1.If else statement

In this program you see we get value from the user using the scanf method

you have to pass the format specifier and variable name

In if (condition) we write a>b or not if yes so if statement is execute and not so else statement is excecutes

#include<stdio.h>

*int* main()

{

*int* a,b=50;

    printf("Enter your number: ");

    scanf("%d",&a);

    if(a>b)

    {

        printf("a is greater than b");

    }

    else

    {

        printf("b is greater than a");

    }

    return 0;

}

1.1 If elseif else statement

In this program we define the else if when we need more than 2 condition then else if statement is used here we define the three condition a>0 means positive a<0 means negative and else if above condition is not true so else is execute means number is zero

#include<stdio.h>

*int* main()

{

*int* a;

    printf("Enter your number: ");

    scanf("%d",&a);

    if(a>0)

    {

        printf("a is positive");

    }

    else if(a<0)

    {

        printf("a is negative");

    }

    else

    {

        printf("a is zero");

    }

    return 0;

}

1.3 Short hand if else

Here we define if else but in short manner a>18 then you vote it is if statement another is else statement

#include<stdio.h>

*int* main()

{

*int* a = 20;

    (a >18) ? printf("You can vote.") : printf("You can not vote.");

    return 0;

}

1.4 Switch case

Switch case statement is used when we have a selected number character or a string just define a case and implement the switch case statement

the default statement is executes when above case is not match.

#include<stdio.h>

*int* main()

{

*int* a ;

    printf("enter your number ");

    scanf("%d",&a);

    switch(a)

    {

        case 1:

            printf("Sunday");

            break;

        case 2:

            printf("Monday");

            break;

        case 3:

            printf("Tuesday");

            break;

        case 4:

            printf("Wednesday");

            break;

        case 5:

            printf("Thursday");

            break;

        case 6:

            printf("Friday");

            break;

        case 7:

            printf("Saturday");

            break;

        default:

            printf("Please enter 1 to 7");

    }

}

**2. Loops**

Loop is used when we need to print or do a task repeatedly then loop is used there are three types of loops is available do- while , while loop, for loop

And for loop we need to initialize it check the condition and increment/decrement it

2.1 While loop

Here first we initialize i=0 then we specify condition i<5 means i will executed until this condition becomes true and when true the loop will terminated then we simple increment i ++;

#include <stdio.h>

*int* main()

{

*int* i = 0;

    while (i < 5)

    {

        printf("%d\n", i);

        i++;

    }

}

2.2 Do-while loop

Here first we initialize i=0 then we specify condition i<=4 means i will executed until this condition becomes true and when true the loop will terminated then we simple increment i ++;

#include <stdio.h>

*int* main()

{

*int* i = 0;

    do

    {

        printf("%d\n",i);

        i++;

    } while (i<=4);

}

2.3 For loop

Here initialize condition and increment is in one line the for loop is used more than above 2.

#include <stdio.h>

*int* main()

{

    for(*int* i=0;i<=4;i++)

    {

        printf("%d\n",i);

    }

}

3. Break and continue

3.1 Break

when you write break statement then the loop will terminated here we define if condition and when i becomes 4 we write break means lop will terminate.

#include <stdio.h>

*int* main()

{

*int* i;

    for (i = 0; i < 10; i++)

    {

        if (i == 4)

        {

            break;

        }

        printf("%d\n", i);

    }

}

3.2 Continue

when we write continue it skip the particular number means continue will skip particular condition and break will terminate the whole loop.

#include <stdio.h>

*int* main()

{

*int* i;

    for (i = 0; i < 6; i++)

    {

        if (i == 4)

        {

            continue;

        }

        printf("%d\n", i);

    }

}

**4. Array**

Array means a one type of data type which can store multiple element of single data type means you can create array of int, float, char, double, string

4.1 Array

Here we define int a[] means a is variable of array type ={1,2,3} then we initialize it with the variable and simple print it with its index which are a[0],a[1],a[2] array index is always start with the 0.

#include <stdio.h>

*int* main()

{

*int* a[]={1,2,3};

    printf("%d",a[0]);

    printf("\n");

    printf("%d",a[1]);

    printf("\n");

    printf("%d",a[2]);

    printf("\n");

}

4.2 Array print using a loop

#include <stdio.h>

*int* main()

{

*int* a[]={1,2,3};

    for(*int* i=0;i<3;i++)

    {

        printf("%d",a[i]);

        printf("\n");

    }

}

4.3 Array scanf using loop

Here we define the 3 size of array then simple take it from user and print it using the for loop.

#include <stdio.h>

*int* main()

{

*int* a[3];

*int* i;

    for(i=0;i<3;i++)

    {

        printf("Enter your number ");

        scanf("%d",&a[i]);

    }

    for(i=0;i<3;i++)

    {

        printf("%d",a[i]);

        printf("\n");

    }

}

**5. String**

String is one type of data type in c if you make a string so you need to make a array of the char so then you can access the string.

Here we make a array of char and initialize it with one string and print it we use %s as format specifier so it prints whole string and then we use %c and a[0] so it print the particular character.

#include <stdio.h>

*int* main()

{

*char* a[]={"Student Bread"};

    printf("%s",a);

    printf("\n");

    printf("%c",a[0]);

}

**6. Memory address**

The variable which we declare it can store in memory so if we have to fetch memory address so we use %p format specifier for the memory address.

#include <stdio.h>

*int* main()

{

*int* a=45;

*float* b=25.25;

    printf("%p",a);

    printf("\n");

    printf("%p",b);

    return 0;

}

**7. Pointer**

Pointer means it can store the memory address of variable here we initialize one variable int a=45; then we define pointer type int variable int\* b=&a; and gave the reference of the a. The we print the value of a then we print address of a by %p and then we print again its address with pointer variable.

#include <stdio.h>

*int* main()

{

*int* a=45;

*int*\* b=&a;

    printf("%d",a);

    printf("\n");

    printf("%p",&a);

    printf("\n");

    printf("%p",b);

    return 0;

}

**8.Function in c**

Function means a block of code which executes when its call if we need function so we need to follow 3 steps function declaration, function call, function definition.

8.1 Function without argument without return type

Here it is a function which print no argument and no return type it will run when you call in main function it will run. It has no argument and no return type because void has no return type.

#include <stdio.h>

*void* Myfunc();

*int* main()

{

    Myfunc();

    printf("\n");

    Myfunc();

    return 0;

}

*void* Myfunc()

{

    printf("no argument and no return type");

}

8.2 Function with argument with return type

Here it is a function with return type and with argument we gave argument int a,int b then we return it a+b then in main function we store it’s value in another variable and simple print it.

#include <stdio.h>

*int* Myfunc(*int* *a*,*int* *b*);

*int* main()

{

*int* ans=Myfunc(12,12);

    printf("Your add is %d",ans);

    return 0;

}

*int* Myfunc(*int* *a*,*int* *b*)

{

    return *a*+*b*;

}

8.3 Function with argument and no return type

Here we declare void type of function o it has no return value but we gave argument so then print it in the function and simple call the function in main function.

#include <stdio.h>

*void* Myfunc(*int* *a*,*int* *b*);

*int* main()

{

    Myfunc(45,5);

    return 0;

}

*void* Myfunc(*int* *a*,*int* *b*)

{

*int* c;

    c=*a*-*b*;

    printf("Your ans is %d",c);

}

8.4 Function without argument and with return type

Here we just define the function with it return type and without argument and simply print the value by calling from main method.

#include <stdio.h>

*int* Myfunc();

*int* main()

{

*int* c=Myfunc();

    printf("%d",c);

    return 0;

}

*int* Myfunc()

{

*int* a=5;

    return a;

}

**9.Structure**

Structures (also called structs) are a way to group several related variables into one place. Each variable in the structure is known as a member of the structure.

You can create a structure using the struct keyword.

Then in main function you need to make its instances and access it’s variable.

#include <stdio.h>

*struct* MyStruct{

*int* a;

*char* letter;

};

*int* main()

{

*struct* MyStruct ms;

    ms.a=45;

    ms.letter='S';

    printf("A in structue is %d\n",ms.a);

    printf("letter in structue is %c",ms.letter);

    return 0;

}