Type of application:-

1.Mobile Application= apk. file, need to install in phone

2.Website = Run by URL (Uniform Resource Locator),: it run on web browser

3.web application = dynamic website ,Run by URL

Method to Creat -

* Designing : html/css/bootstrap
* Programming: c , c++ , java ,j query ,python
* Database: Mysql , mssql ,msaccess , oracle , mongodb…

4.Cansole application: Run By command on blue/black screen

5.Desktop Application : Rn by .exe file, need to install

6.standalone Application : key based software ,like antiviruses

Language: language is a medium to communicate with Other person

Programming language: It id medium to perform any task by computer system

**C programming : C is a high level programming language . it was developed by Dennis Ritchie in 1972 in A&T’s Bell Laboratory. C is a hill (high level language) that means maximum words used in c programming is taken with the real life words.**

High level language that means a developer friendly language

Computer->language->Machine language(Binary Language / Low level language/0&1)

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

**Basic structure of c programs:-**

* **Documentation Section**

Comments : Comments are those line of program, that is ignored by Compiler . it is just for the understanding of developer.

Single line comments: //comments……

Multi line comments :/\*………………….

………………………………………………….\*/

* **Linking Section: Each programming language provide some pre-defined codes, that is saved in the library of programming.**
* **When we need to use those pre-defined codes firstly we need to add library in the programs.**

**C-Header files**

**Java- Package**

**Python-Modules**

**C#- Namespace**

**Header File :** Header file is the collection of some pre-define function.

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

**Stdio.h:** Standard input output : printf(),scanf(),……..

**Conio.**h: console input output : clrscr(),getch(),getche(),getchar(), putchar()………….

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

**String.**h: strcpy(),strlwr(), strupr(), strrev(),strcat(),…………………………

**#-** Pre-processor with # is executed first In the program.

**Include-** Include is a folder that contains all header files

* **Global Declaration Section:** Global declaration that means declaring some variables globally that is accessible in all function of programs

int a=5;//globally declared

void main()//pre-defined function + user-defined function

{

}

Pow();

{

A=a+10;

}

Void techpile()//user-defined function

{

}

* **Main Function:** Main finction is a mandatory function of each c programs main function is declared with named main().

**Void main()**

**{**

**}**

**//void-data type**

**Main()-Function**

* **Other sub programs: you can define other user-defined function in c programs . it is optional.**

**// his is my first program//comment line**

**#include<stdio.h>//header files**

**Void main()//it is main part of program, execution from here**

**{**

**}**

**Keywords:** keyword are some reversed words that has special meaning and functionality . in c program total 32 keywords are defined . in programming keywords are used at special whenever it needs.

**Ex.** If,else,do,int,while,float,double,long,struct,break,continue…….

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

**Rules to declare identifier:**

Identifier can not start with number. It can not be numericAlpha.

Ex. 2var:invalid

identifier can not have space in name.

It can not contain some special symbols like &, \* etc

It can not be same as keywords

It can be Alpha-numeric like :var1 ,number1

Identifier: variable, constant , UDFunction

**Data Type:** Data type are some reserve words that defines the type of value and total memory space . data type defines witch type of value , it can store and how much memory space it will occupy

**There manly Four type of data type in C:**

1. **Primitive data type / basic / fundamental data type**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data type name** | **Format specifier** | **Memory space** | **Ex value** |
| Short | **%d** | **1 byte** | **-32768 to + 32767** |
| Int : whole num | **%D** | **2byte/4byte** | **3** |
| Long int : whole num | **%ld** | **4byte** |  |
| Long long int : whole num | **%ld** | **8byte** |  |
| Float : decimal | **%f** | **4byte** | **3.5** |
| Double : decimal | **%lf** | **8byte** |  |
| Long double : decimal | **%lf** | **10byte** |  |
| char | **%c** | **1byte** |  |

1. **Non-Primitive data type / derived data type** : derived data type are created with the help of basic data type

**Ex**: String , Array ,pointer

1. **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**

1. **Empty data type :** void is the empty data type . it is used to declare functions. It is not used to declare variables.

**Input and output function in c:** Each programming language some pre-defined

Input output function

I/P function: input function is used to take input from the user at run time . where you can store those value in a variable and can use anywhere in the program.

* Formatted input function:

Scanf():

* Un-formatted input function:
* Getch(): use to input a single character
* Getche(): use to input a single character to user
* Getchar(): use to input a single character to user
* Gets(): use to input a multi-word string value from user

Output function : o/p functions are used to print message or value on the screen for user.

* Formatted Output function

Printf()

* Un-formatted output function

Putchar()

Puts()

Dev c++:

**Variable :-** variable are some temporary storage area , witch can hold a value and its may be change any where during the execution of program.

Note-

1. **Declaration of variable:** Each variable in c , should be declaration once at the top of program . without declaration you can not use any variable in your program.

**Syntax:** data type variable name;

1. **Definition of variable:** assign value to the variable

**(A):** direct initialization / compile time initialization

Var=50;

**(b):** User input / run time initialization

Scanf(“formate\_specifier”,variable \_list with &);

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

Int a; ,float f;

Scanf(“%d”,&a);

Scanf(“%f”,&f);

Scanf(“integer %d float %f”);

**(C): By expression/calculation**

**A=b+10;**

**Numeric value: whole number/real number-**

**Character input special function :-**

**getch():** getch() is used to input a single character from user 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 a single key on the output screen , next line the program execute without waiting to press enter key as soon as user inputs a single character

Syntax:

Char ch;

Ch=getch();

Character inputed by user by using getch() function does not appear on output screen

**Getche(): - getche()** also permit user to input only a single character and character inputed by user appear on output screen and next line of program execute without waiting for enter key.

Ex. //WAP to input a charecter and print the inputed charecter

#include<stdio.h>

void main()

{

char ch;

printf("please enter a single charecter :");

//scanf("%c",&ch);

ch=getche();

printf("\n inputed charecter is : %c",ch);

}

**Getchar(): getchar()**  function work same as the scanf(), but getchar() unformatted input function so u do not need to add any forment specifier here.

Syntax:

Char val;

Val=getchar();

**Example:**

//WAP to input a character and print the inputed character

#include<stdio.h>

void main()

{

char ch;

printf("please enter a single character :");

//scanf("%c",&ch);

ch=getchar();

printf("\ninputed character is : %c",ch);

}

C source code -> compile-> object file -> convert into .exe files ->

Output

**Operator:** operator are some special symbol , pre defined in library , that is used for special functionality .

Each operator has its own use when ever we need this we can use the operator in any statement of program

Operator are always used with operands

**Suppose a statement:**

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

**Unary operator**: unary operator used with one operands , like increment – decrement operator

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

**Ternary operator:** used with minimum 3 operands like: conditional operator

Based on the working operator are divided are many category:

**4 .Arithemtic operator**

1. **Relational operator**
2. **Logical operator**
3. **Assignment operator**
4. **Increment & decrement operator**
5. **Conditional operator**
6. **Bitwise operator**
7. **Arithmetic operator:**

This operator I used for mathematical operations it is used with numeric value it is a binary operator that means it need mean 2 operands to be used.

+:Addition

-:subtract

\*:multiply

/:division

%:modular division

Precedence of operator:

\*

/

%

+

-

=

//WAP to demonstrate the use of arithmetic operator

#include<stdio.h>

void main()

{

int n1,n2,n3;

printf("Enter three number :");

scanf("%d%d%d",&n1,&n2,&n3);

printf("some of number is : %d",(n1+n2+n3));

printf("\nsubstract of first & second number is: %d",(n1-n2));

printf("\nmultiply result of last two values is : %d",n3\*(n1-n2));

}

**/:** it returns always a integer number if both operands are integer.

It returns a float value if any one operands is float.

**/:** 20/10=2

**%:** it is called as modulo operator it returns remaining value after the division of two integer type value . it can not be applied on floating type value.

Int n1=40,n2=30;

N1/n2=4

N1%n2=division by remainder

Precedence of arithmetic Operator :

\*,/,%:- same precedence : high : Associativity – left to right

+,- :- same precedence : low : associativity : left to right

Int a=a-10+b;

**Relational operator:** relational operator are used to specify the relation between two operands . it is a binary operator.

**Return type/**result of relational statement is always true/false

Relational statement is a Boolean statement

==:- equals to

>: greater than

<: less then

>=: greater then equal to

<=: less then are equal to

!=: is not equal to

A=a+10-30 :-//arithmetic statement

A=a>10; //true/false: 1/0

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

**Conditional operator :** conditional operatpr is used to execute are used to execute one statement out of two statement based on a condition.

It is optional of if elese statement . when there is only one line statement in if and else then u can use conditional statement is more easy way it is a ternary operator that means it executes with 3 operands

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

Syntax:

Condition ? true\_statement :false \_statement

Condition is a statement which result is always true or false.

**//wap to input total fee of your gradustion and there is 10% discount**

**//if fee is greater than 20000 and there is 20% discount if fee is**

**//less than 20000 calculate the total fee after discount**

**#include<stdio.h>**

**void main()**

**{**

**int fee,disper;**

**printf("Enter the actual fee :");**

**scanf("%d",&fee);**

**disper=fee>20000?10:20;**

**printf("Dicounted fee is : %d",fee-((fee\*disper)/100));**

**}**

**Assignment Operator:-**

Is the assignment operator it 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=40,b=50;

A=a>b;a=?,a=0

A+b=20;//invalid statement

In left hand side there should be a single variable .

**Increment & Decrement operator :-**

This operator is a unary operator that means it is applied on single operands.

It is used to add or subtract the value one from the correct value of operand .

Example:

Int a=3;

A=a+1; //this is normal statement

A++; this is unary statement

**Pre operator:-**  it increment or decrement the value of variable first the updated value is used in the statement .

Int i=10;j;

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

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

**Post-operator:** it firstly allows to use the previous value of variable in the statement and after that 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

Exercise:

Int a=10,b=2,c=5,res;

a. Res=a++ + ++b + c;

b. Res=a++ + a + --b + c++;

c. Res=++a + ++b + ++a + a + b + c;

d. Res=a++ + a++ + b-- + ++a;

e. Res=a + b + ++a + ++b;

f=Res=a++ + ++b + ++c + ++a;

g. Res=a++ + ++a + a + a + b++;

**Compound statement : -**

+=,-=,\*=,/= are some compound statement.

Int a=10;

A=a+1; a++;

A=a+10; //a+=10;

Scope of variable & Scope modifier in c:

Scope: Scope is the lifetime variable , it define aa range in which variable is accessible.

1.Local scope

2.Globle scope

Local scope:

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

Void mas()

{

Int a;//local variable for msg()

}

#include<stdio.h>

void main()

{

int var=20;//local variable

printf("var=%d",var);

fun();

}

void fun()

{

printf("var=%d",var);

}

**Global scope:-**

#include<stdio.h>

int var=20;//Global variable

void main()

{

int var=20;

printf("var=%d",var);

fun();

}

void fun()

{

printf("var=%d",var);

}

* Within same scope you can not declare single variable multiple times
* Variable declare within a block , it destroyed after the block closing.
* You can declare a variable same as global variable again as a local variable of a block.

**Constant in c:-**

Constant is a name that has a value that can not be change anywhere during the program.

In c programming , you can declare constant in two way:

1.By using #define

2.By using const keyword

1. By using #define:

* #does not executes the line but replace a name with a value #define a 89

All over the program a will be replace by 89.

1. Do not put ; after the constant declaration
2. You can not declare a locale variable same as macro
3. You can not change value of macro

#define a 50 // here is a micro

1. You can the define a function as a macro
2. You can add multiple lines in #define by using \
3. There is two pre defined macro for to get current date and time

Bitwise Operator:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Operator | Bit1 | Bit2 | Result | Operator |  | ^ |
| & | 1 | 1 | 1 | 1|1 | 1 | 1^1=0 |
| & | 1 | 0 | 0 | 0|1 | 1 | 0^0=0 |
| & | 0 | 1 | 0 | 1|0 | 1 | 1^0=1 |
| & | 0 | 0 | 0 | 0|0 | 0 | 0^1=1 |

Conditional statement :- conditional statement are lines of program that produce output always as true/false.

Conditional can be applied by using relational or logical operator relational or logical expressions 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 without else is known as simple if programs.

System:-

If(condition)

{

//block of if

}

>,<,>=,<=,!=,&&,||,!

//wap to cheack a persion eligble for hr or not

//only female candidates are allowed

#include<stdio.h>

void main()

{

char gen;

printf("enter m if you are male and enter f if you are female :");

gen=getch();

if(gen=='f')

{

printf("\nYes you are eligible to apply");

}

if(gen!='f')

{

printf("\nNo You are not eligble to apply");

}

}

**Simple if statement :-**

If else statement :-

If there is only one condition is given and two different block of statement has to execute . one if given condition is true and second if the given condition is false . then if-else statement is used .

* no semicolon should be present between if else
* no other statement should be present between if and else block .
* no condition is written else statement , condition always applied with if.
* Else statement can not be used independently in program ,it always used after the if statement .

If(condition)

{

}

Else

{

}

//WAP to input percentage of student and check result status(pass or fail).

#include<stdio.h>

void main()

{

float per;

printf("Enter your percenyage :");

scanf("%f",&per);

if(per>=33)

{

printf("You are pass");

}

else

{

printf("You are fail");

}

}

**OR**

//WAP to input percentage of student and check result status(pass or fail).

#include<stdio.h>

void main()

{

float per;

printf("Enter your percenyage :");

scanf("%f",&per);

if(per>=0 && per<=100)

{

if(per>=33)

{

printf("You are pass");

}

else

{

printf("You are fail");

}

}

else

{

printf("You are type invalid percentage");

}

}­

Ladder if statement :-

WAP to check result status of student based on the percentage.

Ladder else if statement:

When there is multiple condition is given and different statement on given on the different condition.

Then ladder else if statement is used it ensure that out of all statement only one statement will be executed at a time.

Syntax:

If(condition)

{

//statement;

}

Else if

{

//statement ;

}

Else

{

//statement;

}

//WAP to input a character and check character is alphabet or not if character is alphabet then check it is vowel or consonant

#include<stdio.h>

void main()

{

char alpha;

printf("Enter a character :");

alpha=getche();

if((alpha>='a' && alpha<='z') || (alpha>='A' && alpha<='Z'))

{

if(alpha=='a' || alpha=='e' || alpha=='i' || alpha=='o' || alpha=='u' || alpha=='A' || alpha=='E' || alpha=='I' || alpha=='O' || alpha=='U')

{

printf("\nCharecter is vovel");

}

else

{

printf("\nCharecter is consonent");

}

}

else

{

printf("\nYou type invalid charecter");

}

}

|  |  |
| --- | --- |
| **Char Type Value** | **Integer type value** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

//WAP to input a character and print character in uppercase if the inputed value is alphabet and in lowercase

**24/09/2021**

**Looping statement / Iteration statement:-**

Looping statement are some keywords like for while do that is used to execute statement multiple time in program .

**Types of loop :-**

**Entry control loop :-** where given condition is checked first ,the block is executed only if given condition is true.

1.For loop

2.while loop

**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 three important parts:

1.initialization – starting of loop

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

3.Updation- Increment / decrement value on each step.

1. **For loop:** - for loop is entry control loop that means given condition with loop is checked first and then statement is executed by compiler.

**Syntax:**

For(initialization ; condition ; updation)

{

}

Initialization: a=5,a=b+2

Condition: a>5 , a==5, a%2==5%3,

Updation: a++ , a=a+10, a+=10,a=1%2

**Looping statement :**

Write the syntax of following loops and print value of variable assigned by loop

1. Initialization i=6,condition is i<10,execute loop 3 times .
2. Initialization is i=10, updation i=i=7 execute loop 5 times
3. Condition is i<=81 and updation is i=i\*I execute loop 3 times
4. For(i=10; i<=50) execute loop 4 times
5. For(i=-3; ;i+1) execute loop 5 times
6. For (i=10; ;i=i-3) execute loop 5 times
7. For (; i>30 ; i=i\*2) execute loop 4 times
8. For(; i>=1 ; i=i/3)execute loop 5 times

Print following given value by loop

1,2,3,4,5

7,5,3,2,1

2,4,8,16,32,64,128

9,6,3,1

1.5,3.5,5.5,7.5,9.5

8,4,2,1

1,3,6,10,15,21

90,30,10,3,1

**27/09/2021**

1.//WAP to print following statement

* 8,12,17,30,38
* 7,10,15,22,31,42
* 12,11,9,6,2

2.//when ending of loop is not given

* wap to print even value 1 to n

4.//WAP to print all odd number from 10 to n

5.//WAP to print all digits witch is divisible by 4 from 1 to n

6.WAP to print following statement

* 1,2,4,7,11,16,22,………n

7.loop where we have to work on multiple user inputed value

* WAP to ask 5 value from user and print only odd value

8. find sum of value in loop

* WAP to find sum of all value within 1 to 10

9.WAP to print sum of all numbers between 1 to n where number divide by 3 is equal to 2

10.WAP to print sum of all number where number is not divisible by 3 within 1 to n

11.WAP to add all even number within n to 50

**28/09/2021**

**1.** //WAP to find the multiplication of all number that is divisible by 3 and 5 both

within a given series

2. WAP to find multiplication of all number where last digit of number is 3 within

1 to n

3. WAP to find multiplication of all number where last digits of number

is divisible by 2 within 1 to n

Task

1. WAP to count total number divisible by 3 and 5 and 7 within a given series
2. WAP to ask to input gender first character (m/f) from 10 users and count how many female and how many males are present here
3. WAP to read 10 number from keyboard and count number of positive , negative , zeros are present here

//wap to print digits all digits of number in separate lines

//wap to print sum of digits of a number

//wap to reverse a number

//wap to check a number is palindrome are not

//wap to check a number is a Armstrong number or not

//wap to print sum of square of all digits of a number

**29/09/2021**

**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 clear the output buffer by using fflush (stdin) method.

Syntax:

Fflush(stdin);

Int/float-numeric->char/string

**Do while loop:-**

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

In the case the given condition is false then also loop will execute for while and for loop is 0

Syntax:-

Initialization

Do

{

//statement ;

//updation

}while(condition);

**01/10/2021**

**Branching statement :-** branching statement are used to transfer control from one place to another in the program in branching statement mainly 3 keywords are used to transfer control.

1. Break;
2. Continue;
3. Goto;

Break and continue statement is used only within the block of loop .

1. Break: Break can be used with switch statement or within block of looping statement .

Syntax:

Break;

**Switch statement :** switch statement is optional of else – if ladder switch is used to execute blocks based on a single value .

When you have to apply multiple condition based on a single value then instead of using else if ladder you can use switch statement .

Switch statement executes blocks based on a value not on condition

Syntax of switch statement :-

Switch(value/expression)

{

Case value;

Statement;

Break;

Case value;

//statement ;

Break;

Note: switch statement can be used only for integer and character type expression

**Continue:-** Continue statement can only be used within looping block. Continue is used to break the current iteration and continue the loop from the next iteration .

Continue breaks only one iteration but executes for its completes time.

Syntax:

Continue;

void main()

{

int i;

for(i=1;i<=10;i++)

{

if(i==5)

continue;

printf("%d\n",i);

}

//printf("%d",i);

}

**Goto Statement :-**

Goto statement is used to transfer control to a pre-defined label .

Label is a identifier that is used to define a position into program

Syntax:

Main()

{

Point ://label

//statement;

Goto Point;

}

**Program:**

//wap to input a number and print number is even or odd till user input

//a neg number

#include<stdio.h>

void main()

{

int num;

start:

printf("Enter the number :");

scanf("%d",&num);

if(num<0)

goto end;

if(num%2==0)

printf("Even number\n");

else

printf("Odd number\n");

goto start;

end:

printf("##");

}

//wap to input multiple value from user and find sum of all value until user input a negative value

//wap to input number and print factorial of numbers until user input 0 or 1

//wap to find table of all number from 1 to 5

123456789

246810…

369121518…

//wap to print all prime number between 1 to 50

**04/10/2021**

**Nested loop:**

Whenever a loop block is contained inside another loop block , then the set of blocks looping all together are called nested loop.

Find out the set of nested loops

**Important point of the nested loop**

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

For(i=1; i>=10;i++){

For(j=1;j<=10;j++){

}

}

**Explanation:**

Both Syntax appears to be nested in form of loops but they cannot be regarded as nested loop.

Why?, because the condition for the outer loop becomes false , hence no of iterations for the

outer loop becomes =0

inner loop iteration= 0

**05/10/2021**

Syntax of nested loop:-

Outer\_loop(<min-condition-1>){

//some statement

Inner\_loop(<min-condition-2>){

//some statement

}

//some statement

}

For => min-condition initialisation; condition; updation;

While => condition

Do => odd loops

GCC – gnu compiler collection

GPL-general public license

BSD - Berkelian software distribution

**06/10/2021**

Let us make a program using odd loop so that we can get better idea how to run a odd loop program.

Now the question is what is odd loop:-

Odd loop is user defined loop which are subjected to , run according to user defined condition.

**Type of loop :-**

Two type loop are their-

1. Uncontrolled loop(Infinite loop)
2. Controlled loop (finite loop)
3. Entry control loop

Condition will be checked before hand.

1. Exit control loop

Condition will be checked afterwards

1. Odd loop

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

1. Do while
2. Goto

**07/10/2021**

//wap to find table of all numbers from# 2 to 10

//wap to find all prime number from 1 to 50

//wap to input 5 digit from user and count how many digits of are palindrome

//wap to count all prime numbers within 1 to given number

//wap to input 5 subject marks of 3 student and print division of all

**Array :-**

Array is the collection similar data type element when you need to multiple value and same data type it is instead of declaring all variable separately , you can declare a array with fix size.

An array can store multiple value on different index . indexing of array always start from 0 and the last element index is size-1.

Size of array is a integer value , it is assign at the compile time .

Type of array: -

1. One dimensional array: 1D array
2. Two-dimensional array :2D array

Int s1,s2,s3,s4,s5;

Int s[5]; // declaration of array

S[0],s[1],s[2],s[3],s[4],s[5]

1D array :-

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

Syntax of declaration:

Data\_type var\_name[size];

Ex:-

Input arr[5];

**08/10/2021**

//wap to input 10 element and find sum of all.

//wap to find greatest number within 10 element of array.

//wap to find smallest number within 10 elements.

//wap to input 10 element and arrange value in descending order.

1 5 8 4 7 6 3 8 5 9

5 1 8 4 7 6 3 8 5 9

8 1 5 4 7 6 3 8 5 9

9 1 5 4 7 6 3 8 5 8

Step2 :

9 5 1 4 7 6 3 8 5 8

9 8 1 4 5 6 3 7 5 8

Arr[0]=1

// wap to input 10 element and arrange value in ascending order.

//wap to input 10 element and check a given value is present in array or not.

//wap to input 10 element of array and print only unique element of array

//wap to input 10 elements of array and count a given value is how many times present in array.

1,2,3,4,2,3,5,6,4,56

Unique element are: 1,2,3,4,5,6,56

//wap to input 10 element and find sum of two sequential elements

//wap to input 10 element on two array and find sum of both array

//wap to input 10 element of array and check a given of value is how many times present in array and on which position

// //wap to input 10 element and search one value is how many times present in array or not.

//wap to input 10 element and search one value is on which indexing present in array or not.

//wap to input 10 element and print all indexing on which a value is present in array

**2D Array:-**

**T**wo-dimensional array in the memory organization that store value in form of rows and columns. Two-dimensional array is used to store any value of similar data type in form of rows and columns.

Syntax to declare a 2d array: -

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

Int ar[2][3];

|  |  |  |
| --- | --- | --- |
| 0 ar[0][0] | Ar[0][1] | Ar[0][2] |
| 1 ar[1][0] | Ar[1][1] | Ar[1][2] |
|  |  |  |

Total number of element in two dimentional array is : number of row\*number of column

Ar[3][3]=9 unique element of array

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

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

1. WAP to store 3 subject marks of 5 student and find sum of marks of all student.
2. WAP to input all element of a 3\*3 matrix and print all values in matrix form.
3. //wap to input 3\*3 matrix and print “this is acceptable “only if all element of matrix is 0.
4. //wap to input 3\*3 matrix and print “this is acceptable “only if all diagonal element is 1.

**10/10/2021**

**UDF: - (**user defined function**)**

User defined function are a block of statement that is written by developer for the self use . Function are mainly two types:

1. Pre-defined function
2. User-defined function

Function are nothing , it is just a name with a block of statement that is used to perform by specific task .

Pre-defined function:-

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-reusability . so you can declare a UDF for any specific task and you can call this function multiple time whenever you need this code in the program.

Syntax to defined a UDF:-

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

{

//local variable;

//statement;

//return value;

}

Return Type: Return type of function is a data type that is always use before the name of function. this return type defines the value that a function return to caller.

Void fun1()-void data 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 : Argument are some local variable of the function which is defined with the top parenthesis of function.

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 call by caller the caller has to pass values to the function same in length , data\_type and number of argument.

Note: Value of argument variable are assignment by caller.

Void add()

{

Int a,b; //local variables

Printf(“Enter two numbers”);

Scanf(“%d%d”,&a,&b);

Printf(“result =%d”,(a+b));

}

Void add(int a, int b)

{

Printf(“result =%d”,(a+b));

}

A UDF to Find SI (Simple Interest)

Void Find SI()

{

Int pa;

Float rate,time;

Printf(“enter pa,rate,time:”);

Scanf(“%d%d%d”,&pa,&rate,&time);

Res=(pa\*rate\*time)/100;

Printf(“Result = %f” , res);

}//calling:find()

Void find(int pa, float, rate, float time)

{

Int res;

Res=(pa\*rate\*time)/100;

Printf(“result=%f,res);

}//calling : find si(10000,2.5,2.7)

findSI(a,b,c)



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 argument.

3.return type with no passing argument.

4.return type with passing argument.

//WAP to print all number of a series by using UDF

Void series()

Void series(int start , int end)

//WAP to count all even number within a series

Void series()

Void series(int start , int end)

//WAP with UDF to add two integer value and second to add two float value

Void add(int a, int b)

Void fadd(float a, float b)

**11/10/2021**

No passing parameter

Void addition (int a, int b) int a, int b is the formal parameter

{

}

**Calling:**

Addition(20,40); -20,40 is the actual parameter actual parameter and formal parameter should be same and number data type.

Return Type:- A UDF with void return type is known as no return type if a UDF has any return type except void then this function will return a value to the caller.

A single function can return a single value at a time;

Return type function must have return keyword within the block.

//WAP with following UDF

Void checkpalindrome(int num);

Int check palindrome (int num);//return 0 if palindrome and 1 if not palindrome

//wap to get ncr value of a number by using UDF

NCR=factorial of n / (factorial of r\*factorial of (n-r))

N,r=5,3

Ncr=120/6\*2=10

//wap to convert rupee to paisa

Int convert()

Void convert()

Int convert(float rupee)

Void convert(float rupee)

//wap to arrange element of array in ascending order

Void arrangeArray(int arr[])

**String:-**

String is the collection of character . 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 an output function are different.

**22/10/2021**

Syntax to declare a string type variable:

Charvar\_name[size];

Ex:

Char name[50];

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

Input function of string:

Scanf():which can read only one word string rom the user

Gets(): which can read multi-word or multi line string from user

Gets(name);

Char str[100];

Int arr[]={20,39,50,30,10}

Int arr[10]={40,80,50,80,50};

Int arr[5];

For(i=0;i<=4;i++}

Scanf(“%d”,&arr[i]);

Char str[]=”techpile”;

Char str[]={‘t’,’e’,’c’,’h’,’p’,’I’,’l’,’e’,};

Input and output of string type value thare is formate specifier :%s

Printf(“%s”,str);

**User Input to string type variable:**

Char str[50];

Scanf(“%s”,str);//one-word string

Gets(str); multiword string-terminate by enterkey

Int arr[5];

To store value on each index we execute a loop

But in array of character type (string):there is %s. By using %s , you can store and fetch all index value of array at one time.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |

Char name[100];

Reading function=====

String Pre-defined Function : There are some pre defined function in c that makes our task easy . all string function are stored in a library

“string.h”

You just need to know the syntax of function and can directly call it when ever you need it.

Strlen() -> is a function used to find out the length of the string it returns a unsigned integer. Takes a string type argument.

Strcpy() : -is a function used to copy a string to another string. It require two string type argument.

Ex:

Char name[100];

Name=”suraj”;//wrong

Strcpy(name,”suraj”);//right way

Strcpy(name,firstname);//right way

Strncpy() :-

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

Int val=strcmp(string\_1,string\_2);

Val==0 if string\_1 and string\_2 is equal.

Same string different in case are not equal by strcmp() function.

Strcmpi():- It is same as strcmp()

But it ignores the cases of character of both string . only character of string should be same.

It also takes two string type argument and return a integer type value.

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

But it compare both string till a given length. It need 3 argument first two string of type and last one of integer type. And return a integer type value same as strcmp().

Strnicmp(): - ait same as strncmp() but it ignores the cases used in both string

So it compare two string two string till n number of character and ignores the cases of string

It return 0 if first n character are same or return any positive are negative value.

Syntax:-

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

Ex:-

Int result=strnicmp(“techpile”,”technology”,4);

Strrev():- it reverse the value a string . it takes only one argument and after the execution of function string value will be reverse of previous value.

Syntax: strrev(string\_1);

Ex:- char name[]=”ram”;

Printf(“%s”,name);//tech

Strrev(name);

Printf(“%s”,name);//hcet

wap to check a given string is a palindrome string or not by using pre defined function

1.enter the string char name[100]=”div”,rev[100]

2.strcpy(rev,name);

3.reverse the string by using strrev()function

4.compare rev,name:strcmpi(rev,name);

5.check the return value of strcmp()

Strlwr():- is used to convert all character of string to the lowercase.

It accept only one argument and set the argument value to the lowercase

Char name[]=”techpile”

Syntax:strlwr(name);

//name:”techpile”

Strupr():- is a function accept only one string type argument and sets the argument value to its uppercase.

Strcat() :- function is used to concatenate to string type value and assign into first argument .

Syntax : strcat(string\_1,string\_2);

It takes 2 string type value as argument and return 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 string will be concatened to the first string.

Strncat(string\_1,string\_2,N);

Strrev()

Strtkn()

UDF:Difinition,

**26/10/2021**

**Pointer:-**

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

Mainly two symbol used with pointer :-

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

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

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

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

**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();

}

//wap to find factorial by using recursion

Void main()

{

Getfact(5);

}

Void getfact(int n)

{

If(n==1)

Return 1;

Else

Return n\*f=getfact(n-1);

}