The do... while loop

Date
Page

★ Syntax :-

```
do
{
    ....
    ....
    ....
} while( test_cond.);
```

part of syntax

// Missing this " ; " will produce syntax Error !

★ Tracing of "do ... while"

Ex :- using while

```
int main()
{
  int i=1;
  while( i<=10)
  {
      printf("%d \n", i);
      i++;
  }
  return 0;
}
```

Ex :- using do ... while.

```
int main()
{
    int i = 1;
    do
    {
        printf("%d \n", i);
        i++;
    } while(i<=10);
    return 0;
}
```

Output :- Both the loop in this program will provide same out put.

Date
Page

★ Predict the output :-

```
int main()
{
  int i = 11;
  while (i<=10)
  {
    printf("%d\n", i);
    i++;
  }
  return 0;
}
```

output :- No output

```
int main()
{
   int i=11;
   do
   {
      printf("%d\n", i);
      i++;
   } while(i<=10);
   return 0;
}
```

11

★ Important point :-

- Points regarding "while" and "do-- while"

- Minimum iteration of the while loop is 0, and the minimum iteration of do -- while loop is 1.

- "while" loop is also called as Entry controlled loop The test condition is evaluated before the loop body.

Ex:-
```
while (test_cond.)
{
---
---
}
```

Date
Page

o do-- while loop is also called as Exit controlled loop The test condition is evaluated after the loop body.

Ex.. do
{ - - - -
} while (test -- cond);

Q WAP to accept 2 integers from the user, display their sum and again ask the user if he wants to continue. if the answer is yes, then repeat the above process, otherwise terminate the program.

Ex:-

Enter 2 integers : 10 20
Sum is : 30
Do you want to continue/repeat (Y/N) : Y
Enter 2 integers : 7 2
Sum is : 9
Do you want to repeat (Y/N) : N

A Solution from do --- while.

```
int main()
{
   int a, b;
   char choice;
   do
   {
     printf("Enter 2 integers : ");
     scanf("%d %d", &a, &b);
```

```
    printf(" sum is :%d \n ", a+b );
    printf(" Do you want to repeat (Y/N) : ");
    fflush(stdin);
    scanf("%c", &choice);
    } while ( choice == 'Y');
    return 0;
}
```

Solution from while.

Note :- if we will write the same program in while loop and the while condition will same then the output will give unpredictable behavior

So improved version of the code from while.

```
int main()
{
   int a, b;
   char choice = 'Y';
   while (choice == 'Y');
   {
     printf(" Enter 2 integers : ");
     scanf(" %d %d , &a, &b );
     printf(" sum is : ",a+b);
     printf(" Do you want to repeat (Y/N): ");
     fflush (stdin);
     scanf("%c", &choice);
     }
     return 0;
}
```

Q→ WAP to accept integers from the user and print its factorial. Make sure your program prints 1 if the user enters 0.

Solution from while :-

```
int main():
{
    int n, f = 1;
    printf(" Enter a no : ");
    scanf("%d", &n);
    while( n>1)
     {
        f = f * n ;
        n--;
     }
      printf(" fact is %d ", f );
     return 0;
}
```

Solution from do -- while :-

```
int main()
{
    int n, f = 1 ;
    printf (" Enter a no : ");
    scanf("%d", &n );
    do
    {
        f = f * n;
        n--;
    } while(n>1);
```

```
    printf(" fact is %d ", f);
    return 0;
}
```

It will show factorial 0 is 0. which is wrong behavior.

So improved version using do ... while

```
int main()
{
    int n, fact = 1;
    printf(" Enter a no : ");
    Scanf(" %d", &n );
    if (n!=0)
    {
        f = f * n ;
        n--;
    } while (n>1);
    printf(" Fact is %d ", f);
    return 0;
}
```

(n>0) also take place.

Q→ Modify the do while version of factorial program so that without using if statement, the code can handle 0! also.

Q→ WAP to accept an integer from the user and find out the sum of the digits of that number.

Ex:- Enter a no : 294
Sum of digits : 15

Note:- Number can be of any number of digits

The "for" loop.

Date
Page

- Standard syntax of "for" loop :

// for loop is also known as Entry controlled loop.

~~for (<test-co~~ int

```
for(<initialization> ; <test-cond> ; <statement>)
{
  - - -
}
```

Q→ WAP using for loop to print numbers from 1 - 10 .

```
int main()
{
 int i;
 for ( i=1 ; i<=10 ; i++ )
  {
    printf ("%d\n" , i);
  }
  return 0;
}
```

"IMP Question for Inter."

Q→ How many types of loops are there in C?

Ans:- 2

(a) Entry controlled loop

1) while
2) for

(b) Exit controlled loop.

1) do ... while

Q→ WAP a program to print all the number from 10 to 1.

```
int main()
{
 int i;
  for (i=10; i>=1; i--)
  {
     printf("%d\n", i);
  }
     return 0;
}
```

Q→ WAP to accept an integer from the user and print the series from 1 to that number. Assume the user will input positive number only.

Solution :-

```
int main()
{
    int i, n;
    printf("Enter a no : ");
    scanf("%d", &n);
    for(i=1; i<=n; i++)
    {
      printf("%d\n", i);
    }
    return 0;
}
```

Q> WAP to accept an integer from the user and print the sum of the series from 1 to that number. Assume the user will input positive number only.

Note > ~~Ase~~ Ans will take two variables for better program.

Q> WAP to calculate the factorial of a number given by the user. Use only 2 variables. Make sure that if the user inputs 0, the program displays 1.

Date
Page

Q> WAP to accept an integer from the user and calculate and print the sum of the digits of that number use for loop.

```
int main()
{
   int n, sum , x;
   printf ("Enter a no : ");
   scanf("%d ", &n);
   for (sum =0; n>0; n = n/10)
   {
      x = n%10;
      sum = sum + x;
   }
    printf("sum of digits is %d ", sum);
   return 0;
}
```

Q> WAP to accept an integer from the user and check wether it is an ~~Ams~~ Armstrong number or not.

=> Armstrong numbers are those numbers that have a special property. the sum of the cube of all the digits of the number is equal to the number itself.

Solution:-

```
Void main()
{
    int n, Sum, x, a;
    clrscr();
    printf("Enter a no : ");
    scanf("%d", &n);
    a = n;
    for(sum =0; n>0; n = n/10)
    {
      n = n%10;
      Sum = sum + x * x * x;
    }
    if( a == Sum)
        printf("Number is Armstrong ");
    else
      printf(" Number is not Armstrong ");

  getch();
}
```

Verieties of "for" loop

Standard syntax of for loop :-

```
for ( initi ; test_cond ; stmt )
{
    ---
}
```

☆ First Variety of "for" loop.

(1) Initialization is optional :-

Syntax :-

```
for( ; test_cond ; stmt)
{
    :
}
```

Ex :-

```
int main()
{
    int n ;
    printf("Enter a no. ");
    scanf("%d ", &n);
    for (; n>0 ; n--)
    {
        printf("%d\n", n);
    }
    return 0;
}
```

Date
Page

★ Second veriety of "for" loop

(2) Statement is opetional :-

Syntax :-

```
for( ; test-condition; )
{
    - - -
    - - -
}
```

Ex :-

```
int main()
{
    int n;
    printf("Enter a no. ");
    scanf("%d", &n);
    for(; n>0;)
    {
        printf("%d\n", n);
        n--;
    }
    return 0;
}
```

★ Third variety of "for" loop.

(3) Even condition in for loop can be left blank

Syntax :-

```
for( ; ;)
{
    - - -
    - - -
}
```

It always represents true condition

Ex:-

```
int main()
{
  int n;
  printf(" Enter a no : ");
  scanf("%d", &n);
  for(;;)
   {
     printf(" %d\n", n);
     n--;
   }
  return 0;
}
```

Runtime Error

★ Imp Point :-

following are the syntax Error :-

```
while() X        if() X        do X
{                {             {
}                }             } while();
```

★ —: Using " Break " with loop :—

- in C language, the "break" keyword can also be used with (all loops).
- But there are 2 points we must remember before using "break" in the loop.

(1) we must enclose " break " inside an if statement.

(2) When "break" will run, it will terminate the entire loop body.

Syntax :-

(A) Using " break with for :-

```
for ( initi ; test-cond ; Stmt )
  {
    - - - - -
    if ( test - cond )
      {
          break ;
      }
    - - - - -
  }
```

⇒ When will the loop terminate ?

- Either the loop will when the parent condition false. This termination of the loop is the loop is also known as Mature Termination.

/OR/

- The loop might also terminate because of "break". This termination of the loop is also know as pre-Mature Termination.

Ex :-

• Demo of for loop using break inside the loop.

```
int main()
{
  int i;
  for(i=1 ; i<=10 ; i++)
     {
        if(i==6)
           break;
        printf("%d ",i);
     }
     return 0;
}
```

Output :-

1
2
3
4
5

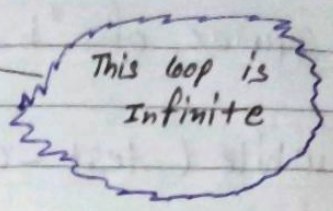
Syntax :- Using "break" with for :

```
for( initi ; ; stmt)
  {
    - - - - - -

    if( test-cond)
      {  break;
      }
    - - - - - -

  }
```

★ Important point :-

• If the internal break is not present the loop will infinite which is a Run time Error :-

```
for( initi ; ; stmt)
  {
    - - - - - ;
    - - - - - - -
  }
```



Ex :- Ex of break in for loop :-

- Improved version of the following example using break

Example

```
Ex :- int main()
{
   int n ;
   printf (" Enter a no. ");
   scanf(" %d ", &n);
   for(; n>0 ;)
   {
      printf(" %d\n ", n);
      n--;
   }
   return 0;
}
```

Improved version

```
int main()
{
   int n ;
   printf("Enter a no ");
   scanf(" %d", &n);
   for (; ;)
   {
      if(n<=0)
         break ;
      printf(" %d \n", n);
      n--;
   }
   return 0;
}
```

(i) Syntax of "break" with while :

```
while ( test_cond )
{
    - - - - - -
        if ( test_cond )
            break ;
    - - - - - -
    - - - - -
}
```

(ii) Syntax of "break" with do --- while :

```
do
{
    - - - - - -
    if ( test_cond )
        break ;
    - - - - - -
} while ( test_cond );
```

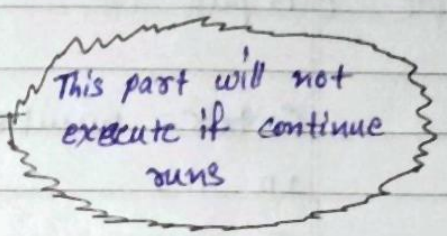
★ Using Keyword "continue"

In C language, the "continue" keyword can also be used with (all loops.)

(1) The keyword "continue" can only be used within a loop and not without it.

Syntax :-

```
for( initi ; test_condi ; stmt )
{
    - - - -
        if( test_cond )
        {
            continue ;
        }
    - - - - - -
}
```



- "continue" never stops the loop but "continue" can stop the current iteration of the loop and moves the control to the statement part.

⇒ Following are the examples of continue in for loop.

(1)

```
int main()
{
 int i;
 for(i=1; i<=10; i++)
  {
      if(i==6)
        break;
      printf("%d\n",i);
  }
  return 0;
}
```

(2)

```
int main()
{
int i;
for(i=1; i<=10; i++)
{
   if(i==6)
    continue;
   printf("%d\n",i);
}
 return 0;
}
```

Q→ WAP to ask the user to input numbers continuously until 0 is entered. As soon as the user enters 0, stop accepting numbers and display the sum of all the Numbers entered before 0.

Sample output :—

```
Enter numbers and press 0 to stop :
10
5
7
0
Sum is : 22
```

A "Solution"

```
int main()
{
  int n, sum = 0 ;
  printf(" Enter numbers and press 0 to stop : ");
  for( ; ; )
   {
      Scanf(" %d ", &n);
      if( n == 0)
          break ;
       sum = sum + n ;
   }
  printf(" sum is : %d ", sum);
  return 0;
}
```

Q→ Modify the previous code so that now your program ignores negative numbers and adds positive numbers

Ex :- Enter numbers and press 0 to stop :

10
5
-7
8
3
2
-6
0

Sum is : 28

"Solution"

```
int main()
{
   int n, Sum = 0;
   printf(" Enter numbers and press 0 to stop : ");
   for (; ;)
   {
        scanf(" %d ", &n);
        if (n==0)
            break;
        if (n<0)
            continue;
        Sum = sum + n;
   }
   printf(" Sum is : %d ", sum);
   return 0;
}
```

Exercise :-

Q→ Again modify the previous code so that alone with sum your program calculates and prints average of the numbers Also.

★ Fourth variety of "for" loop

(4) Multiple initialization, test condition and statements are also allowed

Syntax :-

```
for( multi-init ; multi-condi ; multi-Stmt )
 {
    ---  - ---
    ---  - ---
 }
```

Ex :-

```
int main()
{
  int i, j;
  for(i=1, j=10 ; i<=10 && j>=1 ; i++, j--)
  {
     printf(" %d  %d\n", i, j);
  }
  return 0;
}
```

Date
Page

☆ Predict the Output :-

(1)
```
int main()
{
 int i, j;
 for(i=1, j=10; i<=10 && j>=5 ; i++, j--)
  {
      printf("%d  %d\n", i, j);
  }
  return 0;
}
```

(2)
```
int main()
{
  int i, j;
  for (i=1, j=10; i<=10 || j>=5 ; i++, j--)
  {
      printf("%d  %d\n", i, j);
  }
   return 0;
}
```

Date
Page

☆ Predict the output :-

⇒ Programmer's view :-

```
int main()
{
    int i;
    for(i=1 ; i<=10 ; i++);
    printf("%d\n",i);
    return 0;
}
```

output :- 11

⇒ compiler's view :-

```
int main()
{
    int i;
    for(i=1 ; i<=10 ; i++);
    {

    }
    printf("%d\n",i);
    return 0;
}
```

Date
Page

A Predict the Output :-

⇒ Programmer's view :-

```
int main()
{
    int n=5;
    if(n%2==0);
      printf("Even No.");
    return 0;
}
```

⇒ compiler's view :-

```
int main()
{
    int n=5;
    if(n%2==0);
    {

    }
    printf("Even No.");
    return 0;
}
```

☆ Predict the output :-

⇒ Programmer's view :-

```
int main()
{
    int n = 5;
    if(n%2 == 0);
        printf("Even No.");
    else
        printf("Odd No");
    return 0;
}
```

← Syntax Error! Misplaced else

⇒ Compiler's view :-

```
int main()
{
    int n=5;
    if(n%2 == 0);
    {

    }
        printf("Even No.");
    else
        printf("Odd No.");
    return 0;
}
```

Syntax Error! Misplaced else

Q→ WAP to accept an integer from the user and check whether it is a prime number or not. Assume that the user will not enter numbers less than 2.

Solutions-

```
int main()
{
    int i, n;
    printf(" Enter a no. ");
    scanf("%d", &n);
    for(i=2 ; i<=n-1 ; i++)
    {
        if(n%i==0)
            break;
    }
    if(n==i)
        printf(" Number is prime ");
    else
        printf(" Number is not prime ");
    return 0;
}
```

⇒ Now,

Improved version of the above code.

P.T.O.

```
int main()
{
  int i, n;
  printf(" Enter a no : ");
  scanf(" %d ", &n);
  for( i=2 ; i<n/2 ; i++)
   {
       if(n%i==0)
         break;
   }
   if(i== n/2)
     printf(" Number is prime ");
   else
     printf(" Number is not prime ");
   return 0;
}
```

"NESTED Loop"

Syntax :-

```
for(init ; test_cond ; stmt)
  {
      for( init ; test_cond ; stmt)
      {
        - - -
      }
    - - -
  }
```

Example :-

```
int main()
{
    int i,j;
    for( i=1; i<=3; i++)
    {
        for(i=10; j<= 15; j++)
        {
            printf(" %d ", j);
        }
        printf("\n");
    }
    return 0;
}
```

Output :-

```
10  11  12  13  14  15
10  11  12  13  14  15
10  11  12  13  14  15
```

Exercise :-

Q) WAP to print the following pattern in console

```
* * *
* * *
* * *
* * *
```

Solution :-

```
int main()
{
    int i, j;
    for (i=1; i<=4; i++)
    {
        for (j=1; j<=3; j++)
        {
            printf("* ");
        }
        printf("\n");
```

```
    }
    return 0;
}
```

O→

```
*
* *
* * *
* * * *
```

Solution :-

```
int main()
{
    int i, j;
    for(i=1; i<=4; i++)
    {
        for(j=1; j<=i; j++)
        {
            printf("* ");
        }
        printf("\n");
    }
    return 0;
}
```

O→

```
1
1 2
1 2 3
1 2 3 4
```

O→

```
1
2 2
3 3 3
4 4 4 4
```

O→

Date
Page

Q=> 4 3 2 1
4 3 2
4 3
4

Solution:-
```
int main()
{
    int i, j;
    for (i=1; i<=4; i++)
    {
        for (j=4; j>=i; j--)
        {
            printf("%d ", j);
        }
        printf("\n");
    }
    return 0;
}
```

Q=> 4 3 2 1
3 2 1
2 1
1

Solution:-
```
int main()
{
    int i,j;
    for (i=4; i>=1; i--)
    {
        for(j=i; j>=1; j--)
        {
            printf("%d ", j);
        }
        printf("\n");
    }
    return 0;
}
```

Date
Page

Q>
A
A B
A B C
A B C D

Solution:-

```
int main()
{
   char i,j;
   for(i=65 ; i<=68 ; i++)
   {
     for(j=65; j<=i ; j++)
     {
        printf("%c", j);
     }
     printf("\n");
   }
   return 0;
}
```

Q>
4
4 3
4 3 2
4 3 2 1

Q> 1
2 3
4 5 6
7 8 9 10
(Hint : use 3 variables)

Q> D C B A
D C B
D C
D

Q>
1
1 2
1 2 3
1 2 3 4

(Hint: use 3 loops)

Q>
1
1 2 1
1 2 3 2 1
1 2 3 4 3 2 1

Q→ WAP to accept an integer from the user and print all the ARMSTRONG NUMBERS from 1 to that number.

Ex:- Sample Output :-

(a) Enter a no : 200
1
153

(b) Enter a no : 500
1
153
370
371
407

Solution:-

```
int main()
{
  int n, i, x, rem, sum;
  printf("Enter a no : ");
  scanf("%d", &n);
  for(i=1, i<=n; i++)
  {
      x = i;
      Sum = 0;
      while(x>0)
      {
          rem = x%10;
          Sum = Sum + rem*rem*rem;
          x = x/10;
      }
      if(Sum == i)
          printf("%d\n", i);
  }
  return 0;
```

"Arrays"

Date
Page

⇒ To understand the array we have to solve the following program using the concept we have discussed till now.

⇒ WAP to accept 5 integers from the user and display them back on the screen.

Sample output:-

```
Enter a number : 10
Enter a number : 7
Enter a number : 25
;
You inputted :
10
7
25
;
```

Approch 1:-

```
int main()
{
  int a, b, c, d, e;
  printf(" Enter a number : ");
  scanf(" %d", &a);
  printf(" Enter a number : ");
  scanf(" %d ", &b);
  printf(" Enter a number : ");
  scanf(" %d ", &c);
  printf(" Enter a number : ");
  scanf(" %d ", &d);
  printf(" Enter a number : ");
  scanf(" %d ", &e);
```

```
printf(" You inputted: \n ");
printf(" %d \n %d \n %d \n %d \n %d \n", a, b, c, d, e );
return 0;
}
```

★ Another solution to the above problem. :-

- Now we have the same problem as follows :

Q⇒ WAP to accept 50 integers from the user and display them back on the screen.

⇒ Now, 50 variables and the 50 pairs of printf, scanf will be required which is too time-consuming.

- So, we have another solution provided by the C language.

—: Array :—

Definition :- In C language, an array is always defined as a collection of similar kinds of data elements stored at CONTINUOUS memory location.

- Similar kinds : In C language array is a collection of homogeneous data.
- Continuous :- compiler in C language in the array always store element one after the other in continuous form

Date
Page

★ Syntax of Declaring an Array :

<data_type> <array_name>[<size>];

Number of values an array can hold

for Ex:-

```
int roll[10];
char grade[15];
float per[20];
```

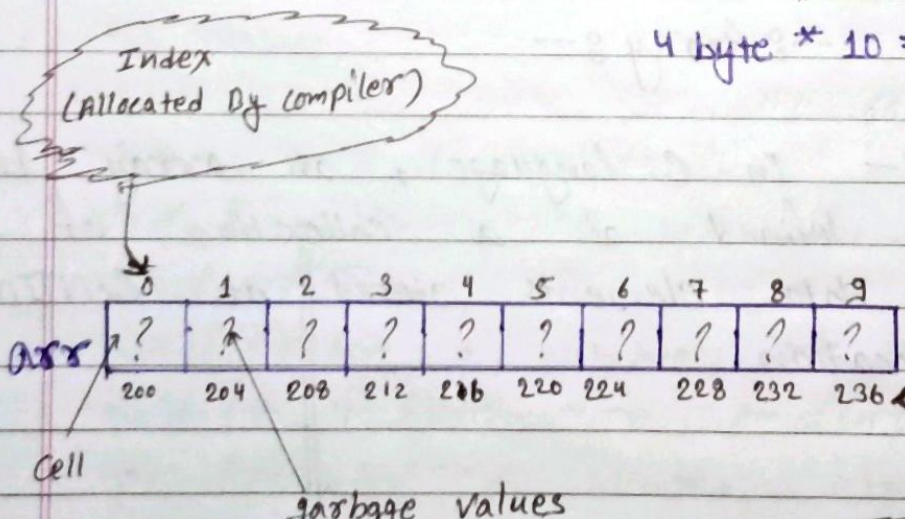
Example of Array 1or1
Examples of single Dimensional Array.

o Single dimensional integer Array :-

Supose, we write the following statement

int arr[10]; → How compiler deals with this:

4 byte * 10 = 40 Bytes



Date
Page

★ How do we access elements of the array :-

int arr[10]; → square Bracket

arr = 25; ✗ Syntax Error

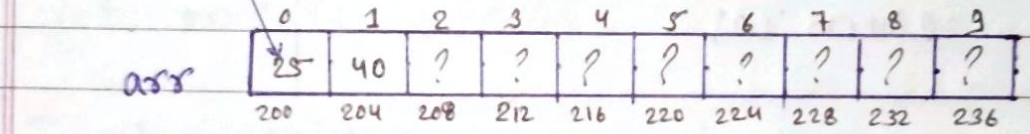
↳ can be written as :-

```
arr[0] = 25;
arr[1] = 40;
```

Syntax :-

<array>[<index>] = <value>;

we want to store 25 → Subscript operator



★ Accepting Input from the user :-

Sample output :-

```
Enter a number : 40
Enter a number : 25
Enter a number : 30
!
!
```

You inputted :-

```
40
25
30
!
!
```

Solution :-

```
int main()
{
  int arr[10];
  int i;
  for(i=0; i<=9; i++)
  {
    printf("Enter a number :");
    scanf("%d", &arr[i]);
  }
  printf("\n You inputted ");
  for(i=0; i<=9; i++)
  {
    printf("\n%d", arr[i]);
  }
  return 0;
}
```

	arr	
200	40	0
204	25	1
208	30	2
212	45	3
216	10	4
220	18	5
224	27	6
226	49	7
228	51	8
232	23	9

★ You also know this :-

- In actual indexes are not physically available the are converted to address through using formula used only by the compiler on the address of the array.

arr[i] → formula → Address

- what if iterated loop on the array more than its size ?

- Unpredictable Output, i.e. if the space on the successor memory cell is free then the program displays the output correctly but

If the successor cell occupied by another program the code might be crash and can give absurd behavior or runtime error.

* Syntax Errors in Array Declaration :-

1). The size is given to an array while declaring it Must compulsorily be AN INTEGER CONSTANT and

```
int n = 10;
int arr[n]; //ERROR
```

⇒ This restriction is only in turbo compiler not in Gcc or clang or LLVM or MSVC (Microsoft visual compiler) compilers

2.) The size of an array is compulsory i.e we cannot leave the array size blank.

```
int arr[]; //ERROR
```

⇒ In one special case this is allowed.

3.) The minimumum size we can give to an array is 1.

int arr[0]; //ERROR

★ Initializing An Array :-

1.)
```
int months[12];
months[0] = 31;
months[1] = 28;
months[2] = 31;
months[3] = 30;
months[4] = 31;
months[5] = 30;
months[6] = 31;
months[7] = 31;
months[8] = 30;
months[9] = 31;
months[10] = 30;
months[11] = 31;
```

If any of the index is missed than it will by default contain GARBAGE VALUE

2.)
```
int months[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31,
                   30, 31 };
```

Initializers or Initializer list

3.)
```
int months[12];
months = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
```
//ERROR → we can not initialize a array

like a variable.

4.) int months[12] = {31, , 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

Hole

holes are not allowed

5.) int arr[5] = {10, 20, 30, 40, 50}; ⇒ OK.

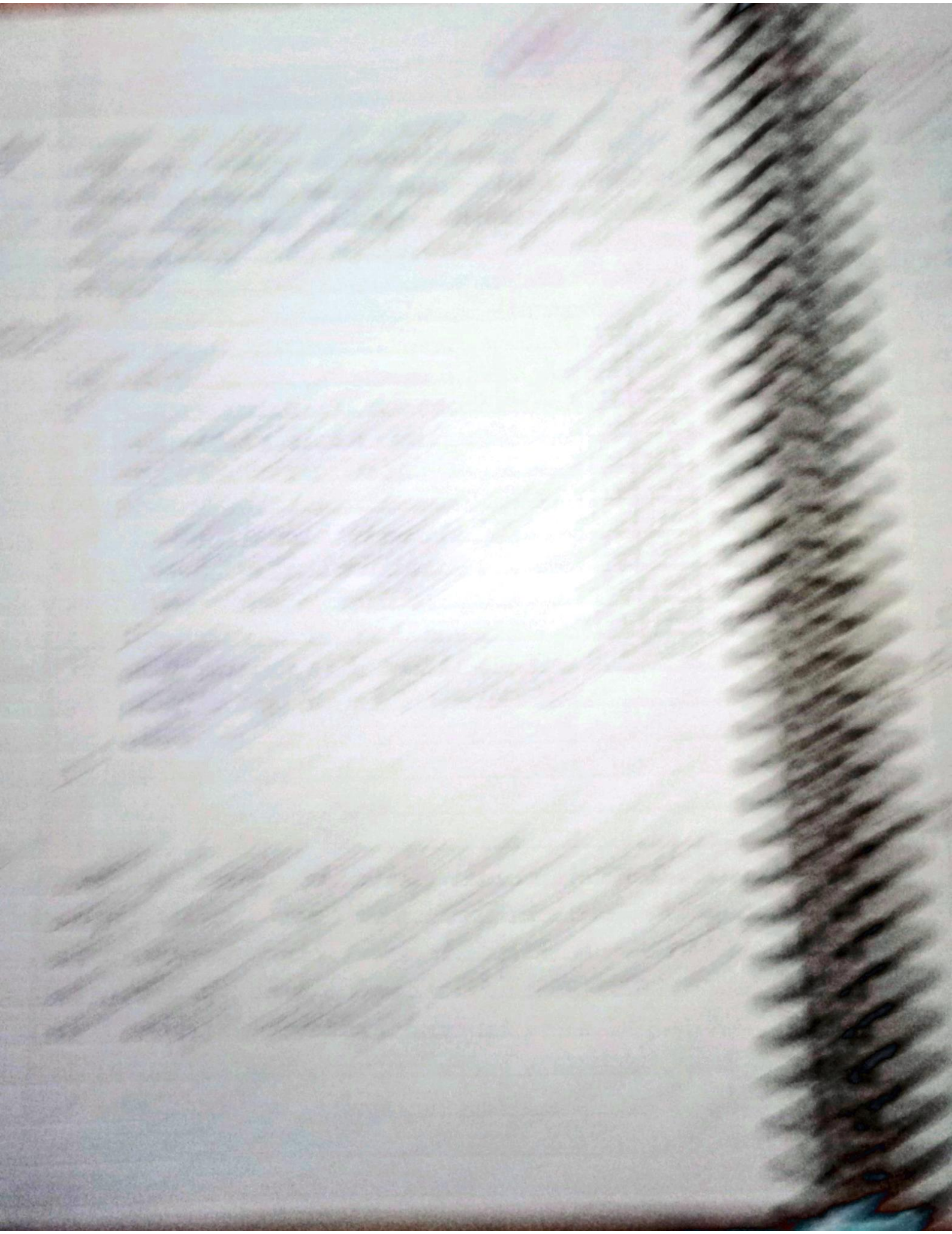
6.) int arr[5] = {10, 20, 30} ⇒ OK.

⇒ output will be → 10 20 30 0 0

7.) int arr[5]; = {10, 20, 30, 40, 50, 60}; // Syntax ERROR

8.) int arr[] = {10, 20, 30, 40, 50, 60, 70};

⇒ OK. compiler will automatically count the number of initializers and make it the size of the array.



Solution:-

```
int main()
{
    int arr[10], i, soe=0, sod=0;
    for (i=0; i<=9; i++)
    {
      printf("Enter Elements : ");
      scanf("%d", &arr[i]);
      if (arr[i]%2==0)
          soe = soe + arr[i];
      else
        sod = sod + arr[i];
    }
    printf("Sum of even is %d", soe);
    printf("\nSum of odd is %d", sod);
    return 0;
}
```

	arr	
0	12	2000
1	3	2004
2	15	2008
3	21	2012
4	18	2016
5	5	2020
6	22	2024
7	24	2028
8	11	2032
9	19	2036

Q→ Modify the previous code so that now your programs calculates the average of even and odd numbers also separately.

Solution2-

```
int main()
{
    int arr[10], i, soe=0, sod=0, count=0;
    for (i=0; i<=9; i++)
    {
      printf("Enter Element : ");
      scanf("%d", &arr[i]);
      if (arr[i]%2==0)
      {
          soe = soe + arr[i];
          count = count + 1;
      }
```

	arr	
0	15	2000
1	13	2004
2	11	2008
3	6	2012
4	19	2016
5	3	2020
6	18	2024
7	21	2028
8	23	2032
9	13	2036

else
Sod = Sod + arr[i];
}
}
printf("Sum of even is %d", Sae);
printf("\n Sum of odd is %d", Sod);
printf("\n Avg of even %f", (float) Sae/count);
printf("\n Avg of odd is %f", (float) Sod/(n-count));
}

(Q.3) Modify the previous code so that all the numbers of the array are even or all the numbers are odd, the program should handles the situation carefully.

(Q.4) write to create an array of 10 integers and accept values from the user in that array.

Now again ask the user to input another number and search it in the array. If the number is present then print its position, otherwise print the message number not found.

Assume that the array contains unique numbers only.

```
int main()
{
    int arr[10], i, n;
    for (i=0; i<=9; i++)
    {
        printf("Enter a number: ");
        scanf("%d", &arr[i]);
    }
printf("Enter number to search:");
scanf("%d", &n);
for (i=0; i<=9; i++)
{
    if(arr[i] == n)
    {
    printf("Number found at position %d", i+1);
    break;
    }
}
if (i == 10)
    printf("Number not found");
return 0;
}
```

Q→ WAP to create an array of 10 integers, ask the user to input values in the array and find out the maximum number in this array. Make sure your code should not change the original order of elements in the array.

Solution :-

```
int main()
{
  int arr[10], i, max = 0;
  for (i=0; i<=9; i++)
  {
    printf("Enter number : ");
    scanf("%d", &arr[i]);
  }
  for (i=0; i<=9; i++)
  {
    if(arr[i] > max)
      max = arr[i];
  }
  printf("Max Number is %d", max);
  return 0;
}
```

Q=> modify the previous program so that your code should handle negative numbers also.

Solution :-

```
int main()
{
  int arr[10], i, max;
  for (i=0; i<=9; i++)
  {
    printf("Enter number : ");
    scanf("%d", &arr[i]);
  }
  max = arr[0];
  for(i=1; i<=9; i++)
  {
```

```
        if(arr[i] > max)
            max = arr[i]
    }
    printf(" Max Number is %d ", max);
    return 0;
}
```

★ Shorting the Array :-

```
int main()
{
  int arr[5];
  int i, j, temp;
    for (i=0; i<=4; i++)
    {
      printf("Enter Element: ");
      scanf("%d", &arr[i]);
    }
    for (i=0; i<4; i++)
    {
      for( j=i+1; j<5; j++)
      {
          if(arr[i]>arr[j])
          {
              temp = arr[i];
              arr[i] = arr[j];
              arr[j] = temp;
          }
      }
    }
```

```
for (i=0; i<=4; i++)
    printf("\n %d", arr[i]);
return 0;
}
```

	0	1	2	3	4
arr	10	7	11	9	2

	0	1	2	3	4
arr	7	10	11	9	2

	0	1	2	3	4
arr	2	10	11	9	7

	0	1	2	3	4
arr	2	9	11	10	7

	0	1	2	3	4
arr	2	7	11	10	9

	0	1	2	3	4
arr	2	7	10	11	9

	0	1	2	3	4
arr	2	7	9	11	10

	0	1	2	3	4
arr	2	7	9	10	11

The Required Shorted Array

Q> WAP to create an array of 10 integers and find out the maximum element, the minimum element as well as their positions in the array.

Q) WAP to create an array of 5 integers, accept input from the user in the array and calculate the sum of the digits of every element of the array.

☆ Double Dimensional integer Array.

- whenever we want to store multiple groups of data in our program, then we take or use a double dimensional integer array.

Ex:-

(i) For storing marks of 3 students, where each student is having 5 subjects.

(ii) For storing sale data of 5 salesmen of 1 week.

Syntax of 2-D integer Array :-

<data_type> <array_name>[<row_size>][<column_size>]

Ex:-

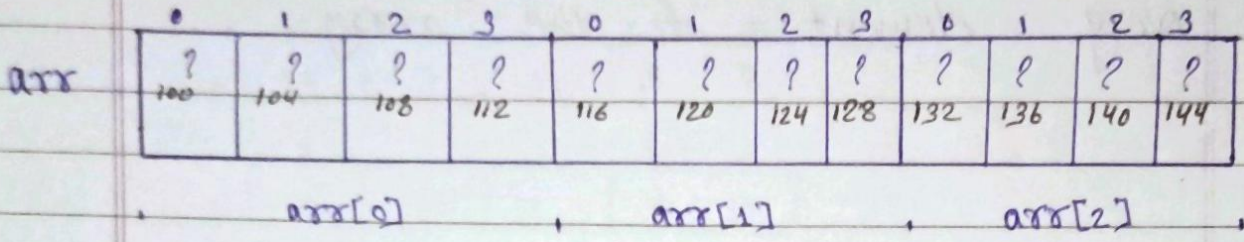
```
int arr[3][4];
```

row column

	0	1	2	3
0	100 ?	104 ?	108 ?	112 ?
1	116 ?	120 ?	124 ?	128 ?
2	132 ?	136 ?	140 ?	144 ?

Logical view

Physical View of 2D integer Array :-



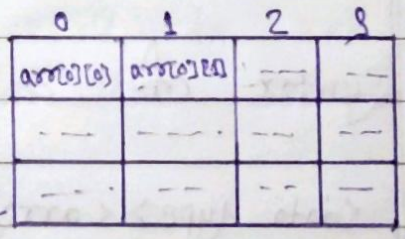
★ How do we access elements of a 2D array :-

Syntax of Accessing a 2D Array :-

<array_name>[<row_index>][<column_index>] = <value>;

Ex :-
```
int arr[3][4];
arr[0] = 25 ; ⇒ Error
```

Ex :-



arr[0][0] = 25 ⇒ OK
arr[0][1] = 35 ⇒ OK
!
!

arr[2][3] = 100 ⇒ OK

```
int main()
{
 int arr[3][4];
 int i,j;
 for( i=0; i<3; i++)
  {
      for(j=0; j<4; j++)
      {
       printf("Enter a number");
        scanf("%d", &arr[i][j]);
      }
 }
 for(i=0; i<3; i++)
 {
   for (j=0; j<4; j++)
   {
     printf("%d", arr[i][j]);
   }
   printf("\n");
 }
   return 0;
}
```

arr

Logical view

	0	1	2	3
0	15 / 100	25 / 104	8 / 108	12 / 112
1	18 / 116	35 / 120	40 / 124	25 / 128
2	29 / 132	31 / 136	48 / 140	28 / 144

Q) WAP to accept values from the user in a 2D array of size 3*4, and display the sum and average of all the numbers of the array.

Solution 2-

```
int main()
{
  int arr[3][4], i, j, sum=0;
  for (i=0; i<3; i++)
  {
    for(j=0; j<4; j++)
    {
      printf("Enter Element:");
      scanf("%d", &arr[i][j]);
      sum = sum + arr[i][j];
    }
  }
  printf("Sum is %d", sum);
  printf("\n Average is %f", (float) sum/12);
  return 0;
}
```

Q3 WAP to accept values from the user in 2, 2D arrays of 2*2 size each. Then add their corresponding using matrix addition logic and store the result in the 3rd 2D array finally print the 3rd array.

Solution 3-

```
int main()
{
  int arr[2][2], brr[2][2], crr[2][2], i, j;
  printf("Enter value for the 2D array: \n");
  for (i=0; i<2; i++)
  {
    for (j=0; j<2; j++)
    {
```

```
            printf("Enter Number ");
            scanf("%d", &arr[i][j]);
        }
}
printf("Enter value for the second 2D array: \n");
for (i=0; i<2; i++)
{
    for(j=0; j<2; j++)
    {
        printf("Enter number: ");
        scanf("%d", &brr[i][j]);
    }
}
for (i=0; i<2; i++)
{
    for( j=0; j<2; j++)
    {
        crr[i][j] = arr[i][j] + brr[i][j];
        printf("%d", crr[i][j]);
    }
    printf("\n");
}
return 0;
}
```

Q> WAP to accept marks of 3 students each having 4 subjects and do the following:

1. Find out the total marks obtained by each student.

Solution 2—

```
int main()
{
 int  arr[3][4] , i, j, sum = 0;
 for (i=0 ; i<3 ; i++)
  {
     for(j=0 ; j<4; j++)
     {
          printf("Enter Marks :");
          scanf("%d", &arr[i][j]);
     }
  }
for (i=0; i<3; i++)
 {
     sum = 0;
     for (j=0 ; j<4; j++)
     {
       Sum = sum + arr[i][j];
     }
  printf("\nstudent number = %d , Total  = %d , i+1,
        sum);
  }
 return 0;
}
```

2.) Find out the highest marks scored by every student :

Sample Output :—

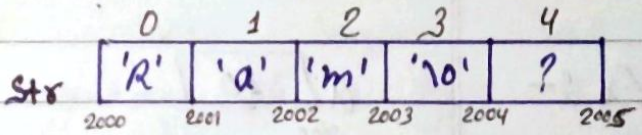
```
Student  No: 1          Highest : 90
Student  No: 2          Highest : 85
Student  No: 3          Highest : 81
```

3) Modify the previous code to display the subject number also in which the student has scored the highest marks. sample Output :—

```
Student No. 1    Highest : 90    Subject : 1
Student No. 2    Highest : 85    Subject : 4
Student No. 3    Highest : 81    Subject : 1
```

★ "Strings"

⇒ Single Dimensional character Array (Strings)




```
int main()
{
    char str[5];
    printf(" Enter your name ");
    scanf("%s", str);

    printf(" your name is %s ", str);
    return 0;
}
```

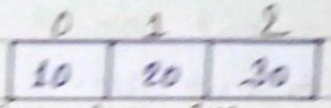
To scan multiple characters at once

Address of operator "&" not required

★ Important Point :-

In C/C++ an array name always represents its BASE ADDRESS

Guess the output :-



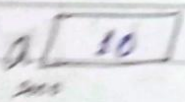
1) printf("%d", arr); ——— 2000

2) printf("%d", &arr[0]); ——— 2000

3) printf("%d", &arr); ——— 3000

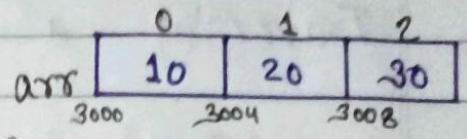
4) printf("%d", arr[0]); ——— 10

Ex :-



int a = 10;
printf("%d", a); ——— variable name always represents its value

Guess the Output :—



1) printf("%u ", arr); → 3000

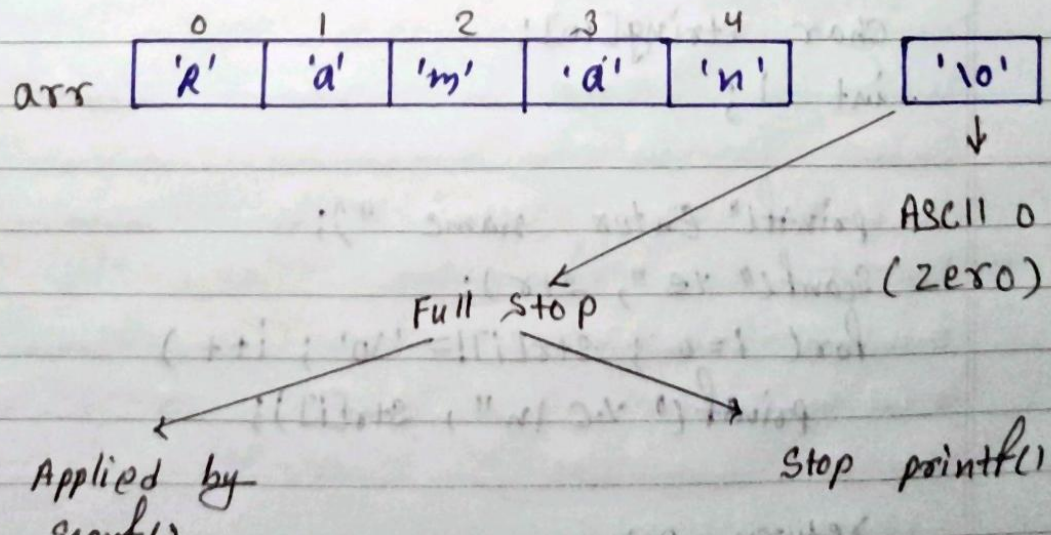
2) printf("%u ", &arr[0]); → 3000

3) printf("%u", &arr); → 3000

4) printf("%d", arr[0]); → 10

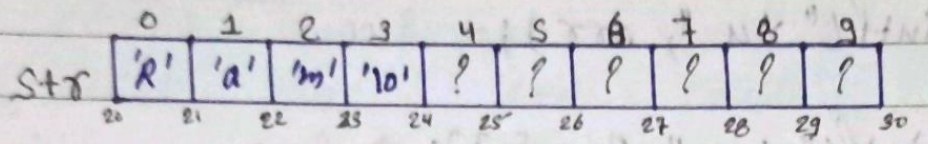
Note :— Always use %u for printing addresses unsigned int type and its range is from 0 to 2147483648 and it is possible that the address of array may be of range beyond the range of signed int type in Turbo Compiler i.e. -32768 to 32767.

=>



Date
Page

Traversing a character Array :—




```
int main()
{
    char str[5];
    printf(" Enter your name ");
    scanf(" %s", str);
    printf(" Your name is %s ", str);
    return 0;
}
```

Output :— Enter your name Ram
Ram—

⇒ Printing string vertically :—

1)
```
int main()
{
    char string[10];
    int i;

    printf(" Enter name ");
    scanf(" %s", str);
    for( i=0 ; str[i]!= '\0' ; i++ )
        printf(" %c \n", str[i]);

    return 0;
}
```

Output :- Enter name : Raman

R
a
m
a
n

improved way to print string vertically

2)
```
int main()
{
    Char str[10];
    int i;
    printf(" Enter name ");
    scanf(" %s ", str);
    for( i=0 ; i<3 ; i++)
        printf(" %c \n ", str[i]);
    return 0;
}
```

Output :- Enter name Ram

R
a
m

This code works fine when only 3 letter string stored in array

Note :- If we store "Raman" in Store then it will print only first 3 characters

3)

```
int main()
{
    char str[10];
    int i;

    printf("Enter name:");
    scanf("%s", str);
    for(i=0; i<9; i++)
        printf("%c\n", str[i]);
    return 0;
}
```

Output :- Enter name : Raman

```
R
a
m
a
n
?
?
?
```

This code will print whole string including unpredictable values

04) WAP to accept a string from the user, store it in a character array and calculate and print its length.

Solution

```
#include <stdio.h>
int main()
{
    char str[10];
    int i, x=0;
```

	0	1	2	3	4	5	6	7	8	9
str	'R'	'a'	'm'	'\0'	?	?	?	?	?	?
	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009

```
printf("Enter name ");
scanf("%s", str);
for (i=0; str[i]!='\0'; i++)
    x++;
printf("Length = %d", x);
return 0;
}
```

improved version of previous program :-

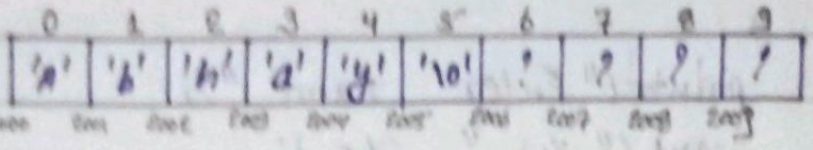
```
int main()
{
    char str[10];
    int i;
    printf("Enter name ");
    scanf("%s", str);
    for (i=0; str[i]!='\0'; i++);
    printf("Length = %d", i);
    return 0;
}
```

Q-> WAP to accept a string from user, store it in a character array and print its reverse.

Sample out put :-

```
Enter name : RAMA
AMAR
```

11 The drawback with the scanf()




```
Int main()
{
  char string[5];

  printf(" Enter your name ");
  scanf(" %s ", str );
  printf(" Hello %s ", str);
   return 0;
}
```

Output :- Enter your name Abhay Rai
Hello Abhay

Note :- The function scanf() in c language cannot accept spaces when we use %s . As soon as it finds a space in the input it stops reading the input data .

A solution to the problem is using a new function of c language called as gets()

Q) what is gets() ?

1) gets() is predefined library function .

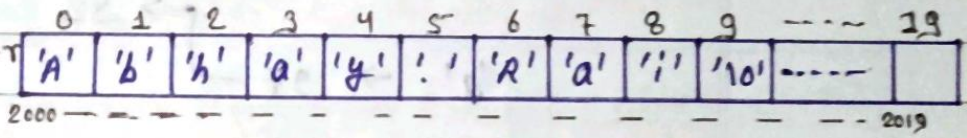
2) It is available in the header file stdio.h .

3) The good thing about is that :

a) It doesn't require any format specifier
b) It can read all the spaces, special characters as well as any string.

4) But remember it is only with strings and not for any other type like char, int, float.

Example of gets() :-




```
int main()
{
  char Str[20];

  printf(" Enter name ");
  gets(str);
  printf(" Hello %s ", str);
  return 0;
}
```

output :-

```
Enter name Abhay Rai
Hello Abhay Rai
```

Exercise :-

Q-> WAP to accept a multiword string from the user the user and print it by converting all upper case characters to lower case and vice versa

Sample output :-

Enter Name : Abhay Rai
Hello aBHAY rAI

Hint :- 'A' → 65 → 32
'a' → 97

★ "String Functions"

- String functions in C Language are those important functions that have been provided by C language to work with strings.

- These are predefined functions and perform very useful operations on strings so that we have to write their logic.

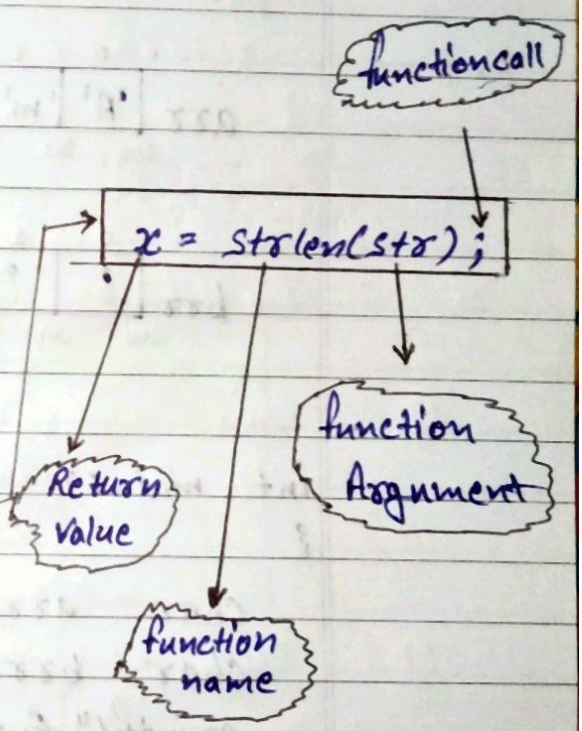
- All these string functions are available in the header file called as string.h .

- Following are the most common of them:-

1) Strlen() :- Calculates and returns the length of the string.

Example :-

```
#include<stdio.h>
#include<string.h>
int main()
{
    char str[20];
    int x;
    printf("Enter a string!");
    gets(str);
    x = strlen(str);
    printf("length : %d", x);
    return 0;
}
```



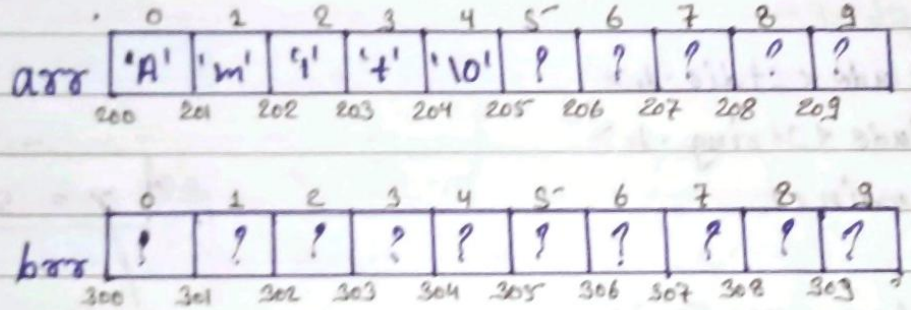
Improved version :-

We can remove the variable "x" and put the strlen() function in the printf() fun^n.

Q) Using strlen(), print the reverse of the string. The string has to be accepted by the user.

2) strcpy() :- Used to copy one string to another

Ex :-




```
int main()
{
    char arr[10];
    char brr[10];
    printf(" Enter a string : ");
    gets(arr);
    brr = arr; → Not Possible
    return 0;
}
```

300 = 200

↓

Non-sense

⇒ in C/C++ language we can never write array name on left Hand side of the assignment operator.

To copy strings we can use the string function →

strcpy() .

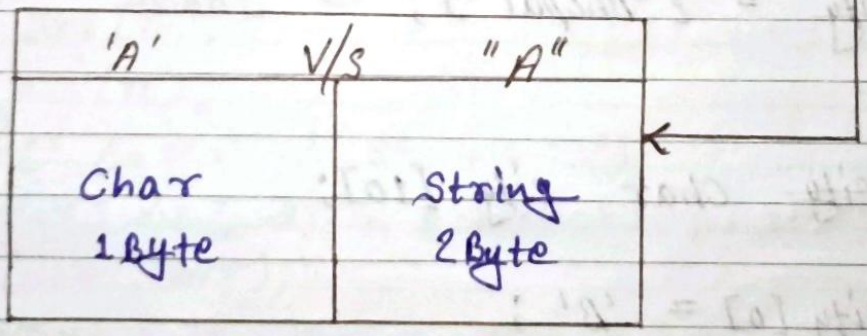
Syntax :- strcpy(<destination>, <source>);

The array in which we have to copy the data

An array whose data is to be copied

★ Initializing a character array :-

3) Char city[10] = {"Bhopal"};



4) Char city[] = "Bhopal";

5) Char city[10] = {'B', 'h', 'o', 'p', 'a', 'l', '\0'};

6) Char city[] = {'B', 'h', 'o', 'p', 'a', 'l', '\0'}

Ex. of strcpy() :—

```
int main()
{
  char arr[10];
  char brr[10];
  printf(" Enter a string ");
  gets (arr);
  strcpy ( brr , arr);
  printf(" in y.s ", brr);
  return 0 ;
}
```

★ Initializing a character array :—

1) `char city [10];`
`city = {"Bhopal"};` → ERROR

2) ~~city~~ `char city [10];`

```
city [0] = 'B' ;
city [1] = 'h' ;
city [2] = 'o' ;
city [3] = 'p' ;
city [4] = 'a' ;
city [5] = 'l' ;
city [6] = '\0' ;
```
→ compulsory otherwise garbage values will be printed.

Initialization of character array using strcpy() :-

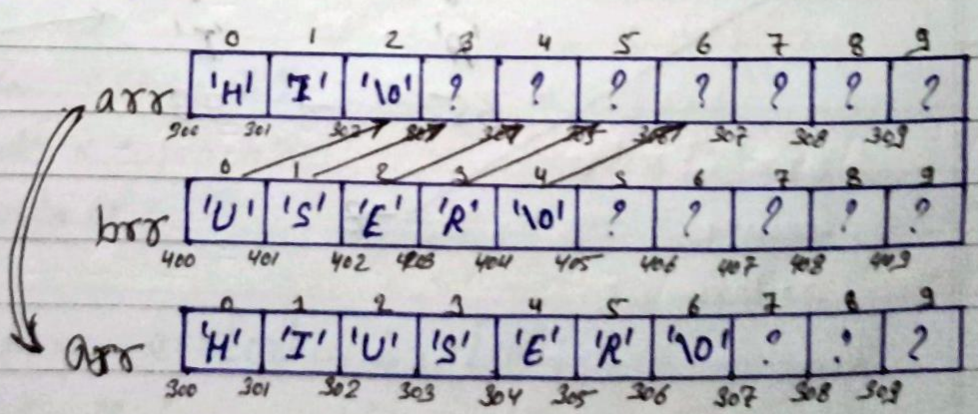
⇒
```
char city[10];
strcpy(city, "Bhopal");
```
↳ String constant

(3) strcat() :- String concatenation :-

This function joins a string at the end of another string.

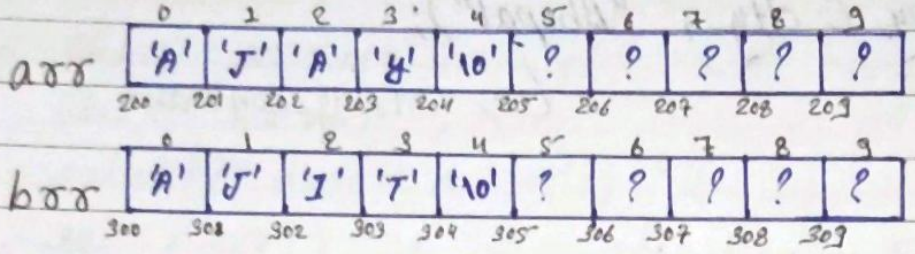
Syntax :- `strcat(<destination>, <source>);`

Ex :-
```
int main()
{
    char arr[10];
    char brr[10];
    printf("Enter 2 strings : ");
    scanf("%s %s", arr, brr);
    strcat(arr, brr);
    printf("%s", arr);  // HIUSER
    return 0;
}
```



(4) strcmp() :- string comparision.

Ex:-




```
int main ()
{
    char arr[10] , brr[10];
    printf (" Enter 2 string ");
    scanf(" %s %s ", arr , brr);
    if( arr == brr)  →  200==300 → 0(False)
       printf(" string are equal");
    else
       printf(" string are not equal ");
    return 0;
}
```

Syntax:- strcmp (<array 1> , <array 2>);

i) x = strcmp ("AJAY" , "AJIT");

'A' 'I'

65 → 73

mismatch

-8

ii) x = strcmp("AJAY", "ajay");

65 - 97 ← Mismatch

-32

iii) x = strcmp("RAM", "RAM");

'R' - 'R' ← Mismatch

82 - 82

0

iv) x = strcmp("Rama", "Ram");

'a' '\0' ← Mismatch

97 - 0

97

Ex of strcmp() :-

```
#include <stdio.h>
#include <conio.h>
int main()
{
    char arr[10], brr[10];
    int x;
```

```
    printf(" Enter 2 strings : ");
    scanf(" %s %s ", arr, brr );
    x = strcmp( arr, brr);

    if( x == 0)  /  if(strcmp(arr,brr)==0)
     printf(" strings are equal ");
    else
     printf (" strings are not equal ");
  return 0;
}
```

Q→ WAP to declare 2 character arrays arr & brr. Accept a string from the user in arr and copy it into brr. Do not use any string function.

Solution:-

```
int main ()
{
  char arr[10], brr[10];
  int i;
  printf(" Enter a string : ");
  gets( arr);
  for( i=0; arr[i]!='\0'; i++)
       brr[i] = arr[i];
   brr[i] = '\0';
   printf(" %s ", brr);
   return 0;
  }
```

"Exercise"

Date
Page

Qs 1 WAP ~~a program~~ to accept a multiword String from the user and print it by displaying the initial character of every word, and the full last word.

Qs 2 write a program to accept a string from the user and check wether it is palindrome or not.

Sample outputs :—

Qs 1 Enter a String : Ram Kumar Sharma
R. K. Sharma

Qs 2 Enter a String : NITIN.
It is a palindrome

Enter a String : JATIN
It is not a palindrome