter price of the item: 50

Enter name of the item: Jam
Enter price of the item: 45

Purchased Information:
Bread 50
Jam 45

Exercise 9:

2. Write a program in C to demonstrate how to handle the pointers in the program.

```
/*
Write a program in C to demonstrate how to handle the pointers in the
program.
*/

#include <stdio.h>
int main()
{
int* myPtr;
int val;
val=41;
printf("\n\n Pointer usage in C :\n");
printf(" \n");
printf(" Here in the declaration myPtr = int pointer, int val= 41\n\n");
printf(" Address of val : %p\n",&val);
```

```
printf(" Value of val : %d\n\n",val);
myPtr=&val;
printf(" Now myPtr is assigned with the address of val.\n");
printf(" Address of pointer myPtr : %p\n",myPtr);
printf(" Content of pointer myPtr : %d\n\n",*myPtr);
val=75;
printf(" The value of val assigned to 75 now.\n");
printf(" Address of pointer myPtr : %p\n",myPtr);
printf(" Content of pointer myPtr : %d\n\n",*myPtr);
*myPtr=9;
printf(" The pointer variable myPtr is assigned the val 9 now.\n");
printf(" Address of val : %p\n",&val);//as myPtr contain the address of
val
//so *myPtr changed the value of val and now val become 9
printf(" Value of val : %d\n\n",val);
return 0;
}
```

Output:

Pointer usage in C :

Here in the declaration myPtr = int pointer, int val= 41

Address of val : 0061FF18
Value of val : 41

Now myPtr is assigned with the address of val.
Address of pointer myPtr : 0061FF18
Content of pointer myPtr : 41

The value of val assigned to 75 now.
Address of pointer myPtr : 0061FF18
Content of pointer myPtr : 75

The pointer variable myPtr is assigned the val 9 now.
Address of val : 0061FF18
Value of val : 9

Exercise 10:

1. Write a program in C to add numbers using call by reference.

```
//Write a program in C to add numbers using call by reference.
#include <stdio.h>
long addTwoNumbers(long *, long *);
int main()
{
long fno, sno, *ptr, *qtr, sum;
printf("\n\n Addition of two numbers - call by reference using
Pointers:\n");
printf(" \n");
printf(" Input the first number : ");
scanf("%ld", &fno);
printf(" Input the second number : ");
scanf("%ld", &sno);
sum = addTwoNumbers(&fno, &sno);
printf(" The sum of %ld and %ld is %ld\n\n", fno, sno, sum);
return 0;
}
long addTwoNumbers(long *n1, long *n2)
{
long sum;
sum = *n1 + *n2;
return sum;
}
```

Output:

Addition of two numbers - call by reference using Pointers:

Input the first number : 45
Input the second number : 16
The sum of 45 and 16 is 61

Exercise 10:
2. Write a program in C to find the largest element using Dynamic Memory Allocation.

```
//Write a program in C to find the largest element using Dynamic Memory
Allocation.

#include <stdio.h>
#include <stdlib.h>
int main()
{
int i,n;
float *element;
printf("\n\n Pointers : Find the largest element using Dynamic Memory
Allocation:\n");
printf(" \n");
printf(" Input total number of elements(1 to 100): ");
scanf("%d",&n);
element=(float*)calloc(n,sizeof(float)); // Memory is allocated for 'n'
elements
if(element==NULL)
{   printf(" No memory is allocated.");
    exit(0);
}
printf("\n");
for(i=0;i<n;++i)
{   printf(" Number %d: ",i+1);
    scanf("%f",element+i);
}
for(i=1;i<n;++i)
{   if(*element<*(element+i))
    *element=*(element+i);
}
printf(" The Largest element is : %.2f \n\n",*element);
return 0;
}
```

Output:
Pointers : Find the largest element using Dynamic Memory Allocation:

Input total number of elements(1 to 100): 5
Number 1: 5

Number 2: 27
Number 3: 1
Number 4: 67
Number 5: 4
The Largest element is : 67.00

Exercise 11:
1. Write a program in C to swap elements using call by reference.

```
//Write a program in C to swap elements using call by reference.
#include <stdio.h>
void swapNumbers(int *x,int *y,int *z);
int main()
{
int e1,e2,e3;
printf("\n\n Pointer : Swap elements using call by reference :\n");
printf(" \n");
printf(" Input the value of 1st element : ");
scanf("%d",&e1);
printf(" Input the value of 2nd element : ");
scanf("%d",&e2);
printf(" Input the value of 3rd element : ");
scanf("%d",&e3);
printf("\n The value before swapping are :\n");
printf(" element 1 = %d\n element 2 = %d\n element 3 = %d\n",e1,e2,e3);
swapNumbers(&e1,&e2,&e3);
printf("\n The value after swapping are :\n");
printf(" element 1 = %d\n element 2 = %d\n element 3 = %d\n\n",e1,e2,e3);
return 0;
}
void swapNumbers(int *x,int *y,int *z)
{
    int tmp;
    tmp=*y;
    *y=*x;
    *x=*z;
    *z=tmp;
}
```

Output:
Pointer : Swap elements using call by reference :

Input the value of 1st element : 37
Input the value of 2nd element : 23
Input the value of 3rd element : 55

The value before swapping are :
element 1 = 37
element 2 = 23
element 3 = 55

The value after swapping are :
element 1 = 55
element 2 = 37
element 3 = 23

Exercise 11:
2. Write a program in C to count the number of vowels and consonants in a string using a pointer.

```
/* Write a program in C to count the number of vowels and consonants in a
string
using a pointer.
*/
#include <stdio.h>
int main()
{
char str[100];
char *p;
int vCount=0,cCount=0;
printf("Enter any string: ");
fgets(str, 100, stdin);
//assign base address of char array to pointer
p=str;
//'\0' signifies end of the string
while(*p!='\0')
{
    if(*p=='A' ||*p=='E' ||*p=='I' ||*p=='O' ||*p=='U'
        ||*p=='a' ||*p=='e' ||*p=='i' ||*p=='o' ||*p=='u')
        vCount++;
    else
        cCount++;
```

```
//increase the pointer, to point next character
p++;
}
printf("Number of Vowels in String: %d\n",vCount);
printf("Number of Consonants in String: %d",cCount);

return 0;
}
```

Output:
Enter any string: Principles of programming
Number of Vowels in String: 7
Number of Consonants in String: 19

Exercise 12:
1. Write a C program to find sum of n elements entered by user. To perform this program, allocate memory dynamically using malloc( ) function.

```
/*Write a C program to find sum of n elements entered by user. To perform
this program, allocate memory dynamically using malloc( ) function.
*/
#include <stdio.h>
#include <stdlib.h>
int main()
{
int n, i, *ptr, sum = 0;
printf("Enter number of elements: ");
scanf("%d", &n);
ptr = (int*) malloc(n * sizeof(int));
// if memory cannot be allocated
if(ptr == NULL)
{   printf("Error! memory not allocated.");
    exit(0);
}
printf("Enter elements: ");
for(i = 0; i < n; ++i)
{   scanf("%d", ptr + i);
    sum += *(ptr + i);
}
printf("Sum = %d", sum);
```

```
// deallocating the memory
free(ptr);
return 0;
}
```

Output:
Enter number of elements: 4
Enter elements: 10
20
30
40
Sum = 100

Exercise 13:
1. Write a program in C to check whether a number is a prime number or not using the function.

```
/*Write a program in C to check whether a number is a prime number or not
using the function.
*/
#include<stdio.h>
int PrimeOrNot(int);
int main()
{
int n1,prime;
printf("\n\n Function : check whether a number is prime number or not
:\n");
printf(" \n");
printf(" Input a positive number : ");
scanf("%d",&n1);
prime = PrimeOrNot(n1);
if(prime==1)
    printf(" The number %d is a prime number.\n",n1);
else
    printf(" The number %d is not a prime number.\n",n1);
return 0;
}
int PrimeOrNot(int n1)
{
    int i=2;
```

```
    while(i<=n1/2)
    {
    if(n1%i==0)
        return 0;
    else
        i++;
}
return 1;
}
```

Output:
Function : check whether a number is prime number or not :

Input a positive number : 7
The number 7 is a prime number.

Exercise 13:
2. Write a program in C to get the largest element of an array using the function.

```
//C program to get the largest element of an array using the function.
#include<stdio.h>
#define MAX 100
int findMaxElem(int []);
int n;
int main()
{
int arr1[MAX],mxelem,i;
printf("\n\n Function : get largest element of an array :\n");
printf(" \n");
printf(" Input the number of elements to be stored in the array :");
scanf("%d",&n);
printf(" Input %d elements in the array :\n",n);
for(i=0;i<n;i++)
{
    printf(" element - %d : ",i);
    scanf("%d",&arr1[i]);
}
mxelem=findMaxElem(arr1);
printf(" The largest element in the array is : %d\n\n",mxelem);
```

```
return 0;
}
int findMaxElem(int arr1[])
{
int i=1,mxelem;
mxelem=arr1[0];
while(i < n)
{
if(mxelem<arr1[i])
mxelem=arr1[i];
i++;
}
return mxelem;
}
```

Output:

Function : get largest element of an array :

Input the number of elements to be stored in the array :5
Input 5 elements in the array :
element - 0 : 55
element - 1 : 33
element - 2 : 77
element - 3 : 22
element - 4 : 11
The largest element in the array is : 77

Exercise 14:

1. Write a program in C to append multiple lines at the end of a text file.

```
//C Program to append multiple lines at the end of a text file.
#include <stdio.h>
int main ()
{
FILE * fptr;
int i,n;
char str[100];
char fname[20];
char str1;
printf("\n\n Append multiple lines at the end of a text file :\n");
printf(" Input the file name to be opened : ");
```

```
scanf("%s",fname);
fptr = fopen(fname, "a");
printf(" Input the number of lines to be written : ");
scanf("%d", &n);
printf(" The lines are : \n");
for(i = 0; i < n+1;i++)
{     fgets(str, sizeof str, stdin);
      fputs(str, fptr);
}
fclose (fptr);
//----- Read the file after appended -------
fptr = fopen (fname, "r");
printf("\n The content of the file %s is :\n",fname);
str1 = fgetc(fptr);
while (str1 != EOF)
{     printf ("%c", str1);
      str1 = fgetc(fptr);
}
printf("\n\n");
fclose (fptr);
// End of reading
return 0;
}
```

Output:

Append multiple lines at the end of a text file :

Input the file name to be opened : file1.txt
Input the number of lines to be written : 3
The lines are :
Line1
Line2
Line3

The content of the file file1.txt is :
Hello1
Hello2

Line1
Line2
Line3

Exercise 14:

2. Write a program in C to copy a file into another name.

```
//Write a program in C to copy a file into another name.
#include <stdio.h>
#include <stdlib.h> // For exit()
int main()
{
FILE *fptr1, *fptr2;
char filename[100], c;
printf("Enter the filename to open for reading \n");
scanf("%s", filename);
fptr1 = fopen(filename, "r");
if (fptr1 == NULL)
{   printf("Cannot open file %s \n", filename);
    exit(0);
}
printf("Enter the filename to open for writing \n");
scanf("%s", filename);
fptr2 = fopen(filename, "w");
if (fptr2 == NULL)
{   printf("Cannot open file %s \n", filename);
    exit(0);
}
c = fgetc(fptr1);
while (c != EOF)
{   fputc(c, fptr2);
    c = fgetc(fptr1);
}
printf("\nContents copied to %s", filename);
fclose(fptr1);
fclose(fptr2);
return 0;
}
```

Output:

Enter the filename to open for reading
file1.txt
Enter the filename to open for writing
file2.txt
Contents copied to file2.txt

Exercise 14:

3. Write a program in C to remove a file from the disk.

```
//Write a program in C to remove a file from the disk.
#include <stdio.h>
void main()
{
int status;
char fname[20];
printf("\n\n Remove a file from the disk :\n");
printf(" \n");
printf(" Input the name of file to delete : ");
scanf("%s",fname);
status=remove(fname);
if(status==0)
{
    printf("\nFile %s has been removed.",fname);
}
else
{
    printf("\nFile %s hasn't been found or does not exist.",fname);
}
}
```

Output1:

Remove a file from the disk :
Input the name of file to delete : file3.txt
File file3.txt has been removed.

Output2:

Remove a file from the disk :
Input the name of file to delete : file3.txt
File file3.txt hasn't been found or does not exist.

Additional Programs for Practice:

1. Write a c program to illustrate the Fibonacci series with recursion & without recursion

```
#include<stdio.h>
#include<stdlib.h>
int recfib(int); //function declaration
void nonrecfib(int);   //function declaration
int main()
{ int i,n;
  printf("\n Enter the numbers:");
  scanf("%d",&n);
  printf("\n The Fibonacci series using recursion is: ");
  for(i=0;i<n;i++)
      printf("%d\t",recfib(i));
  printf("\n The Fibonacci series without using recursion is: ");
  nonrecfib(n);
  return 0;
}
int recfib(int x)      //function definition - recursive
{     if(x==0)
            return 0;
      else if(x==1)
            return 1;
      else
            return recfib(x-1)+recfib(x-2);
}
void nonrecfib(int c)  //function definition - non recursive
{     int a=0,b=1,d,i;
      printf("%d\t%d",a,b);
      for(i=3;i<=c;i++)
      {   d=a+b;
          printf("\t%d",d);
          a=b;
          b=d;
      }
}
```

Output:

Enter the numbers:5
The Fibonacci series using recursion is: 0 1 1 2 3
The Fibonacci series without using recursion is: 0 1 1 2 3
