MODULE-5 Array, String and Structure

ARRAY :-

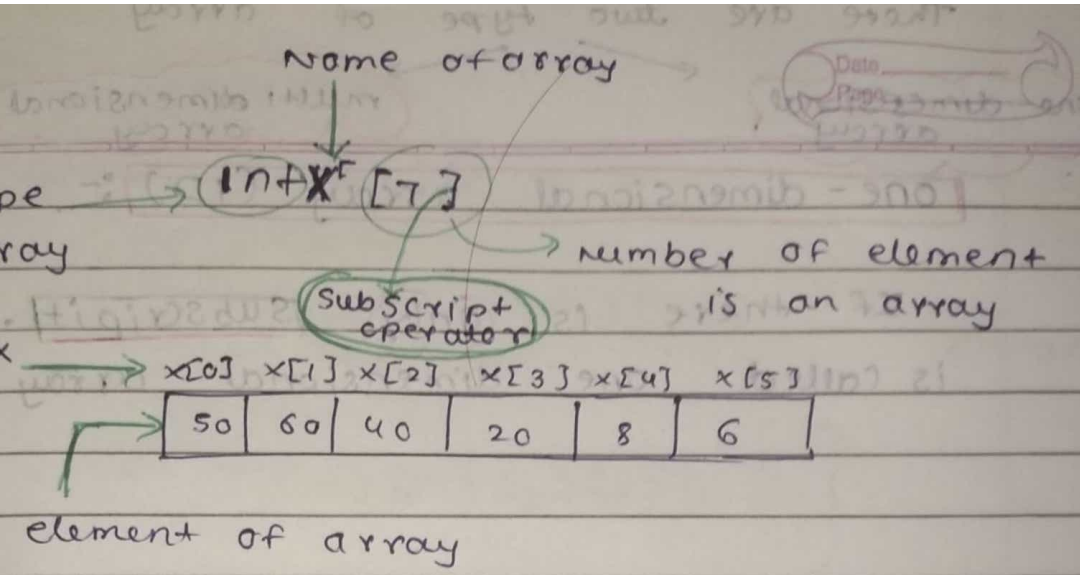
- Array is a way to take → multiple value by using → single identifier.
- Array is a variable which → store → more than one value of some datatype.
- array is ~~derived~~ user-defined datatype which is constructed → the help of → primitive datatype
- May be of type → int, float, char, double
- Array can't contain dis-similar type of data
- All these element ~~ares~~ store → in consecutive memory locations.
- An array starts with → subscript 0
- ends with → array size minus one

Syntax : type variable — name [length of array];

For example :-

```
double height [15];
float width [25];
int min [10];
char name [25];
```

Advantage
- make our code more optimized and clean
- ~~#~~ modifies
- are used to present two dimensional array
- we can store multiple ~~data~~ elements in single array.



e.g

```
#include<stdio.h>
int main(){
int a[5];
printf("Enter Element in array :");
```

For take value from user

```
for (i=0; i<=4; i++)
{
    scanf ("%d", &a[i]);
}
```

```
printf ("Array elements :");
```

Print array value/element which is taken by user

```
for (i=0; i<=4; i++)
{
    printf ("The value of a[%d] is %d\n", i, a[i]);
}
return 0;
}
```

output :

Enter element in array : 10 20 30 40 50

The value of a[0] is 10
" a[1] is 20
" a[2] is 30
" a[3] is 40
" a[4] is 50

There are two type of array

one-dimensional array

multi dimensional array

one-dimensional array (1D) :-

If there is only subscript. is called one-dimensional array.

{ same as Array content }

tmp

write a program to create a 1D array and display the average of all element of an array

```
#include<stdio.h>
int main(){
int a[5]; sum=0, avg;
printf("enter the five number\n");
for (i=0; i<5; i++)
    {
        scanf("%d", &a[i]);
        sum += a[i];
    }
avg = sum/5;
printf("average of given number %d\n", avg);
return 0;
}
```

for store value Taken and print sum

output : enter the five number 5
5
5
5
5
average of given number 5

3-1 Function

```
#include<stdio.h>
int average (int a[]); // function prototype

int main(){
    int avg, a[5];
    printf("Enter five number :");
    for (int i=0; i<5; i++){
        scanf("%d", &a[i]);
    }
    avg = average (a); // function call
    printf("Average of given number : %d\n", avg);
    return 0;
}

int average (int a[]){
    int avg, sum=0;
    for (int i=0; i<5; i++){
        sum += a[i];
    }
    avg = sum/5;
    return avg;
}
```

for Take input and store value

for sum of all element → then get average

average = Sum / 5

output :
5
5
5
5
5
average of given number : 5

Multidimensional array:

↳ multidimensional array is an array that has more than one dimensional

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

representation 2D array

let assume the name of matrix a

To acess a particular element from the array
- we have to use two subscripts. [ ] [ ]
  ↳ subscripts operator
  on for row number and another for column number

notation is a[i] [j]

i stand for raw subscripts
j stand for Column subscripts

Syntax :-

type variable_name [raw][column];

e.g float table [50][50];
Char line [24 [40];

e.g 3×3 matrix

```
#include<stdio.h>
int main() {
    int matrix[3][3] = {{1,1,1}, {2,2,2}, {3,3,3}};
    ~~int i~~ j ;   initialize int for loop
```

for row ⟹

```
for (int i=0 ; i<3 ; i++)
{
```

For Column ⟹

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

with use

output

```
1 1 1
2 2 2
3 3 3
```

output

```
1 1 1 2 2 2 3 3 3
```

write a program to read and display 3×3 matrix in array. and addition / sum of diagonal

```
# include<stdio.h>
int main(){
    int sum=0 , a[3][3] = {{1,1,1,},{2,2,2},{3,3,3}};
    printf("The Given 3x3 matrix is:");
    for (int i=0 ; i<3 ; i++)
    {
        for (int j=0 ; j<3 ; j++)
        {
            printf(" %d ", a[i][j]);
            if (i==j)
            {
                sum = sum + a[i][j];
            }
        }
        printf("\n");
    }
    printf ("sum of diagonal matrix is %d", sum);
}
```

output:

The Given 3×3 matrix is :

1	1	1
2	2	2
3	3	3

Sum of diagonal matrix is 6

c[i][j] = a[i][j] + b[i][j]

addition / multiply of two matrix in array

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

    int a[3][3] , b[3][3] ;

    Printf("Enter element
            for 1st matrix :\n");

    for (int i=0 ; i<3 ; i++)
    {
        for (int j=0 ; j< 3 ; j++)
        { printf("Enter the value [%d][%d] : ", i,j);
          scanf(" %d", &a[i][j]);
        }
    }

    printf(" Enter element
            for 2nd matrix :\n");

    for (int i=0 ; i<3 ; i++)
    {
        for (int j=0 ; j<3 ; j++)
        {
          printf("Enter the value [%d][%d] : ", i, j);
          scanf("%d" , &b[i][j]);
        }
    }
```

for matrix a

$$\begin{bmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{bmatrix}$$

For matrix b

$$\begin{bmatrix} 2 & 4 & 6 \\ 2 & 4 & 6 \\ 2 & 4 & 6 \end{bmatrix}$$

multiply two matrix in array.

```
printf("addition of two given matrix.");

For (int i=0 ; i<3 ; i++){
 {
  for(int j=0 ; j<3 ; j++)
   {
    c[i][j] = a[i][j] +b[i][j];
   }
 }
```

for addition of two matrix

for addition of two matrix

for multiply two matrix

and

```
c[i][j] = a[i][j] * b[i][j];
```

print addition of two matrix

```
For (int i=0 ; i<3 ; i++)
 {
  for (int j=0 ; j<3 ; j++)
   {
    printf("%d", c[i][j]);
   }
  printf("\n");
 }
```

output :

Enter element for 1st matrix :

Enter the value [0][0] : 1
" [0][1] : 1
" [0][2] : 1
" [1][0] : 2
" [1][1] : 2
" [1][2] : 2
" [2][0] : 3
" [2][1] : 3
" [2][2] : 3

addition of two given matrix

4 4 4
4 4 4
4 4 4

Enter element for 2nd matrix :

Enter the value [0][0] : 3
" [0][1] : 3
" [0][2] : 3
" [1][0] : 2
" [1][1] : 2
" [1][2] : 2
" [2][0] : 1
" [2][1] : 1
" [2][2] : 1

multiplication of two given matrix

3 3 3
4 4 4
3 3 3

String

↳ A string is stored in one-dimensional array of type char

terminated by a null character '\0'

Syntax :-

```
data type string_name [size];
```

(char)

Declaration and initialization of create a string

```
char great[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
```

String input

```
scanf("%s", &s)   OR   gets(ch);
```

String output

```
printf("%s", s)   OR   puts(ch);
```

Function → Purpose

(1) **Strlen**("String"); → This function is used to determine the length of given string

(2) **Strcpy**
(source_variable, destination_variable);
→ This function is used to copies the string from one variable to another variable

(3) **Strcmp** (string1, string2);
→ This function is used to compare two string. {They are case sensitive}

(4) **Strrev** (string);
→ This function is used to reverse the character in a given string

(5) **Strlwr**
→ This function is used to convert string upper to lower

(6) **Strupr**
→ This function is used to convert string lower to upper

(7) **Strcat**(string1, string2)
→ This function is used to join / merge two string.

with String function

(1) using strupr

```
void main()
{
  char str[] = "ankit";
  clrscr();
  printf("%s", strupr(str));
  getch();
}
```

output ANKIT

(2) using strlwr

```
void main()
{
  char str[] = "ANKIT";
  clrscr();
  printf("%s", strlwr(str));
  getch();
}
```

output ankit

(3) using strrev

```
void main()
{
  char str[] = "ANKIT";
  clrscr();
  printf("%s", strrev(str));
  getch();
}
```

output TIKNA

(3) using **Strcat**

```
void main()
{
  char str1[] = "ANKIT", str2[] = "AMAN"
  clrscr();
  printf("%s", strcat(str1, str2));
  getch();
}
```

output : ANKIT AMAN

(4) using **Strcmp**

```
void main()
{
  char str1[]="ANKIT", str2[] = "AMAN"
  clrscr();
  if( strcmp[str1, str2) == 0 );
    printf("string are equal");
  else
    printf("Not equal");
    getch();
}
```

output :- Not equal

(5) using **Strcpy**

```
void main()
{
  char str1[] = "ANKIT", str2[10];
  clrscr();
  strcpy(str2, str1);

  printf("%s", str2);
  getch();
}
```

source variable, destination variable

output : ANKIT

(6) using **Strlen**

```
void main()
{
  char str[] = "ANKIT";
  clrscr();
  Printf("%d", strlen(str));
  getch();
}
```

output : 5

Find length of string (strlen) without using string function

(1) without using Strlen()

```
int main()
{
    char s[100];
    printf("Enter a string : ");
    scanf("%s", s);
    for (int i=0 ; s[i] != '\0' ; i++)
        printf("Length of string : %d", i);
    return 0;
}
```

output : Enter a string :
ANKIT
length of string : 6

(2) without using Strcpy()

```
int main() {
    char str1[20] = "COMPUTER";
    char str2[20];
    for (int i=0 ; str1[i] != '\0' ; i++)
    {
        str2[i] = str1[i];
    }
    str2[i] = '\0';
    printf("%s", str2[i]);
}
```

output :
COMPUTER

- karna ka without using string function
Concatate - to 1st length find kar ke one by one 2nd str1 me str2 ka character store kardege ar last me null store kardege

(3) Concatati with using strcat()

```
#include<stdio.h>
int main()
{
    char str1[50];
    char str2[50];
    printf("enter 1st string:");
    scanf("%s", str1);
    printf("enter 2nd string:");
    scanf("%s", str2);
  For(i=0 ; str1[i] != '\0' ; i++);
  For(j=0 ; str2[j] != '\0' ; j++, i++)
  {
      str1[i] = str2[j];
  }
  str1[i] = '\0';
  printf("\n concatenated string is %s", str1);
}
```

for checking length str1 to be store

for terminate

output :- enter 1st string : Hello
enter 2nd string : world

concatenated string is Helloworld

~~(4)~~ Copy

② It can be store → more than one value
→ different type of datatype
of different - different memory location

Structure ①

→ structure is used define datatype

2nd which is constructed by the help of primitive and derived datatype

Syntax :-

```
Struct structure_name
{
    data_type member 1;
    data-type member 2;
    data-type member 3;
};
```

→ termination of structure

2 byte int + 1 byte char + 4 byte float ⇒ Total = 7-byte

③ The size of structure is sum of it's all data member size

④ The minimum size of structure, is 1-byte

⑤ The structure is declared by the help of Struct keyword

⑥ Every structure should be terminate by the help of semi-colon (;)

for take input and store value from user

for display input →

most most imp (10M)

write a program to read and display student information (roll_no, name, address) using structure.

```
#include<stdio.h>                          // keyword
Struct student                             // structure name
{
    int roll;
    char name[20];
    char address[20];
};
int main()                                 // creat array structure
{
    Struct student s[3];
    Printf("Enter student record\n\n");
    For(int i=0; i<3; i++)
    {
        Printf("Enter Sroll : ");          // for specific member you want to interact
        Scanf("%d", &s[i].roll);
        printf("Enter Sname :");
        Scanf("%s", &s[i].name);
        Printf("Enter Saddress :");
        Scanf("%s", &s[i].address);
    }
    Printf("student record\n\n");
    For(int i=0; i<3; i++)
    {
        Printf("Sroll : %d\n", s[i].roll);     // for integer (int)
        Printf("Sname : %s\n", s[i].name);     // for string
        Printf("Sadd : %s\n", s[i].address);
    }
    return 0;
}
```

output :

```
Enter student record

enter Sroll : 10
enter Sname : ankit
enter Saddress : mumbai

enter Sroll : 20
enter Sname : amon
enter Saddress : kolapur

enter Sroll : 30
enter Sname : sam
enter Saddress : New York
```

student record

```
Sroll : 10
Sname : ankit
Saddress : mumbai

Sroll : 20
Sname : amon
Saddress : kolapur

Sroll : 30
Sname : sam
Saddress : New York
```