C Language

Hardware

It is a physical component which you can see and touch.

Like CPU,mouse,monitor,keyboard,hard disk,ram etc.

Software

It is a logical component.

It is set of many programs.

Program

It is set of instructions or orders.

Computer = hardware + software

User understands---decimal system—10 Nos—(0 to 9)

Computer understands-binary system—2 Nos(0-off and 1-on)

Software—It provides the interface between user and computer. (Operating System)

Software Categories

1)System Software

These are necessary software to operate the computer.

All the operating systems like dos,windows-95, 98, 2000, 2003, XP, 7, vista, 8, 10

Unix,Linux,Sun Salaries,Mac

2)Application Software

It is a software to do any project, or application or any work easily.

Ms Office, CorelDraw, Photo Shop, Flash, Dream viewer

VB—6.0---To Develop Projects.

3)Programming Languages.

COBOL, FORTRAN, Pascal, Basic

C, C++, VC++, Java, C# (C Sharp)

C,C++

Java------Fundamentals of Java(OCJA), Core Java(OCJP),

Advance Java(JSP, Servlet, Java Beans, EJB)

JSP and Servlet-------OCEWCD

Struts, Hibernate, Spring----Framework

Dot Net----VB.NET, ASP.NET(VB & C#)-----wcf, wf, wpf, Silver light

PHP----Core PHP----Advance PHP

Framework----Joomala, Wordpress, Drupal, Magento

Android----Fundamentals of Java, Core Java---Android

Iphone--- Fundamentals of java, Core Java---Iphone

Database----

Oracle---DBA---Database Administrator

SQL Server---DBA

My SQL-----DBA

Web based Application Development---

(HTML, DHTML(CSS), Scripting Languages(Java Script, VB Script, AJAX, JQUERY)---It is common for all Technology.

4)Database Software

DBMS—Database Management System

Insert, Delete, Update, Sorting, Filtering----

Microsoft Access

Oracle

SQL Server

MY SQL

Informix

Latest Technology

Big Data and Hadoop

Cloud Computing

IOT—Internet Of Things

Programming Languages

C, C++, Java, C#

Syntax---Rules and Regulations and Keywords or Reserve Words

Computer—Binary Language

Translator

1)Compiler

2)Interpreter

3)Assembler

1)Compiler

Source Code------------Compile-----------Machine Code

First.c--------------------Compile-----------First.obj(Object Code)

Source Code---Your Program written in any Language

It converts whole source code into machine code and then display the list of errors.

2)Interpreter

Source code---------------Interpreter------------Machine Code

It converts line by line and also execute or run it.

3)Assembler

Assembly Language-------Assembler-----------Machine Code

(Micro Processor Language)

Programming Languages

1)High Level

It is similar to the English language.

Like C, C++, Java, C#, Python

2)Middle Level(Op Code)

Assembly Language

3)Low Level (0 and 1)

Binary Language

C Language

History of C Language

Unix---Operating System------1965------Not Portable

Dennis Ritchie----1972-----c Language Develop

1972---1982-----Unix----Developed in C Language(50 to 60%)

Features of C Language

Procedure Oriented Language or

Structured Programming Language or

Modular Programming Language

Simple Program of C

```
#include<stdio.h>
#include<conio.h>
void  main()
{
        clrscr();
        printf("Hello");
        getch();
}
```

Output-----Hello

#include<Header File Name>---It is a Pre Processor Directive.

To include any Header File in your program.

Other Pre Processor Directive in C Language

#if

#define

Header file-----Predefined Functions or In Built Functions

stdio.h----Standard Input/Output Header File

printf(), scanf()------stdio.h

Standard input---Keyboard

Standard Output---Monitor or Screen

conio.h----Console(Screen) Input Output Header File

clrscr(), getch()----conio.h

```
void main()
{
//program
}
```

Functions are of two types

1)In Built or Library Functions

Like printf, scanf, clrscr, getch

2)User Defined Functions

main is a user defined function—It is a starting point of program execution

clrscr()----To clear the screen.

printf("any message")----To print any message on screen.

getch()—It stops the program execution until any key is pressed.

C Programs

Editor for Program

Simple Text Editor---Notepad, Word Pad

Write+Save with .c extension===First.c

Special Editor

Turbo C 3.0

Write+Save+Compile+Run+Output

Compile---Alt + f9

Run---------Ctr + f9

Output--- Alt + f5

Escape Character or Backslash Character

'\n'------New Line Character

'\t'------Tab Character

Comments in C Language

1)Single Line Comments

//comments

2)Multi Line Comments

```
/*------------------------
        --------------------------------
        ----------------------
```

*/

Data Types in C Language

int---Integer-----To store whole no----Range(-32768 to 32767)
Format Specifier-------%d---Memory—(2 Byte)
float---To store Floating Point Value or Decimal Point Value
Format Specifier----%f, Memory--(4 Byte)
char—Character----To store only single character
Format Specifier---%c---Memory---(1 Byte)
You can store character like 'a' to 'z' or 'A' to 'Z' or '#','@'
'0' to '9'
long int—It is an extension of Integer
Memory---(4 Byte), Format Specifier---%ld
double---It is extension of Float
Memory(8 Byte)—Higher Precision Values
Format Specifier---%lf
unsigned int----Only Positive Value----0 to 64000
Format Specifier---%u, Memory(2 Byte)

Variable

It is a name of memory location.
Ram----Temporary Storage Device

Hexadecimal No System(16 nos—0 to 9 and a to f)---Address

Variable Declaration

Data Type Variable Name;

Data Type –int, float, char, long int, double etc.

Variable Name---Rules for Variable Name

1)It must starts with alphabet or underscore.

2)Space is not allowed.

3)It should be meaningful and short name.

4)It may be a alphanumeric---jay2910.

5)It should not be any keyword.

Variable Declaration

int maths;

Variable Initialization

maths=57;----Assignment Operator

maths=67;

maths=78;

Declaration and Initialization

int maths=57;

Multiple Variable Declaration and Initialization

int maths=67, sci=78, eng=89;

All the variables must be declared at the starting point of main function before the clrscr()

To print the values of any variable on screen

printf("message format specifier message",variablename);

How to take input from user

scanf("format specifier",&variabelname);

&---Address operator

&maths----Address of maths memory location

Operators in C Language

1)Arithmetic Operator

+, - ,*, /, %

int a=10, b=20;

int c;

c=a+b;---30

c=a-b--- -10

c=a*b--- 200

c=a/b--

c=a%b

Division---Quotient---int/int---Result is int

27/5------5

int/float----result is float

27/5.0----5.-----

27.0/5----5.-----

Modulus---Reminder

27%5------2

2)Comparison operator or Relational Operator

< , >, <=, >=,==, !=

int a=10,b=20

a==b	a=b
false	b is assigned to a

3) Logical Operator (Between Different Conditions)

and --- &&

or-----||

not-----!

ex or-----^

Condi1	Cond2	And &&	Or ||	Ex-or ^
False—0	0	0	0	0
0	1	0	1	1
1	0	0	1	1

1	1	1	1	0

and----All the conditions must be true----Result is True
Otherwise ---False
or—Any one condition must be true----True
ex-or----The condition must be exclusive then the result is True
not-----0---1
1---0
int a=10,b=20;
!(a==b)

Bit wise Operator (0 and 1)
1 Byte==8 Bits
00101010---1 Byte
Bit wise and----&
Bit wise or-----|
Bit wise not----!
Bit wise left shift----<<
Bit wise right shift- >>
int a=5,b=8;
int c=a & b;
c----Result---0
Increment Operator and Decrement Operator

++ **--**

int a=10;

a=a+1;-----11---a++(PostIncrement) or ++a(PreIncrement)

a=a-1------9-----a--(PostDecrement) or --a(PreDecrement)

int a=10,b=20;

int c;

1)

c = a++ + b++;

a= 11 b= 21 c= 30

2)

c = ++a + ++b;

a=11 b=21 c=32

int b=20;

```
int a=b++;-----a=20
               b=21
```

int a = ++b;

a=21

b=21

Special Operator

int l = sizeof(Data Type or Variable Name)

int l = sizeof(int);

int l = sizeof(a);

Decision making structure or Control Structure

1)Simple if

```
if(Condition)
{
	//Statements
}
--Next Statements
```

```
If-----check condition-----true---execute all statements----next
statement                  false-----next statements
```

2)if else

```
if(Condition)
{
	//statements
}
else
{
	//statements
}
Next
```

```
If---check condition---true----execute if statements---next
                             False-execute else statements--next
```

3)ladder if else---To check series of conditions or multiple conditions

```
if(Condition)
{
}
else if(Condition)
{
}
else if(Condition)
{
}
else
{
}
```

-Next

4)Nested if else

One if else within another if else

```
if(Condition)
{
        if(Condition)
        {
        }
        else
        {
        }
}
else
{
        if(Condition)
        {
        }
        else
        {
        }

}
```

Ternary Operator or Conditional Operator

Condition ? Ans1 : Ans2;

True------ans1;

False-----ans2;

Switch---To test for multiple conditions—Alternative of ladder if else-----Equality check only

You can pass only int or character expression in switch

switch(int or char)

```
{
        case value:statements;
                        break;
        case value:statements;
                        break;
        case value:statements;
                        break;
        default:statements;
                        break;
}
```

Loop Structure or Iterative Structure

To do any task repeatedly

1)do while

2)while

3)for

1)do while

It is post tested loop or exit controlled loop.

It must be executed at lest once when condition is false.

```
do
{
        Statements
        ------------------
        ------------------
}
while(Condition);
-Next
```

do------execute all statements----while—check condition

True-------execute all statements---while—check condition

False---next

2)while loop

It is pretested loop or entry controlled loop.

```
while(Condition)
{
        Statements
        ------
        -----------
}
```

while----check condition-----true----execute all statements-

False----next

3)for loop

It is pretested loop or entry controlled loop.

```
for(initialization;  condition;  expression)
```

```
{
        Statements
        -------
        --------------
}
```

For---initialization---check condition----true---execute statements

expression

false----next

1)sum of digit

sum=0;

1234------10

1234%10------4

sum = sum(0) + rem(4)-----sum=4

1234/10------123

123%10------3

sum = sum(4) + rem(3)-----sum=7

123/10-----12

12%10-----2

sum = sum(7) + rem(2)-----sum=9

12/10------1

1%10-----1

sum = sum(9) + rem(1)-----sum=10

1/10----0

2)Armstrong Number

153------(1)^3+(5)^3+(3)^3====153

3)To find the factorial of given no

5-----1*2*3*4*5==120

4)To print the multiplication table of any no

5 * 1=5

5* 2=10

5)To check the given no is prime or not

2 3 5 7 11 13 17 19

39------2 to 38---divide-----not divisible---prime

6)To print the Fibonacci series

0 1 1 2 3 5 8---------

7)To print the no between 1 to 200 that is divisible by 7 and 5

To print following pattern

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

1
1 2
1 2 3
1 2 3 4
1 2 3 4 5

1
2 2
3 3 3
4 4 4 4
5 5 5 5 5

1 2 3 4 5
  2 3 4 5
    3 4 5
      4 5
        5

 1 2 3 4 5
   1 2 3 4
```

```
1 2 3
  1 2
    1

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

```
        5
      4 5
    3 4 5
  2 3 4 5
1 2 3 4 5
```

```
5 4 3 2 1
  4 3 2 1
    3 2 1
      2 1
        1
```

```
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

break and continue

1)break---To break current loop.

Statement after break will not be executed**.**

2)continue-----To continue the current loop for next tern.

Statement after continue will not be executed.

Array

It is a collection of homogeneous data types.

It is a set of similar type of values.

To store more than one values of same data type under single variable name.

int a=10;

int b=20;

int c=30;

Array Declaration

Variable Declaration

int a;

a=78;

Array Declaration

int a[5];

a

56	78	534	34	82
0	1	2	3	4

Array Initialization

a[0]=56;---array index—starts with 0

a[1]=78;

a[2]=534;

a[3]=34;

a[4]=82;

int a[5000];

Array Declaration and Initialization

int a[ ]={10,20,30,40,50};

char name1[]={‘k’,’i’,’r’,’a’,’n’,’\0’};

char name2[]=”kiran”;

char name3[10];

jay

j	a	y	\0						

‘\0’----null character---automatic placed by compiler

Maximum no from Array

34 78 12 90 23

Max=90

Sorting of an array

Selection sort

34 78 12 90 19------12 19 34 78 90

12 78 34 90 19—first pass----minimum value

12 34 78 90 19

12 19 78 90 34---second pass---

12 19 34 90 78---third pass

12 19 34 78 90—forth pass

Two dimensional array

```
int a[][]={
            {10,20,30},
            {40,50,60},
            {70,80,90}
      };
```

3*3=9 values

Index	0	1	2
0	10	20	30
1	40	50	60
2	70	80	90

int a[3][4];---3*4==12 values

Index	0	1	2	3
0				
1			45	
2				78

a[2][3]=78;

a[1][2]=45;

Addition of two matrix

```
1 2 3          1 2 3          2    4    6
4 5 6          4 5 6          8    10   12
7 8 9          7 8 9          14   16   18
```

Multiplication of matrix

```
1 2 3          1 2 3          30   36   42
4 5 6          4 5 6
7 8 9          7 8 9
```

First row*First column----1+8+21---30

First row*Second column---2+10+24--36

First row*Third column—3+12+27--42

String----Sequence of characters enclosed in double quotes.

"shgsd nghsg sdnmgsdgbjhdsb"

"5"---string

5—int

'5'—character

string.h –Header File—Predefined Functions

1)strlen()—To find the length of string.

int l = strlen(string);

2)strupr()----To convert into uppercase.

strupr(string)-----Capital

3)strlwr()---lowercase

strlwr(string)

4)strrev()---To reverse the string.

strrev(string)

5)strcmp()---To compare two string.

int l = strcmp(str1,str2);

If(l>0)-----str1>str2

If(l<0)-----str1<str2

If(l=0)----both are equal

Comparison is based on ASCII values

American standard code for information interchange.

A-----65

B---66

a—97,b=98-------

0---48,1—49-----------

6)strcpy()---To copy one string into another string.

strcpy(str1,str2)------

Right string--------Left string(overwrite)

7)strcat()---to concatenate(add) two string.

strcat(str1,str2)

Right string------Left string(add)

Pointer

Pointer is a variable which stores the address of another variable.

int a=10;

Pointer Declaration

int *p;---Pointer Variable

p=&a;

&a=---address of a

printf("%d",*p)-----10

*p-----DE referencing pointer variable------value

printf("%u",p)-----unsigned---address of a

Pointer with an array

int a[]={10,20,30,40,50};

int *p;

p=a(array name itself is a address of first memory location) or p=&a[0]

printf("%u\t%d",p,*p)----address(address of 10) value(10)

p++----p=p+1

printf("%u\t%d",p,*p)----address(address of 20) value(20)

p++

printf("%u\t%d",p,*p)----address(address of 30) value(30)

I=0,1,2,3,4

printf("%u\t%d",p+I,*(p+i));

Pointer Arithmetic or Pointer Manipulation

Function

Two Types Of Functions

1)In Built or Predefined function or Library Function

Like printf, scanf, getch, strlen, strcmp etc.

2)User Defined Function----user has to make his own function depends upon requirement

main—user defined function

C Language------- Modular Programming

Advantages Of Function

You can make function to perform any task

Duplicate coding reduces

Program length reduce

Memory save

Time save

Any function contains four parts

1)Return Type

2)Function Name

3)Parameter List or Arguments List

4)Body Of Function

1)Return Type-----int, float,char,array,pointer,structure,void

int strlen(char array)

2)Function Name----Meaningful and Short

3)Parameter List or Argument s List

int,float,char,array,pointer ,structure

4)Body Of Function-----Coding Of Function

Four Categories Of Function

1)Without Return Type Without Parameters

void clrscr()

2)Without Return Type With Parameter

void gotoxy(int,int)

3)With Return Type Without Parameter

int getch()

4)With Return Type With Parameter

int strlen(string)

Steps to make function

1)Function Prototype or Function Declaration

2)Function Calling

3)Function Body or Function Definition

Or

1)Function Body or Function Definition

2)Function Calling

Recursion

To perform any task repeatedly

When function call itself it is called recursion.

You can pass parameter in function by two methods

1)Pass By Value or Call By Value

```
int sum(int,int)
a=10,b=20;
int c = sum(a,b);---Actual Parameter

int sum(int c,int d)----Formal Parameter
{
        int r;
        r=c+d;
        return r;

}
Actual Parameter---Copy----Formal Parameter
Operation---on ---Formal Parameter
```

2)Pass By Reference or Call By Reference

```
int a=10,b=20
c = sum(&a,&b);
int sum(int *c,int *d)----Formal Parameter
{
        int r;
        r=*c+*d;
```

```
            return r;

    }
```

Actual Parameter Address-Copy---Formal Parameter(Pointer)
Operation---on ---Actual Parameter

Function with Array

void sort(int [],int)-----Pass By Reference

Storage Classes(Types of Variables)

1)Local Variable or Automatic Variable
2)Global Variable or External variable
3)Static Variable
4)Register Variable

1)Local Variable:---The variables declared in any function.

Scope—visibility-----In the body of that function.

Life Time:---At the end of function execution.

2)Global Variable:----The variables declared outside any function before the main function.

Scope----It can be accessible in any function

Life Time---At the end of program

3)Static Variable:----The variables declared inside any function with static keyword.

Scope-----Scope of local variable

Life Time-----Life Time of Global Variable

It can be initialized only at once then there after it retains its value

Default value of static variable is 0

4)Register Variable

The variable with register keyword.

It is faster then any other variable.

Frequently access------variable value

for(i=1;i<=5000;i++)

Structure

Array---It is a collection of homogenous data types.

Structure----It is a collection of different data types

You can use structure to store related information of different data types.

Student

Rollno-----int

Name---char array

Address---char array

Phoneno---long int

Per;----float

Grade---char or char array

Declaration of Structure

```
struct structurename
{
        Information or data or variables;
};
```

Structure variable

struct structurename variablename;

structurevariable.member

Structure Pointer

struct student *s1;

s1.rollno---------------s1->rollno

Array of Structure and Array in Structure

Nested Structure

One structure within another structure

Function with Structure

File I/O-----File Input and Output

File Input-----To read data from file

File Output-----To write data into file

You can use file i/o to store data permanently.

Data Read

1)Open

2)Read

3)Close

Data Write

1)Open

2)Write

3)Close

To open File for Reading and Writing

FILE *p------File Type Pointer

p = fopen("filename with path","file mode")

File Modes

r----Read Only Mode(File must be exists otherwise fopen function return NULL)

w—Write Only Mode—(If file exist then it will be overwrite. If file does not exist then new file will be created automatically)

a—Append Mode--(If file exist then new data is inserted after the old data. If file does not exist then new file will be created automatically)

r+---Read + Write

w+---Write + Read

a+---Append + Read

b—Binary Mode

Two Types of File

    1)Text File

        12345678------8 bytes

    2)Binary File

        12345678-----4 bytes

    rb—Read Binary

    wb---Write Binary

2)Reading and Writing

1)char i/o---Single character by character reading and writing

Reading

        char ch = fgetc(File Pointer)

Writing

        fputc(Character,File Pointer)

2)Line I/O or String I/O

Reading

    fgets(Char Array, Size ,File Pointer)

Writing

    fputs(Char Array, File Pointer);

3)Formatted I/O----int ,char, float, read and write

Reading

fscanf(File Pointer, Format Specifier, &Variable Name)

Writing

fprintf(File Pointer,Format Specifier, Variable Name);

4)Structured I/O

Reading

fread(&Stru Var, SizeOfStruct,NoOfRecords, File Pointer)

Writing

fwrite(&Stru Var, SizeOfStruct,NoOfRecords, File Pointer)

3)To Close any File

fclose(File Pointer)

To set the file pointer at desired position

fseek(File Pointer, Offset, Position)

Offset--+ or - No of Bytes

Position—0---From Beginning

Position---1---From Current

Position---2---From End

To know the current position of file pointer

long int l = ftell(File Pointer)

To set file pointer at the beginning of the file

rewind(File Pointer)

Commandline Arguments or Run time Arguments

```
void main(int argc, char *argv[ ])
{
        //statements;
}
arv[0]===Fixed---FileName.exe
```

Pre Processer Directive

#define-----Macro Substitution

To declare the constant

```
#define  PI 3.14
#define p printf
#define s scanf
```

typedef

To provides alternative name to any data type

Dynamic Memory Allocation

int a[5];--Compile time memory allocation

Two Problems

Memory Wastage

You can not store more values than array size

Dynamic Memory Allocation

malloc----Single Block Of Memory
calloc----Multiple Block Of Memory
realloc----To reallocate exiting Memory Allocation
free---To free the Memory