* ARRAYS:-

The primary datatypes like int, float and chas are used to store a single value at any given time

→ In some applications we need to store large amount of data in terms of reading, accessing. In such cases we can use one power full derived datatype is called an array.

ARRAY:-

It is a Homogeneous (similar) collection of related data items or elements which shares a common name.

Declaration:-

syntax:

```
Datatype   arrayname[size];
```

eg:- int a[5];

where datatype specifies the type of data contained in an array like int, float etc

→ ARRAY name means any user defined variable

→ size represents the number of elements in an array

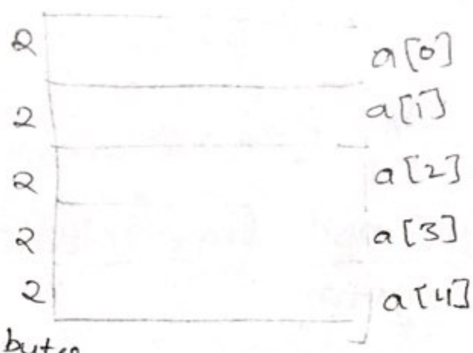
→ ARRAY index starts from zero. Each element in an array is uniquely identified by its subscript [] or index []

→ Memory is allocated sequencially for each element of 2 bytes.

→ If there are n [illegible] [illegible] then subscript range is: from 0 to [illegible]

→index
↓
a[0], a[1] - - - a[4]

→range
0, 1, 2, 3, 4



→Memory allocation = 10 bytes

*Types of arrays :-

→These are of 3 types

1). one dimensional array 2) two dimensional array
3) Multi dimensional array.

1) One dimensional array :-

→if an array contains only one subscript is called one dimensional array.

→ declaration: ~~eg:~~ Like any other variable array ~~syn~~ must be declared before its usage in the program

syntax:

Datatype arrayname[size];

eg: int group [10];

declares a group as an array to contain 10 integer constants.

→ initialization: The process of assigning the values to the array elements is called array initialization.

There are 2 types 1)Compile time initialization (during declaration)

2) Runtime initialization (with help of keyboard)

1) Compile time initialization

syntax:

```
Datatype arrayname [size] = {list of values};
```

eg: int a [5] = {35, 40, 69, 57, 20};

→ where list of values are separated by comma and should enclosed within the flower braces. and ends with a semi column.

eg: int a[5] = {1, 2};

→Note: if we miss any value to initialize, compiler automatically initializes it with zero or with garbage value

eg: int a[ ] = {1, 2, 3, 4, 5};

a[0]	a[1]	a[2]	a[3]	a[4]
1	2	3	4	5

eg: char a[3] = {'3', 'a', 'i'};

Q) write a c-program to ~~read~~ store and accessing of one dimensional array elements. using compile time initialization

Program

```
#include<stdio.h>
# include <conio.h>
void main()
{
    int a[5] = {30, 25, 57, 68, 79};
    clrscr();
    ~~for~~
    Printf("\n the array elements are:");
for (i=0; i<5 ;i++)
 {
   printf("%d \t", a[i]);
 }
 getch();
}
```

Output:

```
the array elements are
  30 25 57  68  79
```

* Run time initialization:

→ if we want to enter the elements at run time the following procedure must be followed.

1) Read the size : i.e how many elements you want to insert in an array.

2) To read the oned elements : one for loop and scanf is required

3) To print 1D array elements : one for loop & 1 printf is required

Reading of one-dimensional array elements

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

Accessing of one dimensional array elements

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

Q) Write a c program to read and display one dimensional array elements using run time initialization

Program

```
# include <stdio.h>
# include <conio.h>
void main()
{
  int a[10], i, n;
  clrscr();
  printf("\n enter the value of n:");
  scanf("%d", &n);
  printf("\n enter %d elements", n);
  for(i=0; i<n; i++)
  {
    scanf("%d", &a[i]);
  }
  printf("\n the array elements are");
  for (i=0; i<n; i++)
  {
    printf("%d\t", a[i]);
  }
  getch();
}
```

input: enter the value of n:6
enter 6 elements → 15, 20, 25, 30, 35, 40

Output

```
the array elements are

15     20     25   30   35   40.
```

8) write a 'c'-program to read n integers and find sum of the average of the integers in a one dimension array.

program

```
/* a program to display average of sum*/
#include <stdio.h>
#include <conio.h>
void main()
{
 int a[10];
 int sum=0, avg, i, n;
 clrscr();
 printf(" enter the value of n=\n");
 scanf("%d", &n);
 printf("enter the array elements =\n");
 for (i=0;i<n;i++)
 {
  scanf("%d", &a[i]);
 }
 for (i=0;i<n;i++)
 {
  sum = sum+a[i ];
 }
```

0<5→Sum = 0+a[0]=0+5 = 5

1<5(✓)→Sum= 5+ [illegible]

5+15= 30

+20

<5(✓)→ Sum= [illegible]

5.

<5(✓)→ Sum 5 [illegible]

5<5(✗)

```
Printf ("sum=%d", sum);
avg = sum/n;
~~Printf~~
printf ("avg = %d", avg);

getch();
}
```

Result

input: enter the value of n=5

enter the array elements

```
5
10
15
20
25
```

output

```
avg =15
```

write a 'c' program to read n [illegible] and find the largest and smallest [illegible] list of integers. (8) How

** 3) write a 'c' program To find out maximum and minimum elements in the given list of integers

(week 7(a))

Program

```
/* a program to display max & min integer*/
#include <stdio.h>
#include <conio.h>
void main ()
{
    int a[20], max, min;
    int i,n;
    clrscr();
    printf("enter the size=\n");
    scanf("%d", &n);
printf("enter the array values \n");
 for (i=0;i<n;i++)
 {
  scanf("%d", &a[i]);
 }
 max=a[0];
 min=a[0];
 for (i=0; i<n; i++)
 {
   if (a[i]>=max)
     max=a[i];
   if(a[i]<=min)
     min=a[i];
 }
 printf("\n the max=%d & min=%d", max,min);
 getch();
}
```

0<5(T)
if(a(0)>=1)=> 1>=1(T)
So, max=1;
if(a(0)<=1)=> 1<=1(T
So, min=1;
i++=i+1=0+1=1
1<5(T)
if(a(1)>=1)=> 2>=1(T)
So max=2,
if(a(1)<=1) => 2<=1(F).
So, min=1;
i++=1+1=2
2<5(T) if(a(2)>=1) => 3>=1(T). So, max=3,
3<5(T);i++=3=>if(a(3)>=1) => 4>=1(T). So, max=4
4<5(T);i++=4=> if(a(4)>=1) => 5>=1(T). So, max=5
i++=5 =>, 5<5(F)

Result.

```
enter the size=
5 -> n
enter array values
1  a[0]
2 -> a[1]
3 -> a[2]
4 -> a[3]
5 -> a[4]

the max=5 & min=1.
```

So, max=a[0]=1
min=a[0]=1

max=1
max=2
max=3
max=4
max=5

min=1

Date: 30/9/15

5. Write a c program to check whether the given number is prime or not ~~(~~

Algorithm

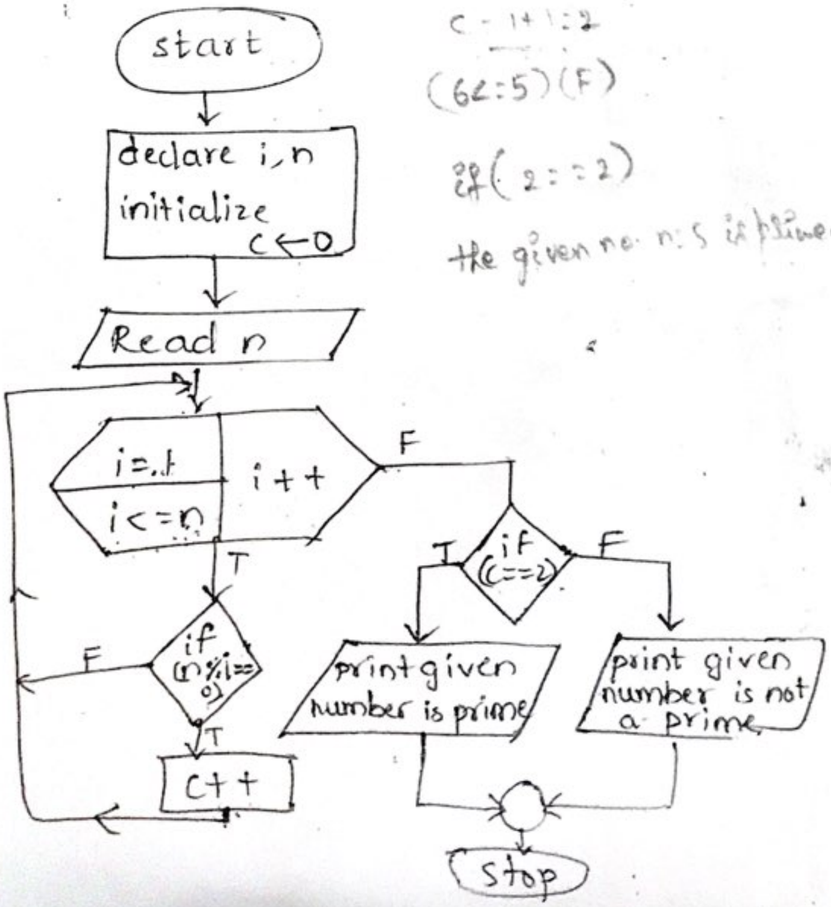
```
step1: start
Step2: Declare i, n, initialize c with 0
step3: Read n
Step4: for (i=1; i<=n; i++)
          if (n % i == 0)
            c++;
          go to step 4
Step5: if (c == 2)
          print the given number is prime
          otherwise
          print the given number is not a prime.
Step6: stop
```

flow chart:

1)5(5
5
(0)

c → count

```
(1<=5)(T).
(5%1==0)=>(0==0)(T)
c=0+1=1
(2<=5)(T)
(5%2==0)=>(1==0)(F)
(3<=5)(T)
(5%3==0)=>(2==0)(F)
(4<=5)(T)
(5%4==0)=>(1==0)(F)
(5<=5)(T)
(5%5==0)=>(0==0)(T)
c=1+1=2
(6<=5)(F)
if(2==2)
the given no. n=5 is prime.
```



program:

6✓

```
#include <stdio.h>
#include <conio.h>
void main()
{
 int i,n, c=0;
 clrscr();
 printf ("\n enter n value:");
 scanf ("%d", &n);
 for (i=1; i<=n; i++)
   {
        if (n%i==0)
          c++;
   }
   if(c==2)
   printf("\n the given number is prime)
   else
    printf("\n the given number is not a prime)
   getch();
}
```

Result:

input: enter the n value
          5

output: the given number is prime.

* Two dimensional array.

If an array contains two subscripts or two indice then it is called two dimensional array

The data items are stored in this array in the form of matrix or table

Declaration

Syntax : Datatype arrayname [row size][col size

int a [2][2]; 2* 4

No of elements = 4

subscript range = a[0,0],a[0,1],a[1,0],a[1,1]

Memory allocation = 4 x 2 = 8 bytes

Initialization:

There are 2 ways 1) during declaration(or) compile time
2) during execution (or) Runtime initialization
with help of keyboard

1) Compile time:

Syntax

Datatype arrayname [rowsize][colsize] = {set of elements}
" " " (or) = {{set1},{set2}.....{setn}}

eg: int a[2][2] = {1,2,3,4};
(or) int a[2][2] = {{1,2},{3,4}};

	0	1
0	a[0,0]	a[0,1]
1	a[1,0]	a[1,1]

	0	1
0	1	2
1	3	4

$$\begin{array}{c|cc} & 0 & 1 \\ 0 & 1 & 2 \\ 1 & 3 & 4 \end{array}$$

eg:/* a program to display 2 dimensional aeeayelements using compile time initialization*/

```
#include < stdio.h>
# include<conio.h>
void main()
{
int a[3][3]= {5,10,15,20,25,30,35,40,45};
int i,j;
clrscr();
printf("\n The array elements are:");
 for(i=0;i<3;i++)
 {
   for (j=0;j<3;j++)
   {
    printf("%d\t",a[i][j]);
   }
  printf("\n");
 }
 getch();
}
```

Result : the array elements aee

```
5   10  15
20  25  30
35  40  45
```

```
i=0 => 0< 3
      j=0 => 0<3 => a [0][0] = 5
      j=1 => 1< 3 => a [0][1] = 10
      j=2 => 2 <3 => a [0][2] = 15
i=1 => 1< 3
      j= 0   0<3  => a[1][0] = 20
      j= 1   1<3  => a [1] [1] = 25
      j= 2   2<3 => a [1] [2] = 30
i= 2 => 2< 3
      j= 0   0<3  => a[2] [0] = 35
      j= 1   1< 3 => a [2][1] = 40
      j= 2   2 < 3 => a [2][2] = 45
```

2) Runtime initialization:

Procedure

1) Read the rowsize and column size; i,e, how many elements want to insert in 2 dimensional array.

2) Read the array elements: with the help of 2 for loops and one scanf functions

3) Print or display the array elements; with help of 2 for loop and one printf function

for reading

rs → row size

cs → column size

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

for printing

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

eg.

Q) Write a 'c' program to read and display twodimensional array elements

Program:

```
void main()
 {
  int a[3][3], i,j;
  clrscr();
  printf("\n enter the array elements");
  for (i=0; i<3; i++)
   {
    for(j=0;j<3;j++)
     {
     scanf("%d", &a[i][j]);
     }
pointf("\n the array elements are");
for(i=0;i<3;i++)
 {
  for(j=0; j<3; j++)
   {
    printf("%d\t", a[i][j])
   }
  printf("\n");
 }
getch();
}
```

```
Result:  i/p  Enter the array element .
              1   2   3   4   5  6  7  8  9
         o/p the array elements are  -  1   2   3
                                        4   5   6
                                        7   8   9
```

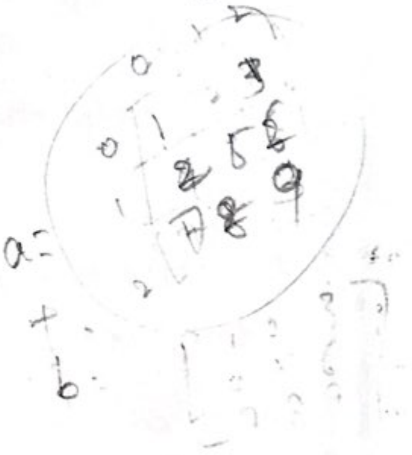
8) Write a C program to display the addition of two 3X3 matrices

```
/* a program to display addition of 2 3x3 matrices*/
# include <stdio.h>
# include< conio.h>
void main()
{
 int a[3][3], b[3][3], c[3][3], i, j;
 clrscr();
 pointf("\n enter the array elements ~~in a~~ of a");
 for (i=0; i<3; i++)
  {
   for (j=0; j<3; j++)
    {
     scanf("%d", &a[i][j]);
    }
  }
 for (i=0; i<3; i++)
  {
```

printf("\n enter the array elements of b");

```
    for(j=0;j<3; j++)
    {
    scanf("%d", &b[i][j]);
    }
  }
 printf("\n array elements are\n");
 printf("a:\n");
 for(i=0;i<3; i++)
 {
 for(j=0; j<3; j++)
  {
  printf("%d\t", a[i][j]);
  }
  printf("\n");
 }
 printf("b:\n");
  for(i=0;i<3;i++)
  {
   for(j=0; j<3;j++)
   {
   printf("%d\t", b[i][j]);
   }
   printf("\n");
  }
for(i=0; i<3; i++)
{
 for(j=0; j<3; j++)
 {
 c[i][j]=a[i][j] + b[i][j];
 }
}
printf("c=\n");
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{ printf("%d\t", c[i][j]);
}
printf("\n");
}
getch();
```

$$C = \begin{bmatrix} 2 & 4 & 6 \\ 8 & 10 & 12 \\ 14 & 16 & 18 \end{bmatrix}$$



a[3][3] = {{1,2,1}, {2,1,2},
{1,2,1}, {2,1,2},
{1,2,1}, {2,1,2}};

Result.

enter array elements
1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2

o/p array elements are

```
a: 1  1  1
   1  1  1
   1  1  1
b: 2  2  2
   2  2  2
   2  2  2
c = 3  3  3
    3  3  3
    3  3  3
```

Multiplication of two matrices

Program:

```
#include <conio.h>
# include < stdio.h>
void main()
{
    int a[10][10], b[10][10], c[10][10];
    int i, j, r1, r2, c1, c2, k ~~[illegible]~~;
    clrscr();
    printf("Details of matrix A");
    printf("\n enter row size and col size of matrix A");
    scanf("%d%d", &r1, &c1);
    printf("\n enter the elements of matrix A");
    for(i=0; i<r1; i++)
     {
      for(j=0; j<c1; j++)
       {
        scanf("%d", &a[i][j]);
       }
     }
     printf("\n details of matrix B");
     printf("\n enter row size and col size of matrix B");
     scanf("%d %d", &r2,&c2);
     printf("\n enter the elements of matrix B");
     for(i=0; i<r2; i++)
     {
       for(j=0; j<c2; j++)
        {
         scanf("%d",&b[i][j]);
        }
     }
     if (c1 ==r2)
      {
       for (i=0; i<r1; i++)
        {
         for(j=0; j<c2; j++)
          {
                c[i][j]=0;
                for(k=0; k<c1;k++)
                 {
                   c[i][j]=c[i][j]+a[i][k]*b[k][j];
                 }
            }
        }
    printf("\n The Resultant matrix is");
    for(i=0;i<r1;i++)
    {
     for(j=0;j<c2;j++)
       {
        printf("%d \t", c[i][j])
       }
       printf("\n");
    }
  }
  else
  printf("\n matrix multiplication is not possible");
  getch();
}
```

Result

Date: 27/10/15

5) Write a 'c' program to determine the addition of two matrices

program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
 int a[10][10], b[10][10], C[10][10];
 int i, j, r1, r2, c1, c2;
 clrscr();
 printf("\n Details of Matrix A");
 printf("\n Enter row size & col size of matrix A");
 scanf("%d %d", &r1 , &c1);
 printf("\n enter the elements of matrix A");
 for (i=0 ; i<r1 ; i++)
  {
   for(j=0 ; j<c1 ; j++)
    {
     scanf("%d", &a[i][j]);
    }
  }
 printf("\n Details of matrix B");
 printf("\n Enter row size & col size of matrix B");
 scanf("%d %d", & r2 , &c2);
 printf("\n enter the elements of matrix B");
 for(i=0;  i<r2 ; i++)
  { for(j=0; j<c2 ; j++)
    {
     scanf("%d", &b[i][j]);
    }.
  }
if(( r1==r2)&&(c1==c2))
{
  for(i=0;i<r1;i++)
  {
    for(j=0;j<c1;j++)
    {
      c[i][j]=a[i][j]+b[i][j];
    }
  }
  printf("\n the resultant matrix is");
  for(i=0;i<r1;i++)
  {
    for(j=0;j<c1;j++)
    {
      printf("%d\t", c[i][j]);
    }
    printf("\n");
  }
}
else
printf("\n matrix addition is not possible");

getch();
}
```

Result: i/p details of matrixA

```
enter rowsize and col size of matrix A
2
2
enter the elements of matrix A
1  2  3  4
 details of matrix B
enter row size and col size of matrix B
2
2
enter the elements of matrix B
1   2   3   4
```

o/p the resultant matrix is

```
2   4
6   8
```

```
#include <stdio.h>
void main()
{
int A[2][3], B[3][4], c[2][4];
int i, j, k;
pf("enter 'A' elements");
for(i=0; i<2; i++)
 {
 for(j=0; j<3; j++)
  {
  sf("%d", & A[i][j]);
  }
 }
 pf("enter B elements");
 for(i=0; i<3; i++)
  {
  for(j=0; j<4; j++)
   {
   sf("%d", & B[i][j]);
   }
  }
 if (col of 1st mat == rows of 2nd mat)
 {
 for(i= 0; i<2; i++)
   {
   for(j=0; j<4; j++)
    {    c[i][j]=0;
    for(k=0; k<3; k++)
     {                          0   0
      C[i][j]= C[i][j]+ A[i][k]* B[k][j];
     }
    }
   }
   pf("multiplication is");
   for(i=0; i<2; i++)
    {
    for(j=0; j<4; j++)
     {
     pf("%d", c[i][j]);
     }
     pf("\n");
    }
 }
```

$$\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 4 & 1 & 2 \\ 1 & 6 & 0 & 3 \end{array}_{2\times3}$$

-A-MATRIX

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 5 & 7 & 2 & 9 \\ 1 & 8 & 3 & 4 & 0 \\ 2 & 6 & 1 & 2 & 7 \end{array}_{3\times4}$$

B-MATRIX.

Multi-dimensional Array:

→If an array contains more than two subscripts then it is called multi-dimensional array.

syntax for declaration

Datatype arrayname[size1][size 2]......[sizen];

eg: int a[3][3][3];

No of elements = 27

subscript range = a[0][0][0], a[0][0][1], ... a[2][2][2]

Memory allocation = 27 X 2 = 54 bytes

Compile time initialization

Program:-

```
#include<stdio.h>
#include<conio.h>
void main()
{
  int a[2][2][2]={10,15, 20,25},{30, 35, 40, 45}}, i, j, k;
  clrscr();
  for(i=0 ; i<2 ; i++)
   {
    for(j=0 ; j< 2;j++)
     {
      for(k=0 ; k<2 ; k++)
       {
        printf("%d \t", a[i][j][k]);
       }
     }
   }
   getch();
}
```

Evaluation

```
i=0              i=0
j=0              j=0
k=0              k=1
a[0][0][0]=10    a[0][0][1]=15

i=0              i=0
j=1              j=1
k=0              k=1
a[0][1][0]=20    a[0][1][1]=25

i=1              i=1
j=0              j=0
k=0              k=1
a[1][0][0]=30    a[1][0][1]=35

i=1              i=1
j=1              j=1
k=0              k=1
a[1][1][0]=40    a[1][1][1]=45
```