Development of "C" (Introduction and history)

"C" is a programming language developed at AT & T Bell Laboratories of USA in 1972. It was developed Dennis Ritche in late 1970's. it began to replace the more familiar languages of that time like PL/1, ALGOL etc.

1. "C" became popular because of its reliability, simple and easy to use
2. It was friendly capable and reliable
3. ALGOL 60 was developed and did not become popular because it was too general and too abstract.
4. They developed "CPU" (Combined Programming Language)
5. Next as it could not come up to make ALGOL 60 better one they moved to "BCPL" (Basic Combines Programming Language. Developed by martin Richard Cambridge university)
6. At the same time a language called "B" written by ken Thompson at AT & T'S. Bell laboaratories as a further simplification of BCPL.
7. "C" s compactness and coherence is mainly due to it's one man language. Ex-LISP, AASCA

Year	Lang	Developed by	Remarks
1960	ALGOL	International Committe	too general, too abstract
1963	CPL	camebridge university	Hard to Learn & implementation
1967	BCPL	Camebridge university	could deal only special problem
1970	B	AT&T	could deal only special problem
1972	C	AT & T	Lost Generality of BCPL, B restored

Note : C is a middle level language because it was due to have both a relatively good

programming efficiency and relativity good machine effecience.

Features of "C" Language :

1. It is robust language because of rich set of binary in – function
2. It is efficient and fast because of its variant data-types and powerful operation.
3. It is highly Portable i.e., programs written in one computer can be run on another
4. It is well suited for structure program, thus allows the user to think about the problem in the terms of functional blocks.
5. Debugging, testing and maintenance is easy
6. ability to extend itself, we can continuously add our own functions to the program.

Complier : This reads the entire source program and converts it to the object code. It provides error not of one line, but errors of the entire program. It executes as a whole and it is fast

Interpreter : It reads only one line of a source program at a time and converts it into an object code. In case of errors/same will be indicated instantly. It executes line by line and it is slow.

Linker is a function which links up the files that an present in the operating system, it also links the files for the hardware and makes the system ready for executing.

Preprocessor : This is a program, that processes the source program before it is passed on to the compiler. The program typed in the editor is the source code to the preprocessor, then it passed the source code to the compiler. It is not necessary to write program with preprocessor & activity

Preprocessor directories are always initialized at the beginning of the program. it begins with the symbol (#) hash. It place before the main() function

Eg: # include <station>

define PI 3.14

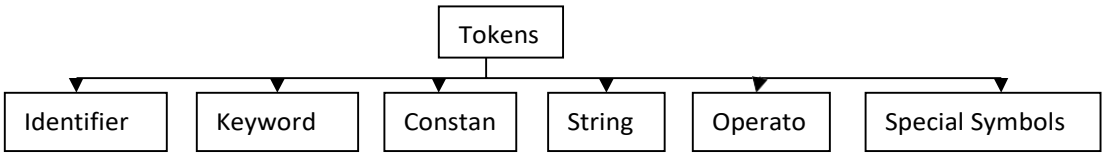
Character Set : The characters that can be used to form words and expressions depends upon the computer to which the program is run

The Characters in C are

1. Letters A-X, a-z, both upper and lower
2. Digits 0-9
3. Special character, +,-,*,",;,./,
4. which spaces newline, horizontal tab, carriage return ,blank space

"C" Tokens :

Individual words and punctuation marks are characters. In a "C" program the smallest individual units are known as "C" tokens. It has 6 types of token's.



Keywords : Keywords are reserved words by compiler. Keywords are assigned with fixed meaning and they cannot be used as variable name. No header file is needed to include the keywords.

There are 32 keywords

Eg : auto, break, double, int, float

Identifiers :

These are the names of variables ,functions and arrays, these are the user defined names

Eg : # define NUM 10
define A 20
"NUM", "A" are user – defined id

Constants : constants in "C" are applicable to the values which not change during the execution of a program.

Integer Constants : Sequence of numberr 0-9 without decimal points, fractional part or any other symbols. It requires two or four bytes, can be +ve, -ve or Zero the number without a sign is as positive.

Eg: -10, +20, 40

Real Constants : Real constants are often known as floating constants.

Eg: 2.5, 5.521, 3.14 etc.

Character Constants :

1. Single character const : A single character constants are given within a pair of single quote mark.
   Eg : ‘a’, ‘8’ , etc.

String Constant : These are the sequence of character within double quote marks

Eg : “Straight” , “India”, “4”

Variables : This is a data name used for storing a data, its value may be changed during the execution. The variables value keep’s changing during the execution of the program

Eg : height, average, sum, etc.

Rules for Defining Variable name :

1. First character must be an alphabet.
2. Must consist of only letters, digits or underscore.
3. Can not use keyword.
4. Must not contain white space.

Data types :

1. C language is rich in its data types. The variety of data types available allows the programmer to select the type appropriate to the needs to the application as well as the machine.
   ANSI C supports four classes of data types:
1. Primary data type. Example, Integral type, Floating point type.
2. User-defined data type. Example, Main.
3. Derived data type. Example, arrays, structures, functions.
4. Empty data set.

All C compilers support four primary data types, namely (int), character (char), floating point (float) and double-precision floating point double.

The primary data types in C are tabulated as follows:

PRIMARY DATA TYPES		
INTEGRAL TYPE		
INTEGER		CHARACTER
SIGNED TYPE	UNSIGNED TYPE	Signed char
int	Unsigned int	Unsigned char
Short int	Unsigned short int	
Long int	Unsigned long int	

FLOATIING DATA TYPE		
Float	Double	Long Double

Size and Range of Basic Data Types	
DATA TYPE	RANGE OF VALUES
char	–128 to 127
int	–32,768 to 32,767
float	3.4 e –38 to 3.4 e +38
double	1.7 e –308 to 1.7 e +308

Sl no	Type	Size (bytes)	Range
1	Char or signed char	1	–128 to 127
2	Unsigned char	1	0 to 25
3	Int or signed int	2	–32769 to 32767
4	Unsigned int	2	0 to 65535
5	Short int or signed short int	1	–128 to 127
6	Unsigned short int	1	0 to 255
7	Long int or signed long int	4	–2147483648 to 2147483647
8	Unsigned long int	4	0 to 4294967295
9	Float	4	3.4E–38 to 3.4E38
10	Double	8	1.7E–308 to 1.7E+308
11	Long double	10	3.4E–4932 to 1.1E+432

Character Data type : Character are usually stored in 8 bits

Void data type : A void type has no value this is usually used to specify the return type of function , this function does not return any value to calling function

Operators

Operator: An operator is a symbol that tells the Computer to perform certain mathematical or logical manipulations.

Expression: An expression is a sequence of operands and operators that reduces to single value

Eg: 10+25 is an expression whose value is 35

C operators can be classified into a no. of categories.

They include:

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

Arithmetic Operators: C provides all the basic arithmetic operators, they are +, -, *, /, % Integer division truncates any fractional part. The modulo division produces the remainder of an integer division.

Eg: a + b a – b a * b

-a * b a / b a % b

Here ‘a’ and ‘b’ are variables and are known as operands. % cannot be used for floating point data. C does not have an operator for exponentiation.

Integer Arithmetic: When the operands in an expression are integers then the expression is an integer expression and the operation is called integer arithmetic. This always yields an integer value. For Eg. a = 14 and n = 4 then

a - b = 10 Note : During modulo division,the

a + b = 18 sign of the result is always the sign

a * b = 56 of the first operand (the dividend)

a / b = 3 - 14 % 3 = -2

a % b = 2 -14 % - 3 = 2

14 % -3 = 2

1). Write a program to illustrate the use of all Arithmetic operator

```
main ( )
{
        int sum, prod , sub, div, mod, a, b ;
        printf("Enter values of a, b :") ;
        scanf(" /.d %d", & a, & b) ;
                sum = a+b ;
               printf("sum = %d", sum);
                        sub = a-b;
        printf("sub = %d",  sub);
                 prod = a * b ;
        printf("prod = %d", a* b);
                 div = a/b;
        printf(" Div = %d", div);
                 mod = a % b ;
        printf(" mod = %d",a % b);
}
```

2). WAP to convert given no. of days into years, months days

3). WAP to use various relational operators and display their return values.

```
main ( )
{
        printf(" in condition :        Return Values In");
        printf(" In 10! = 10           :      %5d", 10! = 10);
        printf(" In 10 = 10            :      %5d" , 10 == 10);
        printf(" In 10>=10             :      %5d", 10>=10);
        printf(" In 10<+100            :      %5d", 10<100);
        printf(" In 10! = 9            :      %5d", 10!=9);
}
```

Real Arithmetic / Floating Pont Arithmetic:

Floating Point Arithmetic involves only real operands of decimal or exponential notation. If x, y & z are floats, then

x = 6.0/7.0 = 0.857143

y = -1.0/3.0 = 0.333333

z = 3.0/2.0 = 1.500000

% cannot be used with real operands

Mixed mode Arithmetic: When one of the operands is real and the other is integer the expression is a mixed mode arithmetic expression.

Eg: 15/10.0 = 1.500000

15/10 = 1

10/15 = 0

-10.0/15 = -0.666667

Relational Operator: These are the operators used to Compare arithmetic, logical and character expressions.the value of a relational express is either one or zero .it is 1 if one is the specified relation is true and zero if it is false For eg:

a>b a>=b

The relational operators in C are

Operator	**Meaning**
<	is less than
< =	is less than or equal to
>	is greater than or equal to
> =	is greater than or equal to
= =	is equal to
! =	is not equal to

O/P

Condition	:	Return values
10! = 10	:	0
10 = = 10	:	1
10> = 10	:	1
10! = 9	:	1

4). WAP to illustrate the use of Logical Operators

```
void main ( )
 {    clrscr ( );
      printf("In    5>3 && 5<10        :    %3d", 5>3&&5<10);
      printf(" In   8<5 || 5= =5       :    % 3d", 8<5 || 5= =5);
      printf("In    !(8 = =8)          :    %3d", !(8= =8) ;
 }
```

O/P 5>3 && 5<10 : 1

8<5 || 5= =5 : 1

!(8 = =8) : 0

5). WAP to show the effect of increment and decrement operators

```
main ( )
{
        int     x = 10,         y = 20,         z, a ;
        z= x * y ++;
        a = x * y ;
        printf(" %d %d\n", z,a);
        z = x * ++y;
        a = x * y;
        printf(" %d %d\n", z, a);
        printf(" ++ x = %d, x++=%d", ++x, x++);
}
```

O/P 200 210

220 220

12 10

Logical operator : Logical Operators are used when we want to test more than one condition and make decisions. here the operands can be constants, variables and expressions

Logical operators are &&, ||, !

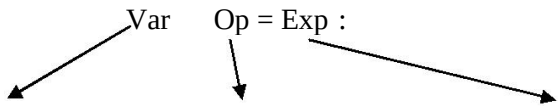
Eg:a > b && x = = 10

Logical or compound relational Expression

Truth Table && ||, !

OP1	OP2	OPJ&OP2	OP1 ||OP2	OP	!
1	1	1	1	0	1
1	0	0	1	1	0
0	1	0	1		
0	0	0	0		

Assignment Operator: Used to assign the result of an expression to a variable. '= 'is the assignment operator. In addition C has a set of 'short hand' assignment operators of the form



Variable operator shorthand assignment

Binary arithmetic operator

var op = exp;

is equivalent to

var = var op exp;

Eg: x + = 1 == > x = x+1

x+ = y+1 == > x = x+y+1

6) WAP to print whether a given number is even or odd

```
main()
{
    int a, b
    printf(" Enter a number ");
    scanf(" %d", &a);
        b = a%z;
    ((b = =o)? printf("Even"): printf("odd");
}
```

7) WAP to print logic 1 if input character is capital otherwise o

```
main ( )
{
    char x ;
    int y;
    printf((" \n Enter a character" );
```

```
        scanf(" %C ", &x);
        y = (x>=65 && x <=90)?
        1:0); printf(" y : %d", y);
}
```

O/P

1) Enter a character : A

2) Enter a character : a

y : o

Shorthand operator	Assignment operator
a + = 1	a = a+1
a - = 1	a=a-1
a * = n+1	a = a* (n + 1)
a / = n+1	a = a/(n+1)
a % = b	a = a % b

Increment and Decrement Operators:

++ and - -

The Operator + + adds 1 to the operand while -- subtracts 1, Both are unary operators

Eg : ++x or x ++ == > x+=1 == > x=x+1

. -- x or x- - == > x-=1 == > x=x-1

A Profix operator first adds 1 to the operand and then the result is assigned to the variable on left. A postfix operator first assigns the value to the variable on the left and the increments the operand.

Eg: 1) m = 5; 2). m = 5

y = ++m; y = m++

O/P m =6, y=6 m=6, y=5

Conditional operator: is used to check a condition and Select a Value depending on the Value of the condition.

Variable = (condition)? Value 1 : Value 2:

If the Value of the condition is true then Value 1 is e valued assigned to the varable, otherwise Value2.

Eg: big = (a>b)? a:b;

This exxp is equal to

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

Bitwise operator : are used to perform operations at binary level i. e. bitwise. these operators are used for testing the bits, or Shifting them right or left . These operators are not applicable to float or double. Following are the Bitwise operators with their meanings.

Operator	Meaning
&	Bitwise AND
|	Bitwise OR
^	Bitwise Exclusive – OR
<<	Left Shift
>>	Right Shift
~	Complement

Eg' consider a = 13 & b = 6 as 8 bit short int (1byte)

<< Left Shift

a = 13 Binary 00001101

b = 6 00000110

Consider a << 2 which Shifts two bits to left , that is 2 zeros are inserted at the right and two bits at the left are moved out.

00001101

Moved

00110100

Finally the result is 00110100 . Deci 52 (13x4)

Note : when you shift a bit towards left its Decimal Value is multiplied by Two (2).

a =13 13= 00001101

a >>2 00000011

000000 11 Decimal 3 (13/4)

~ Complement

convert 0's & to 1's and 1's to O's .

& (Bitwise logical and)

a = 13 0000 1101

b = 6 0000 0110

a & b 0000 0100

op1	op2	&
0	0	0
0	1	0
1	0	0
1	1	1

| (Bitwise Logical or)

a = 13 0000 1101

b = 6 0000 0110

a 1b 0000 1111

^ (Bitwise Exclusive or)

a = 13 0000 1101

op1	op2	^
0	0	0
0	1	0

b = 6	0000 0110	1	0	1
a ^ b	0000 1011	1	1	0

Special operators : these are two other operators in c.

sizeof operator : is used to find the on. of bytes occupied by a variable / data type in computer memory.

eg : sizeof (float) returns 4

int m, x [50]

sizeof (m) returns 2

sizeof (x) returns 100 (50 x 2)

9) WAP to illustrate the use of size of operator

```
main ( )
{
   int x = 2;
   float y = 2;
   printf (" in size of ( x ) is %d bytes ", sizeof ( x ));
   printf (" in size of ( y ) is %d bytes ", sizeof ( y ));
   printf (" in Address of x = % u and  y = % u ", &x, &y);
}
```

o/p sizeof (x) = 2

sizeof (y) = 4

Address of x = 4066 and y = 25096

i) **comma operator** : can be used to link the related expressions together. A comma- linked: list of expressions are evaluated left to right and the value of right-most exp is the value of combined expression.

Eg : value = (x = 10, y = 5, x = y)

First 10 is assigned to x

then 5 is assigned to y

finally x + y i .e. which 15 is assigned to value .

since comma has the lowest precedence of all operator, the parantheses are necessary .

Operator - precedence & Associativity

precedence is nothing but priority that indicates which operator has to be evaluated first when there are more than one operator.

Associativity : when there are more than one operator with same precedence [priority] then we consider associativity , which indicated the order in' which the expression has to be evaluated. It may be either from Left to Right or Right to Left.

eg : 5 * 4 + 10 / 2

1 2

= 20 + 5

3

=25

Type Casting: Normally before an operation takes pace both the operands must have the same type. C converts One or both the operands to the appropriate date types by "Type conversion". This can be achieved in 3 ways.

Implicit Type conversion : In this the data type /Variable of lower type (which holds lower range of values or has lower precision) is converted to a higher type (which holds higher range of values or has high precision). This type of conversion is also called "promotion".

If a char is converted into int it is called as Internal promotion

Eg: int I;

char C;

C = "A";

I = C;

Now the int Variable I holds the ASCII code of the char ‘A

a) An arithmetic operation between an integer and integer yields an integer result.

b) Operation b/w a real yields a real

c) Operation b/w a real & an integer always yields a real result

Eg:5/2 = 2	2/5 = 0
5.0/2 = 2.5	2.0/50. = 0.4
5/2.0 = 2.5	2/5.0 = 0.4
5.0/2.0 = 2.5	2.0/5.0 = 0.4

Assignment Type Conversion: If the two Operands in an Assignment operation are of different data types the right side Operand is automatically converted to the data type of the left side.

Eg: Let k is an int variable & a is a float variable

k= 5/2	2	k=2/5	0	a = 5/2	2.0
k=5.0/2	2	k=2.0/5	0	a = 5.0/2	2.5
k=55.0/2	2	k=2/5.0	0	a = 5/2.0	2.5
k=5.0/2.0	2	k=2.0/5.0	0	a = 2/5	0.0
				a = 2.0/5	0.4
				a = 2.0/0.5	0.4

Explicit Type Conversion: When we want to convent a type forcibly in a way that is different from automatic type conversion, we need to go for explicit type conversion.

(type name) expression;

Type name is one of the standard data type. Expression may be a constant variable Or an expression this process of conversion is called as casting a value.

Eg: x = (int) 7.5

A = (int) 21.3/(int) 4.5

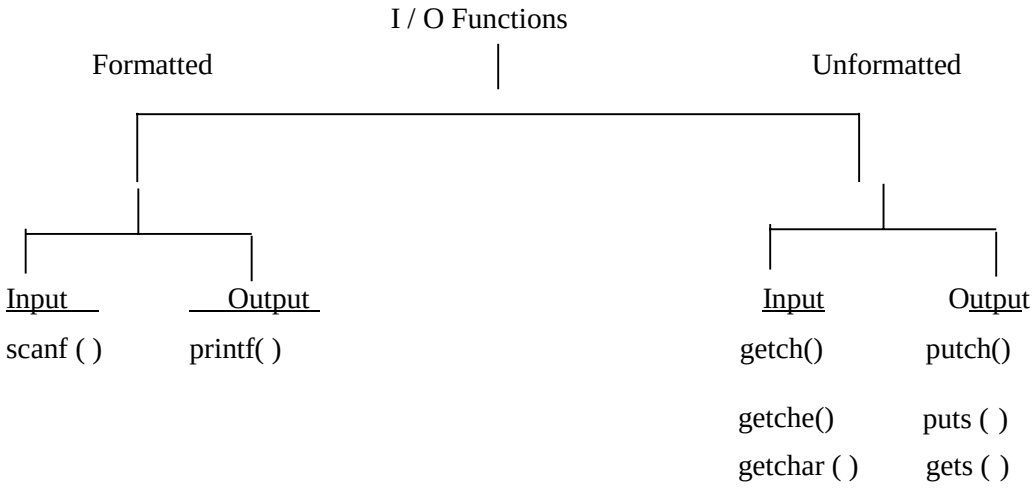
Y =(int) (a + b)

P = (double)sum/n

Basic Input output : C has many input output functions in order to read data from input devices and display the results on the screen.

Formatted Functions: These functions read and write all types of data values. They require a conversion symbol to indents the data type using these functions the O/P can be presented in an aligned manner.

Classification:



Data Types with conversion symbol (format string)

	Data Type	Conversion symbol		
Integer	Short integer	%d	or	% i
	Short unsigned		% u	
	long signed		% ld	
	Long unsigned		% lu	
	Unsigned hexadecimal		% x	
	Unsigned octal		% 0	
real	float	% f	or	% g
	double		% lf	
	signed character		%c	
	unsigned char		%c	
	string		%s	

a) scanf ( ) function is used to read values using key board. It is used for runtime assignment of variables.

The general form of scanf( ) is

scanf("format String " , list_of_addresses_of_Variables);

The format string contains

- Conversion specifications that begin with % sign

Eg: Scanf(" %d %f%c", &a,&b,&c)

'& is called the "address" operator. In scanf( ) the '& operator indicates the memory location of the variable. So that the Value read would be placed at that location.

printf( ): function is used to Print / display values of variables using monitor:

The general form of printf( ) is

printf("control String " , list_of_ Variables);

- Characters that are simply printed as they are
- Conversion specifications that begin with a % sign
- Escape sequences that begin with a '\' sign.

Eg: Program

```
main ( )
{
int avg = 346;
float per = 69.2;
printf(" Average = %d \n percentage = %f", avg, per);
}
```

O/P : Average = 346

Percentage = 69.200000

printf( ) examines the format string from left to right and prints all the characters until it encounter a '% d or '\f on the screen. When it finds % (Conversion Specifier) it picks up the first value. when it finds '\n (escape sequence) it takes appropriate action (\n-new line). This process continues till the end of format string is reached.

Eg:

```
main ( )
{
float per;
printf("Enter values for avg & per");
scanf(" %d%f", &avg, &per);
printf( " Average = %d \n Percentage = %f", avg. per);
}
```

O/P: Enter values for avg & per 346 69.2

Average = 346

Percentage = 69.200000

Unformatted functions character I/O functions

getchar ( ) function is used to read one character at a time from the key board

Syntax ch = getchar ( ); where ch is a char Var.

```
main ( )
{
char ch;
printf("Enter a char");
ch = getchar ( );
printf("ch =%c", ch);
}
```

O/P Enter a char M

M

ch = M

When this function is executed, the computer will wait for a key to be pressed and assigns the value to the variable when the "enter" key pressed.

putchar ( ): function is used to display one character at a time on the monitor.

Syntax: putchar (ch);

Ex char ch = 'M'

putchar (ch);

The Computer display the value char of variable 'ch' i.e M on the Screen.

getch ( ): function is used to read a char from a key board and does not expect the "enter" key press.

Syntax: ch = getch ( );

When this function is executed ,computer waits for a key to be pressed from the dey board. As soon as a key is pressed, the control is transferred to the nextline of the program and the value is assigned to the char variable. It is noted that the char pressed will not be display on the screen.

getche ( ): function is used to read a char from the key board without expecting the enter key to be pressed. The char read will be displayed on the monitor.

Syntax: ch = getche ( );

Note that getche ( ) is similar to getch ( ) except that getche ( ) displays the key pressed from the dey board on the monitor. In getch ( ) 'e stands for echo.

Strng I/O functions

gets ( ) function is used to read a string of characters including white spaces. Note that wite spaces in a strng cannot be read using scanf( ) with %s format specifier.

Syntax: gets (S); where 'S' is a char string variable

Ex: char S[20];

gets (S);

When this function is executed the computer waits for the string to be entered