Flowchart

Start / Stop

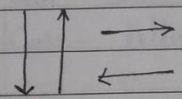
Input / Output

Processing

Checking / Decision making

Looping

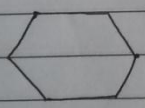
Connector



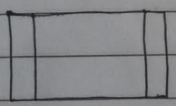
Arrows

Comments / Explanation

Additional Symbols Related to more advanced Programming-



Preparation (may be used with "do loops")



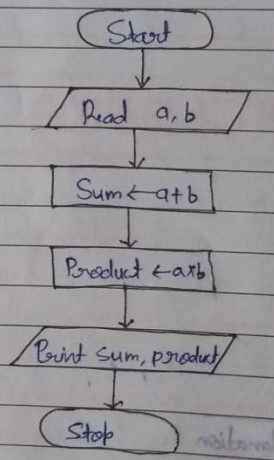
Refers to separate flowchart

Q- Write the algorithm and draw the flowchart to find the sum and product of given two numbers.

Solution- Algorithm-

1- Read a, b
2- Sum ← a+b
3- Product ← a×b
4- Print Sum, Product
5- Stop

Flowchart-



C Language

Difinition- The C Language is one of the powerful programming language used to write computer programs for variety of application. C has flexible features to write programs for numerical, commercial and graphical application.

History - It was developed by Dennis Ritchie at Bell Lab during the 1970's. This language was developed from B language which was modified from a language called Basic Combined Programming Language (BCPL).

Characters used in C -

1- Alphabets -

Upper Case letters A to Z
Lower Case letters a to z

2- Numbers -
0 to 9

3- Special Characters.

DATE
PAGE No. 4

Special Characters:-

+	Plus
-	Minus
*	Asterisk
/	Slash
\	Back slash
%	Percent
|	Vertical bar
~	Tilde
?	Question mark
!	Exclamation mark
,	Comma
.	Full stop
;	Semicolon
:	Colon
'	Apostrophe
"	Double Quote
&	Ampersand
#	Hash
$	Dollar
^	Caret
<	Lesser than
>	Greater than
=	Equal to
(	Open Parenthesis
)	Close Parenthesis
[	Open bracket
]	Close bracket
{	Open Set bracket
}	Close Set bracket
_	Underscore

DATE
PAGE No. 5

Identifier - An identifier is a name having a few letters, numbers and special characters _ (underscore). It is used to identify a variable, function, symbolic constant and so on.

Ex- X2 , PI , Sigma , matadd

Keywords or Reserved Words-

auto	break	case	enum
volatile	while	char	extern
switch	typedef	const	struct
short	union	continue	return
Signed	unsigned	register	goto
sizeof	void	for	if
static	float	double	int
default	do	else	long

Tokens-

C Tokens	Examples
Keywords	Auto , float , int etc.
Constants	-25, 3.24 etc.
Identifiers	Sum , PI etc.
String literals	"Total Amount is", "Sum = " etc.
Operators	+, -, * etc.
Separators	, ;

DATE
PAGE No. 6

Constants

1- Integer constant } Numeric Constant
2- Real or Floating point constant }
3- Character Constant } String or Character Constant
4- String Constant }

1- Integer Constant- An integer Constant is signed or unsigned whole number.
Ex:- 25, -5.6, +30 etc

2- Real or Floating point constant- An signed or unsigned number with fractional part is called real or floating point constant.
Ex- 0.25, -2.56, 0.23, 0.34e3 etc.
where e = exponential

3- Character constant- Any letter or character enclosed in single apostraphe is called single character, string constant or character constant.
Ex- 'y', '+', '5' etc.

4- String constant- Any string of characters consisting of letter, digits and symbols enclosed in double quotes is called, string of character constant or string constant.
Ex- "Total amount is", "Average =", "Jail No. - 420".

Variable - A variable is an identifier or a name which is used to refer a value and this value varies or changes during the program execution. A variable is written with a combination of letters, numbers and special characters _ (underscore) with the first letter being an alphabet. Maximum of 31 letters can be used to write a variable.
Ex- c, fact, b33, total_amount etc.

Basic Data Type -

Data type	Bytes Occupied in RAM
Char	1 byte
int	2 bytes
float	4 bytes
double	8 bytes

Additional Data types -

Data type	Bytes Occupied in RAM
unsigned char	1 byte
short int	1 byte
unsigned short int	1 byte
unsigned int	2 bytes
long int	4 bytes
unsigned long int	4 bytes
long double	10 bytes

True - 1
False - 0

Operators and Expressions -

1- Arithmetic Operators
2- Relational Operators
3- Logical Operators
4- Increment and Decrement Operators
5- Assignment Operators
6- Conditional Operators
7- Bitwise Operators

1- Arithmetic Operators -

Operators	Meaning	Exp Example	Result
+	Addition	4+2	6
−	Subtraction	4-2	2
*	Multiplication	4*2	8
/	Division	4/2	2
%	Modulus operator to get remainder in integer division	5%2	1

2- Relational Operators -

Operators	Meaning	Example	Result
<	Less than	5<2	False
>	Greater than	5>2	True
<=	Less than or equal to	5<=2	False
>=	Greater than or equal to	5>=2	True
==	Equal to	5==2	False
!=	Not equal to	5!=2	True

DATE
PAGE No. 9

3- Logical Operators-

Operators	Meaning	Example	Result
&&	Logical and	(5>6)&&(5>3)	False
||	Logical or	(5<2)||(5>3)	True
!	Logical not	!(5<2)	True

4- Increment and Decrement operators-

Operators	Meaning
++a	Increment a by 1, then use the new value of a
a++	Use value of a, then increment a by 1
--b	Decrement b by 1, then use the new value of b
b--	Use the current value of b, then decrement by 1

Ex- a = 15; b = 15;

++a	a++	b--	--b
a = 16	a = 15	b = 15	b = 14
a = 17	a = 16	b = 14	b = 13

5- Assignment Operators-

Operators =	Example	Equivalent Expression (m = 15)	Result
+=	m += 10	m = m + 10	25
-=	m -= 10	m = m - 10	5
*=	m *= 10	m = m * 10	150
/=	m /= 10	m = m / 10	1
%=	m %= 10	m = m % 10	5

6- Conditional / Ternary Operator-

Variable = (Condition) ? Value 1 : Value 2;

Ex- big = (a>b) ? a : b;
big = (5>2) ? 5 : 2;
big = 5

```
if (a>b)
  big= a;
else
  big = b;
```

7- Bitwise Operators-

Operators	Meaning
<<	Shifts the bits to left
>>	Shifts the bits to right
~	Bitwise inversion (one's complement)
&	Bitwise logical and
|	Bitwise logical or
^	Bitwise exclusive or

Additional operators-

1- Size of operator -

int mx[50]

2- Comma (,) operator -

Ex- temp = x;
x = y;
y = temp;

temp = x, x = y, temp;

Library Function (Intrinsic functions or Math Function)

Mathematical Notation	C function	Meaning
$\sqrt{a}$	sqrt(a)	Square root of a
$\|a\|$	fabs(a)	Absolute value of a
e^x	exp(x)	Exponential of x
$\log x$	log(x)	Logarithm of x
x^y	pow(x,y)	x raised to y
$\sin x$	Sin(x)	Sine of x (x in radians)
$\cos x$	Cos(x)	Cosine of x (x in radians)
$\tan x$	tan(x)	Tangent of x (x in radians)
	ceil(x)	Round off a float value to the nearest integer upward (eg. 5.34 will be rounded off to 6.0)
	floor(x)	Rounded off a float value to the nearest integer downwards (eg. 5.34 will be round off to 5.0)

Type conversion-

1- Mixed mode operation and Automatic (implicit) Conversion-

char → int → long int → float → double
(Low Ranking) (High ranking)

Ex- int m = 15;
float x = 3.1, y;
y = m * x;
↳ 9ts type float is high ranking
↳ 9ts type int is converted to float
y = 46.500000

25 Cast or Explicit Conversion –

(type) expression
or
(type) (expression)

Ex- int 5;
float y;
y = (float) m/2;
y = 2.5

int 5;
int y;
y = m/2;
y = 2

Structure of C –

< header file >
< Global declaration of variables >
main ()
{
< Local declaration of variables >
< statements >
}
< Sub program - function blocks >

Comment –

Single line – //
Multiple line – /* */

Simbolic constant –

define

- WAP to print Hellow World.

```
#include <stdio.h>
main()
{
  clrscr();
  printf("Hellow World");
  getch();
}
```

Output- Hello World

Formatted Input / Output function-

Scanf() function- scanf(.) function is used to read/input value of variable using the standard input device (keyboard). It has following form-

scanf("format string" & Variable1 &V2, ... &Vn);

Ex- i) scanf("%d %d", &a, &b); to read of int variable a and b

ii) scanf("%f, %d", &x, &n); to read float value of x and int value of n.

iii) scanf("%c", &sex); to read char value for variable sex

iv) scanf("%s", sname); to read string of char variable sname.

DATE
PAGE No. 14

Format Specifier

Data Type	Format	Meaning
int	%d	Represents a decimal integer value
	%hd	short int
	%ld	long int
	%u	Represent an unsigned integer value
	%o	Represent an unsigned octal value
	%x	Represent an unsigned hexadecimal value
float/double	%f	Represent a floating point value
	%lf	long double
	%e	Represent a floating point value in decimal or exponential form.
char	%c	Represent a single character value
	%s	Represent a string of value of character

printf() function - printf() function is used to print/display value of variable using the standard out put device (monitor). It has following form-

printf("format string", variable 1, v2, ——— vn);

Ex- printf("%f", s);
printf("\n sum = %.6.3f", s);
printf("\n %d factorial is %d", k, kfact);

Escape sequences - Escape sequences are control characters used to move the cursor and print characters such as ?, ", \ and so on.

Character Constant	Meaning
\a	Audible bell (Alert)
\b	Backspace
\f	Form feed
\n	Move to new line
\r	Carriage return (Enter)
\t	Horizontal tab
\v	Vertical tab
\\	Print back slash
\?	Print question mark
\'	Print single quote
\"	Print double quote
\0	Null character

Assignment Statement - An assignment statment is used to assign value to a variable. It has following form:

Variable = value or expression.

Ex- m = 25;
y = a * a + 4 * a + 5;
S = S + a;

Multiple assignment statement - A multiple assignment is used to single a value to more than one variable. It has following form:

Variable 1 = Variable 2 = ... = Variable n = Value or expression

Ex- m = n = 3;
nb = nm = nz = 0;
x = y = (a * a + b * b) / 2;

More about formatted output function - Output values can be printed with specific width using commands for formatted output.

Ex- (i) int a=28;

(a) printf("%5d", a);

It will print a value right justified with three preceding spaces.

				2	8

(b) printf("%-5d", a);

It will print a value left justified.

2	8			

(c) printf("%+5d", a);

It will print a value right justified with + sign.

		+	2	8

(ii) float x = 28.358416;

(a) printf("%6.2f", x);

It will print x value right justified with a space preceding it.

	2	8	.	3	6

(b) printf("%0.2f", x);

It will print x value left justified without any space preceding it.

2	8	.	3	6

- It should be noted that even if the total width may be assigned wrong (eg. zero), the computer prints the result with the required number of decimal digits.

(c) printf("%0.3e", x);
It will print x value left justified in exponential form.

2	.	8	4	e	+	0	1

(iii) Char st[10] = "NEW YORK";
(a) printf("%s", st);
It will print st value left justified

N	E	W	Y	O	R	K

(b) printf("%10s", st);
It will print st value right justified

			N	E	W	Y	O	R	K

(c) printf("%10.3s", st);
It will print st value right justified and only the first 3 letters are printed.

							N	E	W

(d) printf("%-10.3s", st);
It will print st value left justified and only the 3 letters are printed.

N	E	W							

(e) printf("%3s", st);
It will print only the first 3 letters

N	E	W

Operators -

1- Arithmetic Operators -

(i) + (addition) -

```
/* Arithmetic operator Addition */
# include <stdio.h>
# include <conio.h>
main()
{
  clrscr();
  int a, b, sum;
  scanf("%d%d", &a, &b);
  sum = a + b;
  printf("%d", sum);
  getch();
}
```

or float a, b, sum;
or scanf("%f %f", &a, &b);
or printf("%f", sum); or printf("%0.2f", sum);

Output -

5 3	3.2 3.1	3.12 3.21
8	6.300000	6.33

```
# include <stdio.h>
# include <conio.h>
main()
{
  clrscr();
  int a, b, sum;
  printf("\n Enter the value to A: ");
  scanf("%d", &a);
  printf("\n Enter the value to B: ");
  scanf("%d", &b);
  sum = a + b;
  printf("\n Sum = %d", sum);
  getch();
}
```

Output - Enter value to A : 3
Enter value to B : 4
Sum = 7

DATE
PAGE No. 19

2(ii) - (Subtraction) -

```
/* Arithmetic operator Subtraction */
# include < stdio.h >
# include < conio.h >
main ()
{
   clrscr ();
   int a, b, sub;
   scanf ("%d %d", &a, &b);
   sub = a - b;
   printf ("%d", sub);
   getch ();
}
```

Output -

8
3
5

(iii) * (Multiplication) -

```
/* Arithmetic operator multiplication */
# include < stdio.h >
# include < conio.h >
main ()
{
   clrscr ();
   int a, b, product;
   scanf ("%d %d", &a, &b);
   product = a * b;
   printf ("%d", product);
   getch ();
}
```

output -

3
2
6

DATE
PAGE No. 20

Task-1

1- WAP to find the subtraction of given two decimal numbers. (User friendly)

```
# include <stdio.h>
# include <conio.h>
void main()
{
  clrscr();
  int a, b, sub;
  printf("Enter the value to A:");
  scanf("%d", &a);
  printf"Enter the value to B:");
  scanf("%d", &b);
  sub = a-b;
  printf("A-B = %d", sub);
  getch();
}
```

Out put -

(i) Enter the value to A: 3
Enter the value to B: 2
A - B = 1

(iii) Enter the value to A: 5
Enter the value to B: 7
A - B = -2

DATE
PAGE No. 21

2- WAP to find the subtraction of given two decimal numbers.

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
int a, b, sub;
scanf("%d", &a);
scanf("%d", &b);
sub=a-b;
printf("%d", sub);
getch();
}
```

Output- 4
4
0

3- WAP to find the multiplication of given two decimal numbers.

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
int a, b, multi;
scanf("%d", &a);
scanf("%d", &b);
multi= a* b;
printf("%d", multi);
getch();
}
```

Output- 3
4
12

4- Write a user friendly program to find to Multiplication of given two decimal numbers. (After decimal, digits should be only 3)

```
#include<stdio.h>
#include<conio.h>
void main()
{
 clrscr();
 float a, b, product;
 printf("Enter the value to A:");
 scanf("%f",& a);
 printf("Enter the value to B:");
 scanf("%f",& b);
 product= a*b;
 printf("%0.3f", product);
 getch();
}
```

Output-

Enter the value to A: 4.7543
Enter the value to B: 5.1476
24.473.

(iv) / (Division) –

```
#include <stdio.h>
#include <conio.h>
void main()
{
 clrscr();
 int a, b, division;
 scanf("%d %d", &a, &b);
 division = a/b;
 printf("%d", division);
 getch();
}
```

Output – 10 2 5

5 2 2

10 3 3

जब हम int का प्रयोग करें किन्तु Ans float हो तो Ans wrong आयेगा।

(v) % (Modulus) –

```
#include <stdio.h>
#include <conio.h>
void main()
{
 clrscr();
 int a, b, mod;
 scanf("%d %d", &a, &b);
 mod = a % b;
 printf("%d", mod);
 getch();
}
```

Output – 10 3 1

DATE
PAGE No. 24

Task-2

1- WAP user friendly to find the Division of given two numbers.

```
#include <stdio.h>
#include <conio.h>
void main()
{
 clrscr();
 float a, b, division;
 printf("Enter the value to A:");
 scanf("%f", &a);
 printf("Enter the value to B:");
 scanf("%f", &b);
 division = a/b;
 printf("A/B= %f", division);
 getch();
}
```

Output -

```
Enter the value to A: 2.66
Enter the value to B: 3.99
A/B = 0.666667
```

2- Write
given

```
#inc
#inc
voi
{
clr
int
pr
sc
p
2
m
g
}
```

Out

En
En
Mo

DATE
PAGE No. 25

2- Write a user friendly program to find the Modulus of given two numbers.

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
int a, b, mod;
printf(" Enter the value to A:");
scanf("%d", &a);
printf(" Enter the value to B:");
scanf("%d", &b);
mod = A % B;
printf(" Modulus = %d", mod);
getch();
}
```

Output-

```
Enter the value to A: 11
Enter the value to B: 3
Modulus = 2
```

Math.h –

1- sqrt(a) –

```
/* Library Function */
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
 clrscr();
 int a, square;
 scanf("%d", &a);
 square = sqrt(a);
 printf("%d", square);
 getch();
}
```

Output –

(i) 36
6

(ii) 100
10

(iii) 49
7

(iv) 169
13

2- fabs(a) -

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
    clrscr();
    int a, b;
    printf("Enter a number:");
    scanf("%d", &a);
    b = fabs(a);
    printf("%d", b);
    getch();
}
```

Output -

(i) Enter a number: -45
45

(ii) Enter a number: 33
33

2- Relation Operators -

(i) < (Less than) -

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
 clrscr();
 int a, b, result;
 printf("Note: 1= true and 0=false \n");
 scanf("%d %d", &a, &b);
 result = a<b;
 printf("The result is %d", result);
 getch();
}
```

Output - Note: 1=true and 0=false

(i) 4
2
The result is 0

Note: 1 = true and 0 = false

(ii) 1
4
The result is 1

(ii) > (Greater than) ←

```
#include<stdio.h>
#include<conio.h>
void main()
{
 clrscr();
 int a, b, result;
 printf("Note: 1 = true and 0 = false");
 scanf("%d %d", &a, &b);
 result = a > b;
 printf("The result is %d", result);
 getch();
}
```

Output-

(i) Note: 1 = true and 0 = false
5
2
The result is 1

(ii) Note: 1 = true and 0 = false
2
5
The result is 0

(iii) <= (Less than or equal to) -->

```
#include <stdio.h>
#include <conio.h>
void main()
{
  clrscr();
  int a, b, result;
  printf("Note: 1= true and 0= false\n");
  scanf("%d %d", &a, &b);
  result= a<=b;
  printf("The result is %d", result);
  getch();
}
```

Output -

(i) Note: 1 = true and 0 = false
2
5
The result is 1

(ii) Note: 1 = true and 0 = false
5
2
The result is 0

(iii) Note: 1 = true and 0 = false
2
2
The result is 1

(iv) >

(iv) >= (Greater than or equal to)

```
#include<stdio.h>
#include<conio.h>
void main()
{
  clrscr();
  int a, b, result;
  printf("Note: 1= true and 0= false\n");
  scanf("%d %d", &a, &b);
  result = a >= b;
  printf("The result is %d", result);
  getch();
}
```

Output -

(ii) Note: 1= true and 0= false
5
2
The result is 1

(iii) Note: 1= true and 0= false
2
5
The result is 0

(iii) Note: 1= true and 0= false
2
2
The result is 1

(v) == (Equal to) :-

```
#include <stdio.h>
#include <conio.h>
void main()
{
  clrscr();
  int a, b, result;
  printf("Note: 1 = true and 0 = false\n");
  scanf("%d %d", &a, &b);
  result = a==b;
  printf("The result is %d", result);
  getch();
}
```

Output -

(i) Note: 1 = true and 0 = false
3
3
The result is 1

(ii) Note: 1 = true and 0 = false
3
4
The result is 0

(vi) != (Not equal to) -

```
#include<stdio.h>
#include<conio.h>
void main()
{
  clrscr();
  int a, b, result;
  printf("%d %d", &a, &b);
  result = a != b;
  printf("The result is %d", result);
  getch();
}
```

Output -

Note: 1 = true and 0 = false

4
3

The result is 1

Note: 1 = true and 0 = false

4
4

The result is 0

3- Logical Operators-

(i) && (Logical and)-

Operand 1	Operand 2	Result
True	True	True
True	False	False
False	True	False
False	False	False

```
#include<stdio.h>
#include<conio.h>
void main()
{
 clrscr();
 int a, b, c, d, e, f, result;
 printf("Note: 1 = true and 0 = false \n");
 scanf"%d %d %d %d", &a, &b, &c, &d);
 e = a<b;
 f = c>d;
 result = e&&f;
 printf"The result is %d", result);
 getch();
}
```

Output-

```
Note: 1 = true and 0 = false
5
2
5
3
The result is 0
```

(ii) || (Logical or) -

Operand 1	Operand 2	Result
True	True	True
True	False	True
False	True	True
False	False	False

```
#include <stdio.h>
#include <conio.h>
void main()
{
    clrscr();
    int a, b, c, d, operand1, operand2, result;
    printf("Note : 1 = true and 0 = false\n");
    scanf("%d %d %d %d", &a, &b, &c, &d);
    operand1 = a<b;
    operand2 = c>d;
    result = operand1 || operand2;
    printf("The result is %d", result);
    getch();
}
```

Output -

(ii) Note : 1 = true and 0 = false

2 5 5 3

The result is 1

(iii) Note : 1 = true and 0 = false

4 2 3 5

The result is 0

(iii) ![Logical not) -

Operand	Result
False	True
True	False

```
#include<stdio.h>
#include<conio.h>
void main()
{
  clrscr();
  int a, b, result;
  printf("Note: 1 = true and 0 = false \n");
  scanf("%d %d", &a, &b);
  result=!(a>b);
  printf("The result is %d", result);
  getch();
}
```

Output -

(i) Note: 1 = true and 0 = false
5
2
The result is 0

(ii) Note: 1 = true and 0 = false
2
5
The result is 1

4- Increm

(i)

```
++a
#in
#inc
void
{
cl
i
}
```

Outp

Th
Th
Th
Th
Th
The

4- Increment and Decrement operators-

(i) ++a (ii) a++

// Difference between a++ and ++a.

```
#include <stdio.h>
#include <conio.h>
void main()
{
  clrscr();
  int a=3;
  printf("The value of a1 = %d\n", a);
  ++a;
  printf("The value of a2 = %d\n", a);
  printf("The value of a3 = %d\n", ++a);
  printf("The value of a4 = %d\n", a);
  printf("The value of a5 = %d\n", a++);
  printf("The value of a6 = %d\n", a);
  getch();
}
```

Output -

The value of a1 = 3
The value of a2 = 4
The value of a3 = 5
The value of a4 = 5
The value of a5 = 5
The value of a6 = 6

(i) ++a

```
#include <stdio.h>
#include <conio.h>
void main()
{
clrscr();
int a=3, b=2, total;
total = ++a * b;
printf("The value of a= %d \n", total);
getch();
}
```

Output-

The value of a = 8

(ii) a++

```
#include <stdio.h>
#include<conio.h>
void main()
{
clrscr();
int a=3, b=2, total;
total = a++ * b;
printf("The value of a= %d \n", total);
getch();
}
```

Output-

The value of a = 6

(iii) --a and (iv) a--

```
#include <stdio.h>
#include <conio.h>
void main()
{ // Difference between --a and a--
  clrscr();
  int a=3;
  printf("The value of a1 = %d\n", a);
  printf("The value of a2 = %d\n", --a);
  printf("The value of a3 = %d\n", a);
  printf("The value of a4 = %d\n", a--);
  printf("The value of a5=%d\n", a);
  getch();
}
```

Output -

The value of a1 = 3
The value of a2 = 2
The value of a3 = 2
The value of a4 = 2
The value of a5 = 1

(iii) --a

```
#include<stdio.h>
#include<conio.h>
~~clrscr();~~ void main()
{
  clrscr();
  int a=3, b=2, c;
  c= --a*b;
  printf("The value of c=%d\n", c);
  getch();
}
```

Output -

The value of c = 4

(iv) a--

```
#include<stdio.h>
#include<conio.h>
void main()
{
  clrscr();
  int a=3, b=2, c;
  c= a--*b;
  printf("The value of c = %d\n",c);
  getch();
}
```

Output -

The value of c = 6

5-
(i)

(ii)

5- Assignment Operators -

(i) +=

```
#include <stdio.h>
#include <conio.h>
void main()
{
clrscr();
int m=15;
m+=10;                    or m=m+10;
printf("Value of m=%d\n",m);
getch();
}
```

Output -

Value of m = 25

(ii) -=

```
#include <stdio.h>
#include <conio.h>
void main()
{
clrscr();
int m=15
m-=10;                    or m=m-10;
printf("Value of m=%d\n",m);
getch();
}
```

Output -

Value of m = 5

(iii) *=

```
#include <stdio.h>
#include <conio.h>
void main()
{
clrscr();
 int m=15;
 m*=10;                    or   m=m*10;
 printf("Value of m=%d", m);
 getch();
}
```

Output-

Value of m = 150

(iv) /=

```
#include <stdio.h>
#include <conio.h>
void main()
{
 clrscr();
 int m=10;
 m /=2;                    or   m=m/2;
 printf("Value of m=%d", m);
 getch();
}
```

Output-

Value of m = 5

(v) %=

```
#include <stdio.h>
#include <conio.h>
void main()
{
  clrscr();
  int m=15;
  m %= 10;
  printf("Value of m = %d", m);
  getch();
}
```

Output -

Value of m = 5

6- Conditional / Ternary Operator -

Syntax -

```
Variable = (Conditional) ? Value1 : Value2;
```

++

```
#include <stdio.h>
#include <conio.h>
void main()
{
  clrscr();
  int x, y;
  printf("Value of y = ", y);
  scanf("%d", &y);
  x = y<20 ? 9 : 10
  printf("Result : %d", x);
  getch();
}
```

Output -

(i) Value of y = 12
9

(ii) Value of y = 21
10

```
#include <stdio.h>
#include <conio.h>
void main()
{
clrscr();
int marks;
printf("Marks: ", marks);
scanf("%d", marks);
printf("%s\n", mark>=50?"Passed":"Failed");
getch();
}
```

Output- (i) Marks = 27
Fail
(ii) Marks = 50
Passed

```
#include <stdio.h>
#include <conio.h>
void main()
{
clrscr();
int a, b;
printf("Value of a:");
scanf("%d", &a);
printf("Value of b:");
scanf("%d", &b);
(a>b)? printf("a is greater than b\n"): printf("b is greater than a");
getch();
}
```

Output- (i) Value of a: 8
Value of b: 2
a is greater than b

(iii) Value of a: 2
Value of b: 8
b is greater than a

7. Bitwise operators -

(i) << (Left shifts) -

```
#include <stdio.h>
#include <conio.h>
void main()
{
clrscr();
int a=60;                    // 00111100
printf("Result = %d", a<<2);  // 11110000
getch();
}
```

Output - Result = 240

(ii) >> (Right shifts) -

```
#include <stdio.h>
#include <conio.h>
void main()
{
clrscr();
int a=60;                    //00111100
printf("Result = %d", a>>2);  // 00001111
getch();
}
```

Output -

Result = 15

(iii) ~ (Bitwise inversion or Bitwise not or one's complement)

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
int a=5;                    //00000101
printf("Result = %d", ~a);  //11111010
getch();
}
```

Output-

Result = -6

Note- (one's complement of number में 1 add करने पर Negative value आती है।) (Maximum

(iv) & (Bitwise logical AND) -

Operand 1	Operand 2	Result (Operand 1 & Operand 2)
True 1	True 1	True 1
True 1	False 0	False 0
False 0	True 1	False 0
False 0	False 0	False 0

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
int a = 60;          // Binary of 60 is 00111100
int b = 17;          // Binary of 17 is 00010001
printf("Result of bitwise AND operation : %d", a&b); //00010000
getch();
}
```

Output - Result of bitwise AND operation : 16

DATE
PAGE No. 55

(v) | (Bitwise logical OR) -

Operand 1	Operand 2	Result (operand 1 | operand 2)
True 1	True 1	True 1
True 1	False 0	True 1
False 0	True 1	True 1
False 0	False 0	False 0

```
# include < stdio.h>
#include < conio.h>
void main ()
{
clrscr();
int a=60;          // Bainary of 60 ie 00111100
int b= 17;         // Bainary of 17 is 00010001
printf("Result = %d", a|b);   // 00111101
getch();
}
```

Outpt -

Result = 61

(vi) (Bitwise XOR)

Operand 1	Operand 2	Result (Operand 1 ^ Operand 2)
True 1	True 1	False 0
True 1	False 0	True 1
False 0	True 1	True 1
False 0	False 0	False 0

Note- Same false (0) and different True (1)

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
int a=60;          // Binary of 60 is 00111100
int b=17;          // Binary of 17 is 00010001
printf(" Result = %d", a^b);   // 01101101
getch();
}
```

Output-

Result = 45

Additional operators -

1- Size of operator -

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
int a;
short b;
printf("%d\n\n", sizeof(a));
printf("%d\n\n", sizeof(b));
printf("%d\n\n", sizeof(float));
getch();
}
```

Output -

4
2
4

2- Comma operator (,) —

```
#include<stdio.h>
#include<conio.h>
void main()
{ clrscr();
int a, b;
a= (b=2, b-1);
printf("%d\n", a);
getch();
}
```

Output - 1

(Work Left to right)

- Without use Comma operator

```
#include <stdio.h>
#include <conio.h>
void main()
{
clrscr();
int a;
int b;
int c;
int d;
a=3;
b=4;
c=2;
d=a+b+c;
printf("%d", d);
getch();
}
```

Output:- 9

Use comma operator

```
#include <stdio.h>
#include <conio.h>
void main()
{
clrscr();
int a, b, c, d;
a=3, b=4, c=2;
printf("%d", d); d=a+b+c;
printf("%d", d);
getch();
}
```

Output:- 9

(same out but ...

Type conver...

1- Mixed m...

Char →
(Low Ranking)

```
#include<
#include<c
void mai
{ clrscr
int m=
float
y = m
printf
getch();
}
```

Output-

2- Cast or

Syntax- (type) ex

```
without
#include<
#include<
void mai
{ clrscr
int m=
float y
y = m/n
printf("
getch();
}
```

Output-

DATE
PAGE No. 59

Type conversion-

1- Mixed mode operation and Automatic (implicit) Conversion

Char → int → long int → float → double
(Low Ranking) (High Ranking)

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
int m=15;
float y, x=3.1;
y=m*x;
printf("%f\n",y);
getch();
}
```

Output- 46.500000

or

```
int y;
float x=3.1;
```

Output- 46

2- Cast or Explicit Conversion-

Syntax- (type) expression

without cast conversion

```
#include<stdio.h>
#include<conio.h>
void main()
{ clrscr();
  int m=5, n=2;
  float y;
  y=m/n;
  printf("%f\n",y);
  getch();
}
```

Output- 2.000000
(Waring)

or (type) (expression)

with cast conversion

```
#include<stdio.h>
#include<conio.h>
void main()
{ clrscr();
  int m=5, n=2;
  float y;
  y=m/(float)n;
  printf("%f\n",y);
  getch();
}
```

Output- 2.500000
(Right)

DATE
PAGE No. 60

Precedence Order -

Precedence Order	Operator	Associativity
1	() [] →	Left to Right
2	++ -- -(unary) ! ~ * & sizeof	Right to Left
3	* / %	Left to Right
4	+ -	Left to Right
5	<< >>	Left to Right
6	< <= > >=	Left to Right
7	== !=	Left to Right
8	& (bitwise AND)	Left to Right
9	^ (bitwise XOR)	Left to Right
10	| (bitwise OR)	Left to Right
11	&& (logical AND)	Left to Right
12	|| (logical OR)	Left to Right
13	?: (conditional)	Right to Left
14	= += -= *= /= %= (assignment operators)	Right to Left
15	, (comma operator)	Left to Right

Simbolic Constants -

```
#include <stdio.h>
#include <conio.h>
#define two 3
int main ()
{
 clrscr ();
 int a=1, b=4, c, d;
  c = two + a;  //3 + 1
  d = two + b;  //3 + 4
  printf("%d\n", c);
  printf("%d\n", d);
 return 0;
 getch ();
}
```

Output - 4
7

Type	Storage size	Value range
char	1 byte	-128 to 127 or 0 to 255
unsigned char	1 byte	0 to 255
signed char	1 byte	-128 to 127
int	2 or 4 bytes	-32768 to 32767 or -2147483648 to 2147483647
unsigned int	2 or 4 bytes	0 to 65535 or 0 to 4294967295
short or short int	2 bytes	-32768 to 32767
unsigned short	2 bytes	0 to 65535
long	4 bytes	-2147483648 to 2147483647
unsigned long	4 bytes	0 to 4294967295

→
```
#include <stdio.h>
#include <conio.h>
void main()
{ clrscr();
printf("Memory of each Data type \n");
printf("Storage size for int: %d byte\n", sizeof(int));
printf("Storage size for unsigned int: %d byte\n", sizeof(unsigned
printf("Storage size for short int: %d byte\n", sizeof(short int));
printf("Storage size for unsigned short int: %d byte\n", sizeof(
printf("Storage size for long int: %d byte\n", sizeof(long
printf("Storage size for unsigned long int: %d byte\n", sizeof(unsigned
printf("Storage size for char: %d byte\n", sizeof(char));
printf("Storage size for unsigned char: %d byte\n", sizeof(
printf("Storage size for signed char: %d byte\n", sizeof(signed
getch();
}
```

Output –

```
Memory of each Data type
Storage size for int: 4 byte
Storage size for unsigned int: 4 byte
Storage size for short int: 2 byte
Storage size for unsigned short int: 2 byte
Storage size for long int: 4 byte
Storage size for unsigned long int: 4 byte
Storage size for char: 1 byte
Storage size for unsigned char: 1 byte
Storage size for signed char: 1 byte
```

Type	Storage Size	Value range	Precision
float	4 byte	1.2e-38 to 3.4e+38	6 decimal
double	8 byte	2.3e-308 to 1.7e+308	15 decimal
long double	10 byte	3.4E-4932 to 1.1e+4932	19 decimal

```
#include<stdio.h>
#include<conio.h>
void main()
{ clrscr();
  printf("Memory of each Data Type \n\n");
  printf("Storage size for float : %d byte \n", sizeof(float));
  printf("Storage size for double : %d byte \n", sizeof(double));
  // comparison
   float floatn;
   double doublen;
   floatn = 8.0/3.0;
   doublen = 8.0/3.0;
  printf("float value : %f \n", floatn); or %0.10f
  printf("double value : %f \n", doublen); or %0.10f
  getch();
}
```

Output- Memory of each Data Type

Storage size of float : 4 byte
Storage size of double : 8 byte
floatn value : 2.666667 or 2.6666667461
double value : 2.666667 or 2.6666666667

Address of Operator -

```
#include<stdio.h>
#include<conio.h>
void main()
{ clrscr();
  int m=5;
printf("Result: %d \n", m);
printf" Address of variable in Memory %d \n", &m);
 getch();
}
```

or, `int m=20;`

Output- Result: 5
Address of variable in Memory 2686748

Output- Result: 20
Address of variable in Memory

(Note: Address same, value change).

printf() function-

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
 int K=5, Kfact=120;
 float g=2.8;
 double p=8.0;
printf("** Explanation of printf function ** \n");
printf(" print value of an integer %d \n", K);
printf(" print value of float %f \n", g);
printf(" print value of double %f \n", p/3);
printf(" %d factorial is %d \n", K, Kfact);
printf(" %f %d %d \n", g, K, Kfact);
getch();
}
```

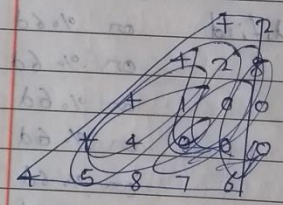
Output - ** Explanation of printf function **
print value of an integer 5
print value of float 2.800000
print value of double 2.666667
5 factorial is 120
2.800000 5 120

```
/* printf() function */
#include<stdio.h>
#include<conio.h>
void main()
{  // int
   clrscr();
   int a=2, b=28, c=100, d=4000, e=58763, f=670000;
   printf("%5d\n", a); // right justified      or %6d
   printf("%5d\n", b);                        or %6d
   printf("%5d\n", c);                        or %6d
   printf("%5d\n", d);                        or %6d
   printf("%5d\n", e);                        or %6d
   printf("%5d\n", f);                        or %6d
   printf("%-5d\n", a); // left justified
   printf("%-5d\n", b);
   printf("%-5d\n", c);
   printf("%-5d\n", e);
   printf("%-5d\n", f);
   printf("%+5d\n", a); // right justified with + sign   or %+6d
   printf("%+5d\n", b);                                  or %+6d
   printf("%+5d\n", c);                                  or %+6d
   printf("%+5d\n", d);                                  or %+6d
   printf("%+5d\n", e);                                  or %+6d
   printf("%+5d\n", f);                                  or %+6d
   getch();
}
```

Output-

```
      2
     28
    100
   4000
  58763
 670000
```

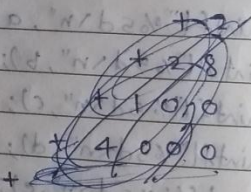
```
2
28
100
4000
58763
670000
```



```
  +2
 +28
+100
+4000
+58763
```

```
      2
     28
    100
   4000
  58763
 670000
```

```
2
28
100
4000
58763
670000
```



```
  +2
 +28
+100
+4000
+58763
```

Note- '+' one block
'.' one block
'e' one block

```
/* printf() function */
#include <stdio.h>
#include <conio.h>
void main()
{ // float
  clrscr();
  float a=8.358416, b=35.434565, c=134.456576, d=2654.453456;
printf("%0.2f\n", a); // left justified without any space providing or %0.3f or %0.3f
printf("%0.2f\n", b);
printf("%0.2f\n", c);                                    or %0.3f
printf("%0.2f\n", d);                                    or %0.3f
printf("%6.2f\n", a); // right justified                 or %7.2f
printf("%6.2f\n", b);                                    or %7.2f
printf("%6.2f\n", c);                                    or %7.2f
printf("%6.2f\n", d);                                    or %7.2f
printf("%0.3e\n", a); // left justified in exponential form
  getch();
}
```

Output -

```
8.36                    8.358
35.43                   35.435
134.46                  134.457
2654.45                 2654.453
  8.36                    8.36
 35.43                   35.43
134.46                  134.46
2654.45                 2654.45
8.358e+000              8.358e+000
```

→ /* printf() function */

```
#include <stdio.h>
#include <conio.h>
void main()
{ // char
  clrscr();
  char a[7] = "INDIA";
  char b[10] = "AMERICA";
  char c[10] = "PAKISTAN";
  char d[5] = "UAE";
 // left justified
 printf("%s\n", a);
 printf("%s\n", b);
 printf("%s\n", c);
 printf("%s\n\n", d);
 // right justified
 printf("%10s\n", a);      or %6s
 printf("%10s\n", b);      or %6s
 printf("%6s\n", c);       or %6s
 printf("%10s\n\n", d);    or %6s
 // right justified and only first 3 letters
 printf("%.10.3s\n", a);
 printf("%.10.3s\n", b);
 printf("%.10.3s\n\n", c);
 // left justified and only first 3 letters
 printf("%-10.3s\n", a);
 printf("%-10.3s\n", b);
 printf("%-10.3s\n\n", c);
 // print only first 3 letters
  printf("%.3s\n", a);     or %.4s
  printf("%.3s\n", b);     or %.4s
```

Output

```
INDIA
AMERICA
PAKISTAN
UAE

IND
AME
PAK
```

```
printf("%.3s \n\n", c);
getch();
}
```

Output -

```
INDIA
AMERICA
PAKISTAN
UAE
```

```
INDIA
 AMERICA
  PAKISTAN
   UAE
```

```
INDIA
AMERICA
PAKISTAN
UAE
```

```
IND
AME
PAK
```

```
IND
AME
PAK
```

```
INDI
AMER
PAKI
```

DATE
PAGE No. 70

Scanf() function -

```
#include <stdio.h>
#include <conio.h>
void main()
{  clrscr();
   int a, b, sum;
   printf("\n Enter Value to A: ");
   scanf("%d", &a);
   printf("Enter value to B: ");
   scanf("%d", &b);
   sum=(a+b);
   printf("\n SUM= %d", sum);
   getch();
}
```

Output- Enter value to A: 6
Enter value to B: 3
SUM = 9

```
#include <stdio.h>
#include <conio.h>
void main()
{  clrscr();
   float a, b, sum;
   printf("\n Enter Value to A: ");
   scanf("%f", &a);
   printf("\n Enter Value to B: ");
   scanf("%f", &b);
   sum = a+b;
   printf("\n SUM = %f", sum);
   getch();
}
```

Output- Enter value to A: 3.1
Enter value to B: 2.1
SUM = 5.200000

→
```
#include < stdio.h >
#include < conio.h >
void main()
{
  clrscr();
  char gender;
  printf("Enter your Gender: ");
  scanf(" %c", & gender);
  printf("Your Gender is %c", gender);
  getch();
}
```

Output - Enter your Gender: m
Your Gender is m

→
```
#include < stdio.h >
#include < conio.h >
void main()
{
  clrscr();
  int age;
  float height;
  char gender;
  scanf("%d %f %c", & age, & height, & gender);
  printf("Your Details are %d %f %c", age, height, gender);
  getch();
}
```

Output-

23
158.0
m

Your Details are 23 158.000000 m

DATE
PAGE No. 71

Escape Sequence - see chart on page no. 15

→ /* Escape Sequence */
#include <stdio.h>
#include <conio.h>
void main()
{
clrscr();
printf("New Line: backslash n \n");
printf("Audible Bell: backslash a \a\n");
printf("Backspace: backslash b Gurkyshowa \b \n");
printf("Form fud: backslash f \f \n");
printf("Carriage return (Enter): backslash r \r\n");
printf("Horizontal Tab: backslash t \t Gurkyshow \n");
printf("Vertical Tab: backslash v \v Gurkyshow \n");
printf("Print Backslash: backslash backslash \\ \n");
printf("Print Question Mark: backslash ? \? \n");
printf("Print Single Quote: backslash ' \' \n");
printf("Print Double Quote: backslash double_quote \" \n");
printf("Null character: backslash 0 \0 \n");
getch();
}

Output - New Lin: backslash n
Audible Bell: backslash a (2 बीप की ध्वनि)
(for printer) → Form fud: backslash f ♀ (Current page को अगले नए page को print करने के लिए)
Backspace: backslash b Gurkyshow (5 remove)
carriage return <Enter>: backslash r (C remove) Gurkyshowa
(for printer) Horizontal tab: backslash t
Vertical Tab: backslash v or Gurkyshowa
Print Backslash: backslash backslash \
(For old compiler) Print Question Mark: backslash ? ?
Print Single Quote: backslash ' '
Print Double Quote: backslash double_quote "
Null Character: backslash 0

Syntax
Syntax-

Assignment Statement :-

Syntax: Variable = Value or Expression;

```
#include <stdio.h>
#include <conio.h>
void main()
{ clrscr();
  int m=25;
  int x= 2;
  int s= 4;
  int y;
  y = x * x + 4 * x - 6;
  s = s + x;
  printf("%d\n", y);
  printf("%d", s);
  getch();
}
```

Output- 6
6

Multiple Assignment Statement :-

Syntax- Variable1 = Variable2 = ——— = Variable n = Value or Expression

```
#include <stdio.h>
#include <conio.h>
void main()
{ clrscr();
  int n=m=25;
  int np = x=2;
  int y, s;
  y = n * x;
  s = np + m;
  printf("%d", y);
  printf("%d", s);
  getch();
}
```

Output- 50
27

Assignment Statement -

Syntax: Variable = Value or Expression;

```
#include <stdio.h>
#include <conio.h>
void main()
{ clrscr();
  int m=25;
  int x= 2;
  int s= 4;
  int y;
  y = x * x + 4 * x - 6;
  s = s + x;
  printf("%d\n", y);
  printf("%d", s);
  getch();
}
```

Output- 6
6

Multiple Assignment Statement -

Syntax- Variable1 = Variable2 = ——— = Variable n = Value or Expression

```
#include <stdio.h>
#include <conio.h>
void main()
{ clrscr();
  int n=m=25;
  int np = x=2;
  int y, s;
  y = n * x;
  s = np + m;
  printf("%d", y);
  printf("%d", s);
  getch();
}
```

Output- 50
27

Use of getchar(), getche() and getch() function:-

```
#include<stdio.h>
#include<conio.h>
void main()
{
  clrscr();
  char a;
  printf("\nEnter  a Value:");
  a= getchar();
  printf("You Entered:%c", a);
  printf("\n\nEnter  a Value:");
  a = getche();
  printf("\n You Entered: %c", a);
  printf("\n\nEnter  a Value: ");
  a= getch();
  printf("\n You Entered:%c",a);
 getch();
}
```

Output-

(i)

(read one time one char.) (Wait करता है Enter press करने का)

Enter a Value: A
You Entered: A

(ये एक समय पर एक शब्द लेता है और Enter का Wait नहीं करता है।)

Enter a Value: a
You Entered: a

(यह दिखाता नहीं है पासवर्ड लेने में होता है ताकी Parent न दिखे।)

Enter a Value:
You Entered: a

(ii)

Enter a Value: ABC
You Entered: A

DATE
PAGE No. 75

Display the ASCII Value in C -

```
/* Program to display ASCII Value */
#include <stdio.h>
#include <conio.h>
void main()
{
  char ch;
  printf("Enter a character: ");
  ch = getchar();
  printf("\n The ASCII Value of %c is %d\n", ch, ch);
  getch();
}
```

Output - Enter a character: A
The ASCII Value of A is 65

6. putchar() and putch() Function -

```
#include <stdio.h>
#include <conio.h>
void main()
{
  clrscr();
  char a='G';
  putchar(a);
  putchar('\n');
  putchar('S');
  putch('\n');
  putch(a);
  getch();
}
```

Output - G
S
G

Note -

1- यह केवल एक समय में एक ही Character को प्रिन्ट करता है।

2- putch Linix को में नहीं काम करता है।

puts() -

Syntax- puts(Variable);
or puts("String");

gets() -

Syntax- gets(Variable);

DATE
PAGE No. 77

Difference between printf() and puts() -

```
/* Difference between printf and puts */
#include <stdio.h>
#include <conio.h>
void main()
{
  clrscr();
  char name[50];
  printf("Enter Your Name : ");
  scanf("%s", &name);
  printf("Your name is %s \n", name);
  getch();
  puts("Enter Your Name : ");  // It create a new line.
  scanf("%s", &name);
  puts("Your Name is");
  puts(Name);
  getch();
}
```

automatic

Output- Enter Your Name : Geekyshows
Your name is Geekyshows

Enter Your Name :
Geekyshows
Your Name is
Geekyshows

Note- Puts automatic new line create करके देता है और यह केवल string प्रिंट करता है।

Difference between scanf() and gets() —

```
/* difference between scanf and gets */
#include<stdio.h>
#include<conio.h>
void main()
{ clrscr();
  char name[50];
  char fullname[50];
// gets can scan string with space ex: Gurky shaw
  printf("Enter Your Name:");
  gets(fullname);
  printf("Your Name is %s \n", fullname);
// scanf can not scan string with space ex: Gurky shaw
// scanf will be only scan Gurky
  printf("Enter your Name:");
  scanf("%s", &name);
  printf("Your name is % \n", name);
  getch();
}
```

Output - Enter Your Name : Gurky Shaw
Your Name is Gurky Shaw

Enter Your Name : Gurky Shaw
Your name is Gurky

Note - gets() space को भी store कर सकता है और scanf() space नहीं store कर पाता है।

Kreator

Clrscr() vs system("cls");

Clrscr() - Old compiler में ही काम करता है।

```
#include <stdio.h>
main()
{
printf("Hello!");
getch();
system("cls");
printf("Bye!");
}
```

Output:

Hello!
(हट जायेगा)

Bye!
(ये आजायेगा)

```
#include <stdio.h>
main()
{
print f(" Hello!");
printf(" Bye!");
}
```

Output-

Hello! Bye!

Basic Control Statements-

1- **Sequence Instruction-** executing one instruction after another, in the order in which they occur in the source code. (Simple)

2- **Selection/Decision Instruction-** It means executing different sections of code depending on a specific condition or the value of a variable. Ex- if, if-else, switch.

3- **Repetition or Loop Instruction-** It means executing the same section of code more than once. Ex- while, do-while, for.

If Statement

Definition- It is used to execute an instruction or sequence/block of instructions only if a condition is fulfilled.

1- Simple if statement
2- if - else statement
3- Nested if - else statement
4- Else if statement.

1- Simple if Statement-

It is used to execute an instruction or block of instructions only if a condition is fulfilled.

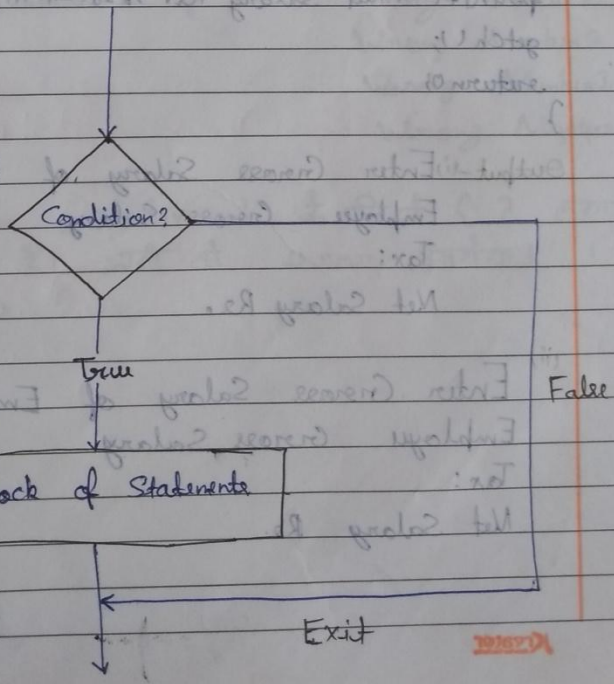
Syntax- (i) For one statement-

```
if (Condition)
    statement;
```

(ii) For more than one statement-

```
if (Condition)
{
    Block of statements;
}
```

Flow chart-



```
/* Program to calculate the net Salary of an employee.
#include <stdio.h>
#include <conio.h>
int main()
{
  float tax, gross_salary, net_salary;
  printf("Enter Gross Salary of Employee: ");
  scanf("%f", &gross_salary);
  printf("Employee Gross Salary %9.2f\n", gross_salary);
  if(gross_salary<10000)
     net_salary=gross_salary;
  if(gross_salary>=10000)
   {
     tax=gross_salary*0.15;
     net_salary=gross_salary-tax;
   }
  printf("Tax: %26.2f", tax);
  printf("\nNet Salary Rs. %16.2f\n", net_salary);
  getch();
  return 0;
}
```

Output-i) Enter Gross Salary of Employee: 9000
Employee Gross Salary 9000.00
Tax: 0.00
Net Salary Rs. 9000.00

(ii) Enter Gross Salary of Employee:
Employee Gross Salary 10000.00
Tax: 1500.00
Net Salary Rs. 8500.00

→
```
#include <stdio.h>
#include <conio.h>
void main()
{
  clrscr();
  int n=2;
  if (n!=2)
    printf("Wrong Number 1\n");
    printf("Wrong Number 2\n");
  if (n!=2)
  {
    printf("Wrong Number 3\n");
    printf("Wrong Number 4\n");
  }
  getch();
}
```

or `if(n=2)`

or `if(n=2)`

Output -

Wrong Number 2

Wrong Number 1
Wrong Number 2
Wrong Number 3
Wrong Number 4

Note - Multiple statement के लिए { } लगाना जरूरी है। नहीं तो error आयेगा।

(ii) Nested if statement -

Syntax -

```
if (condition)
{
    block of statements;
    if (condition)
    {
        block of statements;
    }
    if (condition)
    {
        block of statements;
    }
}
if (condition)
{
    block of statements;
}
```

2- if ... else statement -
if ... else statement is used when a different sequence of instructions is to be executed depending on the logical value (True/False) of the condition evaluated.

Syntax -

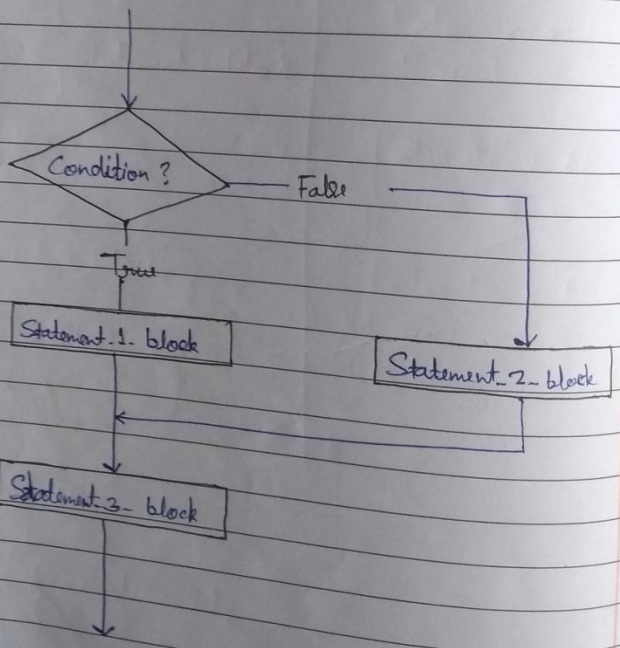
For single statement

```
if (condition)
    statement_1;
else
    statement_2;
statement_3;
```

For more than one statement

```
if (condition)
{
    statement_1_block
}
else
{
    statement_2_block
}
statement_3_block;
```

Flowchart -



→ /* Program to print whether the given number is even or odd */

```
#include<stdio.h>
#include<conio.h>
int main()
{
  clrscr();
  int n;
  printf("Enter a Number: ");
  scanf("%d", &n);
  if(n%2==0)
     puts("\n Number is Even");
  else
     puts("\n Number is odd");
  return 0;
  getch();
}
```

Output -

(i) Enter a Number: 4
Number is Even

(ii) Enter a Number: 3
Number is odd

3- Nested if ... Else statement-

In nested if ... else statement, an entire if ... else construct is written within either the body of the if body of an else statement.

Syntax-1

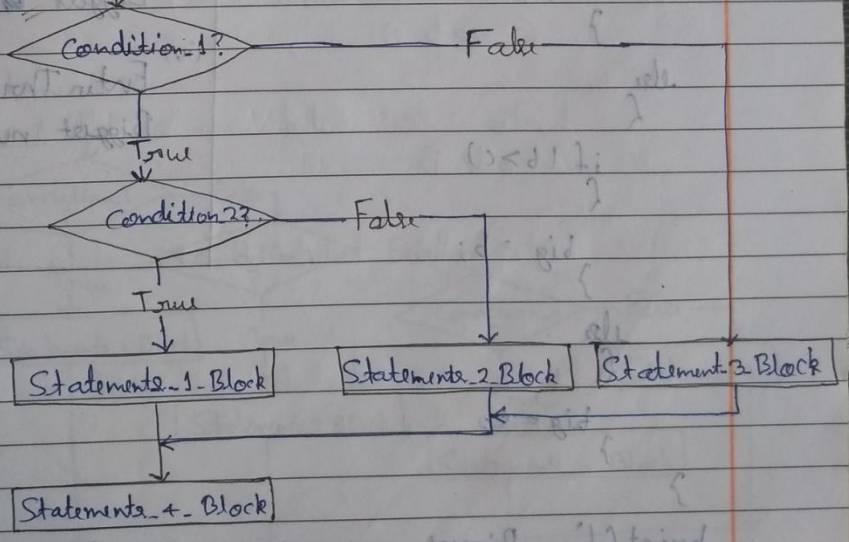
```
if (condition_1)
 {
    if (condition_2)
          {
            statements_1_Block;
          }
    else
          {
            statements_2_Block;
          }
 }
else
 {
   statements_3_Block;
 }
Statements_4_Block;
```

Syntax-2

```
if (condition_1)
 {
      if (condition_2)
          {
            statements_1_Block;
          }
```

```
        else
        {
            statements_2_Block;
        }
    }
    else
    {
        if (condition_3)
        {
            statements_3_Block;
        }
        else
        {
            statements_4_Block;
        }
    }
    Statements_5_Block;
```

Flow chart



→ /* Program to find biggest of three numbers */
/* Nested if else Statement */

```
#include <stdio.h>
#include <conio.h>
void main()
{ clrscr();
  int a, b, c, big;
  printf("Enter Three Numbers: ");
  scanf("%d %d %d", &a, &b, &c);
  if (a>b)
   {
      if (a>c)
       {
         big = a;
       }
      else
       {
         big = c;
       }
   }
  else
   {
      if (b>c)
       {
         big = b;
       }
      else
       {
         big = c;
       }
   }
printf("\n Biggest number is %d\n", big);
getch();
}
```

Output -

(i) Enter Three Numbers:
Biggest number is 10

(ii) Enter Three Numbers: 10
Biggest number is 12

(iii) Enter Three Numbers: 3
Biggest number is 10

(iv) Enter Three Numbers: 3
Biggest number is 12

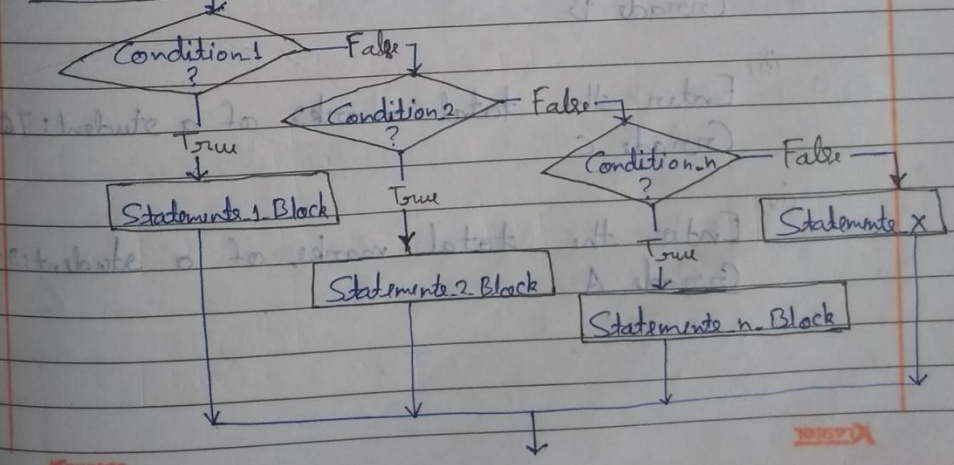
Else if statement :-

To show a multi-way decision based on several conditions, we use else if statement.

Syntax-

```
if (condition_1)
  {
    statements_1_Block;
  }
  else if (condition_2)
    {
      statement_2 Block;
    }
    else if (condition_n)
      {
        statement_n_Block;
      }
else
    statements X;
```

Flowchart-



→ /* Program to Award Grade */

```
#include <stdio.h>
#include <conio.h>
void main()
{
 clrscr();
 printf("Enter the total marks of a student: ");
 scanf("%d", &result);
 if(result<=50)
    printf("Grade D\n");
    else if (result<=60)
       printf("Grade C \n");
       else if (result <= 75)
          printf("Grade B \n");
 else
   printf(" Grade A \n");
 getch();
}
```

Output - (i) Enter the total marks of a student: 50
Grade D

(ii) Enter the total marks of a student: 60
Grade B

(iii) Enter the total marks of a student: 70
Grade C

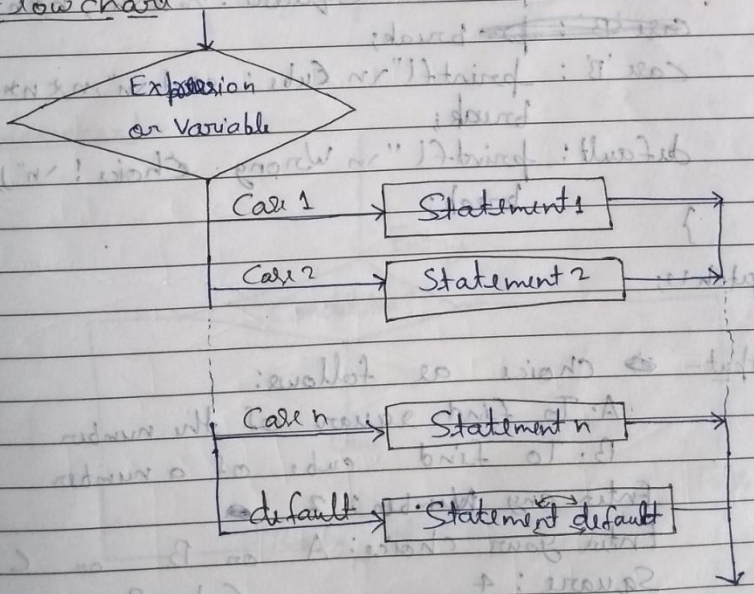
(iv) Enter the total marks of a student: 90
Grade A

Switch Statement- check several possible constant value for an expression.

Syntax-

```
switch(expression)
{
  case expression 1:
      block of statements 1
      break;
  case expression 2:
      block of statement 2
      break;
  default:
      default block of instructions
}
```

Flowchart-



```
#include <stdio.h>
#include <conio.h>
void main()
{ clrscr();
  puts("Choice as follows:");
  puts("A. To find square of the number");
  puts("B. To find cube of a number");
  int n;
  char choice;
  printf("\n Enter any Number:");
  scanf("%d", &n);
  printf("Enter Your choice:");
  scanf("%c", &choice);
  switch(choice)
  {
    case 'A': printf("\n Square : %d\n", n*n);
              break;
    case 'B': printf("\n Cube : %d\n", n*n*);
              break;
    default: printf("\n Wrong Choice !\n");
             break;
  }
getch();
}
```

Output- Choice as follows:
A. To find square of the number
B. To find cube of a number
Enter any Number: 2
Enter your choice: A or B or C
Square : 4
Cube : 8
Wrong ch

DATE
PAGE No. 97

Loop Control Statements- Loop Control Statements are used when a section of code may either be executed a fixed number of times, or while some condition is true.

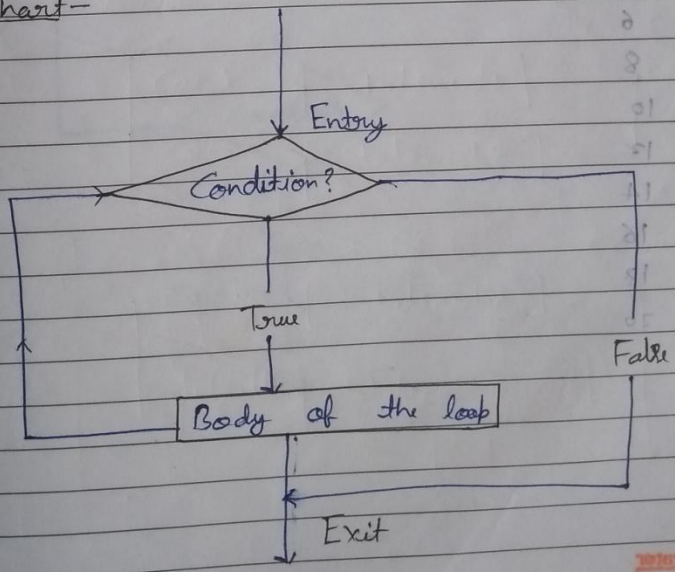
1- While
2- do-while
3- for

1- **While loop**- The while loop keeps repeating an action until an associated condition returns false.

Syntax-

```
while (test condition)
{
    body of the loop;
}
```

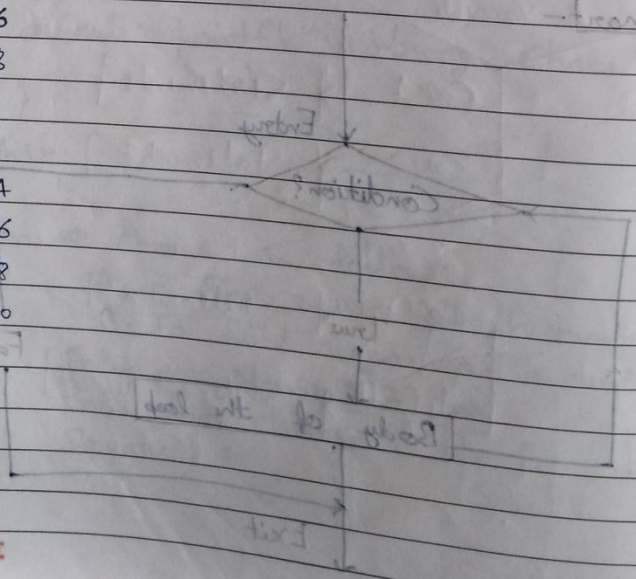
Flowchart-



```
/* Program to Print Table of 2 */
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
 int a=2;
 while (a<=20)
 {
 printf("%d\n", a);
  a+=2;
 }
 getch();
}
```

Output-

2
4
6
8
10
12
14
16
18
20



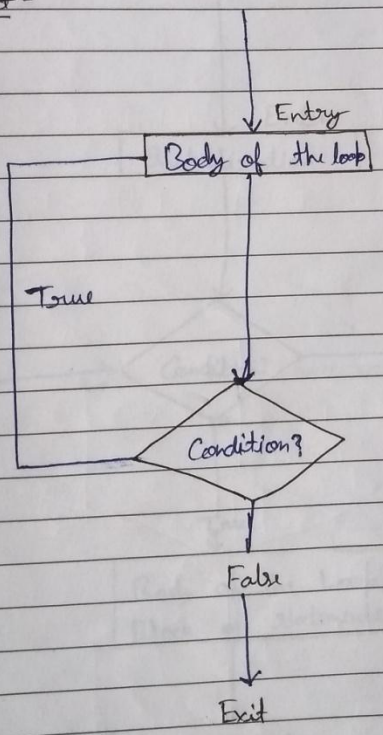
DATE
PAGE No. 99

2- do.....while Loop- The do while loop is similar to while loop, but the condition is checked after the loop body is executed. This ensure that loop body is run at least once.

Syntax-

```
do
{
    statements;
} while (test condition);
```

Flowchart-

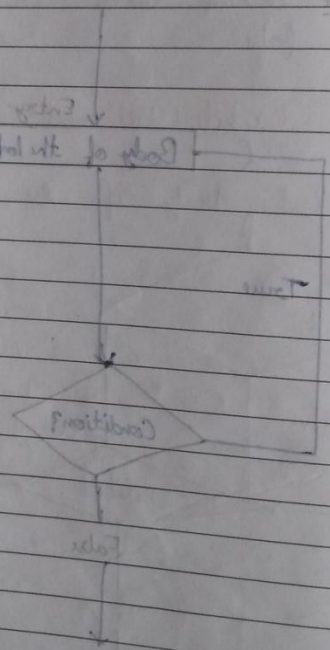


DATE
PAGE No. 100

```
#include<stdio.h>
#include<conio.h>
void main()
{
  int a=0,b=2;
  do
  {
   printf("%d\n",b);
   b+=2;
   a++;
  }while(a<10);
  getch();
}
```

Output-

2
4
6
8
10
12
14
16
18
20

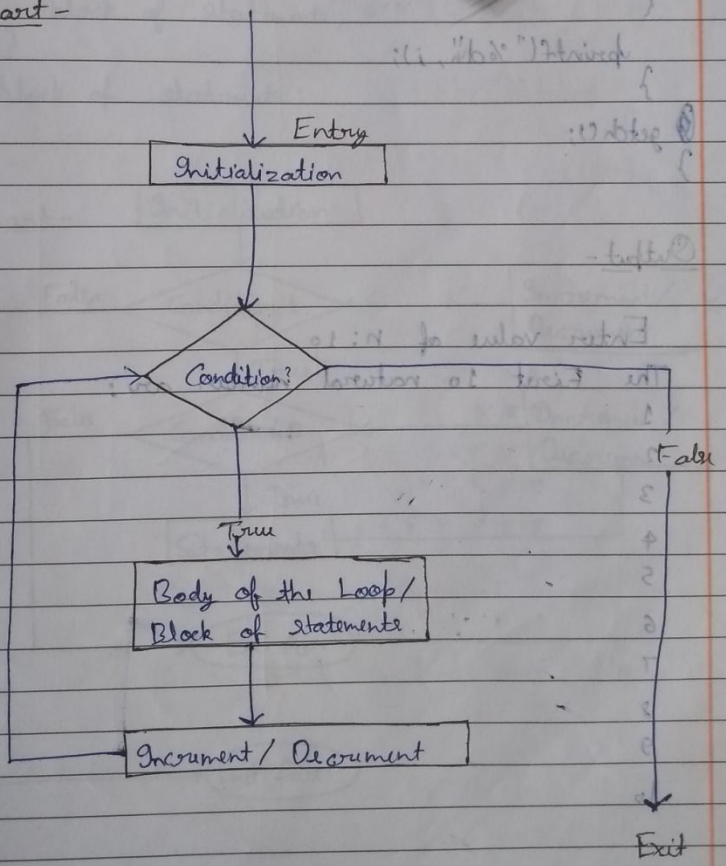


for loop – The for loop is frequently used, usually where the loop will be traversed a fixed number of times.

Syntax-

```
for(initialization; test condition; increment or decrement)
{
    block of statements;
}
```

Flowchart -



DATE
PAGE No. 102

// Program to print first n natural number

```
#include <stdio.h>
#include <conio.h>
void main()
{
  int i, n;
  printf("Enter value of n: ");
  scanf("%d", &n);
  printf("\n The First %d natural number are: \n", n);
  for(i=1; i<=n; ++i)
  {
    printf("%d\n", i);
  }
  getch();
}
```

Output -

```
Enter value of n: 10
The First 10 natural number are:
1
2
3
4
5
6
7
8
9
10
```

Kreator

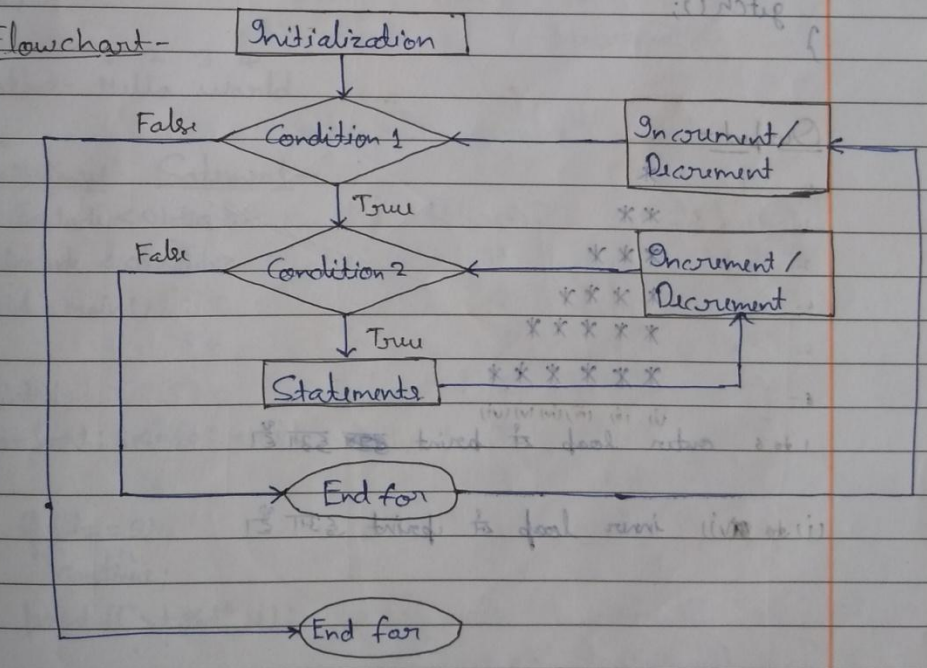
(ii) Nested for loop – The statement block of a for loop lies completely inside the block of another loop.

Syntax –

```
for(initialization; test condition; increment or decrement)
{
    for(initialization; test condition; increment or decrement)
    {
        block of statements;
    }
    block of statements;
}
```

Outer loop
Inner loop

Flowchart –



```
#include <stdio.h>
#include <conio.h>
void main()
{
  clrscr();
  int i, j;
  for(i=1; i<=6; i++)
  {
    for(j=1; j<=i; j++)
    {
      printf("* ");
    }
    printf("\n");
  }
  getch();
}
```

Output -

```
1.  *
2.  * *
3.  * * *
4.  * * * *
5.  * * * * *
6.  * * * * * *
```

(i) (ii) (iii) (iv) (v) (vi)

1 to 6 outer loop से print हुआ है।

(i) to (vi) inner loop से print हुआ है।

DATE
PAGE No. 105

Break Statement-

```
#include <stdio.h>
#include <conio.h>
void main()
{
  clrscr();
  int i;
  for(i=1;i<=10;++i)
  {
    if(i=5)
      break;
    printf("\t%d",i);
  }
  printf("Hello world");
getch();
}
```

Output- 1 2 3 4 Hello world

Continue Statement-

```
#include <stdio.h>
#include <conio.h>
void main()
{
  int i;
  for(i=1;i<=10;++i)
  {
    if(i==5)
      continue;
    printf("\t%d",i);
  }
getch();
}
```

Output- 1 2 3 4 6 7 8 9 10

Array - Array provide a mechanism for declaring and accessing several data items with a one identifier, thereby simplifying the task of data management.

Syntax - ~~data type name~~ data type array name [size];
data type array name[5] = {1,2,3,4,5};

- Subscript - Subscript is an integer type constant or variable name whose value ranges from 0 to size-1, where size is the total number of elements in the array.

Ex - int a[5] , int a[5][6] , int a[2][3][5]
Subscript Subscript Subscript

Memory Creation

Element No.	Subscript
1	0
2	1
3	2
4	3
5	4

A Array

Reference	Value
a[0]	23
a[1]	45
a[2]	5
a[3]	65
a[4]	234

Type of Array -

1- One Dimensional Array
2- Two Dimensional Array
3- Multi-Dimensional Array

DATE
PAGE No. 107

1- One Dimensional Array- Single subscripted variable is referred to a one dimensional or Linear Array.

Initialization-

Syntax- datatype arrayname [size]={val1, value ---- valn};
datatype array_name[ ]={val1, val2, ---- valn};
datatype array_name[size];

Ex- int digits[10]={1,2,3,4,5,6,7,8,9,10};
int digits[ ]={1,2,3,4,5,6,7,8,9,10};
int digits[10], val[8];
float ap[100];

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
int a[3];
scanf("%d %d %d", &a[0], &a[1], &a[2]);
printf("You Entered: %d %d %d, a[0], a[1], a[2]);
getch();
}
```

Output- 3
45
34
You Entered: 3 45 34

/* Find Maximum marks among the marks of 10 students */
/* one dimensional array */

```
#include <stdio.h>
#include <conio.h>
void main()
{ clrscr();
  int i, maxi=0, mark[10];
  int sn=1;
  for(i=0; i<10; i++)
  {
    printf("\n Student No. %d: ", sn++);
    printf("\n Marks obtained out of 50: ");
    scanf("%d", &mark[i]);
  }
  for(i=0; i<10; i++)
  {
    if(mark[i]>maxi)
      maxi = mark[i];
  }
  printf("\n Maximum Marks: %d", maxi);
  getch();
}
```

Output-

Student No.1:
Marks obtained out of 50: 20
Student No.2:
Marks obtained out of 50: 30
Student No.3:
Marks obtained out of 50: 12
Student No.4:
Marks obtained out of 50: 23
Student No.5:
Marks obtained out of 50: 24
Student No.6:
Marks obtained out of 50: 27
Student No.7:
Marks obtained out of 50: 23
Student No.8:
Marks obtained out of 50:
Student No.9:
Marks obtained out of
Student No.10:
Marks obtained out of 50
Maximum Marks: 49

2- Two Dimensional Array- Two subscript variable is referred to two dimensional or Matrix array.

Initialization-

Ex-(i) /* Valid declaration */

```
int a[3][2] = {
                {1,6},
                {5,2},
                {3,4}
              };
```

	a[0]	a[1]
R→a[0]	1	6
R→a[1]	5	2
R→a[2]	3	4

(ii) /* Valid declaration */

```
int a[3][2] = {1,6,5,2,3,4};
```

(iii) /* Valid declaration */

```
int a[3][2] = {
                {1},      → 1 0
                {5,2},    → 5 2
                {6},      → 6 0
              };
```

(iv) /* Valid declaration */

```
int abc[ ][2] = {1,2,3,4};      → 1 2
                                → 3 4
```

(v) /* Invalid declaration - you must specify second dimension */

```
int abc[ ][ ] = {1,2,3,4};   ✗
```

(vi) /* Invalid because of the same reason mentioned above */

```
int abc[2][ ] = {1,2,3,4};   ✗
```

(vii) /* Valid */
int a[2][3];

(viii) /*Invalid */
int a[ ][ ]; ✗

(ix) /* Invalid */
int a[ ][3]; ✗

(x) /* Invalid */
int a[2][ ]; ✗

```
- #include<stdio.h>
#include<conio.h>
void main()
{
 clrscr();
 int a[2][3];
 scanf("%d %d %d %d %d %d", &a[0][0], &[0][1], &a[0][2], &a[1][0], &a[1][1]
                                                    , &a[1][2]);
 printf("%d %d %d %d %d %d", &a[0][0], &[0][1], &a[0][2], &a[1][0], &a[1][1]
                                                    , &a[1][2]);
 getch();
}
```

Output - 2
3
4
6
5
7

2 3 4 6 5 7

```
/* Two Dimensional Array */
#include<stdio.h>
#include<conio.h>
void main()
{ clrscr();
  int i, j, total = 0, a[2][3];
  printf("Enter 6 numbers: \n");
  for(i=0; i<2; i++)  // outter loop for row
  {
    for(j=0; j<3; j++) // inner loop for column
    {
      scanf("%d", &a[i][j]);
    }
  }
  for(i=0; i<2; i++)
  {
    for(j=0; j<3; j++)
    {
      printf("\t %d", a[i][j]);
      total = total + a[i][j];
    }
    printf("\t %d", total);
    total = 0;
    printf("\n");
  }
  getch();
}
```

Output- Enter 6 numbers:

10
10
10
20
20
20

10	10	10	30
20	20	20	60

Symbolic Constant in Array

```
/* One diamentional array */
#include <stdio.h>
#include <conio.h>
#define SIZE 10
void main()
{
  clrscr();
  int i, maxi=0, mark[SIZE];
  int sn=1;
  for(i=0; i<SIZE; i++)
  {
    printf("\n Marks obtained out of 50: ");
    scanf("%d", &mark[i]);
  }
  for(i=0; i<SIZE; i++)
  {
    if (mark[i]>maxi)
      maxi = mark[i];
  }
  printf("\n Maximum Marks: %d", maxi);
  getch();
}
```

Output - Same as page no.

String- String in c are group of characters, digits and symbols enclosed in quotation marks. The end of the string is marked with a spacial character, the \0 (null character), which has the decimal value 0. The string functions operate on null terminated array of characters and require the header <string.h>

We use %s format of string.

Declaration of string-

Syntax- char string_name[size];

Ex- char number[6];

A	P	P	L	E	\0

Initialization of String-

Ex- char name[5] = {'A','P','P','L','E'}; ✗
char name[6] = {'A','P','P','L','E','\0'}; ✓
char name[6] = {'A','P','P','L','E'}; ✓
char name[ ] = {'A','P','P','L','E'}; ✗
~~char name[ ] = {'A','P','P','L','E'};~~
char name[ ] = "APPLE"; ✓
char name[5] = "APPLE"; ✗
char name[6] = "APPLE"; ✓

A	P	P	L	E	\0

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
  clrscr();
  char name[12];
  printf("Enter Name: ");
  scanf("%s", name);
  printf("Welcome to %s", name);
  getch();
}
```

(यहाँ & लगाना जरूरी नहीं)

Output -

(i) Enter Name: SatyamSeth
Welcome to SatyamSeth

(ii) Enter Name: Satyam Seth
Welcome to Satyam

Array of String -

Array of String are multiple strings stored in the form of table.

Syntax-

Char array_name [num_string][length_string]

Ex- char mato[2][5];

	0	1	2	3	4
0	W	O	R	D	\0
1	b	O	O	K	\0

Intialization- char mato[3][10] = {"computer", "book", "log"};

char mato[ ][10] = {"computer", "book", "log"};

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
  clrscr();
  char name[2][10];
  printf"Enter Two String: ");
  scanf"%s %s", name[0], name[1]);
  printf"\nFirst String: %s \nSecond String: %s, name[0], name[1]);
  printf"\n%c\n", name[0][2]);
  getch();
}
```

Output- Enter Two String: Satyam Seth

First String: Satyam
Second String: Seth
a

String Handling Function-

1- strlen() function
2- strcpy() function
3- strcmp() function
4- strcat() function
5- strlwr() function
6- strupr() function
7- strrev() function
8- strncpy() function
9- stricmp() function
10- strncmp() function
11- strchr() function
12- strset() function
13- strncat() function
14- strspn() function
15- strstr() function
16- strcmpi() function
17- atoi() function
18- strnset() function
19- strrchr() function
20- strrstr() function
21- strdup() function

Character Handling Function-

1- isupper() function
2- islower() function
3- isalpha() function
4- isalnum() function
5- isdigit() function
6- isxdigit() function
7- tolower() function
8- toupper() function
9- isspace() function
10- ispunct() function
11-
12-

String Handling Function

DATE
PAGE No. 117

1- strlen( ) function-

Syntax:-

n = strlen (string_name);

where n is an integer variable

```
/* strlen( ) function */
/* find the length of string */
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
   int n;
   char web[20] = "Google";
   n = strlen(web);
   printf("%d", n);
   getch();
}
```

Output - 6

(यह अक्षरों की संख्या बताता है।)

2- ~~Syntax~~ Strcpy() function-

Syntax- strcpy (str1, str2);
where str1 and str2 are two strings.

```
/* strcpy() function*/
/* copy one string to another*/
#include <stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
  ~~clrscr~~ clrscr();
   char web[10] = "News";
   char site[10] = "Hunt";
   strcpy(web, site);
   printf("%s", web);
   printf("\n%s\n", site);
   getch();
}
```

Output- Hunt
Hunt

~~Output~~ News
Hunt

(यह दूसरे String को पहले में copy कर देता है)।(न)

13- Strncpy() function-

Syntax- strncpy (str1, str2, 10);
where str1 and str2 are two string. The 10 characters of string str2 are copied onto string str1.

```
/* Strncpy() function */
/* Copy one string to another with specified length*/
#include<string.h>
#include<conio.h>
#include<stdio.h>
void main()
{
clrscr();
char web[10] = "News";
char site[10] = "Hunt";
strncpy(web, site, 2);
printf("%s" web);
printf("\n%s\n", site);
getch();
}
```

Output- Hews
Hunt

12- strset() function -

Syntax- strset(str, ch); where str is string and ch is character

```
/* strset() function */
/* replace string to one character */
#include <stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
clrscr();
char str[10]="Apple";
strset(str,'*');
printf("%s", str);
getch();
}
```

or // strset(str,'x');

Output-

****

Output :- Apple

यह string को किसी एक char से बदल देता है

18. strnset() function –

Syntax – strnset(str, ch, n); Where str is string and ch is character (and n is number in int type.)

```
/* strnset() function */
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
clrscr();
char str[10] = "Apple";
strnset(str,'*',3);
printf("%s", str);
getch();
}
```

Output – ***le

3- strcmp ( ) function -

Syntax- n = strcmp (str1, str2);
where str1 and str2 are two strings; n is an integer variable.

```
/* strcmp() function */
/* compare two string character by character */
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
clrscr();
int n;
char str[10]= "Gruk";      or "Gruksharma"; or "Gr"
char str1[10]= "Gruk";     or "Gruk"; or "Gruk"
n=strcmp(str, str1);
printf("%d", n);
getch();
}
```

Output- 0 Output- 1 Output- -1

First string = Second string then 0
First string > Second string then +ive
First strin < Second string then -ive.

(Output ASCII value में आयेगा)
(Case sensitive होता है।)

DATE
PAGE No. 123

9. strcmp() function-

Syntax- `n = stricmp(str1, str2);`

where str1 and str2 are two strings and n is an integer variable.

```
/* stricmp() function */
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
   clrscr();
   int n;
   char str[10] = "GEEK";
   char str1[10] = "Geek";
   n = stricmp(str, str1);
   printf("%d", n);
   getch();
}
```

Output- 0

10- strcmpi() function-

Syntax- `n = strcmpi(str1, str2);`

where str1 and str2 are two strings and n is an integer variable.

```
/* strcmpi() function */
n = strcmpi(str, str1);
```

Output- 0

First string = Second string then 0
First string > Second string then +ve
First string < Second string then -ve
(It Case sensitive नहीं होता है।)

10- strncmp() function -

Syntax- n = strncmp(str1, str2, 10);
where str1 and str2 are two strings. The 10 character of string str1 and str2 will compare. n is an intiger variable.

```
/* strncmp() function */
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
  clrscr();
  int n;
  char Str[20] = "Geekyshows"; or "GEEKyshows";
  char str1[10] = "Geeky";
  n= strncmp(str, str1, 4);
  printf("%d", n);
  getch();
}
```

Output- 0

Output- -32

First string = Second string then 0
First string > Second string then +ive
First string < Second string then -ive
(1st Case sensitive होता है)।

string2. The
str2 will be

EEKyshower";

2

ive
ive

~~strcat~~ ~~Strcatpi~~ () function -

Syntax - strcat (str1, str2);
where str1 and str2 are two strings.

```
/* strcat() function
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
clrscr();
char web[20] = "Geeky";
char site[20] = "Shows";
strcat(web, site); or (web, "Shows"); // strcat(web, site);
printf("%s", web);
getch();
}
```

Output - GeekyShows Output GeekyShows Output - Geeky

(दो String को जोड़ता है।)

13- strncat() function -

Syntax - strncat(str1, str2, 5);
where 5 characters of the str2 string is added into str1 string.

```
/* strncat() function */
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
  clrscr();
  char web[20] = "Geeky"; or "Geeky ";
  Char site[20] = "Shows"; or " Shows"
  strncat(web, site, 4);
  printf("%s", web);
  getch();
}
```

Output - GeekyShows

Output - GeekyShows

Output - Geeky Shows

Output - Geeky Sho

5. strlwr(

Syntax-

/* strl
#includ
#includ
#includ
void m
{
clrscr
char
strlw
print
getch
}

Output-

(

DATE
PAGE No. 127

5. strlwr() function -

Syntax strlwr(str);
where str is string.

```
/* strlwr() function */
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
  clrscr();
  char web[20] = "GEEKYSHOWS";
  strlwr(web);
  printf("%s", web);
  getch();
}
```

Output - geekyshows
(Upper case to lower case.)

14- strupr() function -

Syntax - strupr(str);
where str is string.

```
/* strupr() function */
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
  clrscr();
  char web[20] = "geekyshows";
  strupr(web);
  printf("%s", web);
  getch();
}
```

Output -

GEEKYSHOWS

(Lower case to upper case)

6- Strrev() function-

Syntax- strrev(str);
Where str is string.

```
/* strrev() function */
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
   clrscr();
   char web[20] = "gukyshows";
   strrev(web);
   printf("%s", web);
   getch();
}
```

Output- swohsykug

(यह string को reverse करता है।)

7- strspn() function -

Syntax- n = strspn (str1, str2);
where str1 and str2 are two strings.
n is an integer variable.

```
/* strspn() function */
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
  clrscr();
  char web[20] = "Gurbey";   or "Gurbyw";
  char site[20] = "Gurbyshow";
  n = strspn (web, site);
  printf ("%d", n);
  getch();
}
```

Output- 5

Output- 6

(कितने character match कर रहे हैं गिनकर बताना)

11- strchr() function-

- strchr() function returns pointer to the first occurrence of the character in a given string.

Syntax- p = strchr(str, 'c');
where str is string and c is character and p is character pointer.

```
/* strchr() function */
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
  char web[100] = "Gurkyshows is an awesom website for gurkz.");
  char *p;
  p = strchr(web, 'i');
  printf("%s", p);
  getch();
}
```

Output- is an awesom website for gurkz.

(पहले ; से प्रिंट करेगा)

19- strrchr() function -

strrchr() function returns pointer to the last occurrence of the character in a given string.

Syntax- p = strrchr(str, 'c');
Where c is character and p is character pointer. and str is string

```
/* strrchr() function */
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
clrscr();
char web[50]= "Gurkyshows is an awesome website for geeks.";
char *p;
p = strrchar(web,'i');
printf("%s", p);
getch();
}
```

Output- ite for geeks.
(अंतको ; से प्रिंट करेगा)

15- strstr() function -

strstr() function returns pointer to the first occurrence of the string in a given string.

Syntax - p = strstr(str1, "str");
where str1 and str are two string, p is character pointer.

```
/* strstr() function */
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
  clrscr();
  char web[100]="Gukyghows is an awsome website for geks";
  char *p;
  p = strstr(web, "an");
  printf("%s", p);
  getch();
}
```

Output - an awsome website for geks.
(an से शुरू होने वाला प्रिंट करेगा।)

20- strrstr() function -

strrstr() function returns pointer to the last occurrence of the string in a given string.

Syntax- p = strrstr(str1, "str");
where str1 and str are two strings,
p is character pointer.

```
/* strrstr() function */
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
 clrscr();
 char web[100]= "Gurkyshouse is for awesome website for geeks";
 char *p;
 p = strrstr( web," for ");
 printf("%s", p);
getch();
}
```

Output- for geeks.
(यह Last के for से प्रिंट करेगा।)
(सभी Compailers में Support नहीं करता है।)

17- atoi() function :-

Syntax- n = atoi(str);
where str is string and n is an integer variable.

```
/* atoi() function */
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<stdlib.h>
void main()
{
  clrscr();
  char[20] = "241736"; or "Gurkyashowa";
  int n;
  n = atoi(str);
  printf(" %d", n);
  getch();
}
```

Output- 241736

Output- 0
(यदि Valid input नहीं होगा तो 0 देता है।)

(यह String datatype को integer में convert कर देता है।)

21- strdup() function -

Syntax- p1 = strdup(p);
where p is string and p1 is character pointer.

```
/* strdup() function */
#include <stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
  clrscr();
  char p[20] = "Geekyshows"; or //char p[20]="Geekyshows
  char *p1;
printf("%s",p); p1 = strdup(p);   or p1 = strdup("Geekyshows");
  printf("%s", p1);
  getch();
}
```

Output- Output - Geekyshows.
Geekyshow
Geekyshow

Output Geekyshows.

(ये string का एक Duplicate String बनाता है)

Character Handling Function

DATE
PAGE No. 137

3- isalpha() function -

Syntax- isalpha(ch); (where ch is character variable)

```
/* isalpha() function */
/* Test if a character is an alphabet */
#include <stdio.h>
#include <conio.h>
#include <ctype.h>
void main()
{
  clrscr();
  char ch;
  printf("Enter any character:");
  scanf("%c", &ch);
  if(isalpha(ch))
  {
    printf("\nValid character\n");
  }
  else
  {
    printf("\nInvalid character\n");
  }
  getch();
}
```

Output- (i) Enter any character: A
Valid character
(ii) Enter any character: a
Valid character
(iii) Enter any character: 2
Invalid character
(iv) Enter any character: #
Invalid character.

(यह check करता है कि character Alphabet है कि नहीं)

1- isupper() function -

Syntax - isupper(ch); Where ch is character variable

```
/* isupper() function */
/* Test if a character is a upper case latter */
#include <stdio.h>
#include<conio.h>
#include <ctype.h>
void main()
{ clrscr();
  char ch;
  printf("Enter any character:");
  scanf("%c", &ch);
  if (isupper(ch))
  {
    printf("\n You Entered upper case character\n");
  }
  else
  {
  printf("\n Invalid character");
  }
  getch();
}
```

Output - (i) Enter any character: A
You Entered upper case character
(ii) Enter any character: a
Invalid character
(iii) Enter any character: 2
Invalid character
(iv) Enter any character: #
Invalid character

(It check करता है कि character upper case है कि नहीं।)

2- islower() function -

Syntax- islower(ch); where ch is character variable

```
/* islower() function */
/* test if a character is lower case letter */
#include <stdio.h>
#include <conio.h>
#include <ctype.h>
Void main()
{ clrscr();
  char ch;
  printf("Enter any character:");
  scanf("%c", &ch);
  if (islower(ch))
  {
    printf("\n You Enter lower case character\n");
  }
  else
  {
    printf("\n Invalid character");
  }
  getch();
}
```

Output- (i) Enter any character: a
You Enter lower case character
(ii) Enter any character: A
Invalid character
(iii) Enter any character: 3
Invalid character
(iv) Enter any character: @
Invalid character

Kreator (यें check करता है कि character ~~character~~ lower ~~character~~ case है कि नहीं)

4- isalnum() function -

Syntax- isalnum(ch); where ch is character variable

```
/* isalnum() function */
/* Test if a character is an alphabet or number */
#include <stdio.h>
#include <conio.h>
#include <ctype.h>
void main()
{ clrscr();
  char ch;
  printf("Enter any character");
  scanf("%c", &ch);
  if (isalnum(ch))
  {
    printf("\n Valid character\n");
  }
  else
  {
    printf("\n Invalid character\n");
  }
  getch();
}
```

Output - (i) Enter any character: A
Valid character
(ii) Enter any character: a
Valid character
(iii) Enter any character: 4
Valid character
(iv) Enter any character: #
Invalid character

Kreator (It check करता है की character number or alphabet है या नहीं)

5. isdigit() function –

Syntax – isdigit(ch) where ch is character variable

```
/* isdigit() function */
/* Test if a character is a number */
#include<stdio.h>
#include<conio.h>
#include<ctype.h>
Void main()
{ clrscr();
  char ch;
  printf("Enter any character: ");
  scanf("%c", &ch);
  if (isdigit(ch))
  {
    printf("\n Valid character\n");
  }
  else
  {
    printf("\n Invalid character\n");
  }
  getch();
}
```

Output – (i) Enter any character: 3
Valid character
(ii) Enter any character: A
Invalid character
(iii) Enter any character: b
Invalid character
(iv) Enter any character: %
Invalid character.

Creator (यह check करता है कि character number है कि नहीं।)

6- isxdigit() function -

Syntax- isxdigit(ch) where ch is character variable

```
/* isxdigit() function */
/* Test if a character is hexa-decimal Number */
#include<stdio.h>
#include<conio.h>
#include<ctype.h>
void main()
{ clrscr();
  Char ch;
  printf("Enter any character: ");
  scanf("%c", &ch);
  if(isxdigit(ch))
  {
    printf("\n Valid character\n");
  }
  else
  {
    printf("\n Invalid character\n");
  }
  getch();
}
```

Output- (i) Enter any character: 4
Valid character

(ii) Enter any character: !
Invalid character

(यें Check करता है कि Character hexadecimal है की नहीं)

character variable

9. isspace() function -

Syntax - isspace(ch) where, ch is character variable

```
/* isspace() function */
/*test if a character is a space */
#include<stdio.h>
#include<conio.h>
#include<ctype.h>
void main()
{ clrscr();
  char ch;
  printf("%c", &ch); \nEntered character is space");
  printf("Enter any character: ");
  scanf("%c", &ch);
  if(isspace(ch))
  {
    printf("\n Entered character is space");
  }
  else
  {
    printf("\n Entered character is not space");
  }
  getch();
}
```

Output - (i) Enter any character: Ab
Entered character is not space

(ii) Enter any character:
Entered character is space

(यह check करता है कि character space है कि नहीं।)

10- ispunct() function -

Syntax - ispunct(ch) where ch is character variable

```
/* ispunct() function */
/* Test if a character is a punctuation character */
#include <stdio.h>
#include <conio.h>
void main()
{ clrscr();
  char ch;
  printf("Enter any character:");
  scanf("%c", &ch);
  if (ispunct(ch))
  {
    printf("\nCharacter is a punctuation character");
  }
  else
  {
    printf("\n Character is not a punctuation character");
  }
  getch();
}
```

Output - (i) Enter any character: a
Character is not a punctuation character
(ii) Enter any character: 3
Character is not a punctuation character
(iii) Enter any character: $
Character is a punctuation character

(यह check करता है कि character punctuation character है कि नहीं।)

DATE
PAGE No. 14-5

7- tolower() function –

Syntax- tolower(ch) where ch is a character variable.

```
/* include to_lower() function */
/* convert upper case to lower case */
#include<stdio.h>
#include<conio.h>
#include<ctype.h>
 void main()
{     clrscr();
      char ch;
      printf("Enter any character:");
      scanf("%c", &ch);
      printf("\n Output: %c\n", tolower(ch));
      getch();
}
```

Output –

(i) Enter any character: A
Output: a

(ii) Enter any character: a
Output: a

(iii) Enter any character: 2
Output: 2

(iv) Enter any character: %
Output: %

(यदि character को lower case में convert करता है।)

8- toupper() function-

Syntax- toupper(ch)
where ch is character variable or any character. (it is valid for prototype)

```
/* toupper() function */
/* Convert lower case to upper case */
#include <stdio.h>
#include <conio.h>
#include <ctype.h>
void main()
{
  clrscr();
  char ch;
  printf("Enter any character: ");
  scanf("%c", &ch);
  printf("\nOutput: %c\n", toupper(ch));
  getch();
}
```

Output- (i) Enter any character: a
Output: A

(ii) Enter any character: A
Output: A

(iii) Enter any character: 2
Output: 2

(iv) Enter any character: %
Output: %

Function

Function are subprograms which are used to compute a value of perform a task.

Type of Function-

1- Library or Built-in function
Ex- scanf(), printf() etc.

2- User-defined functions -

Function Prototype Declaration -

Syntax- datatype functionname(type d1, type d2 ---- type dn);

Ex- int fact(int a);
int add(int a, int b);
int add(int, int);

{ Variable name ~~optional~~ लिखना जरूरी नहीं है।

Function Declaration-

Syntax- Datatype Function_name(type d1, type d2, ---- type n);
{
Local Declaration; (जरूरी नहीं)

block of statement;

return(variable or expression); (जरूरी नहीं)
}

Note- No data type = integer by default
(सभी type एक ही क्रम में हर स्थान पर लिखें)

re used to

type dn);

ble name
[illegible] लिखना
है नहीं है।)

ype n);

रती नहीं)

रती नहीं)

Ex- int fact(int n, int m)
{
int c;
c = n+m;
return(c);
}

Return Statement -

Syntax- return (variable or expression);
Ex- return (3);
return (x+y);
return (y);

{ return (2,4); ✗
return (x,y); ✓

- No argument but return integer value -
int show(void);
- Argument ~~are~~ available but return no value -
void add(int, int);
- No argument and No return value -
void add(void);
- Arguments available and return an integer value -
square(int a);
int square(int a);

DATE
PAGE No. 150

Actual and Formal Argument-

* Function call arguments are actual arguments.
* Function declaration arguments are formal arguments.

Actual → Formal.

Calling a function-

Syntax- function_name( );

Ex- add( );

```
#include<stdio.h>
#include<conio.h>
int add(int n, int m);  // Function Prototype
void main()
{
  clrscr();
int a, b, ans;
printf("\n Enter Two numbers: ");
scanf("%d %d", &a, &b);
ans = add(a,b);          // Function called
                  // a and b are actual arguments
printf("Addition: %d \n", ans);
 getch();
}
/* function declaration */
int add(int n, int m)   // n and m are formal arguments
{
   int c;
   c= m+n;
   return(c);
}
```

Output- Enter Two numbers: 5
4

Addition: 9

Category of Function calling-

1- With no arguments and with no return value
2- With no arguments and with return value
3- With arguments and with no return value
4- With arguments and with return value.

1- With no arguments and with no return value-

```
/* Function with no argument and with no return value */
#include <stdio.h>
#include <conio.h>
void massage(void);    // Function Prototype
void main();
{
  clrscr();
  printf("\n Main Function\n");
  massage(); // Function called
  printf("\n After calling function\n");
  getch();
}
/* function declaration */
void massage(void)
{
  printf("\n I am subprogram\n");
}
```

Output- Main Function

I am subprogram

After calling program

With no arguments and with return value -

```
/* Function with no arguments and with return value */
#include<stdio.h>
#include<conio.h>
int add(void);   // Function Prototype
void main()
{
   clrscr();
   int ans;
   ans=add();  // Function called
   printf("\n Addition: %d\n", ans);
   getch();
}

/* function declaration */
int add(void)
{
  int a, b, c;
  printf("\n Enter Two Numbers:");
  scanf("%d %d", &a, &b);
  c= a+b;
  return(c);
}
```

Output- Enter Two Numbers: 3
5

Addition: 8

3- With arguments and with no return value-

```
/* Function with argument and with no return value */
#include<stdio.h>
#include<conio.h>
void add(int, int); // Function prototype
void main()
{
  clrscr();
  int a, b, c;
  printf("\n Enter Two numbers: ");
  scanf("%d %d", &a, &b);
  add(a,b); // Function called
  getch();
}
/* function declaration */
void add(int n, int m)
{
  int s;
  s = n+m;
  printf("\n Addition: %d\n", s);
}
```

Output - Enter Two numbers: 3
5

Addition: 8

DATE
PAGE No. 155

4- With arguments and with return value -

```
/* Function with argument and with return value */
#include <stdio.h>
#include <conio.h>
int add(int, int); // Function prototype
void main().
{
  clrscr();
  int a, b, ans;
  printf("\n Enter Two numbers: ");
  scanf(" %d %d", &a, &b);
  ans = add(a, b); // Function called
  printf("Addition: %d\n", ans);
  getch();
}
/* function declaration */
int add(int n, int m)
{
   int s;
   s = n+m;
   return(s);
}
```

} or return(n+m);

Output- Enter Tow numbers: 3
5

Addition: 8

Local / Internal / Static Variable -

* Inside the body of a function or main program.
* These variables are active in the block in which they are declared.

Global / External variable -

* Outside the body of a function or main program.
* These variable are commonly available for the main program and function and are throughout C program.

```
/* Local Variable and Global Variable */
#include <stdio.h>
#include <conio.h>
int add (int , int);
int se = 100;          // Global variable
void main()
{
  clrscr();
  int a=10, b=5, ans;  // Local variable
  ans = add(a, b);
  printf("Addition : %d \n", ans);
  getch();
}
int add(int n, int m)
{
  return(n+m+se);
}
```

Output - 116

DATE
PAGE No. 157

Storage Classes in C -

1- Auto or Automatic Storage Class.
2- Extern or External Storage Class
3- Register Storage Class
4- Static Storage Class
<i> Static auto storage class
<ii> static external storage class

1- Auto Storage class - यह local variable के लिये स्टोरेज

यदि Variable को किसी Value नहीं दें तो यह गारबेज Value ले लेगा। auto लिखना जरूरी नहीं है।

```
/*auto Storage class*/
#include<stdio.h>
#include<conio.h>
void main()
{ clrscr
auto int a=10, b=5, ans;   or int a=10, b=5, ans;   or a, b ans;
  ans = a+b;
printf("Addition : %d", ans);
 getch();
}
```

Output- Addition : 15 Output- Addition : 15 Output- Addition : 2647

2- Extern Storage Class- यह Global variable के लिये होता है

यदि Variable को कोई Value न दी जाये तो यह उसे 0 से initialize कर देगा। extern data type variable लिख कर हम किसी program में भी Global variable का उपयोग कर सकते

```
/* Extern Storage class*/
Include <stdio.h>
Include <conio.h>
int a=10;                    or int a;
Void main()
{
 clrscr();
 int b = 2; c;
 c = a+b;
 printf("Addition: %d", c);
 getch();
}
```

Output - 12 Output - 2

3- Register Storage Class - यह लोकल Variable के लिये होता है Variable declare करने से पहले register लिखना पड़ता है। यह Register में Store करता है। यदि वहां जगह नहीं रहती यह auto में रख देगा।

```
/* register storage class*/
#include <stdio.h>
#include <conio.h>
Void main()
{ clrscr
  register int a=10, b=7, ans;
  ans = a+b;
printf("Addition: %d", a);
  getch();
}
```

Output - 17

~~Static Static Storage class-~~
~~Static Auto storage class-~~

4- Static Storage class -

(i) Static local / auto storage class- यह लोकल की value को Block खत्म होने के बाद Destroy होता है, variable नहीं होने देता है।

```
/* Static local storage class */
#include <stdio.h>
#include <conio.h>
void num(void);
void main()
{
    clrscr();
    num();
    num();
    num();
    getch();
}
void num(void)
{
static int a=1;
    printf("%d\n",a);
    a++;
}
```

Output-

1
2
3

```
int a=1;
```

Output-

1
1
1

(iii) Static Global storage class -

main.c

page 1 →

```
/* Static global storage class */
#include <stdio.h>
#include <conio.h>
static int a = 1;        or  int a=1;
void num (void);
main()
{ clrscr();
  num();
  getch();
}
```

page 2 →

```
* mu3.c
extern int a;
void num (void)
{ int b=10, ans;
  ans = a+b;
  printf("\n Addition : %d", ans);
}
```

Output - नहीं आएगा।
(Error)

Output -
Addition : 11

Static variable को ~~Access~~ पर नहीं कर सकते लेने से दूसरे program में

Recursion - A function calling itself again to compute a value is referred to recursive function or recursion.

→ /* Program to find factorial using recursion */

```
#include <stdio.h>
#include <conio.h>
int fact (int);  // function prototype
void main()
{
  clrscr();
  int n, facto;
  printf("Enter any number: ");
  scanf("%d", &n);
  facto = fact(n);
  printf("\n Factorial is %d\n", facto);
  getch();
}
/* function declaration */
int fact (int k)
{
  int f;
  if (k==1||k==0)  // Terminating Condition
  {
    return (1);
  }
  else
  {
    f = k * fact(k-1); // Recursive call
    return (f);
  }
}
```

Output-

Enter any number: 4
Factorial is 24

Noting of Function - एक function दूसरे और function को call करती है।

```
/* Noting of function */
#include <stdio.h>
#include <conio.h>
int add (int, int);   // Function Prototype
void out (void);      // Function Prototype
void main ()
{
  clrscr ();
  out ();  // function called
  getch ();
}
/* display function declaration */
void out (void)
{
   int a, b, ans;
   printf("Enter Two numbers : ");
   scanf("%d %d", &a, &b);
   ans = add (a, b);   // function called
   printf(" Addition: %d", ans);
}
/* addition function declaration */
int add (int x, int y)
{
   int c;
   c = x+y;
   return (c);
}
```

Output - Enter Two number : 3
5
Addition : 8

Passing Array as Function Argument -

(i) Function Prototype -

Syntax - Return_type function_name (datatype[], datatype);
(For array, For array size)

Ex - int add (int [], int);

(ii) Function Call -

Syntax - Function_name (array_name);
or Function_name (array_name, arraysize);

Ex - add (arr);
or add (arr, 5);

(iii) Function Declaration -

Syntax - return_type function_name (datatype array_name[], datatype size);
(array size)

Ex - int add (int arr[], int s)

Note - (i) Function call के द्वारा Array को Base address pass होता है। Ex - & arr[0]
(ii) यह Call by reference होता है।
(iii) function के अन्दर array के Value में change होने पर उसका प्रभाव main function के array की Value में भी होगा।

Ex Program

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
int add (int[], int);   // Function Prototype
void main ()
{
  clrscr();
  int arr[5], i, sum;
  printf("Enter five Numbers: \n");
  for(i=0; i<5; i++)
  {
    scanf("%d", &arr[i]);
  }
  sum = add(arr, 5);  // Function call
  printf("Addition: %d", sum);
  getch();
}
int add(int arr[], int s)  // Function declaration
{
  int i, total=0;
  for(i=0; i<s; i++)
  {
    total = total + arr[i];
  }
  return(total);
}
```

Output- Enter five Numbers: 1
2
3
4
5
Addition: 15

Structure -

Declaration of Structure -

Globaly Declare करते हैं

Method 1 -

```
struct structure_tag
{
    Data type variable 1;
    Data type variable n;
} structure_variable 1, structure_variable n;
```

Structure Members.

Ex-

```
struct student
{
    char name[20];
    int roll;
    int class;
} stu1, stu2, stun;
```

Method 2 -

```
struct structure_tag
{
    Data_type variable 1;
    Datatype variable n;
};
main()
{
struct structure_tag structure variable 1, structure_variable n;
```

Ex- struct student
{
    char name[20];
    int roll;
    int class;
};
main()
{
    struct student stu1, stu2, stu3;

Initializing Structure -

Mathod 1-

struct student
{
    char name[20];
    int roll;
    int class;
} stu1 = {"Raj", 101, 12};

Mathod 2-

struct student
{
    char name[20];
    int roll;
    int class
};
main()
{
    struct student stu1 = {"Raj", 101, 12};

DATE
PAGE No. 168

Accessing Member of Structure -

Syntax- structure_variable.member

```
#include <stdio.h>
#include <conio.h>
struct student
{
    char name[20];
    int roll;
} stu = {"Raj", 101}; or };
void main()
{
  clrscr();
                    or struct student stu = {"Raj", 101};

  printf("Student Name : %s", stu.name);
  printf("\n Student Roll No. : %d\n", stu.roll);
  getch();
}
```

or stu.roll;

Output- Student Name : Raj
Student Roll No. : 101

Same output Same output

```
#include <stdio.h>
#include <conio.h>
struct student
{
   char name[20];
   int roll;
};
void main()
{
 clrscr();
 struct student stu;
 printf("Enter Name: ");
 scanf("%s", &stu.name);
 printf("Enter Roll No.: ");
 scanf("%d", &stu.roll);
 printf("\n Student Name: %s", stu.name);
 printf("\n Student Roll No.: %d", stu.roll);
 getch();
}
```

Output -

```
Enter Name: Raj
Enter Roll No.: 101
Student Name: Raj
Student Roll No.: 101
```

DATE
PAGE No. 170

Array within Structure –

```
#include <stdio.h>
#include <conio.h>
struct student
{
  char name[20];
  int mark[5];
};
void main()
{
  clrscr();
  struct student stu;
  int i;
  printf("Enter Name:");
  scanf("%s", &stu.name);
  printf("Enter Roll No.:");
  scanf("%d", &stu.roll);
  printf("Enter 5 Subject Marks:\n");
  for(i=0; i<5; i++)
  {
    scanf("%d", &stu.mark[i]);
  }
  printf("\n Student Name: %s", stu.name);
  printf("\n Student Roll No.: %d\n", stu.roll);
  printf("Marks:");
  for(i=0; i<5; i++)
  {
    printf("%5d", stu.mark[i]);
  }
  printf("\n\n");
getch();
}
```

Output

```
Enter Name: Raj
Enter Roll No.: 101
Enter 5 Subject Marks:
1
2
3
4
5
Student Name: Raj
Student Roll No.:
Marks: 1 2 3 4 5
```

Array of Structure -

```
#include <stdio.h>
#include <conio.h>
struct student
{
  char name[20];
  int roll;
};
void main()
{ clrscr();
  struct student stu[3];
   int i;
  for(i=0; i<3; i++)
  {
    printf("Enter Name: ");
    scanf("%s", &stu[i].name);
    printf("Enter Roll No.: ");
    scanf("%d", &stu[i].roll);
  }
  for(i=0; i<3; i++)
  {
    printf("\n Student Name: %s", stu[i].name);
    printf("\n Student Roll No.: %d\n", stu[i].roll);
  }
getch();
}
```

Output -

```
Enter Name: Raj
Enter Roll No: 101
Enter Name: Rani
Enter Roll No.: 102
Enter Name: Sam
Enter Roll No.: 103

Student Name: Raj
Student Roll No.: 101
Student Name: Rani
Student Roll No.: 102
Student Name: Sam
Student Roll No.: 103
```

Array within Structure and Array of Structure

Bingo Combination —

```
#include<stdio.h>
#include<conio.h>
struct student
{
 char name[20];
 int roll;
 int mark[5];
};
void main()
{
 clrscr();
 struct student stu[3];
 int i,j;
 for(i=0; i<3; i++)
 {
  printf("Enter Name: ");
  scanf("%s", &stu[i].name);
  printf("Enter Roll No.: ");
  scanf("%d", &stu[i].roll);
  printf("Enter 5 subject Marks:\n");
  for(j=0; j<5; j++)
  {
   scanf("%d", &stu[i].mark[j]);
  }
 }
 for(i=0; i<3; i++)
 {
  printf("\n Student Name: %s", stu[i].name);
```

```
printf("\n Student Roll No.: %d\n", stu[i].roll);
printf(" Marks :\n");
for (j=0; j<5; j++)
{
printf("%5d", stu[i].mark[j]);
}
getch();
}
```

Output- Enter Name: Satyam
Enter Roll No.: 101
Enter 5 Subject Marks:
99
99
99
99
99
Enter Name: Seth
Enter Roll No.: 102
Enter 5 Subject Marks:
99
98
97
96
97
Enter Name: Soni
Enter Roll No.: 103
Enter 5 Subject Marks:
97
96
98
99
95

Student Name: Satyam
Student Roll No.: 101
Marks:
99
99
99
99
99
Student Name: Seth
Student Roll No.: 102
Marks:
99
98
97
96
97
Student Name: Soni
Student Roll No.: 103
Marks:
97
96
98
99
98

Nesting of Structure -

```
#include <stdio.h>
#include <conio.h>
struct student
{
   char name[20];
   int roll;
   struct student1
   {
      int tm;
   } instu;
};
void main()
{
   clrscr();
   struct student outstu;
   printf("Enter Name: ");
   scanf("%s", &outstu.name);
   printf("Enter Roll No.: ");
   scanf("%d", &outstu.roll);
   printf("Enter Total Marks: ");
   scanf("%d", &outstu.instu.tm);          // This is syntax
   printf("\n Student Name : %s", outstu.name);
   printf("\n Student Roll No.: %d\n", outstu.roll);
   printf("Total Marks: %d\n", outstu.instu.tm);
   getch();
}
```

Output - Enter Name : Satyam
Enter Roll No.: 5
Total Marks: 100
Student Name : Satyam
Student Roll No.: 5
Total Marks : 100

Passing Structure to function -

(ii) Function Prototype -

Syntax- return_type function_name (struct structure_tag &structure_variable);

Ex- Void disp (struct student &stu);

(iii) Function Call -

Syntax- Function_name (function_variable);

Ex- disp (stu);

(iii) Function Declaration -

Syntax- return_type function_name (Struct structure_tag structure_variable)

Ex- void disp (struct stu)

Note- (i) चाहे तो एक ही Variable का उपयोग Function Prototype, Function Call, Function Declaration में कर सकते हैं।

(ii) Function prototype से पहले Structure Define करें।

→
```
#include<stdio.h>
#include<conio.h>
struct student
{
    char name[100];
    int roll;
};
void disp(struct student stu);
void main()
{
  clrscr();
  struct student st;
  printf("Student's Name: ");
  scanf("%s", &st.name);
  printf("Roll No.:");
  scanf("%d", &st.roll);
  disp(st); // passing structure variable st as argument
  getch();
}
void disp(struct student stu)
{
  printf("\nName: %s", stu.name);
  printf("\nRoll: %d\n", stu.roll);
}
```

Output-

```
Student's Name: Raj
Roll No.: 101

Name: Raj
Roll: 101
```

DATE
PAGE No. 177

Union-

Declaration of Union-

Method 1- union union_tag
{
Data_type variable 1;
Data type variable 2;
} union_variable 1, union variable 2;

Ex- union student
{
char name[20];
int roll;
int class;
} stu1, stu2, stun;

Method 2- union union_tag
{
Data type variable 1;
Data type variable 2;
};

main()
{
union union_tag union variable 1, union_variable n;

Ex- union student
{
char name[20];
int roll;
int class;
};

main()
{
union student stu1, stu2, stun;

→
```
#include <stdio.h>
#include <conio.h>
union student
{
    char name[20];
    int roll;
}
void main()
{
    clrscr();
    union student stu = {"Raj"};
    stu.roll = 101;
    printf("Student Name : %s", stu.name);
    printf("\n Student Roll No: %d \n", stu.roll);
    getch();
}
```

or ↓

Output - Student Name : Raj
Student Roll No.: 101

Note - Structure में Memory सभी Member के Size के Addition के बराबर Allocate होती है।
Union में Memory सबसे बड़े Size वाले Member के Size के बराबर Allocate होती है।

Difference between Structure and Union -

```
#include<stdio.h>
#include<conio.h>
struct student
{
   char name[20];
   short int id;
} stu;
union person
{
 char name[20];
 short int roll;
} u;
void main()
{ clrscr()
printf("\n Size of Structure: %d\n", sizeof(stu));
printf("Size of Structure Member Name: %d\n", sizeof(stu.name));
printf("Size of Structure Member ID: %d\n", sizeof(stu.id));
printf("Size of ~~Structure~~ Union: %d\n", sizeof(u));
printf("Size of Union Member Name: %d\n", sizeof(u.name));
printf("Size of Union Member Roll: %d\n", sizeof(u.roll));
 getch();
}
```

Output -

Size of Structure: 22
Size of Structure Member Name: 20
Size of Structure Member ID: 2
Size of ~~Structure~~ Union: 20
Size of Union Member Name: 20
Size of Union Member Roll: 2

DATE
PAGE No. 180

Pointer- Pointer holds an address than a value

Syntax- data_type *pointer_name;

Ex-

```
int a = 100;
int *p;
p = &a;
```

How to it Work-

int a = 100

a ← Variable name

100 ← Variable value

2653 ← Variable Address

int a = 100

int *p;

p = &a;

p ← Pointer Variable name

2653 ← ~~Pointer Variable~~ Address of Variable

6743 ← Pointer Variable Address.

```
#include <stdio.h>
#include <conio.h>
void main()
{
clrscr();
int a=100;
int *p;
p=&a;
printf"\n Value of a          : %d", a);
printf"\n Address of a        : %d", &a);
printf"\n Address of a using pointer: %d", p);
printf"\n Value of a using pointer: %d", *p);
printf"\n Address of pointer  : %d", &p);
getch();
}
```

Output:-

```
Value of a                  : 100
Address of a                : 2686748
Address of a using pointer: 2686748
Value of a using pointer  : 100
Address of pointer          : 2686744
```

Change Value of Variable using Pointer

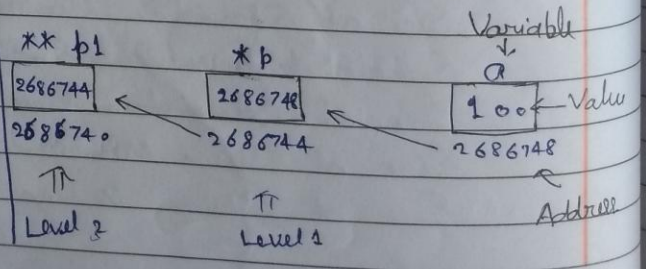
```
#include <stdio.h>
#include <conio.h>
void main()
{
  clrscr();
  int a = 100;
  int *p;
  p = &a;
  *p = 50;
  printf("\n Value of a          : %d", a);
  printf("\n Address of a        : %d", &a);
  printf("\n Address of a using pointer: %d", p);
  printf("\n Value of a using pointer : %d", *p);
  printf("\n Address of pointer  : %d", &p);
  getch();
}
```

Output -

Value of a : 50
Address of a : 2686748
Address of a using pointer : 2686748
Value of a using pointer : 50
Address of pointer : 2686744

Pointer to Pointer -

```
int a = 100;
int *p;
int **p1;
p = &a;
p1 = &p
```



```
#include <stdio.h>
#include <conio.h>
void main()
{
clrscr();
int a=100;
int *p;
int **p1;
p= &a;
p1= &p;
printf("\n Value of a                    :%d, a);
printf("\n Value of a using pointer p    :%d, *p);
printf("\n Value of a using pointer p1   :%d, **p1);
printf("\n Address of a                   :%u, &a);
printf("\n Address of a using pointer p   :%u, p);
printf("\n Address of a using pointer *p1 :%u, *p1);
printf("\n Address of pointer p           :%u, &p);
printf("\n Add. of point. p using point. p1 :%u, p1);
printf("\n Add. of point. **p using point. p1 :%u, &p1);
getch();
}
```

Output-

```
Value of a                              : 100
Value of a using pointer p              : 100
Value of a using pointer p1             : 100
Address of a                            : 2686748
Address of a using pointer p            : 2686748
Address of a using pointer *p1          : 2686748
Address of pointer p                    : 2686744
Add. of point. p using point. p1        : 2686744
Add. of point. **p using point. p1      : 2686740
```

Passing Pointers To Functions -

1- Pass / Call by Value Method - When arguments are passed by value a copy of the values of actual arguments is passed to calling function. Thus, any change made to the variable inside the function will have no effect on variable used in the actual argument list.

(i) Function Prototype :-
Syntax- return_type function_name(argument list);
Ex- void value (int, int);

(ii) Function Definition -
Syntax- return_type funtion_name(argument list)
{
Function body;
}
Ex- void value (int x, int y)
{
function body;
}

(iii) Function Call -
Syntax- function_name (argument list)
Ex- Value (x, y);

2-

arguments
by value
arguments
any
the
variable


```
#include <stdio.h>
#include <conio.h>
int add (int, int);
void main ()
{ clrscr();
  int a=10, b=20, sum;
  sum = add (a,b);
  printf("Addition = %d", sum);
  getch(); }
int add (int x, int y)
{
  int total;
  total = x+y;
  return (total);
}
```

Output -

Addition = 30

2- Pass/Call by address method -

When arguments are passed by address (i.e. when a pointer is passed as an argument to a function), the address of a variable is passed. The contents of that address can be accessed freely, either in the called by address can change the value of the variable used in the call.

DATE
PAGE No. 186

(i) Function Prototype -

Syntax- return type function name (argument list with pointer);

Ex- void address (int *, int *);

(ii) Function Definition -

Syntax-

returntype functionname (argument list with pointer)

{

Function body;

}

Ex- Void address (int *fx, int *fy)

{

function body;

}

(iii) Function Call -

Syntax- function_name (argument list &);

Ex- address (&x, &y);

```
#include <stdio.h>
#include <conio.h>
void age (int *, int *);
void main ();
{
clrscr ();
int rani=10, amit = 20;
age (&rani, &amit);
printf("Rani's Age : %d", rani);
printf("\n Amit's Age : %d", amit);
getch ();
}
void age (int *r, int *a)
{
   *r=20;
   *a = 10;
}
```

Output -

Rani's Age: 20
Amit's Age: 10

A Function returning more than one value-

Using call by address method we can make a function return more than one value at a time, which is not possible in the call by value method.

→
```
#include <stdio.h>
#include<conio.h>
void addsubl int, int, int*, int*);
void main()
{
  clrscr();
  int add, sub;
  addsub(20, 10, &add, &sub);
  printf("Addition: %d", add);
  printf("\nSubtraction: %d", sub);
  getch();
}
void addsub(int x, int y, int *pa, int *ps)
{
  *pa = x+y;
  *ps = x-y;
}
```

Output- Addition: 30
Subtraction: 10

DATE
PAGE No. 189

Function Returning Pointer -

```
#include<stdio.h>
#include<conio.h>
float *fun(); /* function prototype */
int main()
{
 float *a;
 a= fun(); // a= &r
 printf("Address =%u ", a);
 getch();
}
float *fun()
{
 float r= 5.2;
 return (&r);
}
```

Output -

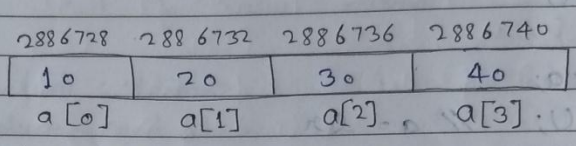
Address = 2686708

DATE
PAGE No. 190

Array of Pointer

```
int a[] = {10, 20, 30, 40};

int *p[3]; // *p[0], *p[1], *p[2], *p[3]
```



→
```
#include <stdio.h>
#include <conio.h>
void main()
{
 clrscr();
 int a[] = {10, 20, 30, 40};
 int *p[4];
 int i;
 printf("Output via pointer:\n");
 printf("Address\t Value:");
 for(i=0; i<4; i++)
 {
  p[i] = &a[i]; // p[0] = &a[0]
  printf("\n%u", p[i]);
  printf("  %d", *p[i]);
 }
 getch();
}
```

Output -

Output via pointer:

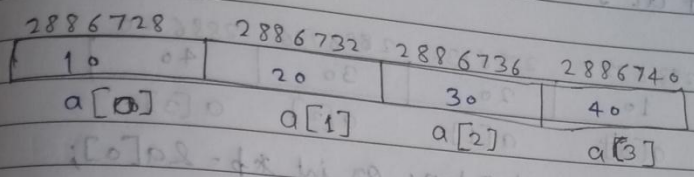
Address	Value
2686732	10
2686736	20
2686740	30
2686744	40

Kreater

DATE
PAGE No. 191

Increment and Decrement Point -

```
int a[] = {10, 20, 30, 40};
```



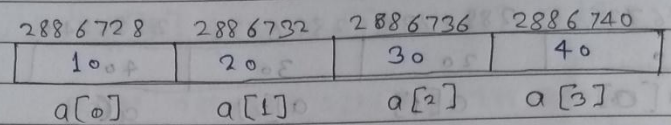

```
int *p;          // int *p=a; or *p=&a[0];
p=a;             // p=&a[0];
p++      or      p=p+1;
p+=2     or      p=p+2;
#include<stdio.h>
#include<conio.h>
void main()
{ clrscr();
  int a[]={10,20,30,40};
  int *p;
  p=&a[0];
  int i;
  printf("Output via Pointer:\n");
  printf("Address  Value");
  for(i=0; i<4; i++)
  { printf("\n%u", p);
    printf("  %d", *p);
    p++;
  }
getch();
}
```

Output - Output via Pointer:

Address	Value
2686728	10
2686732	20
2686736	30
2686740	40

Arrays And Pointers -

```
int a[] = {10,20,30,40};
```




```
int *p; // int *p=a; or int *p = &a[0];
p = a; // p = &a[0];
```

→
```
#include <stdio.h>
#include <conio.h>
void main()
{  clrscr();
   int a[] = {10,20,30,40};
   int i;
   int *p;     // int *p = a; or int *p = a;
   p = &a[0];  // p = a;
   printf("Address \t Value");
   for (i=0; i<4; i++)
   {
    printf("\n%u \t %d", p, *p);
   } p++;

   getch();
}
```

Output-

Address	Value
2686728	10
2686732	20
2686736	30
2686740	40