Unit 4
Array and Strings

Introduction: Array

Array by definition is a variable that hold multiple elements which has the same data type in a continuous memory location. An array is also known as a *subscripted variable.*

Types of array:

There are two types of array:

a) One dimensional array (1D array)
b) Two dimension array (2D array)

1-D Array and Declaration

In this structure arrays are stored either in row or column i.e. either in X-direction or Y-direction. 1D array has only a single subscript.

1-D array should be declared before using it and we can declare an array by specifying its data type, name and the fixed number of elements the array holds between square brackets immediately following the array name.

Syntax:
```
    data type identifier [size];
```

Example:
```
    int marks[5];
```

Note: The size of an array must be pure positive integer value. Followings are invalid in the case of defining size of an array.

- We can‟t use floating point.
  Example: `int marks[5.5];`
- We can‟t use characters. Example: `int marks[five];`
- We can‟t use any special symbols
  Example: `int marks[%];`
- We can‟t use negative value Example: `int marks[-5];`

In above example 10 bytes are immediately reserved in the memory, 2 bytes for each 5 integers and since the array is not being initialized, all five values presents in it would be garbage values. Since the storage class is not mentioned then it is assumed to be **auto**. If the storage class is declared to be **static,** then all the array elements would have a default initial value **zero.** But all the elements are present in consecutive memory location. This arrangement of array element in the memory is illustrated in following figure.

Memory arrangement:

2001	167	231	11223	19803	⟶	Element (Garbage Value)
num[0]	num[1]	num[2]	num[3]	num[4]	⟶	Index Number

Fig: Memory Allocation of 1-D Array

Initialization of Arrays:

If the array elements are not given any specific values, they are supposed to contain garbage values. The values are managed to store values in them during program execution. The process of defining initial values to the array is known as initialization of 1-D array.

Syntax:

```
    data type identifier [size] = {Elements};
```

Example:

```
    int marks[5] = { 50, 65, 34, 45, 87};
```

Memory arrangement:

1000	1002	100	1006	1008	→	Memory Address
50	65	34	45	87	→	Element
num[0]	num[1]	num[2]	num[3]	num[4]	→	Index Number

Fig: 1-D Array Initialization

Accessing of 1-D Array elements:

Once the array is declared, we have to access those array elements for our requirements. Array elements are accessed with its subscript, the number in the brackets followed in the array name i.e. through its index number. This number specifies the element‟s position in the array. All the array elements are numbered, starting with **0** and ending with **size – 1,** where **size** is the total size of the array. Generally a single **for** loop is associated with 1-D array while entering data into an array or reading elements from the array. This is illustrated in following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, num[5];
clrscr();
printf("\n Input any five numbers: - ");
/* entering elements into an array*/
for(i=0;i<5;i++)
scanf("%d", &num[i]);

/*Reading elements from array and displaying to the
monitor*/
for(i=0;i<5;i++)
printf("%5d",num[i]);
getch();
}
```

Multidimensional (2-D) Arrays:

The data in 2-D array are arranged in tabular form i.e. in number of rows and columns. It consists of two subscript in which first gives the number of rows size and the second number gives the column size. The number of elements that a two dimensional array is defined as the product of row size and column size.

Declaration of 2D array:

Syntax:

```
data type identifier [row_size][column_size];
```

Example:

```
int num[4][3];
```

Initialization of 2D array:

Syntax:

```
data type identifier [row_size][column_size] = { Elements };
```

Example:

```
int num[3][4] = {
                   { 10, 12, 34, 40 },
                   { 23, 21, 01, 10 },
                   {100, 91, 83, 49}
                 };
```

Or even we can initialize 2-D array as:

```
int num[3][4] = { 10, 12, 34, 40, 23, 21, 01, 10, 100, 91, 83, 49};
```

Note: In case of initialization of 2-D array the order of row is optional but the order of column is compulsory. Following initialization process is invalid in 2-D array

- `int num[3][ ] = { 10, 12, 34, 40, 23, 21, 01, 10, 100, 91, 83, 49};`
- `int num[ ][ ] = { 10, 12, 34, 40, 23, 21, 01, 10, 100, 91, 83, 49};`

Memory management

	Col[0]	Col[1]	Col[2]	Col[3]
Row [0]	10	12	34	40
Row [1]	23	21	1	10
Row [2]	100	91	83	49

Fig: Memory Arrangement of 2-D Array

Accessing of 2-D Array:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j, num[10][10];
clrscr();

/* entering the array elements to the matrix*/
printf("\n Input the elements of 3 X 3 matrixes :- ");
for(i=0;i<3;i++)
 {
 for(j=0;j<3;j++)
       scanf("%d", &num[i][j]);
 }

/* Displaying the matrix*/
printf(" The Given matrix is:\n");
for(i=0;i<3;i++)
 {
 for(j=0;j<3;j++)
printf("%5d",num[i][j]);  printf("\n");
 }
getch();
}
```

Output:

```
Input the elements of 3 X 3 matrixes: - 1 2 3 4 5 6 7 8 9

The Given matrix is:
      1  2  3
      4  5  6
      7  8  9
```

Introduction to String:

Collection of interrelated characters i.e. arrays of characters is known as string. Such character arrays or string are used by programmer in the programming language to manipulate text such as word, sentence or paragraphs. A string consonant is a 1-D array of characters terminated by a null („\0‟) character. For example:

```
 char name [ ] = { „C‟, „O‟, „M‟, „P‟, „U‟, „T‟, „E‟, „R‟, „\0‟ };
```

or

```
     char name [ ] = "COMPUTER";
```

Each character in an array occupies 1 byte of memory space and the last character is always the null character, which denotes the end of character. The following figure shows the way a character array are stored in contiguous memory locations.

1000	1001	1002	1003	1004	1005	1006	1007	1008	1009
C	O	M	P	U	T	E	R	„\0‟	

Fig : Memory arrangement of string

Manipulation of string:

Let us consider following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i=0;
char name[]="COMPUTER";
clrscr();
while(i<8)
      {
      printf("%c", name[i]);
      i++;
      }
getch();
}
```

Output:

```
        COMPUTER
```

In above program we have initialized a character array, and then we have printed the strings using **while** loop. This method is not very much applicable because we can‟t write the **while** loop without using the final value 8 i.e. we can‟t determine the length of string in advance, so it is impractical. As we know that each character array always ends with a **„\0‟.** Following program shows this process.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i=0;
char name[]="COMPUTER";
clrscr();
while(name[i]!='\0')
      {
      printf("%c",name[i]);
      i++;
      }
getch();
}
```

Output:

```
        COMPUTER
```

Above program doesn‟t depend on the length of the string to print the contents of the string. So it is more general than the earlier program.
Even we can refer to the elements of a character array using **printf()** function (it doesn‟t print the „\0‟) with **%s** format specifier. The same specifier is also used to receive a string form the keyboard. This is illustrated in following program

```
#include<stdio.h>
#include<conio.h>
void main()
{ int i=0;
char name[20];
clrscr();
printf("\n Input any string :- ");
scanf("%s", name);
printf("%s", name);
 getch();
}
```

Output:

```
        Input any string: - COMPUTER
        COMPUTER
```

In above program a character array **name[20]** is declared which reserves **20 bytes** space in memory. The **scanf()** function accepts the characters entered through the keyboard and stores in the array until the enter key is pressed by the user. Once the enter key has been pressed the **scanf()** function places the null character („\0‟) in the array. But the **scanf()** function has some limitations. They are:

- **scanf()** function can‟t receive multiple word strings.
- It can‟t perform bounds checking on character array.

Such limitations are eliminated by **gets()** function. The working mechanism of **gets()** and **puts()** function is illustrated in following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
char name[50];
clrscr();
printf("\n Input your name: - ");
gets(name);
puts(name);
getch();
}
```

Output:

```
Input your name: - Krishna Joshi
Krishna Joshi
```

Note:

- Unlike **printf()** function **puts()** function places the cursor on the next line.
- **Puts()** function can display only one string at a time.
- **[^\n]** can be used within **scanf()** function to receive characters of array until a **\n** is encountered.

String related functions:

In C compiler there are many useful string manipulating library functions. Some of the main functions are described below.

1. strlen()

- This function is used to count the number of characters present in a string (including number of spaces present in the string, excluding null character).
- It takes single argument as an input.
- It returns pure positive integer value.
- **Syntax:** `strlen(argument);`
- **Example:** `strlen("Computer");`
- **Output:** 8

The use of **strlen()** function is illustrated in following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
char string[50];
int length;
clrscr();
printf("\n Input any string: - ");
gets(string);
length=strlen(string);
printf("\n There are %d characters in \"%s\" string", length, string);
getch();
}
```

Output:

```
Input any string: - National Academy of Science and Technology
There are 42 characters in "National Academy of Science and
Technology" string
```

Alternative method: without using string function

```
#include<stdio.h>
#include<conio.h>
void main()
{
char string[50];
int length=0, i=0;
clrscr();
printf("\n Input any string: - ");
gets(string);
while(string[i]!='\0')
 {
 length++;
 i++;
 }
printf("\n There are %d characters in string \"%s\" ",length,string);
getch();
}
```

2. strcpy()

- This function is used to copy the contents of one string to another string.

- It takes two arguments. The base address of the source string (from which we want to copy) and the destination string (into which we want to copy) is supplied to this function. It returns the array of characters.
- **Syntax:** `strcpy(destination, source);`
- **Example:** `strcpy(string, "Computer");`
- **Output:** Computer

The use of **strcpy()** function is illustrated in following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
char source_string[50],target_string[50];
clrscr();

printf("\n Input any string: - ");
gets(source_string);

strcpy(target_string,source_string);
printf("\n The source string is \"%s\" ", source_string);
printf("\n The Copied string is \"%s\" ", target_string);
getch();
}
```

Output:

```
Input any string: - Aishwarya Vidhya Niketan
The source string is "Aishwarya Vidhya Niketan"
The Copied string is "Aishwarya Vidhya Niketan"
```

Alternative method: without using string function

```
#include<stdio.h>
#include<conio.h>
void main()
{
char string1[50],string2[50];
int i=0; clrscr();
printf("\n Input any string: - ");
gets(string1);
while(string1[i]!='\0')
 {
 string2[i]=string1[i];
 i++;
 }
string2[i]=0;
printf("\n The copied string is \"%s\" ",string2);
getch();
}
```

3. strcmp()

- This function is used to compare two strings to find whether they are identical or not.
- It takes two arguments.
- The two strings are compared character by character (i.e. the first character of the first string is compared to the first character of second string, if it matches then the second character of the first string is compared with the second character of the second string and so on) until the end of one of the strings is reached or until there is a mismatch between the strings, whichever comes first. The function returns an integer value.
- The function returns a zero value when both strings are identical otherwise it returns numeric difference between the ASCII values of the first non-matching pair of characters.
- **Syntax:** `strcmp(text1, text2);`
- **Example:** `strcmp("Computer", "Computer");` □ **Output:** 0

The use of **strcmp()** function is illustrated in following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
Char text1[50],text2[50];
 int difference; clrscr();
printf("\n Input first string: - ");
gets(text1);
printf("\n Input second string: - ");
gets(text2);

difference=strcmp(text1,text2);
if(difference==0)
 printf("\n Both strings are identical");
else
 printf("\n Both strings are non-identical");
getch();
}
```

Output:

- **First Run:**

```
Input first string: - NAST
Input second string: - Nast
Both strings are non-identical
```

- **Second Run:**

```
Input first string: - dhangadhi
Input second string: - dhangadhi
Both strings are identical
```

4. strlwr()

- This function is used to convert any upper case text of a string into its equivalent lower case.
- It takes only one argument as an input.
- It returns single argument into its equivalent lower case.
- **Syntax:** `strlwr(argument);`

- **Example:** `strlwr("COMPUTER");` **Output:** computer

The use of **strlwr()** function is illustrated in following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
Char string[50];
clrscr();
printf("\n Input any string: - ");
gets(string);
strlwr(string);

printf("\n The lower case is \"%s\"",string); getch();
}
```

Output:

```
Input any string: - INDRA RANA
The lower case is "indra rana"
```

Alternative method: without using string function

```
#include<stdio.h>
#include<conio.h>
void main()
{
char string[50];
int i=0;
clrscr();
printf("\n Input any string: - ");
gets(string);
while(string[i]!='\0')
 {
 if(string[i]>=65 && string[i]<=90)
   string[i]=string[i]+32;  i++;
 }

printf("\n The lower case string is \"%s\" ",string);
getch();
}
```

5. strupr()

- This function is used to convert any lower case text of a string into its equivalent upper case text.
- It takes only one argument as an input.
- It returns single argument into its equivalent upper case.
- **Syntax:** *strupr(argument);*
- **Example:** `strupr("computer");`
- **Output:** COMPUTER

The use of **strupr()** function is illustrated in following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
char string[50];
clrscr();

printf("\n Input any string: - ");
gets(string); strupr(string);
printf("\n The upper case is \"%s\"",string);
getch();
}
```

Output:

```
Input any string: - Krishna Joshi
The upper case is "KRISHNA JOSHI"
```

Alternative method: without using string function

```
#include<stdio.h>
#include<conio.h>
void main()
{
char string[50];
int i=0;
clrscr();
printf("\n Input any string: - ");
gets(string);
while(string[i]!='\0')
 {
 if(string[i]>=97 && string[i]<=132)
   string[i]=string[i]-32;  i++;
 }

printf("\n The Upper case string is \"%s\" ",string);
getch();
}
```

6. strcat()

- This function is used to concatenate the source string at the end of the target string (without space between source and destination string).
- It takes two arguments.

- It returns a concatenated string.
- **Syntax:** `strcat(argument1,argument2);`
- **Example:** `strcat("Computer", "Science");`
- **Output:** ComputerScience

The use of **strcat()** function is illustrated in following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
char string1[50],string2[20];
clrscr();

printf("\n Input First string: - ");
gets(string1);
printf("\n Input Second string: - ");
gets(string2);

strcat(string1,string2);

printf("\n The complete string is \"%s\"",string1);
 getch();
}
```

Output:

```
Input First string: - I read in
Input Second string: - class 12
The complete string is "I read inclass 12"
```

7. strrev()

- This function is used to reverse the characters in the string.
- It also takes one argument
- It returns single argument is reverse of the given string.
- **Syntax:** `strrev(argument);`
- **Example:** `strrev("Computer");`
- **Output:** retupmoC

The use of **strrev()** function is illustrated in following program.

```
#include<stdio.>
#include<conio.>
void main() {
char string[50];
clrscr();
printf("\n Input any string: - ");
gets(string);
strrev(string);
printf("\n The reverse string is \"%s\"",string);
getch();
}
```

Output:

```
Input any string: - Class BE Civil
The reverse string is "liviC EB ssalC"
```

Alternative method: without using string function

```
#include<stdio.>
#include<conio.>
void main() {
char string[50];
int i=0,j;
clrscr();
printf("\n Input any string: - ");
gets(string);
printf("\nBefore Reverse\n");
puts(string);
while(string[i]!='\0')
i++;
printf("After Reverse\n");
for(j=i;j>=0;j--)
printf("%c",string[j]);
 getch();
}
```

2D array of Strings:

The characters in 2D array are arranged in the memory is shown in following figure. Each string ends with a „\0‟. The arrangement of characters is similar to that of 2D numeric array. In below figure, 2000, 2010, 2020 etc are the base address of successive names. Some of the names do not occupy all the bytes reserved for them. The first name "Krishna" only occupies 8 bytes however for the name total 10 bytes is reserved, so 2 byte is wasted. This wastage increases when the number of name increases. This wastage can be avoided by using *"array of pointers"*.

2000
2010
2020
2030
2049
2049

K	r	i	S	h	N	a	„\0‟		
R	a	m	„\0‟						
S	i	t	A	„\0‟					
I	n	d	R	a	„\0‟				
S	r	i	J	a	N	a	„\0‟		

(Last Location)

Fig: Memory arrangement of 2D array of Characters

Program 1: Write a program to read two matrixes and display the addition of the matrix in its appropriate form.

```
#include<stdio.h>
#include<conio.h> void
main()
{
int i, j, A[10][10], B[10][10], sum[10][10], r1, c1, r2, c2;
clrscr();
printf("\n Input order of first matrix:- ");
 scanf("%d%d",&r1,&c1);
printf("\n Input order of second matrix:- ");
scanf("%d%d",&r2,&c2);
if(r1==r2 && c1==c2)
     {
     printf("\n Input Elements of First %d X %d Matrix:- ",r1,c1);
        for(i=0;i<r1;i++)
          {
          for(j=0;j<c1;j++)
               {
               printf("\n Elements [%d][%d]:- ",i, j);
                 scanf("%d", &A[i][j]);
               }
          }

     printf("\n Input Elements of Second %d X %d Matrix:- ",r2,c2);
      for(i=0;i<r2;i++)
          {
          for(j=0;j<c2;j++)
               {
               printf("\n Elements [%d][%d]:- ",i,j);
             scanf("%d", &B[i][j]);
               }
          }
     for(i=0;i<r1;i++)
          {
          for(j=0;j<c1;j++)
               sum[i][j]=A[i][j]+B[i][j];
          }

     clrscr();
     printf("\n First Matrix: \n\n");
     for(i=0;i<r1;i++)
          {
          for(j=0;j<c1;j++)
printf("%5d",A[i][j]);
printf("\n");
```

```
          }

     printf("\n Second Matrix: \n\n");
     for(i=0;i<r1;i++)
          {
          for(j=0;j<c1;j++)
printf("%5d",B[i][j]);
printf("\n");
          }
     printf("\n Addition of the Matrix: \n\n");
     for(i=0;i<r1;i++)
          {
          for(j=0;j<c1;j++)
printf("%5d",sum[i][j]);
printf("\n");
          }
     }
else
     printf("\n Invalid Matrices");
 getch();
}
```

Program 2: Write a program to read "n" numbers and display those numbers in ascending orders

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, j, num[100],n, temp;
clrscr();
printf("\n How many terms:- ");
scanf("%d", &n);
printf("\n Input %d numbers", n);
for(i=0;i<n;i++)
     {
     printf("\n Number [%d]:- ",i+1);
     scanf("%d", &num[i]);
     }
     for(i=0;i<n;i++)
     {
     for(j=i+1;j<n;j++)
          {
          if(num[i]>num[j])
               {
               temp=num[i];
               num[i]=num[j];
               num[j]=temp;
               }
          }
```

```
      }

printf("\n The Numbers in ascending orders are: \n");
for(i=0;i<n;i++)
printf("%5d",num[i]);
getch();
}
```

Program 3: Write a program to read a matrix and display the sum of an individual column entered by the user.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, j, A[10][10],r, c, sum=0,col;
clrscr();
printf("\n Input order of a matrix:- ");
scanf("%d%d", &r, &c);
      printf("\n Input Elements of First %d X %d Matrix:- ",r,c);
      for(i=0;i<r;i++)
      {
      for(j=0;j<c;j++)
            {
            printf("\n Elements [%d][%d]:- ",i,j);
                  scanf("%d", &A[i][j]);
            }
      }
printf("\n Which Column Do you want to Add (0 to %d)",c-1);
scanf("%d", &col); clrscr();

printf("\n Given Matrix: \n\n");
for(i=0;i<r;i++)
      {
      for(j=0;j<c;j++)
      printf("%5d",A[i][j]);
      printf("\n");
      }

for(i=0;i<r;i++)
      {
      for(j=0;j<c;j++)
            {
            if(col==j)
                  sum=sum+A[i][j];
            }
      }
printf("\n Sum of elements of %d row is %d", col, sum);
getch();
}
```

Program 4: Write a program to read salary of 200 employees and count the number of employees whose salary is in between Rs 5000 and Rs 10000.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, count=0, salary[200];
 clrscr();
printf("\n Input salary of 200 employees:- ");
for(i=0;i<200;i++)
     {
     printf("\n Salary [%d]:- ",i+1);
     scanf("%d", &salary[i]);
     }

for(i=0;i<n;i++)
     {
     if(salary[i]>5000&&salary[i]<10000)
          count++;
     }

printf("\n There are = %d employees", count);
getch();
}
```

Program 5: Write a program to read n numbers from the user and find the greatest number among them.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, num[100],n, greatest;
clrscr();
printf("\n How many terms:- ");
scanf("%d", &n);
printf("\n Input %d numbers", n);
for(i=0;i<n;i++)
     {
     printf("\n Number [%d]:- ",i+1);
scanf("%d", &num[i]);
     }

greatest=num[0];
for(i=0;i<n;i++)
     {
     if(num[i]>greatest)
          greatest=num[i];
```

```
      }

printf("\n The Numbers are: \n");
for(i=0;i<n;i++)
printf("%5d",num[i]);

printf("\n The greatest number is %d", greatest);
getch();
}
```

Program 6: Write a program to input "n" numbers from the user display those numbers in ascending order and also find the number of odds and even numbers in those numbers.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, num[100], n, odd_count=0, even_count=0;
clrscr();
printf("\n How many terms:- ");
scanf("%d", &n);
printf("\n Input %d numbers", n);
for(i=0;i<n;i++)
     {
     printf("\n Number [%d]:- ",i+1);
scanf("%d", &num[i]);
     }

for(i=0;i<n;i++)
     {
     if(num[i]%2==0)
even_count++;
else
          odd_count++;
     }

printf("\n The Numbers in ascending orders are: \n");
for(i=0;i<n;i++)
printf("%5d",num[i]);

printf("\n There are = %d odd and %d even numbers", odd_count,
even_count);
getch();
}
```

Program 7: Write a program to read a matrix and display the sum of a row inputted from the keyboard.

```
#include<stdio.h>
#include<conio.h>
```

```
void main()
{
int i, j, A[10][10], r, c, sum=0, row;
clrscr();
printf("\n Input order of a matrix:- ");
scanf("%d%d", &r, &c);
printf("\n Input Elements of First %d X %d Matrix:- ",r,c);
for(i=0;i<r;i++)
     {
     for(j=0;j<c;j++)
          {
          printf("\n Elements [%d][%d]:- ",i,j);
               scanf("%d",&A[i][j]);
          }
     }
printf("\n Which Row Do you want to Add (0 to %d)",r-1);
scanf("%d", &row); clrscr();

printf("\n Given Matrix: \n\n");
for(i=0;i<r;i++)
     {
     for(j=0;j<c;j++)
printf("%5d",A[i][j]);
     printf("\n");
     }

for(i=0;i<r;i++)
     {
     for(j=0;j<c;j++)
          {
          if(row==i)
sum=sum+A[i][j];
          }
     }
printf("\n Sum of elements of %d row is %d", row, sum);
 getch();
}
```

Program 8: Write a program to generate SLC mark sheet using an array as per Nepal government rule.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, pass=0, marks[8], total=0;
float per; clrscr();
printf("\n Input marks of 8 subjects:- ");
for(i=0;i<8;i++)
```

```
      {
      printf("\n Marks [%d]:- ",i+1);
      scanf("%d", &marks[i]);
total=total+marks[i];
      if(marks[i]>=32)
            pass++;
      }

printf("\n Total Marks = %d", total);
if(pass==8)
      {
      per=(float)total/8;
      printf("\n Percentage = %6.2f",per);
if(per>=80)
            printf("\n Result: Distinction");
else if(per>=60)
            printf("\n Result: First Division");
else if(per>=45)
printf("\n Result: Second Division");
      else
            printf("\n Result: Third Division");
      }
else
      printf("\n Failed in = %d subjects", 8-pass);
getch();
}
```

Program 9: Write a program to read a matrix and display the sum of the diagonal elements of the given matrix.

```
#include<stdio.h>
#include<conio.h>
 void main()
{
int i, j, A[10][10], r, c, sum=0;
clrscr();
printf("\n Input order of a matrix:- ");
scanf("%d %d", &r, &c); if(r==c)
      {
      printf("\n Input Elements of First %d X %d Matrix:- ",r, c);
for(i=0;i<r;i++)
            {
            for(j=0;j<c;j++)
                  {
                  printf("\n Elements [%d][%d]:- ",i, j);
      scanf("%d", &A[i][j]);
                  }
            }
      clrscr();
      printf("\n Given Matrix: \n\n");
      for(i=0;i<r;i++)
            {
            for(j=0;j<c;j++)
      printf("%5d",A[i][j]);
      printf("\n");
            }

      for(i=0;i<r;i++)
            {
            for(j=0;j<c;j++)
                  {
                  if(i==j)
sum=sum+A[i][j];
                  }
            }
      printf("\n Sum of Diagonal elements is %d", sum);
      }
 else
      printf("\n Invalid Matrix");
getch();
}
```

Program 10: Write a program to read a matrix and display the sum of all elements of the given matrix.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, j, A[10][10],r, c, sum=0;
clrscr();
printf("\n Input order of a matrix:- ");
scanf("%d %d", &r, &c);
printf("\n Input Elements of First %d X %d Matrix:- ",r, c);
for(i=0;i<r;i++)
      {
      for(j=0;j<c;j++)
            {
            printf("\n Elements [%d][%d]:- ",i, j);
            scanf("%d", &A[i][j]);
sum=sum+A[i][j];
            }
      }
 clrscr();
printf("\n Given Matrix: \n\n");
for(i=0;i<r;i++)
      {
      for(j=0;j<c;j++)
printf("%5d",A[i][j]);
      printf("\n");
      }

printf("\n Sum of All elements is %d", sum);
 getch();
}
```

Program 11: Write a program to store Kathmandu valley"s 7 days maximum and minimum temperature and calculate average, maximum among minimum temperature and minimum among maximum temperature.

```
    #include<stdio.h>
    #include<conio.h>
    void main()
    {
    int i, max_temp[7], min_temp[7], min, max, tot_max=0, tot_min=0;
    clrscr();
    printf("\n Input Maximum temperature of 7 days:- ");
    for(i=0;i<7;i++)
        {
        printf("\n Maximum Temperature [%d]",i+1);
    scanf("%d",&max_temp[i]);
        tot_max=tot_max+max_temp[i];
        }

    printf("\n Input Minimum temperature of 7 days:- ");
    for(i=0;i<7;i++)
        {
        printf("\n Minimum temperature [%d]",i+1);
    scanf("%d",&min_temp[i]);
        tot_min=tot_min+min_temp[i];
        }

    clrscr();

    printf("\n Minimum Temperature of 7 days are:\n");
    for(i=0;i<7;i++)
        printf("%5d",min_temp[i]);
    max=min_temp[0]; for(i=0;i<7;i++)
        {
        if(min_temp[i]>max)
            max=min_temp[i];
        }
    printf("\n Maximum among minimum temperature is %d", max);

    printf("\n Maximum Temperature of 7 days are:\n");
     for(i=0;i<7;i++)
        printf("%5d",max_temp[i]);
```

```
min=max_temp[0]; for(i=0;i<7;i++)
      {
      if(max_temp[i]<min)
            min=max_temp[i];
      }

printf("\n Minimum among maximum temperature is %d", min);
 getch();
}
```

Program 12: Write a program to read a matrix and display the transpose of the given matrix.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, j, A[10][10],r, c; clrscr();
printf("\n Input order of a matrix:- ");
scanf("%d %d", &r, &c);
printf("\n Input Elements of First %d X %d Matrix:- ",r, c);
for(i=0;i<r;i++)
      {
      for(j=0;j<c;j++)
            {
            printf("\n Elements [%d][%d]:- ",i, j);
scanf("%d", &A[i][j]);
            }
      }
 clrscr();
printf("\n Given Matrix: \n\n");
for(i=0;i<r;i++)
      {
      for(j=0;j<c;j++)
printf("%5d",A[i][j]);
      printf("\n");
      }

printf("\n Transpose Matrix: \n\n");
for(i=0;i<c;i++)
      {
      for(j=0;j<r;j++)
printf("%5d",A[j][i]);
printf("\n");
      }
getch();
}
```

Program 13: Write a program to read “n” numbers and display the total sum of those numbers and also calculate the average of those numbers.

```
#include<stdio.h>
#include<conio.h>
 void main()
{
int i, n, total=0, num[50];
float avg; clrscr();
printf("\n How many terms:- ");
scanf("%d", &n);
printf("\n Input %d numbers", n);
for(i=0;i<n; i++)
     {
     printf("\n Number [%d]:- ",i+1);
     scanf("%d",&num[i]);
     }

for(i=0;i<n;i++)
total=total+num[i];
avg=total/n;

printf("\n Total = %7d\nAverage = %8.2f",total,avg);
getch();
}
```

Program 14: Write a program to read "n" numbers from the user and display the smallest number among those numbers.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, num[100],n, small;
 clrscr();
printf("\n How many terms:- ");
scanf("%d", &n);
printf("\n Input %d numbers", n);
for(i=0;i<n; i++)
     {
     printf("\n Number [%d]:- ",i+1);
scanf("%d", &num[i]);
     }

small=num[0];
for(i=0;i<n;i++)
     {
     if(num[i]<small)
          small=num[i];
     }
printf("\n The Numbers are: \n");
for(i=0;i<n;i++)
printf("%5d",num[i]);

printf("\n The smallest number is %d", small);
getch();
}
```

Program 15: Write a program to read "N" numbers and calculate mean, median and range of those numbers.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, j, num[100], n, temp, range, total=0,median_term;
float mean, median;
clrscr();
printf("\n How many terms:- ");
scanf("%d", &n);
printf("\n Input %d numbers", n);
for(i=0;i<n;i++)
      {
      printf("\n Number [%d]:- ",i+1);
scanf("%d", &num[i]);
      total=total+num[i];
      }
for(i=0;i<n;i++)
      {
      for(j=i+1;j<n;j++)
            {
            if(num[i]>num[j])
                  {
                  temp=num[i];
                  num[i]=num[j];
                  num[j]=temp;
                  }
            }
      }

printf("\n The Numbers in ascending orders are: \n");
for(i=0;i<n;i++)
printf("%5d",num[i]);

mean=total/n;

range=num[n-1]-num[0];
if(n%2==0)
      {
```

```
     median_term=(n+1)/2;
median=(float)((num[median_term-1]+num[median_term]))/2;
     }
     else
     {
     median_term=(n+1)/2;
     median=num[median_term-1];
     }

printf("\n Mean   = %5.2f",mean);
printf("\n Range  = %5d",range);
printf("\n Median = %5.2f",median);
getch();
}
```

Program 22:
Program 23:
Program 24:
Program 25:

Programs Related to string

Program 16: Write a program to input names of "n" students and sort them in alphabetical order

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{ int i, j,
n;
char name[100][100],temp[100];
clrscr();
printf("\n How many students:- ");
scanf("%d", &n);
printf("\n Input name of %d students", n);
for(i=0;i<n; i++)
     {
     fflush(stdin);
printf("\n Name [%d]:- ",i+1);
     gets(name[i]);
     }
for(i=0;i<n-1;i++)
     {
     for(j=i+1;j<n;j++)
          {
          if(strcmp(name[i],name[j])>0)
               {
               strcpy(temp,name[i]);
               strcpy(name[i],name[j]);
               strcpy(name[j],temp);
```

```
                    }
            }
        }
    printf("\n Name in Alphabetical order are: \n");
     for(i=0;i<n;i++)
        printf("\n %s", name[i]);
    getch();
    }
```

Program 17: Write a program to input a text and count the number of vowels, consonants, symbols and words in the given text.

```
    #include<stdio.h>
    #include<conio.h>
    #include<string.h>
    void main()
    {
    int i, v_count=0, c_count=0, b_count=0,l ength;
    char string[100];
    clrscr();
    printf("\n Input any text:- ");
    gets(string);

    strlwr(string);
    length=strlen(string);
    for(i=0;i<length;i++)
        {
        if(string[i]=='a'||string[i]=='e'||string[i]=='i'||string[i]==
    'o'||string[i]=='u')
            v_count++;
        else if(string[i]==' ')

        b_count++;
    else
    c_count++;
        }

    printf("\n Given Text is \"%s\"",string);
    printf("\n There are %d vowels, %d consonants, %d blank spaces and
    %d words",v_count,c_count,b_count,b_count+1);
    getch();
    }
```

Program 18: Write a program to read a string and determine whether it is palindrome or not.

```
    #include<stdio.h>
    #include<conio.h>
    #include<string.h>
     void main()
    {
    int length;
```

```
char string[100],string1[100];
clrscr();
printf("\n Input any text:- ");
gets(string);
strlwr(string);
strcpy(string1,string);
strrev(string);
if(strcmp(string,string1)==0)
printf("\n The text \"%s\" is a palindrome text",string1);
 else
 printf("\n The text \"%s\" is a non-palindrome text",string1);
getch();
}
```

Program 19: Write a program to read a text and print rotation of a word typed into it. For example: nast, astn, stna, tnas.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
 void main()
{
int i, j, length;
char text[20],temp;
clrscr();
printf("\n Input any text:- ");
gets(text); length=strlen(text);
for(i=0;i<length;i++)
     {
     temp=text[0];
for(j=0;j<length-1;j++)
text[j]=text[j+1];
text[length-1]=temp;
puts(text);
     printf("\n");
     }
getch();
}
```

Program 20: Write a program to read a text and count the number of vowels and consonants in the given text.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
 void main()
{
int i, v_count=0, c_count=0, length;
char string[100];
 clrscr();
printf("\n Input any text:- ");
gets(string);
strlwr(string);
length=strlen(string);
for(i=0;i<length; i++)
     {
      if(string[i]=='a'||string[i]=='e'||string[i]=='i'||string[i]==
'o'||string[i]=='u')
v_count++;
else
          c_count++;
     }
printf("\n There are %d vowels and %d consonants in text: \"%s\"",
v_count, c_count, string);
getch();
}
```

Program 21: Write programs to display following patterns.

```
n
na
nas
nast
```

```
#include<stdio.h>
#include<conio.h>
void main()
{
char text[20];
int i=0, j;
clrscr();
printf("\n Input any text: - ");
gets(text);
while(text[i]!='\0')
     {
     printf("\n");
for(j=0;j<=i;j++)
          printf("%c",text[j]);
     i++;
     } getch();
}
```

```
 nast
nas
na
n
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
```

```
char text[20];
int length,i;
clrscr();
printf("\n Input any text: - ");
gets(text);
length=strlen(text);
while(text[length-1]!='\0')
     {
     printf("\n");
for(i=0;i<length;i++)
printf("%c",text[i]);
     length--;
     }
getch();
}
```