­­­­

Development:-

1:- mobile applications:-

.apk , need to install in mobile phone.

2:-websites:-

Run by url , it’s run by web browser.

3:-web applications:

Method to crate:-

Designing

Programming

Database

4:- desktop applications

Run by .exe file ,need to install

5:-standalone Applications

Key based software ,like antivirus.

6:-­­Console applications

Languages:-

languages is a medium to communicate with ­­other person.

Programming language:-

it’s a medium to communicate with computer system.

By coding , you are able to perform any task by computer system.

C programming:-

C is a high level programming language.

It was developed by Dennis’s Ritchie in 1972 in A&T’s bell laboratory .C is a HLL that’s means maximum words in c programming is taken with real life words.

C is also known as mother language just because syntaxes of maximum other programming languages is taken from the c.

**Basic structure of c programming:-**

**1:-Documentation section**

**Comment:-**Comment are those line of program , that is ignored by Compilar.

It just for the understanding for developer.

Single line comment:-

//Comments….

Multiline comments:-

/\*comments…….\*/

**2:- linking section:-**Each programming language provides some pre-defined codes,

That is saved in the library of programming.

When we need to use those pre-defined codes firstly ….

C-header

Java-package

C#-namespace

**Header File:-**

Header file is the collection of some pre-defined functions.

Function are kept in different header files based on the category.

Stdio.h:-standard input output :-printf(),stanf()…..

**Conio.h:-** console Input Output ,clrscr(),getch(),getche(),getchar(),putchar()………..

**Math.h**:-pow(),sqrt(),log(),ceil(),floor(),….

**String.h**:- strcpy(),strwr(),strcat()…

**#-** pre processor directive.

#include”header\_file\_name\_with\_extension”

**Include**:-include is a folder.

**3:-global declaration section:-**

Global declaration that’s mean declaring some variable globally that is accessible in all function of programs.

Int I=5;//global variable

**4:-main function:-** main function is a mandatory function of each C programs.

Main() function is declared with named main().

void main()

{

}

//void- data type

Main-function

**5:-Other sub program:-**

You can define other user –defined function in c programs.

It is optional.

//This is my first programs.

#include<stdio.h>//header files

**Keywords:-**keywords are some reserve words that has special meaning and functionality.

The number of keywords in c programming:-32

Ex:- if else int while float double long break continue, for do do-while……

**Identifier:-**Identifier are used for naming like: user defined functions, variable ,constant.

Rules to declare identifier:-

1:- Can’t start from number.

2:- identifier can’t have space in name.

3:-it can’t contain some special symbol like $,@.

4:-it can’t be same as keywords.

5:-it’s can be alphanumeric.

**Data\_type:**

Data type are some reserve words that defines the type of value and total memory spece. Data type defines which type of value,it can store and how much memory space it will occupy.

**There are mainly four datatype in c.**

**1:-Primitive data type/basic/fundamental data type**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data type name | Format specifier | Memory space | Ex values |  |
| Short | **%d** | **2 byte** | **-32768 to +32768** |  |
| Int | **%d** | **2 byte/4byle** | **3** |  |
| Long int | **%ld** | 4 byte |  |  |
| Long long int | **%ld** | 8 byte |  |  |
| Float | **%f** | 4 byte | **3.5** |  |
| double | **%lf** | 8 byte |  |  |
| long double | **%lf** | 10 byte |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Char | **%c** | 1bye |  |

**2:- Non-primitive data type/derived data type/**Derived data type are created with the help of basic data types.

Ex:- string, array pointer

**3:-User defined data type:-**user defined data type are declared by user.

Which type of value it can store is decides by user.

Ex:-structure Union

**4:-Empty data type:-**void is the empty data type. It is used to declare functions ,it’s not used to declare variables.

**Variable:-** variable are some temporary storage area ,which can hold a value and it’s value may be change any where during the exaction of program.

Note:-

1k/ :- Declaration of variable :-

Each variable in c should be declare once at the top of program .without declaration you can’t use any variable in your program.

**Syntax:- data\_type variable name;**

**2:-definition of variable:-**assign value to the variable.

1. Direct initialization /compile time initialization

Var =50;

1. User input /run time initialization

Scanf(“%f ,&var”)(formet\_specifier,variable\_list\_with &)

Scanf(“%f,%var”):- &denotes the memory address of variable

Int a; float f;

Scanf(“%d ,&a”);

Scanf(“%f,&a”);

Scanf(“%d %f ,&a,&b”);

**TASK:-**

// wap to convert given hours to the minutes.

//wap to convert minutes to the hours and remaining minutes.

150 minute : 2hours :30 minutes

// wap to finf simple intere00st

(ask user for the principle amount ,rate, time)

// wap to input rupess from the user and find how many notes of 200 and 50 is possible in the amount

//wap to input a number from user and print cube of number

// wap to input three number from user and find sum and average of numbers

//wap to input total days and find hours in days.

//wap to input a alphabet and print next alphabet comes a-b

//wa

p to input a alphabet and convert this alphabet into uppercase a-A

0

#include<stdio.h>

void main()

{

char val;

printf(“enter a character :”);

scanf(“%c”,&val);

printf(“character is : %c”,val);

}

**1:-getch():**getch function used to the input a single character. And we have to store a new variable. Getch() function reads the inputed value from the screen and saves the value in a character type variable.

Working:- getch() permits user to input only single key on the output screen and without press the enter key. it’s always excuted.

Syntax:-

ch a;

a=getch();

2:-getchar():- getchar() function works same as the scanf(), but getchar is unformatted input function so you do need to add any

**How to Link External file:-**

#include”file\_name.h”//library user defined header file.

%: include”file\_name.h”//library user defined header file.

#include<stdlib.h>

--------

Ststem(“pause”);

Task:-

//myheaderfile1.h

Getsqure(): input 1 number-> print squre

//myheaderfile2.h

getAddition():input 3 number -> print sum of number

// wap to check grater within 3 numbers by using ternary operator.

// a>b>c.

**Operator:-**

Operator are some special symbols , pre-defined in library,

That is used for special functionality.

Each operator has it’s own use ,when ever we need this we can use the operator in any statement of function.

Operator are used always with operands.

Suppose a statement:-

A=a +b:here +a ,b is operand and + is operator.

And = is a also operator.

Unary operator:- used with one operand ,like increment ,decrement

Binary operator: used with minimum 2 operands like := +,-,\*,>,<,etc.

Ternory Operator:- used with minimum 3 operator like: conditional operator

Based on the working operator are divided into many category.

1: Arithmetic operator

2: Relational Operator

3:Logical

4:Assignment

5:Increment/decrement

6: conditional

7: Bitwise

1:Arithmetic Operator:-

This operator is used for the mathematical operations.

It is used with numeric value. it’s a binary operator

That means it needs min 2 operands to be used.

+ : addition

- : subtract

\* : multiply

/ : divide

% : modular division

Precedence of operator (which one will execute first. ):

\*

/

%

+

-n

=

**% :-** it is called as modulo operator .it return remaining value after the division of two integer type values. It can’t be applied on floating type value.

**Relational Oprerator:-**

Relartional operator are used to specify the relation between two operands. It’s bimary operator .

Return type /result of relational statement is always ture/false.

Relational statement is a Boolean statement.

==:equals to

>: greater than

<: less than

>=:greater than equal to

<=:less than equals to

!=: is not equals to

Conditional Operator:-

Conditional operator is used to execute one statement out of two statement baed condition operator.

It is opetional of if- else statement. When there is only tone line statement in if and else then you can use conditional statement in more easy way.

It is a ternary operator ,that means it execute with 3 operands.

**? and : is conditional or ternary operator.**

Condition **?** true\_statement : false statement;

**Logical operator:-**logical operator is also used in two operands .return value of logical is true or false.

&& : logical AND

|| : Logical OR

! : is not /Logical NOT

……………………………

!=: ignores a single value

|  |  |  |  |
| --- | --- | --- | --- |
| Statement | Output | Statement | Output |
| True&&true | True/1 | True||true | True/1 |
| false&&true | False/0 | False||true | True/1 |
| True&&false | False/0 | True||false | True/1 |
| False&&false | False/0 | False||false | False/0 |

**Assignment Operator :-**

Assignment operator is used to assign right hand side calculation result to the left hand side variable.

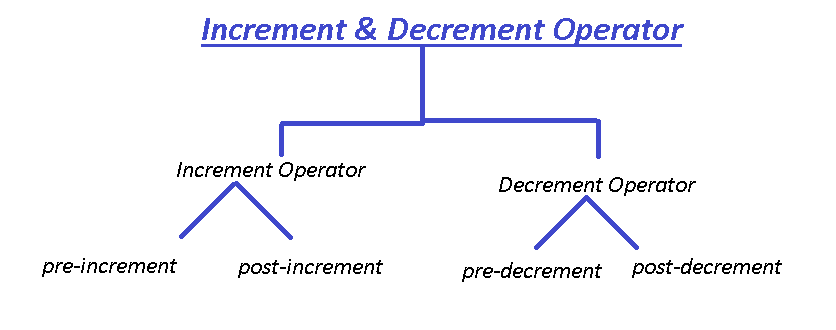
= comes with the lower precedence so it is done after the execution of statement.

Int a=20;

INCREMENT &DECREMENT OPERATOR:-

This is a unary operator. That’s means its applied a single operands.

It is used add or subtract the value.



**Pre-increment/decrement:-**

it increment /decrement the value of variable first then updated values is used un the statement.

Int i=10,J;

J=++I;

J=++i ;// after this statement value of j=11,i=11;

J=--i ;// after this statement value of j=9, i=9;

**Post-increment/Decrement:-**

It firstly allows to use the previous value of variable in the statement and after the value is increment /decrement.

int i=10,J;

J=i++;//after this statement value of J=10,i=11;

J=i--;// after this statement value of J=10,i=9;

***Compound Statement:-***

+=,-=,%=,/=,\*=,

*Scope of variable & Scope Modifier:-*

Scope:- Scope is a life time variable ,it defines a range in which variable is accessible.

1: local variable

2: global variable

Local variable:-

A variable declared within a block , is only executable within a block not outside the block.

#include< stdio.h >

void main()

{

int a=5; //local variable

//statement;

}

Global variable:- a variable declared outside a block ,it’s not only executable within a block also execute out of block.

#include< stdio.h >

int a=5; //global variable

void main()

{

//statement;

}

* multiple time.
* Variable declared within a block ,it destroyed after the block closing.

*Const variable:-*

// wap to find simple interest value:-

#include<stdio.h>

void main(){

const int val=100;

float pa,ra,t,si;

printf("Enter a principal amount , rate ,time\n");

scanf("%f%f%f",&pa,&ra,&t);

si=(pa\*ra\*t)/val;

printf("\nTotal simple interest = %.1f",si);

}

*Static variable:-*

static variable is a type of variable which lifetime does not destroys even after the closing block.

Static variable remains it’s value safe in the memory even block of variable has been completed.

Normal variable destroys after the closing the block,but static variable not.

Syntax:-

static data\_type variable\_name;

#include<stdio.h>

void main()

{

vinay();

vinay();

}

void vinay()

{

static int a=0;

a=a+10;

printf(“The value of a =%d”,a);

}

*Bitwise Operator:-*

Bitwise Operator are used to perform some task on binary digits . like 0 or 1

&-Bitwise AND

|-Bitwise OR

^-Bitwise X-OR

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Operator | Bit1 | Bti2 | Result | Or-Operator | ^-operator |
| & | 1 | 0 | 0 | 0|0=0 | 0^0=0 |
| & | 0 | 1 | 0 | 0|1=1 | 1^0=1 |
| & | 0 | 0 | 0 | 1|0=1 | 0^1=1 |
| & | 1 | 1 | 1 | 1|1=1 | 1^1=0 |

Controls statements:-

1:- conditional Statement:-conditional statement are lines of program that produce output always true or false.

Conditions can be applied by using relational or logical operator.

Relational or logical expression always generates output in 0 or 1.

i.e. true or false

***a:-If statement***: where condition are applied by using if keyword.

1.simple if statement:- where program has only a single pair of if else is known as simple if programs.

Syntax:-

If(condition)

{

// block of statement.

}

2.If-else statement

3.Ladder statement

4.Nested statement

***b:-Switch Statement***

2:-Iteration statement/looping statement

Looping statement are some keyword like (for ,while ,do) that is used to execute statements

Multiple time in program.

Type of loop:-

Entry control loop: where given condition checked

First ,then block is executed only if given condition

Is first.

1. for

2. while

Exit Control loop:-where statement execute first , then condition is checked .Next time statement will execute or not ,depends on the condition.

1. Do while loop

Each loop that there important parts:

1. Initialization-stating of loop

2.Condition –Stoping point of loop/ending of loop

3.Updation: Increment/Decrement value on each step

*For Loop:-*

For loop is entry control loop that means given condition with loop is checked first and then statements is executed by compilar .

**Syntax:**

for(initialization ;condition ; updation )

{

//statement;

}

*While loop*

*While loop is also a entry control loop where given condition with loop is checked in the starting of loop. If the given condition is true then block will be executed or not.*

*Only condition is written with the while keyword and* Updation of initilazed variable can be done.

*First condition then statement will be executed.*

*Syntax:-*

*Initialization;*

*while(condition)*

*{*

*//statement;*

*updation;*

*}*

*Do while loop:-*

Do while loop is a exit control loop that means firstly block of do will be executd and then after condition will checked.

In the case the given conditions is false then also loop will execute for 1 time.

Minimum execution time for do while loop is 1 and for while and for loop is 0.

Syntax:-

Initialization;

*do*

*{*

*//statement;*

*//updation;*

*}*

*while(condition);*

3:-Branching Statement/Transfer Statement/Jumping/statement

Fflush(): fflush(stdin);: fflush is the typically out stream function.

It is used to move output buffer data to the disk memory. So that output buffer can store next inputed value.

If you will get input of any character/string type value from user, then output buffer saves this data and there is no space to store next inputed value.

So firstly we have to fflush() or char the output buffer by using fflush(stdin) method.

Syntax:-

fflush(stdin);

***important point of the nested loops:-***

1. *Every nested block inside a loop statement must be same family.*
2. *Every nested block must execute at least once, then only we can say it is nested loops otherwise they are just nested block written in a manner that they appear to be nested.*

*ODD LOOP:-*

*Odd loop are used to define loop which are subjected to, run according to user define condition.*

*Type of loop:-*

*1.Uncontrolled loop*

*2.Controlled loop*

*a. entry control loop*

*condition will be checked before hand.*

*For ,while*

*b. control loop*

*Condition will be checked after wards.*

*Dowhile.*

1. *Odd loop*

*This is also a finite loops it runs according to user defined input.*

*1.dowhile*

*2.goto*

*ARRAY:-*

Array is a collection of similar data type elements.

When you need to store ,multiple values of same data type instead of declaring all variables separately , you can declare a array with fix size.

An array can store multiple value of different index.

Indexing of array always starts from 0 and the last element index is size -1.

*Types of array---*

1. One dimensional Array :1D
2. Two dimensional Array :2D

*One Dimensional Array:-*

1D array used to store multiple values sequentially. 1D array initialize sequential memory for all indexes of array.

Syntax of declaration :-

Data\_type var\_name[size];

Ex:-

int vinay[10];

2 dimensional Array:-

`````````````````````````````

2 Dimensional array is a memory organization that store values in from of rows and columns .Two dimensional array is used to store any value of similar

Data types in form of rows and columns.

*Syntax to declare 2D array:-*

Data\_type var\_name[rows][column];

Total number of element in two dimensional is :number of row \* number of coloumn.

Indexing of rows starts from 0 and max index of row is row\_size -1.

In same way indexing of column also starts from 0 and max is column\_size-1.

*UDF(User defined Function)*

:- user defined function are of block of statement that is written by developer for the self use.

Function are mainly two type.

1. Pre-defined function
2. User defined function

Function are nothing , it just a name with a block statement this is used to perform any already task.

The use of pre-defined function are already defined in the library of compiler, we just need to call and use whenever we need it.

*User defined function:-*

UDF provides facility of code -reusabilty. You can declare a udf for any specific task and you can call this function multiple time whenever you need this code in the function.

*Syntax:-*

Return \_type function\_name(formal\_argu0ment\_list)

{

//local variable;

//statement;

//return value;

}

Return type: return type of a function is a data type is always used before the name of function. This return type defines the value that a function returns to the caller.

void fun1():-void date type before the function name defines that this function does not return any value to the caller.

{

}

int fun1():- this function will return a integer value to the caller.

{

return 0;

}

==========================================

PASSING ARGUMENT:-

Arguments are some local variable of function which is defined with the top parentheses of function.

*For Example:-*

void func1(int a , int b)

{  
  
}

Here func1()is the UDF with two integer type parameters.

Note:- This argument is known as formal parameters.

When this function will called by caller then has to pass the function same in length , data\_type and number of arguments.

Note:- Value of arguments variable are assigned by caller.

void add()

{

int a,b,res;

printf(“Enter 2 value:”);

scanf(“%d”,&a,&b);

res=a+b;

printf(“Result is =%d”,res);

}

Calling:add();

void add(int a, int b)

{

res=a+b;  
printf(“result is %d”,res);

}

Calling: add(17,20);

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

***Based on the structure of UDF ,UDF can be defined in four different way:-***

1.No return type and no passing arguments

2.No return type with passing arguments.

3.Return type with no passing arguments.

4. Return type with passing arguments.

*Return Type:-* A UDF with void return type is known as no return type UDF. If a UDF any other return type

Except void then this function will return ……

***1.No return type and no passing arguments:-***

*STRING:-*

String is the collection of characters. In C directly

String data type is not supported so we have to declare a character type array to hold a string type value.

----------------------------------------------------------------------

Mostly used data type of programming is string. In C string data type input and output function are different.

*Syntax to declare a string type variable:-*

char var\_name[size];

Ex:

char name[50];

This name variable is a string type variable which can store a value of maximum 50 character.

Input functions of string:

scanf():Which can read only one word string from user.

gets():

Which can read multi- word or multi-line string from user .

gets(name);

string:string data type need-

Direct Initialization of String type variable:-

char str[“techpile”];

­char str[‘T’,’E’,’C’,’H’,’P’,’I’,’L’,’E’];

to input & output of string type value there is a

format specifier : %s

Printf(“%s”,str);

User Input to string type variable:-

Char str[50];

gets(str);

String Pre-defined function:-

There are some pre-defined functions in c that makes our task easy. All string functions are stored in a library.

Some pre-function of string:-

Strln(); Strcpy(); Strncpy(); Strcmp(); Strcmpi();

Strncmp(); Strnicmp(); Strlwr(); Strupr(); Strcat();

Strncat(); Strrev(); Strtkn();

--------------------------------------------------------------

Strcpy();

Strcpy() is function used to copy a string to another string.

It require two string type arguments.

Char name[100];

Char firstname[100]=”vinay”;

Name=”vinay”;//wrong

Strcpy(name,”vinay”);//right way

Strcpy(name,firstname);//right way

-----------------------------------------------------------------

Strcmp():-strcmp() is used to compare two string that is equal or not. If both string are equal this function will return 0 either any positive or negative value it may return.

int val=strcmp(string1,string\_2);

val==0 if string1 and string\_2 is equal.

strcmp():-

it is same as strcmp() but it ignore the casex of character of both . it also takes two string type arguments and returns a integer type value.

strncmp:- it is same as strcmp() so it also compare that two given strings are equal or not.

but it compare both strings till a given length.it need 3 arguments, first two if strings type and last one of integer type. and returns a integer type value same as strcmp().

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

strnicmp():-

\*\*\*\*\*\*\*\*

strnicmp() is same as strncmp() but it ignore the cases used in both string.

so it compare to string till n number of character and ignores the cases of strings.

it returns 0 if first n character are same or returns any positive or negetive value.

Syntax:-

Int res=strnicmp(string\_1, string\_2,n);

-------------------

Strrev():-

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

it reverse the value of a string. It takes only one argument and after the execution of function string value will be reverse of previous value.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

strlwr();-

strlwr() is used to convert all character of string to the lowercase.

it accept only one argument and sets the argument value to the lowercase.

char name[]="TECHPILE";

Syntax: strwr(name);

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

strupr:-

\*\*\*\*\*\*\*\*

strupr() is a function accpets only string type argument

and sets the argument value to it's uppercase.

strcat :-

\*\*\*\*\*\*\*\*\*

strcat() function is used to concatenate two string type values and assing into first argument.

Syntax:-

strcat(string\_1,string\_2);

it takes 2 string type value as argument and returns nothing.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

strncat:-

\*\*\*\*\*\*\*\*\*

It is same as strcat() but takes 3 argument. First two argument is a string type value to be concatenated and third one is a integer type value which shows how many character of second will be concatenated to the first string.

strncat(strint\_1,string\_2,n);

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*Pointer:-*

Pointer is a variable that points to the memory address of any variable. Pointer is also a variable declared with\* but this type of variable can hold memory address of another variable.

Mainly two symbols used with pointer.

& :- & is a symbol used to denote the memory address of a variable.

:- \* is a symbol used to denote the value present of a memory address.

Int a;// this is a normal integer type variable.

Int \*a;// this a pointer type variable . which can hold memory address of a integer type variable.

DECLARATION OF POINTTER VARIABLE

*Int \*i=5;*

Recursion :- Recursion is a concept when a function called by itself. When calling of a function is done within the block of same function then this concept is known as recursion.

int main();

{

//statement;

main();

}

STURCTURE:-

structure can hold multiple values of different data type. That means we can say that a

structure is the collection of heterogeneous elements.

Mainly structure is a data type, you can declare multiple variable of struct type.

And one variable would store which type of value is declared by user at the declaration time of structure.

*Syntax*:

Struct structure\_name;

{

Data type variable\_name;

-

-

--

};

----------------------------------------------------------------------

1.Global Structure:-

Structure Padding:-

*Union:-*

Union is also a user defined data type but it doesn’t declare sperate memory locations for all elements of unions.

It is also the collection of heterogeneous data elements. That means it can have multiple element of different data types same as structure.

But structure declares seprate memory locations for all it’s elements while union declare only one

Memory location for the element having maximum size.

And all other element shares the same memory.

Struct stru

{

int a;

Char name[20];

}// total size of structure is : sum of all member size +structure padding.

Union uni

{

int a;

Char name[20];

}//total size of union =size of element with largest size =20 byte.

……………..