Programming for problem solving using C

123ES

Ms. Urvashi Gohil
Assistant Professor

Birla Vishvakarma Mahavidhyalaya (BVM) Engineering College
Department of Computer Engineering

January 31, 2025

Outline of the course :

Conditional branching

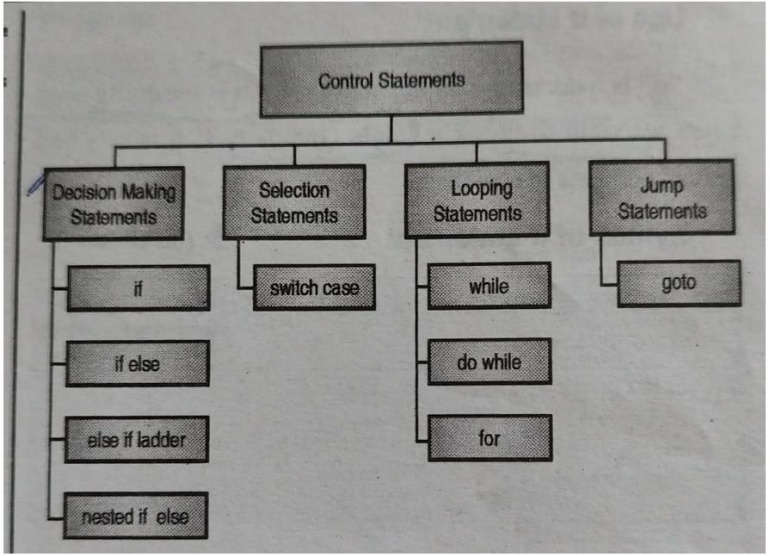
Unconditional branching

Looping

break and continue

Conditional branching

Conditional Statements



Conditional Statements

Conditional Statements in C

If-else

If

```
if(condition)
{
        //true
}
```

If-else

```
if(condition)
{
        //true
}
else
{
        //false
}
```

If-else-if

```
if(condition 1)
{
        //true
}
else if(condition 2)
{
        //true
}
else
{

}
```

Nested-if

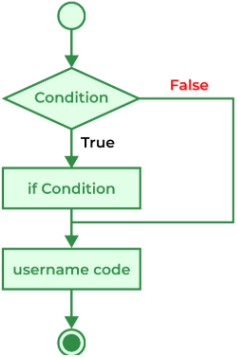
```
if(condition 1)
{
        if(condition)
        {
        }
        else
        {
        }
}
else
{
        if(condition)
        {
        }
        else
        {
        }
}
```

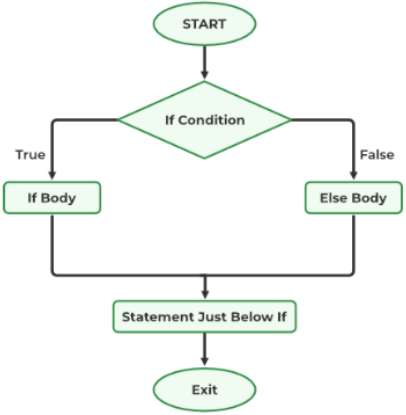
(condition) ? true : false

Switch

```
Switch expression
{
        case 1;
        break;

        case 2;
        break;

        case 3;
        break;

        default;
}
```

Conditional Statements- IF, If-else





Task

Write a program which does the following

- Take input for two integer variables `a` & `b`
- Output "Coding is Fun" to the console if a is greater than b.

Sample 1:

Input	Output
25 20	Coding is Fun

Sample 2:

Input	Output
20 20	

if


```c
#include <stdio.h>

int main() {
    int a;
    int b;

    scanf("%d", &a);
    scanf("%d", &b);

    if (a > b) {
        printf("Coding is Fun");
    }
}
```

Sample Input

```
25
20
```

Your Output

```
Coding is Fun
```

If();

abc > C getc.c > main()

```c
#include<stdio.h>
int main(){
    //if(); -->; makes sentence empty and print next two statements

     if(0); // means what if condition is false-->0?

     printf("its true");

     printf("\nits wrong");

}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'getc.exe'
its true
its wrong
PS F:\code1\abc\output>
```

Task

Write a program which does the following:

- Let's think of a real-life example where we need to find out if a person is old enough to vote.
- Find out if the age (25) is greater than OR equal to the voting age limit, which is set to 18.
- Declare the variables `age` and `voting_age` - and initialize them to the `values` `25` and `18` - i.e. the age and the voting age respectively.
- Compare `age` and `voting_age` using the syntax given above and output the following
  - `"Old enough to vote!"` if `age` is greater than or equal to `voting_age`
  - `"Not old enough to vote."` if `age` is lesser than `voting_age`

if-else


```c
#include <stdio.h>

int main() {
    int age = 25;
    int voting_age = 18;

    if (age >= voting_age) {
        printf("Old enough to vote!");
    } else {
        printf("Not old enough to vote.");
    }
}
```

Your Output

```
Old enough to vote!
```

If without {}

cssp1.html | getc.c 1 | get.c

abc > getc.c > main()

```c
#include<stdio.h>
int main(){
    int a = 10;
    int b = 20;
    if(a>b)
        printf("a is smallest number");
        printf("b is biggest number");

}
```

PROBLEMS 1 | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'getc.exe'
b is biggest number
PS F:\code1\abc\output>
```

If condition without {curly braces/ body} will execute first statement if condition true

If condition false, it will execute next statement.

If else with printf scanf

cssp1.html | getc.c | get.c

abc > getc.c > main()

```c
int main(){
    int a;
if(scanf("%d",&a) && printf("The number is : %d", a)){
    printf("\nyoure correct");
}
else printf("\nhello world");
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'getc.exe'
4
The number is : 4
youre correct
PS F:\code1\abc\output>
```

If else with printf scanf

abc > C ifprin.c > main()

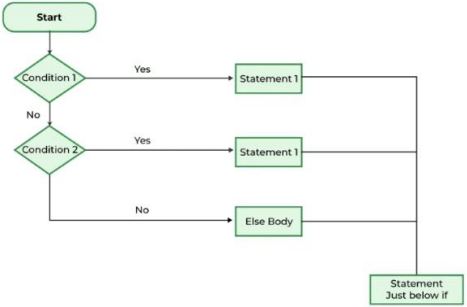
```c
#include <stdio.h>

int main() {
    int a;
    if (printf("Hello, World!\n") && scanf("%d", &a)) {  // printf returns a nonzero value (number of characters p
        printf("Condition is TRUE\n");                   //Scanf also returns int(non zero value)
    } else {
        printf("Condition is FALSE\n");
    }
    return 0;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'ifprin.exe'
Hello, World!
3
Condition is TRUE
PS F:\code1\abc\output>
```

If Else If Statement

if else if ladder is an extension of if else statement used to test a series of conditions sequentially, executing the code for the first true condition. A condition is checked only if all previous ones are false. Once a condition is true, its code block executes, and the ladder ends"



If-Else-if

abc > C elif.c > main()

```c
#include<stdio.h>
int main(){
int grade = 85;
if (grade >= 90) {
    printf("You got an A");
} else if (grade >= 80) {
    printf("You got a B");
} else if (grade >= 70) {
    printf("You got a C");
} else {
    printf("You need to study more");
}
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'elif.exe'
You got a B
PS F:\code1\abc\output>
```

Else-if

Task

Write a program which does the following

- Take two integers `b` and `r` as input
- Print *"Rob scored higher marks than Bob"*, if `r` is greater than `b`
- Print *"Bob & Rob both scored the same"*, if both `b` and `r` are equal

Sample 1:

Input	Output
20 25	Rob scored higher marks than Bob

Sample 2:

Input	Output
15 15	Bob & Rob both scored the same

If-Else-if


```c
#include <stdio.h>

int main() {
    int b, r;
    scanf("%d", &b);
    scanf("%d", &r);
    if(r>b){
        printf("Rob scored higher marks than Bob");
    }
    else if(b == r){
        printf("Bob & Rob both scored the same");
    }

}
```

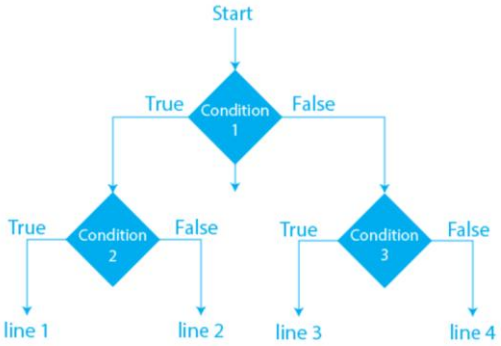
Sample Input

```
20
25
```

Your Output

```
Rob scored higher marks than Bob
```

Nested Else-if



Nested Else-if

abc > C ifels.c > main()

```c
#include <stdio.h>
int main()
{
    int i = 10;
    if (i == 10) {
        // First if statement
        if (i < 15)
            printf("i is smaller than 15\n");

        if (i < 12)
            printf("i is smaller than 12 too\n");
        else
            printf("i is greater than 15");
    }
    else {
        if (i == 20) {

            if (i < 22)
                printf("i is smaller than 22 too\n");
            else
                printf("i is greater than 25");
        }
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
i is smaller than 15
i is smaller than 12 too
```

Nested Else-if

Atm money Withrow

Step-1 check the correct pin (pin_entered = correct_pin)
Step-2 check the amount > 0 or not? If not then invalid amount
Step-3 check the amount <= balance (Sufficient amount)
if yes→ Successful transition
if not→ not enough balance

Q: If the number is even or odd, and then if it is even, whether it is divisible by 4 or not, and if it is odd, whether it is divisible by 3 or not

Nested Else-if

abc > C p-1.c > main()

```c
#include<stdio.h>
int main(){
    int enter_pin, correct_pin, amount, balance;
    correct_pin = 1234;
    balance = 100;
    printf("Enter the pin number: ");
    scanf("%d", &enter_pin);
    if(enter_pin == correct_pin)
    {
      printf("Enter the amount: ");
      scanf("%d", &amount);
        if(amount > 0)
        {
            if(amount <= balance)
            {
                printf("Successful transaction\n");
            }
            else {
                printf("Insufficient balance\n");
            }
        }
        else {
            printf("Enter a valid amount\n");
        }
    }
    else {
        printf("Enter a valid pin number\n");
    }
}
```

abc > C odev.c > main()

```c
#include<stdio.h>
int main(){
    int n;
    printf("Enter the number:");
    scanf("%d", &n);

    if(n%2 == 0){
        printf("Even number");

        if(n %4 == 0){
            printf("\nnumber is div by 4");
        } else printf("\nnumber is not div by 4");
    }
    else{
        printf("Odd number");
        if (n % 3 == 0){
            printf("\nnumber is div by 3");
        }
        else printf("\nnumber is not div by 3");
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'odev.exe'
Enter the number:3
Odd number
number is div by 3
PS F:\code1\abc\output>
```

Operator associativity

3	*,/,%	Multiplication, division, modulus	Left-to-Right
4	+/-	Addition, subtraction	Left-to-Right
5	<< , >>	Bitwise shift left, Bitwise shift right	Left-to-Right
6	< , <=	Relational less than, less than or equal to	Left-to-Right
	> , >=	Relational greater than, greater than or equal to	
7	== , !=	Relational is equal to, is not equal to	Left-to-Right

Common Programming error for Else-if

abc > C cpe1.c > main()

```c
#include<stdio.h>
int main(){
    int x = 5;
    if(0 <= x <=4){
        printf("condition is true!");
    }
    else printf("condition is false");
}
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'cpe1.exe'
condition is true!
PS F:\code1\abc\output>
```

Left to right associativity
For any positive value of x, condition will be true,

0<=x→1
1 <=4 →1

Common Programming error for Else-if

abc > C cpe1.c > main()

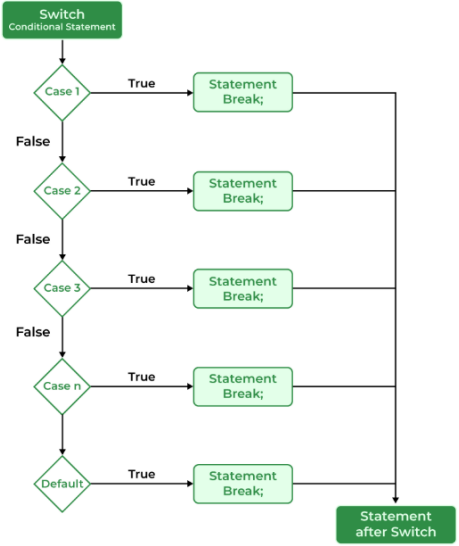
```c
#include<stdio.h>
int main(){
    int x = 15;
    if(x = 10)
    {
     printf("x is 10");
    }
    else printf("x is 15");
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'cpe1.exe'
x is 10
PS F:\code1\abc\output>
```

= is assignment operator

Switch Case



Permitted datatype for switch:
Int, ch, short
Not permitted: float and double

Switch case

abc > C p-3.c > main()

```c
#include<stdio.h>
int main(){
    int grade = 79;
    int score = grade/10;

    switch (score)
    {
    case 9:
        printf("You Got A");
        break;
    case 8:
        printf("you got B");
        break;
    case 7:
        printf("you got C");
        break;
    default:
        printf("you got D");
        break;
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> & .\'p-3.exe'
you got C
PS F:\code1\abc\output>
```

Switch case with Enum

abc > C enum.c > main()

```c
#include<stdio.h>
enum Week {
    sunday, monday, tuesday, wed, thursday, friday, saturday
}w;
int main(){
    printf("Enter the day: ");
    scanf("%d", &w);
    switch (w)
    {
    case sunday:
        printf("Holiday");
        break;
    case monday:
        printf("Working Day");
        break;
    case tuesday:
        printf("Tuesday");
        break;
    case wed:
        printf("early free");
        break;
    case thursday:
        printf("thursday");
        break;
    case friday:
        printf("friday");
        break;
    case saturday:
        printf("saturday");
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'enum.exe'
Enter the day: 3
early free
PS F:\code1\abc\output>
```

Enum

Enumeration (or enum) is a user defined data type in C. It is mainly used to assign **names to integral constants**, the names make a program easy to read and maintain.

abc > C en.c

```c
#include<stdio.h>
enum Os_states {
    new = 1, ready, run, block, terminate
}state;

int main(){
    state = run;
    printf("%d", state); // op is 3 if default (new = starting from 0) op is 2
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'en.exe'
3
PS F:\code1\abc\output>
```

Switch case with Enum

Facts about enum:

1. Two enum names can have same value.
2. If we do not explicitly assign values to enum names, the compiler by default assigns values starting from 0.
3. We can assign values to some name in any order. All unassigned names get value as value of previous name plus one.
4. The value assigned to enum names must be some integral constant, i.e., the value must be in range from minimum possible integer value to maximum possible integer value.
5. All enum constants must be unique in their scope.

Example:

1. Traffic signal system with enum and switch
2. Create a ticket booking system with enum and switch

Switch case with Enum

abc > C en.c > main()

```c
#include<stdio.h>
enum traffic_signals{
    yellow =1 , red, green
}ts;
int main(){
    printf("Enter 1 for yello , 2 for red, 3 for green :");
    scanf("%d", &ts);
    switch (ts)
    {
    case 1:
        printf("Ready to go");
        break;
    case 2:
        printf("Stop");
        break;
    case 3:
        printf("Go");
        break;
    default:
        printf("Study the traffic rules!");
        break;
    }
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'en.exe'
Enter 1 for yello , 2 for red, 3 for green :1
Ready to go
PS F:\code1\abc\output>
```

Switch case with Enum(Typecasting)

cssp1.html | getc.c | ifprin.c | isp.c | q-1.c | get.c

abc > q-1.c > tf

```c
#include <stdio.h>
enum traffic_signals {
    red,
    yellow,
    green
}tf;
int main() {
    int tf;
    printf("Enter the traffic signal (0 for red, 1 for yellow, 2 for green): ");
    scanf("%d", &tf);
    enum traffic_signals ts = (enum traffic_signals)tf;
    switch (ts) {
        case red:
            printf("Stop! Red light.\n");
            break;
        case yellow:
            printf("Caution! Yellow light.\n");
            break;
        case green:
            printf("Go! Green light.\n");
            break;
        default:
```

PROBLEMS | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS

```
Go! Green light.
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'q-1.exe'
Enter the traffic signal (0 for red, 1 for yellow, 2 for green): 1
Caution! Yellow light.
PS F:\code1\abc\output>
```

Compiled

Switch case with if-else

abc > C switchifelse.c > main()

```c
#include <stdio.h>
int main() {
    int choice, beverageType;
    printf("Restaurant Menu:\n");
    printf("1. Burger\n");
    printf("2. Beverage\n");
    printf("Enter your choice (1-3): ");
    scanf("%d", &choice);
    switch (choice) {
        case 1:
            printf("You ordered a Burger. Preparing your order...\n");
            break;
        case 2:
            printf("Choose Beverage Type:\n");
            printf("1. Hot Coffee\n");
            printf("2. Cold Juice\n");
            printf("Enter your choice (1-2): ");
            scanf("%d", &beverageType);

            if (beverageType == 1) {
                printf("Serving a Hot Coffee.\n");
            } else if (beverageType == 2) {
                printf("Serving a Cold Juice.\n");
            } else {
                printf("Invalid choice! Serving water instead.\n");
            }
            break;
        default:
            printf("Invalid selection! Please choose a valid menu item.\n");
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Nested Switch

Restaurant ordering system:

- Two variables category & food choice.
- Category stores two item fast food and beverages. (Use switch case for this 2)
- Food choice: pizza , burger (use switch case for this choice)
- Beverage's choice: coffee, cold drinks(use switch case for this)

```c
#include <stdio.h>
int main() {
    int category, foodChoice;
    // Displaying the main menu categories
    printf("Welcome to the Restaurant!\n");
    printf("Choose a category:\n");
    printf("1. Fast Food\n");
    printf("2. Beverages\n");
    printf("Enter your choice (1-2): ");
    scanf("%d", &category);
    switch (category) {
        case 1: // Fast Food category
            printf("Fast Food Menu:\n");
            printf("1. Burger\n");
            printf("2. Pizza\n");
            printf("Enter your choice (1-2): ");
            scanf("%d", &foodChoice);
            switch (foodChoice) {
                case 1:
                    printf("You ordered a Burger. Preparing your order...\n");
                    break;
                case 2:
                    printf("You ordered a Pizza. Preparing your order...\n");
                    break;
                default:
                    printf("Invalid choice in Fast Food category.\n");
            }
            break;
        case 2: // Beverages category
            printf("Beverage Menu:\n");
            printf("1. Hot Coffee\n");
            printf("2. Cold Juice\n");
            printf("Enter your choice (1-2): ");
            scanf("%d", &foodChoice);
            switch (foodChoice) {
                case 1:
                    printf("Serving a Hot Coffee.\n");
                    break;
                case 2:
                    printf("Serving a Cold Juice.\n");
                    break;
                default:
                    printf("Invalid choice in Beverages category.\n");
            }
            break;
        default:
            printf("Invalid category selection!\n");
    }
}
```

Switch with supportive datatype

As switch does not support float, this example show how to map float values in supportive data type(int, ch)

abc > cgpa.c > main()

```c
#include <stdio.h>

int main() {
    float cgpa;
    int gradeCategory; // Integer mapping for CGPA

   printf("Enter cgpa:");
   scanf("%f", &cgpa);

   if(cgpa >= 8.0 && cgpa <=100.0)
   {
    gradeCategory = 1;
    }
    else if (cgpa >= 7.9 && cgpa <= 7.0)
    {
        gradeCategory = 2;
    }

   printf("To check your award please enter your gradecategory");
   scanf("%d",&gradeCategory);


   switch (gradeCategory)
   {
   case 1:
    printf("You won gold medal");
    break;

   case 2:
    printf("You won silver medal");
    break;
   default:
    printf("you won chocolet box");
    break;
   }
}
```

Goto Statement

- jump to some part of the code

The goto statement allows transfer control of the program to the specified label

abc > C goto.c > main()

```c
#include <stdio.h>

int main() {
    int n = 24;

    // If the number is even, jump to
      // jump_here label
    if (n %2 == 0)
        goto jump_here;

    // This will be skipped
    printf("odd number \n", n);

jump_here:
    printf("Even number\n");
    return 0;
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Even number
PS F:\code1\abc\output>
```

Goto Statement

abc > C gotojump.c > main()

```c
#include <stdio.h>
int main()
{ int i,j;
    for (i = 0; i < 6; i++ )
    {
        printf( "Outer loop executing. i = %d\n", i );
        for (j = 0; j < 2; j++ )
        {
            printf( " Inner loop executing. j = %d\n", j );
            if ( i == 1 )
                goto stop;
        }
    }
    /* This message does not print: */
    printf( "Loop exited. i = %d\n", i );
    stop: printf( "Jumped to stop. i = %d\n", i );
}
```

```
PS F:\code1\abc\output> & .\'gotojump.exe'
Outer loop executing. i = 0
 Inner loop executing. j = 0
 Inner loop executing. j = 1
Outer loop executing. i = 1
 Inner loop executing. j = 0
Jumped to stop. i = 1
PS F:\code1\abc\output>
```

GATE 1994 - Goto Statement [1 mark]

An unrestricted use of the “goto” statement is harmful because

(a) it makes it more difficult to verify programs

(b) it increases the running time of the programs

(c) it increases the memory required for the programs

(d) it results in the compiler generating longer machine code

While loop

While (condition)
statement_to_repeat

While(Condition){
statement 1;
.
.
Statement N;

}

Counter controlled loop & sentinel control loop

abc > C whil.c > main()

```c
#include<stdio.h>
int main(){
    int count;
    printf("Enter the count:");
    scanf("%d",&count);

    printf("countdown starts!");

    while (count >= 0)
    {
        sleep(1);
        printf("\n%d", count);
        count --;
    }

    printf("\nTimes up");
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'whil.exe'
Enter the count:3
countdown starts!
3
2
1
0
Times up
PS F:\code1\abc\output>
```

While loop

abc > C factwhil.c > main()

```c
#include<stdio.h>
int main(){
    int fact = 1 , n;
    printf("Enter the number:");
    scanf("%d", &n);
    while(n>1){
        fact  *=  n ;  //fact = fact * n;
        n--;
    }
    printf("\nThe fact is:  %d ",fact);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'factwhil.exe'
Enter the number:8

The fact is:  40320
PS F:\code1\abc\output>
```

While loop – test cases

abc > C w.c > main()

```c
#include<stdio.h>
int main(){
  int i = 1;
  while(i ==1){
    printf("%d", i);
    i++;
  }
  printf("\nThe value of i is %d", i);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'w.exe'
1
The value of i is 2
PS F:\code1\abc\output>
```

While loop- assignment operator- infinite loop

abc > C w.c > main()

```c
#include<stdio.h>
int main(){
 int i = 1;
 while(i =1){
   printf("%d", i);
   i++;
 }
 printf("\nThe value of i is %d", i);
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
111111111111111111111111111111111111111111111111111111111111111111111111
111111111111111111111111111111111111111111111111111111111111111111111111
111111111111111111111111111111111111111111111111111111111111111111111111
111111111111111111111111111111111111111111111111111111111111111111111111
111111111111111111111111111111111111111111111111111111111111111111111111
111111111111111111111111111111111111111111111111111111111111111111111111
```

While loop – while(1) infinite loop

abc > C w.c > main()

```c
#include<stdio.h>
int main(){
  int i = 1;
  while(1){
    printf("%d", i);
    i++;
  }
  printf("\nThe value of i is %d", i);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
3023473033473043473053473063473073473083473093473103473113473123473131
3183473193473203473213473223473233473243473253473263473273473283473291
3343473353473363473373473383473393473403473413473423473433473443473451
3503473513473523473533473543473553473563473573473583473593473603473611
3663473673473683473693473703473713473723473733473743473753473763473771
3823473833473843473853473863473873473883473893473903473913473923473931
3983473993474003474013474023474033474043474053474063474073474083474091
4143474153474163474173474183474193474203474213474223474233474243474251
4303474313474323474333474343474353474363474373474383474393474403474411
```

1 is considered as true
Control comes to body
I printed and incremented
→ Infinite loop

While loop

abc > C w.c > main()

```c
#include<stdio.h>
int main(){
  int i = -1;
  while(i){
    printf("%d", i);
    i++;
  }
  printf("\nThe value of i is %d", i);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'w.exe'
-1
The value of i is 0
PS F:\code1\abc\output>
```

Not an infinite loop,

Not every time I will be incremented but it will stop at 0

It remove i++ it will infinite loop

While loop

abc > C w.c > main()

```c
#include<stdio.h>

int main(){
  int i = -1;
  while(i){
    printf("The value of i is %d", i);
  }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
value of i is -1The value of i is -1The value of i is -1The value of i is -1The valu
The value of i is -1The value of i is -1The value of i is -1The value of i is -1The
s -1The value of i is -1The value of i is -1The value of i is -1The value of i is -1
 i is -1The value of i is -1The value of i is -1The value of i is -1The value of i i
e of i is -1The value of i is -1The value of i is -1The value of i is -1The value of
value of i is -1The value of i is -1The value of i is -1The value of i is -1The valu
The value of i is -1The value of i is -1The value of i is -1The value of i is -1The
s -1The value of i is -1The value of i is -1The value of i is -1The value of i is -1
 i is -1The value of i is -1The value of i is -1The value of i is -1The value of i i
```

An infinite loop,

Value is not incremented so it considered as n true

While loop without init

abc > C w.c > main()

```c
#include<stdio.h>
int main(){
  int i ;
  while(i<10){
    printf("%d", i);
    i++;
  }
  printf("\nThe value of i is %d", i);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'w.exe'

The value of i is 2543616
PS F:\code1\abc\output>
```

Without init I, it gives garbage value

Loop is not executed

While loop with if and break statement

abc > C whi.c > main()

```c
#include <stdio.h>

int main() {
    int num, sum = 0;
    printf("Enter positive numbers to sum (enter a negative number to stop):\n");
    while (1) {  // Infinite loop
        printf("Enter a number: ");
        scanf("%d", &num);
        if (num < 0) {
            break;  // Exit the loop if the number is negative
        }
        sum += num;  // Add the number to the sum
    }
    printf("Total sum: %d\n", sum);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Enter a number: 4
Enter a number: 4
Enter a number: 4
Enter a number: 4
Enter a number: 4
Enter a number: 4
Enter a number: 4
Enter a number: 4
Enter a number: 4
Enter a number: 4
Enter a number: 4
Enter a number: -9
Total sum: 52
```

While loop with if and break statement

cssp1.html | getc.c | ifprin.c | isp.c

abc > whilee.c > main()

```c
#include<stdio.h>
int main(){
    int i=0;
    while(1){
        // do your work.
        printf("\n%d", i);
        if ( i == 10 ) break;
        i++;
    }
    printf("\nAfter While");
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
0
1
2
3
4
5
6
7
8
9
10
After While
PS F:\code1\abc\output>
```

abc > dem1.c > main()

```c
#include <stdio.h>

int main() {
    int i = 0;

    while (i < 5) {
        printf("\nhello");
        if (i == 3) {
            break;  // Breaks the loop when i reaches 3
        }
        i++;  // Increment i
    }

    printf("\nexit");  // Print "exit" after breaking the loop
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'dem1.exe'

hello
hello
hello
hello
exit
PS F:\code1\abc\output>
```

Inner While loop with break statement

abc > C wh.c > ...

```c

int main() {
    int x = 5;

    while (x-- > 0)
        while (x % 2 == 0) {  // Inner loop runs only when x is even
            printf("%d ", x);
            break;
        }

    return 0;
}

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'wh.exe'
4 2 0
PS F:\code1\abc\output>
```

Printf() while statement


```c
int main() {
    int num = 5;

    while (printf("%d ", num--) && num > 0) {}  // Prints and decrements

    return 0;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'wh.exe'
4 2 0
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'wh.exe'
5 4 3 2 1
PS F:\code1\abc\output>
```

Scanf() while statement


```c

int main() {
    int num;

    printf("Enter numbers (0 to stop):\n");

    while (scanf("%d", &num) && num != 0) {
        printf("You entered: %d\n", num);
    }

    printf("Loop ended.\n");
    return 0;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'wh.exe'
Enter numbers (0 to stop):
6
You entered: 6
4
You entered: 4
7
You entered: 7
1
You entered: 1
0
Loop ended.
PS F:\code1\abc\output>
```

Ch in while statement

abc > C w.c > main()

```c
#include<stdio.h>
//
int main(){
    char ch = 'b';
    while(ch > 0 ) {
        printf("%d ", ch);
        ch++;
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'w.exe'
98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 12
2 123 124 125 126 127
PS F:\code1\abc\output>
```

Sentinel while statement

Loop execute until it found sentinel value(SENTIVAL)

Step-1 Get the line of data
Step-2 while sentival not found:
3. Process the data
4. Get another line of data.

abc > C w.c > main()

```c
#include<stdio.h>
#define SENTIVAL -99
int main(){
    int score , sum = 0;
    printf("Enter the score:");
    scanf("%d", &score);
    while(score!=SENTIVAL){
        sum += score;
        printf("Enter the next score: ", SENTIVAL);
        scanf("%d", &score);
    }
    printf("The sum of the score is %d", sum);
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

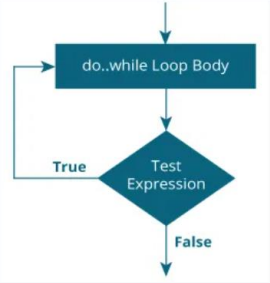
```
PS F:\code1\abc\output> & .\'w.exe'
Enter the score:39
Enter the next score: 34
Enter the next score: 38
Enter the next score: -99
The sum of the score is 111
PS F:\code1\abc\output>
```

Do while statement

The do..while loop is similar to the while loop with one important difference. The body of do...while loop is executed at least once. Only then, the test expression is evaluated.

The syntax of the do...while loop is

```
do {
  // the body of the loop
}
while (Condition/testexpression);
```



→body of loop is executed once→check the condition
→if true then again executed the loop and then again
Check the condition→ continue till true→if false , loop ends.

Do while vs While statement

abc > C c.c > main()

```c
#include<stdio.h>
int main(){
    int i = 0;
    do{
        printf("\nhello");
    }
    while(i>0);
    printf("\nout of dowhile");
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'c.exe'

hello
out of dowhile
PS F:\code1\abc\output>
```

abc > C c.c > main()

```c
#include<stdio.h>
int main(){
    int i = 0;
    while(i>0){
        printf("\nhello");
    }
    printf("\nout of dowhile");
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'c.exe'

out of dowhile
PS F:\code1\abc\output>
```

Do while statement

abc > C dem1.c > main()

```c
// Program to add numbers until the user enters zero

#include <stdio.h>
int main() {
  int number, sum = 0;

  do {
    printf("Enter a number: ");
    scanf("%d", &number);
    sum += number;
  }
  while(number != 0);

  printf("Sum = %d",sum);

}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'dem1.exe'
Enter a number: 3
Enter a number: 4
Enter a number: 5
Enter a number: 6
Enter a number: 0
Sum = 18
PS F:\code1\abc\output>
```

Do while statement– cannot put ; after do

abc > C w.c > main()

```c
#include<stdio.h>

int main(){
  int a=1;
      do;
      {
        printf("%d",a);
        a++;
      }
      while(a<=5);
}
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
f:\code1\abc\w.c:5:9: warning: suggest braces around empty body in 'do'
    5 |       do;
      |         ^
f:\code1\abc\w.c:6:7: error: expected 'while' before '{' token
    6 |       {
      |       ^
```

Do while statement

abc > C d.c > main()

```c
#include <stdio.h>

int main() {
    int num, count = 0;
    printf("Enter a number: ");
    scanf("%d", &num);
    if (num == 0)  count = 1;
    do {
            num /= 10;  // Remove the last digit
            count++;  // Increase the count
        } while (num != 0);  // Continue until all digits are removed

    printf("Number of digits: %d\n", count);

}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'d.exe'
Enter a number: 389
Number of digits: 3
PS F:\code1\abc\output>
```

Do while statement

abc > C dem1.c > main()

```c
int main() {
    int n, a = 0, b = 1, next, i = 0;

    // Taking user input
    printf("Enter the number of terms: ");
    scanf("%d", &n);

    // Printing the first Fibonacci number
    printf("Fibonacci Series: %d, %d", a, b);

    // Using do-while loop for Fibonacci sequence
    do {
        next = a + b;
        printf(", %d", next);
        a = b;
        b = next;
        i++;
    } while (i < n - 2); // Because first two numbers are already printed
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Fibonacci Series: 0, 1, 1, 2, 3, 5, 8, 13
PS F:\code1\abc\output>
```

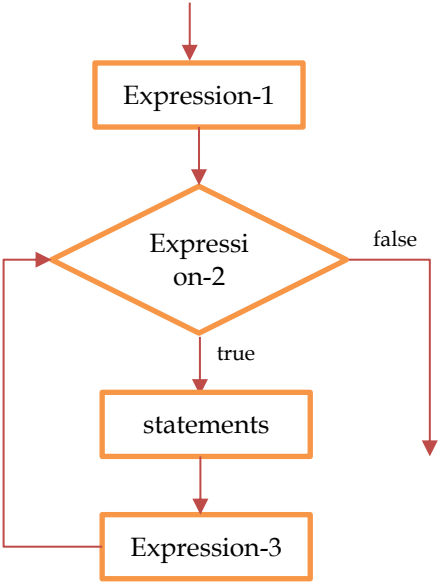
For Loop

In a for loop there are 3 expressions

expression-1 is used to initialize some variable (called index) that controls the loop.

Expression-2 is represent the condition that must be true for the loop to continue

Third expression used to alter the value of index initially assigned by expression-1



For Loop execution

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i;
    for(i = 1; i<=5;i++){
        printf("\ni = %d", i);
    }
    printf("\nThe value of i after for loop is %d", i);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'just.exe'

i = 1
i = 2
i = 3
i = 4
i = 5
The value of i after for loop is 6
PS F:\code1\abc\output>
```

For Loop Properties

All expression are optional

Case:1 without initialization

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i;
    for(; i<=5;i++){
        printf("\ni = %d", i);
    }
    printf("\nThe value of i after for loop is %d", i);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'just.exe'

The value of i after for loop is 3727360
PS F:\code1\abc\output>
```

For Loop Properties

All expression are optional
-- experession-1--

Case:1 without initialization

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i;
    for(; i<=5;i++){
        printf("\ni = %d", i);
    }
    printf("\nThe value of i after for loop is %d", i);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'just.exe'

The value of i after for loop is 3727360
PS F:\code1\abc\output>
```

Case:2 Initialization before for loop

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i=1;
    for(; i<=5;i++){
        printf("\ni = %d", i);
    }
    printf("\nThe value of i after for loop is %d", i);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'just.exe'

i = 1
i = 2
i = 3
i = 4
i = 5
The value of i after for loop is 6
PS F:\code1\abc\output>
```

For Loop Properties

Case:3 two variable initialization in for loop

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i, j;
    for(i = 1, j = 0; i<=5; i++){
        printf("\ni = %d , j = %d", i, j);
    }
    printf("\nThe value of i after for loop is %d and j is %d", i, j);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'just.exe'

i = 1 , j = 0
i = 2 , j = 0
i = 3 , j = 0
i = 4 , j = 0
i = 5 , j = 0
The value of i after for loop is 6 and j is 0
PS F:\code1\abc\output>
```

Case:4 two var, one without init

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i, j;
    for(j = 0; i<=5; i++){
        printf("\ni = %d , j = %d", i, j);
    }
    printf("\nThe value of i after for loop is %d and j is %d", i, j);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'just.exe'

The value of i after for loop is 4169728 and j is 0
PS F:\code1\abc\output>
```

For Loop Properties

--expression-2—

Case:1 Infinite loop- no condition

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i, j;
    for(i = 0; ; i++){
        printf("\ni = %d , j = %d", i, j);
    }
    printf("\nThe value of i after for loop is %d and j is %d", i, j);
}
```

F:\code1\abc • Contains emphasized items

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
i = 4227243 , j = 0
i = 4227244 , j = 0
i = 4227245 , j = 0
i = 4227246 , j = 0
i = 4227247 , j = 0
i = 4227248 , j = 0
i = 4227249 , j = 0
i = 4227250 , j = 0
i = 4227251 , j = 0
```

For Loop Properties

--expression-2—

Case:2 Infinite loop- two condition in for loop, loop will execute until second becomes false(j<4)

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i, j;
    for(i=0, j=1; i<=4, j<4; i++){
        printf("\ni = %d , j = %d", i, j);
    }
    printf("\nThe value of i after for loop is %d and j is %d", i, j);
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
i = 238168 , j = 1
i = 238169 , j = 1
i = 238170 , j = 1
i = 238171 , j = 1
i = 238172 , j = 1
i = 238173 , j = 1
i = 238174 , j = 1
i = 238175 , j = 1
i = 238176 , j = 1
```

For Loop Properties

--expression-2—

Case:3 if logical operators are there then it is considered as a one condition and terminated when both individually become false

abc > just.c > main()

```c
int main(){
    int i, j;
    for(i=0, j=1; i<=100 || j<3 ; i++,j++){
        printf("\ni = %d , j = %d", i, j);
        //if (i == 4) break;
    }
    printf("\nThe value of i after for loop is %d and j is %d", i, j);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
i = 83 , j = 84
i = 84 , j = 85
i = 85 , j = 86
i = 86 , j = 87
i = 87 , j = 88
i = 88 , j = 89
i = 89 , j = 90
i = 90 , j = 91
i = 91 , j = 92
i = 92 , j = 93
i = 93 , j = 94
i = 94 , j = 95
i = 95 , j = 96
i = 96 , j = 97
i = 97 , j = 98
i = 98 , j = 99
i = 99 , j = 100
i = 100 , j = 101
The value of i after for loop is 101 and j is 102
PS F:\code1\abc\output>
```

For Loop Properties

--expression-2—

Case:4 in condition, assignment operator

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i, j;
    for(i=0, j=1; i == 10 ; i++,j++){
        printf("\ni = %d , j = %d", i, j);
    }
    printf("\nThe value of i after for loop is %d and j is %d", i, j);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'just.exe'

The value of i after for loop is 0 and j is 1
PS F:\code1\abc\output>
```

For Loop Properties

--expression-2—

Case:5 in condition, 0 – treated as false, else will treated as true , false will not execute loop body, while true condition must be break!

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i, j;
    for(i=0, j=1; 0 ; i++,j++){
        printf("\ni = %d , j = %d", i, j);
    }
    printf("\nThe value of i after for loop is %d and j is %d", i, j);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'just.exe'

The value of i after for loop is 0 and j is 1
PS F:\code1\abc\output>
```

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i, j;
    for(i=0, j=1; 1; i++,j++){
        printf("\ni = %d , j = %d", i, j);
    }
    printf("\nThe value of i after for loop is %d and j is %d", i, j);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
i = 35683 , j = 35684
i = 35684 , j = 35685
i = 35685 , j = 35686
i = 35686 , j = 35687
i = 35687 , j = 35688
i = 35688 , j = 35689
i = 35689 , j = 35690
```

For Loop Properties

--expression-2—

Case:2 in condition, 0 – treated as false, else will treated as true (25) , false will not execute loop body, while true condition must be break(I == 4)!

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i, j;
    for(i=0, j=1; 25 ; i++,j++){
        printf("\ni = %d , j = %d", i, j);
        if (i == 4) break;
    }
    printf("\nThe value of i after for loop is %d and j is %d", i, j);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'just.exe'

i = 0 , j = 1
i = 1 , j = 2
i = 2 , j = 3
i = 3 , j = 4
i = 4 , j = 5
The value of i after for loop is 4 and j is 5
PS F:\code1\abc\output>
```

For Loop Properties

--expression-3—

Case:1 More than one incremental/decremental

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i, j;
    for(i=0, j=1; i<=100, j<4; i++,j++){
        printf("\ni = %d , j = %d", i, j);
    }
    printf("\nThe value of i after for loop is %d and j is %d", i, j);
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'just.exe'

i = 0 , j = 1
i = 1 , j = 2
i = 2 , j = 3
The value of i after for loop is 3 and j is 4
PS F:\code1\abc\output>
```

For Loop Properties

;

Case: for();

For(); → empty loop &
Body will be executed once
after for statement.

abc > C just.c > ...

```c
#include<stdio.h>
int main(){
    int i, j;
    for(i = 0, j = 1; i < 5, j < 6; j++);
    {
        printf("\ni = %d , j = %d", i, j);
        i++;
    }
    printf("\nThe value of i after for loop is %d and j is %d", i, j);
}
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'just.exe'

i = 0 , j = 6
The value of i after for loop is 1 and j is 6
PS F:\code1\abc\output>
```

For Loop Properties

;

Case: for();

For(); → empty loop & Body will be executed once after for statement.

abc > C gotojump.c > main()

```c
#include<stdio.h>
int main(){
    int i;
    for(i = 0; i<10; i++);
    printf("The value of i is: %d",i);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'gotojump.exe'
The value of i is: 10
PS F:\code1\abc\output>
```

For Loop Properties

--expression-3 –

Case: infinite loop

abc > C just.c > main()

```c
#include<stdio.h>
int main(){
    int i, j;
    for(;;){
        printf("\ni = %d , j = %d", i, j);

    }
    printf("\nThe value of i after for loop is %d and j is %d", i, j);
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
i = 3006464 , j = 4199120
i = 3006464 , j = 4199120
i = 3006464 , j = 4199120
i = 3006464 , j = 4199120
i = 3006464 , j = 4199120
i = 3006464 , j = 4199120
i = 3006464 , j = 4199120
```

For Loop iteration-memory address

abc > C dem1.c > ...

```c
#include <stdio.h>

int main() {
    for (int i = 0; i < 3; i++) {
        printf("Iteration %d: Address of i = %p\n", i, (void*)&i);
    }
    return 0;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'dem1.exe'
Iteration 0: Address of i = 0061FF1C
Iteration 1: Address of i = 0061FF1C
Iteration 2: Address of i = 0061FF1C
PS F:\code1\abc\output>
```

Sum of 10 natural number:


```c
int main(){
  int sum = 0 , i;
  for(i = 1; i<=10; i++){
    sum = sum+i;
  }
  printf("sum of 5 natural n umber is: %d ", sum );
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'dem1.exe'
sum of 5 natural n umber is: 55
PS F:\code1\abc\output>
```

Sum of square of 5 natural number:


```
int main(){
  int sum = 0 , i,n ;
  printf("Enter the number:");
  scanf("%d",&n);
  for(i = 1; i<=n; i++){
    sum = sum+i*i;
  }
  printf("sum of square of 5 natural n umber is: %d ", sum );
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'dem1.exe'
Enter the number:5
sum of square of 5 natural n umber is: 55
PS F:\code1\abc\output>
```

Break statement:

Cause immediate exit from the switch , while, do while and for loop

Common uses:
Escape early form the loop, in a switch statement .

abc > C d.c > ...

```c
#include <stdio.h>

int main() {
    for (int i = 1; i <= 10; i++) {
        if (i == 5) {
            continue;  // Skip printing 5
        }
        if (i == 8) {
            break;  // Stop the loop when i = 8
        }
        printf("%d ", i);
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'d.exe'
1 2 3 4 6 7
PS F:\code1\abc\output>
```

Continue Statement:

Skips the remaining statements in a body of while, do while, for loop
Proceed with the next iteration of the loop

abc > C dem1.c > ...

```c
#include <stdio.h>
int main() {
  int i;

  for (i = 0; i < 10; i++) {
    if (i == 4) {
      continue;
    }
    printf("%d\n", i);
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'dem1.exe'
0
1
2
3
5
6
7
8
9
PS F:\code1\abc\output>
```

Continue Statement:

abc > C dem1.c > ...

```c
#include <stdio.h>

int main() {
  int i = 0;

  while (i < 10) {
    if (i == 4) {
      i++;   // Increment `i` before `continue`
      continue;  // Skips the rest of the loop and jumps to the next iteration
    }
    printf("%d\n", i);
    i++;  // Normal increment for other cases
  }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'dem1.exe'
0
1
2
3
5
6
7
8
9
PS F:\code1\abc\output>
```

Ina case of for loop

The loop structure **automatically increments i** after each iteration.

That's why no need for i++ before continue!

Pattern 1- *:

abc > C pattern1.c > main()

```c
#include<stdio.h>
int main(){
    int i,j;
    for(i=0;i<3;i++){
        for(j=0;j<5;j++){
            printf("* ");
        }
        printf("\n");
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'pattern1.exe'
* * * * *
* * * * *
* * * * *
PS F:\code1\abc\output>
```

Pattern *:

abc > C d.c > main()

```c
int main() {
    int i,j,n;
    scanf("%d",&n);

    for (i = 0; i<n; i++){
        for(j=0;j<=i;j++){
            printf("* ");
        }
        printf("\n");
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'d.exe'
6
*
* *
* * *
* * * *
* * * * *
* * * * * *
PS F:\code1\abc\output>
```

abc > C pattern1.c > main()

```c
#include<stdio.h>
int main(){
    int i,j,n;
    scanf("%d",&n);

    for(i=0;i<n;i++){
        for(j=0;j<n-i;j++){
            printf("* ");
        }
        printf("\n");
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'pattern1.exe'
5
* * * * *
* * * *
* * *
* *
*
PS F:\code1\abc\output>
```

Pattern *:

abc > C pattern1.c > main()

```c
#include<stdio.h>
int main(){
    int i,j,n;
    scanf("%d",&n);

    for(i=0;i<n;i++){
        printf("%*s", i * 2, ""); //
        for(j=0;j<n-i;j++){
            printf("* ");
        }
        printf("\n");
    }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
    *
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'pattern1.exe'
5
* * * * *
  * * * *
    * * *
      * *
        *
```

printf("%*s", width, string);

*→ take the width from a variable instead of a fixed number

Width (i*2)→ The number of spaces allocated before the string

String(“”)→Actual string to print

Pattern numbers:

abc > C pattern2.c > main()

```c
#include <stdio.h>
int main() {
    int n;
    scanf("%d", &n);
    for (int i = 0; i < n; i++) {
        printf("%*s", i * 2, "");
        for (int j = 1; j <= n - i; j++) {
            printf("%d ", j);
        }
        printf("\n");
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'pattern2.exe'
5
1 2 3 4 5
  1 2 3 4
    1 2 3
      1 2
        1
PS F:\code1\abc\output>
```

Pascal Triangle:

abc > pascal.c > main()

```c
#include <stdio.h>
int main() {
    int n, coef = 1;
    scanf("%d", &n);
    for (int i = 0; i < n; i++) {
        printf("%*s", (n - i) * 2, "");
        // Loop for each column (binomial coefficient)
        for (int j = 0; j <= i; j++) {
            // Compute binomial coefficient using iterative method
            if (j == 0 || i == 0)
                coef = 1;
            else
                coef = coef * (i - j + 1) / j;

            printf("%4d", coef);
        }
        printf("\n");
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
          1
        1   1
      1   2   1
    1   3   3   1
  1   4   6   4   1
PS F:\code1\abc\output>
```

Let x be an integer which can take a value of 0 or 1. The statement

```
if (x == 0) x = 1; else x = 0;
```

is equivalent to which one of the following ?

A. $x = 1 + x;$
B. $x = 1 - x;$
C. $x = x - 1;$
D. $x = 1\%x;$

GATE 2004 QNS


```c
main()
{
    int x, y, m, n;
    scanf("%d %d", &x, &y);
    /* Assume x>0 and y>0*/
    m = x; n = y;
    while(m != n)
        {
            if (m > n)
                m = m-n;
            else
                n = n-m;
        }
    printf("%d", n);
}
```

The program computes

A. $x + y$ using repeated subtraction

B. $x \mod y$ using repeated subtraction

C. the greatest common divisor of x and y

D. the least common multiple of x and y

GATE 2004 QNS

Consider the following program fragment for reversing the digits in a given integer to obtain a new integer.

Let $n = d_1\, d_2\, \ldots\, d_m$.

```
int n, rev;
rev = 0;
while(n > 0) {
    rev = rev * 10 + n%10;
    n = n/10;
}
```

The loop invariant condition at the end of the ith iteration is:

A. $n = d_1\, d_2\, \ldots\, d_{m-i}$ **and** $\text{rev} = d_m\, d_{m-1}\, \ldots\, d_{m-i+1}$

B. $n = d_{m-i+1}\, \ldots\, d_{m-1}\, d_m$ **or** $\text{rev} = d_{m-i}\, \ldots\, d_2\, d_1$

C. $n \neq \text{rev}$

D. $n = d_1\, d_2\, \ldots\, d_m$ **or** $\text{rev} = d_m\, \ldots\, d_2\, d_1$

Pattern number

abc > C patternum.c > ...

```c
#include <stdio.h>

int main() {
    int n = 5; // Number of rows

    for (int i = 1; i <= n; i++) {
        // Print leading spaces using %*s
        printf("%*s", (n - i) + 1, "");

        // Print numbers
        for (int j = 1; j <= (2 * i - 1); j++) {
            printf("%d", j);
        }
        printf("\n");
    }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> & .\'patternum.exe'
     1
    123
   12345
  1234567
 123456789
PS F:\code1\abc\output>
```

Pattern alphabet

abc > C patternalpha.c > ...

```c
#include <stdio.h>

int main() {
    int n = 5; // Number of rows

    for (int i = 1; i <= n; i++) {
        // Print leading spaces using %*s
        printf("%*s", (n - i) + 1, "");

        // Print alphabets
        for (int j = 0; j < (2 * i - 1); j++) {
            printf("%c", 'A' + j);
        }
        printf("\n");
    }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS F:\code1\abc\output> cd 'f:\code1\abc\output'
PS F:\code1\abc\output> & .\'patternalpha.exe'
    A
   ABC
  ABCDE
 ABCDEFG
ABCDEFGHI
PS F:\code1\abc\output>
```

Practice Link

Patterns:
https://www.programiz.com/c-programming/examples/pyramid-pattern

https://www.studytonight.com/c-programs/c-program-to-print-hollow-square-pattern-program

https://www.simplilearn.com/tutorials/c-tutorial/c-pattern-programs