* C language was developed by **Dennis Richie** and he belong to **Bell Labs** company.
* C, Java, Python comes under high level language.

**Basic Structure of C language:**

* Comments
* Include header file
* User defined Functions
* main method
* inside main method variable declaration
* at last save the file

example:

#include<stdio.h>

Fun()

{

Stats;

}

void main()

{

// variable declaration code

}

Save the file

Hello, c-->compiling-->execute the program --> output to the program

**IDE:**

* Integrated development environment.
* It provides an environment to write program, to compile and execute.
* IDE is a software application which helps programmers to develop a software code.

**temp=n**

|  |  |  |  |
| --- | --- | --- | --- |
| **Armstrong number** | **Reverse of a number** | **Palindrome number** | **Sum of numbers** |
| **r=temp%10**  **sum=sum +r\*r\*r**  **temp=temp/10** | **r=temp%10**  **sum=sum\*10+r**  **temp=temp/10** | **r=temp%10**  **sum=sum\*10+r**  **temp=temp/10** | **r=temp%10**  **sum =sum+r**  **temp=temp/10** |
| use if and else | Don’t use if and else | use if and else | Don’t use if and else |

**Programs**

1. Prime number
2. Prime series
3. Prime number between A and B
4. Prime whose sum of digits is even
5. Reverse of a number
6. Generating all the series of reverse numbers
7. Palindrome
8. Generating all palindrome number still 1000
9. Fib series
10. Factorial
11. Factorial series
12. Armstrong number
13. Armstrong series still 1000
14. Basic calculator using switch

**KEY WORDS:**

1.INT

2.CHAR

3.FLOAT

4.IF

5.BREAK

6.DOUBLE

7.LONG

8.FOR

9.WHILE

10.DO

11.ELSE

12.signed

13.void

14.switch

15.goto

16.struct

17.auto

18.typedef

19.static

20.continue

21.enum

22.sizeof

23.case

24.register

25.union

26.volatile

27.unsigned

28.short

29.Default

30.Const

31.extern

**Data Types**

* Int
* Float
* Char
* Short int, longed int ,unsigned int
* Long double

**ACCESSING STRUCTURE DATA USING POINTERS AND ARRAYS**

#include<stdio.h>

struct s

{

int roll;

char name[20];

int age;

};

void main()

{

int i;

struct s s1[2];

p=s1;

for(i=0;i<2;i++)

{

scanf("%d",&s1[i].roll);

scanf("%s",&s1[i].name);

scanf("%d",&s1[i].age);

}

for(i=0;i<2;i++)

{

printf("%d\t",p->roll);

printf("%s\t",p->name);

printf("%d\n",p->age);

p++;

}

}

**ACCESSING STRUCTURE DATA USING FUNCTIONS**

#include<stdio.h>

struct s

{

int roll;

char name[20];

int age;

};

void func(struct s s2)

{

printf("%d\t",s2.roll);

printf("%s\t",s2.name);

printf("%d\n",s2.age);

}

void main()

{ int i;

struct s \*p;

struct s s1;

scanf("%d",&s1.roll);

scanf("%s",&s1.name);

scanf("%d",&s1.age);

func(s1);

}

**ACCESSING STRUCTURE DATA USING FUNCTIONS AND POINTERS**

#include<stdio.h>

struct s

{

int roll;

char name[20];

int age;

};

void func(struct s \*P)

{

printf("%d\t",P->roll);

printf("%s\t",P->name);

printf("%d\n",P->age);

}

void main()

{ int i;

struct s \*p;

struct s s1;

scanf("%d",&s1.roll);

scanf("%s",&s1.name);

scanf("%d",&s1.age);

func(&s1);

}

**ADD TWO NUMBERS IN A FUNCTION USING STRUCTURES**

1. **Without return type**

#include<stdio.h>

struct complex

{

int real;

float imaginary;

};

void add(struct complex c1,struct complex c2)

{

struct complex c3;

c3.real= c1.real + c2.real;

c3.imaginary=c1.imaginary+c2.imaginary;

printf("sum of two complex numbers = %d + %.2fi",c3.real,c3.imaginary);

}

void main()

{

struct complex c1,c2;

printf("Enter the 1st complex number : \n");

scanf("%d %f",&c1.real,&c1.imaginary);

printf("Enter the 2nd complex number : \n");

scanf("%d %f",&c2.real,&c2.imaginary);

add(c1,c2);

}

1. **With return type**

#include<stdio.h>

struct complex

{

int real;

float imaginary;

};

struct complex add(struct complex c1,struct complex c2)

{

struct complex c3;

c3.real= c1.real + c2.real;

c3.imaginary=c1.imaginary+c2.imaginary;

return c3;

}

void main()

{

struct complex c1,c2,sum;

printf("Enter the 1st complex number : \n");

scanf("%d %f",&c1.real,&c1.imaginary);

printf("Enter the 2nd complex number : \n");

scanf("%d %f",&c2.real,&c2.imaginary);

sum=add(c1,c2);

printf("sum of two complex numbers = %d + %.2fi",sum.real,sum.imaginary);

}

**STORAGE CLASSES :**

* Storage classes provide the following information to the compiler

1. Storage area of a variable
2. Scope of variable i.e., in which block the variable is visible.
3. The lifetime of a variable i.e., how long the variable will be there in active mode.
4. The default value of a variable if it is not initialized.

* Storage classes are :

1. Auto
2. Static
3. Extern
4. register

* Depending on the behavior and storage area storage classes are classified into two types, such as

1. Automatic storage class
2. Static storage class
3. **Automatic Storage type:** This storage class variables will be created automatically ad destroyed automatically. Automatic storage class variables will be store in the stack area of the data segment or in the CPU register. Under automatic storage class, we have two types of storage class specifiers, such as
4. Auto
5. Register

**What is the static storage class ?**

Static storage class variables will be created only once and throughout

the program, it will be there in active mode, static storage class variables

will be there in the static area of the data segment, under static storage class, we have two of types storage class specifiers

1. Static
2. Extern

Extern can be used both variables and functions

Use extern we can share data and functions between program units

Code – 01 :

#include<stdio.h>

void abc();

int main()

{

abc();

abc();

abc();

return 0;

}

void abc()

{

auto int a=5;

++a;

printf("\n a = %d",a);

}

Code – 02 :

#include<stdio.h>

void abc();

int main()

{

auto int a=5;

++a;

abc();

abc();

++a;

printf("\n a = %d",a);

return 0;

}

void abc()

{

int a =10;

++a;

printf("\n a = %d",a);

}

Code - 03 :

#include<stdio.h>

int a=10;

void abc();

int main()

{

abc();

abc();

printf("\n a = %d",a);

return 0;

}

void abc()

{

static int a=10;

++a;

printf("\n a = %d",a);

}

Code – 04 )

a)

#include<stdio.h>

int main()

{

register int i = 10;

int \*a = &i; // ERROR you can’t do like this

printf("%d",\*a);

getchar();

return 0;

}

b)

#include<stdio.h>

int main()

{

int i = 10;

register int \*a = &i;

printf("%d",\*a);

getchar();

return 0;

}

**FILES IN C :**

* To use a file four essential actions should be carried out. These are,

1. Declare a file pointer variable.
2. Open a file using the fopen() function.
3. Process the file using suitable functions.
4. Close the file using the fclose() and flush() functions.
   1. . Declaration of file pointer:

FILE \*file\_pointer\_name;

Example : FILE \*fp;

2.2. Opening a file :

FILE \*fopen(const char \*fname,const char \* mode);

Example :

* Modes for opening a file :

1. r opens a text file in reading mode
2. w opens or create a text file in writing mode
3. a opens a text file in append mode
4. r+ opens a text file in both reading and writing mode
5. w+ opens a text file in both reading and writing mode
6. a+ opens a binary file in reading mode
7. rb opens a binary file in reading mode
8. wb opens or create a binary file in reading mode
9. ab opens a binary file in append mode
10. rb+ opens a binary file in both reading and writing mode
11. wb+ opens a binary file in both reading and writing mode
12. ab+ opens a binary file in both reading and writing mode

CODE – 01 :

#include<stdio.h>

void main()

{

FILE \*fp;

char fname[30];

scanf("%s",&fname);

fp=fopen(fname,"r");

if(fp==NULL)

{

printf("File does not exist");

}

else

{

printf("File exists = %d",fp);

}

}

CODE – 02 :

#include<stdio.h>

void main()

{

FILE \*fp;

char fname[30],ch;

scanf("%s",&fname);

fp=fopen(fname,"r");

if(fp!=NULL)

{ ch=getc(fp);

while(ch!=EOF)

{

printf("%c",ch);

ch=getc(fp);

}

}

fclose(fp);

}