Problem Solving Through Programming in C

Tutorial Session 2

Prof. Anupam Basu
Dept. of Computer Science & Engg.
IIT Kharagpur

Siddhant Mohapatra
PMRF Scholar
IIT Madras

Operators in C

- **Arithmetic operators**
- Relational operators
- Logical operators
- Assignment operators
- Bitwise operators

Operator	Name	Description	Example
+	Addition	Adds together two values	x + y
-	Subtraction	Subtracts one value from another	x - y
*	Multiplication	Multiplies two values	x * y
/	Division	Divides one value by another	x / y
%	Modulus	Returns the division remainder	x % y
++	Increment	Increases the value of a variable by 1	++x
--	Decrement	Decreases the value of a variable by 1	--x

Courtesy: w3schools.com

Operators in C

- Arithmetic operators
- **Relational operators**
- Logical operators
- Assignment operators
- Bitwise operators

Operator	Name	Example
==	Equal to	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

Courtesy: w3schools.com

Operators in C

- Arithmetic operators
- Relational operators
- **Logical operators**
- Assignment operators
- Bitwise operators

Operator	Name	Description	Example
&&	Logical and	Returns true if both statements are true	x < 5 && x < 10
||	Logical or	Returns true if one of the statements is true	x < 5 || x < 4
!	Logical not	Reverse the result, returns false if the result is true	!(x < 5 && x < 10)

x < 5 && x < 10

x = 1	True	True	True
x = 6	False	True	False
x = 10	False	False	False

x	x OR y	x || y
1	1	1
1	0	1
0	1	1
0	0	0

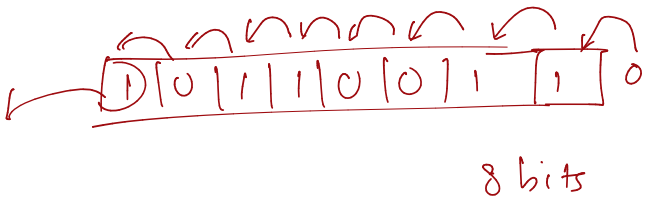
Courtesy: w3schools.com

Operators in C

- Arithmetic operators
- Relational operators
- Logical operators
- **Assignment operators**
- Bitwise operators

Operator	Example	Same As
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
&=	x &= 3	x = x & 3
|=	x |= 3	x = x | 3
^=	x ^= 3	x = x ^ 3
>>=	x >>= 3	x = x >> 3
<<=	x <<= 3	x = x << 3

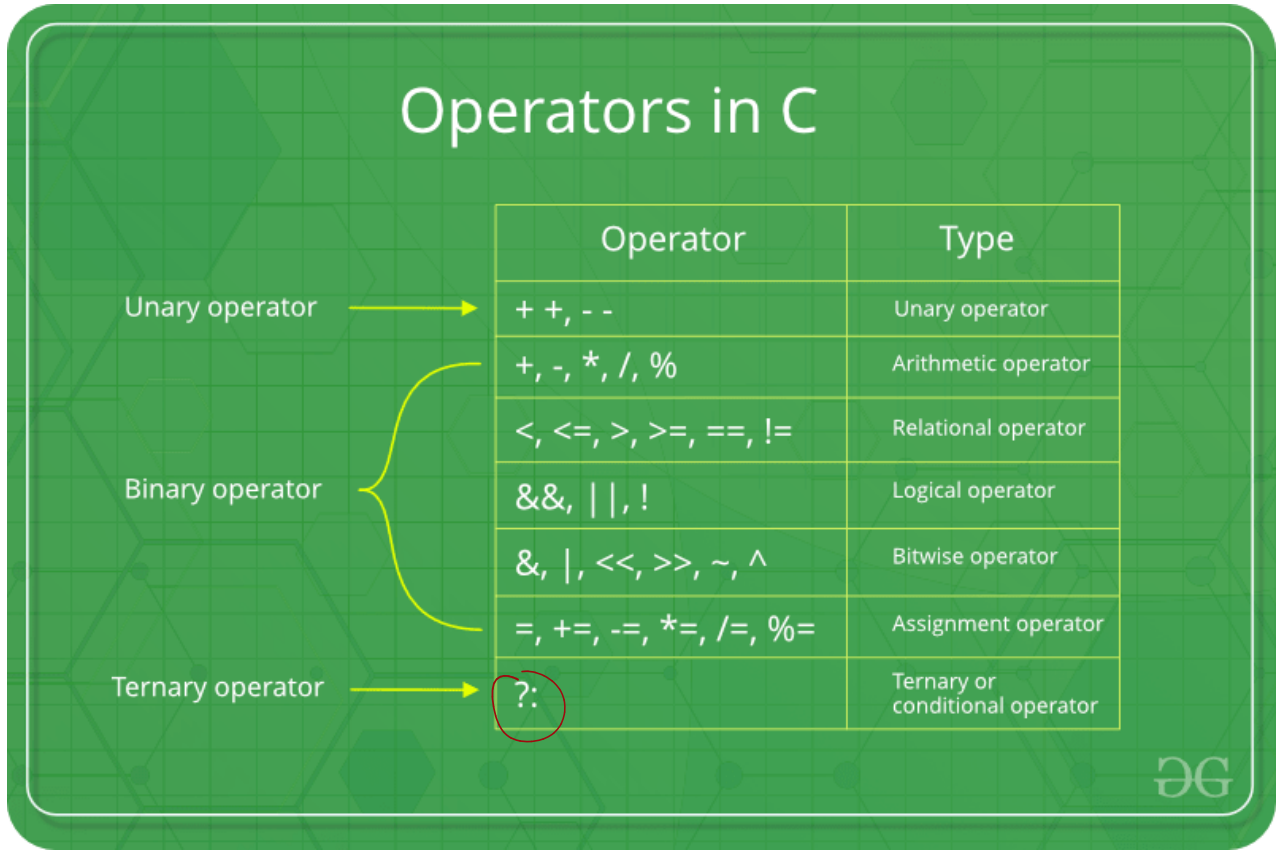
bitwise



Courtesy: w3schools.com

Operators in C

- Unary operators
- Binary operators
- Ternary operators



Courtesy: geeksforgeeks.org

Operators Precedence & Associativity

Precedence	Operator	Description	Associativity
1	++ --	Suffix/postfix increment and decrement	Left-to-right
	()	Function call	
	[]	Array subscripting	
	.	Structure and union member access	
	->	Structure and union member access through pointer	
	(*type*){*list*}	Compound literal(C99)	
2	++ --	Prefix increment and decrement[note 1]	Right-to-left
	+ -	Unary plus and minus	
	! ~	Logical NOT and bitwise NOT	
	(*type*)	Cast	
	*	Indirection (dereference)	
	&	Address-of	
	sizeof	Size-of[note 2]	
	_Alignof	Alignment requirement(C11)	
3	* / %	Multiplication, division, and remainder	Left-to-right
4	+ -	Addition and subtraction	
5	<< >>	Bitwise left shift and right shift	
6	< <=	For relational operators < and ≤ respectively	
	> >=	For relational operators > and ≥ respectively	
7	== !=	For relational = and ≠ respectively	
8	&	Bitwise AND	
9	^	Bitwise XOR (exclusive or)	
10	|	Bitwise OR (inclusive or)	
11	&&	Logical AND	
12	||	Logical OR	
13	?:	Ternary conditional[note 3]	Right-to-left
14[note 4]	=	Simple assignment	
	+= -=	Assignment by sum and difference	
	*= /= %=	Assignment by product, quotient, and remainder	
	<<= >>=	Assignment by bitwise left shift and right shift	
	&= ^= |=	Assignment by bitwise AND, XOR, and OR	
15	,	Comma	Left-to-right

$x = 5 * 3 + 2 * 5 / 2 \ \% \ 5 < 3 \ ? \ 5 : 3;$

$15 < 3 \ ? \ 5 : 3$ → $x = 3$

Courtesy: cppreference.com

Operators Precedence & Associativity

```
int x = 1,2,3;
```

separator, integer literals, assignment operation, compilation error (illegal)

```
int x = {1,2,3};
```

initialization of array, x=1, legal (warning) ← excess values to variable

```
int x = (1,2,3);
```

operator, x=3

```
int a = 54;
int b = (a, ++a);
```

b = ? b = 55

a = 54

b = (a-15, a-5, a*5)

b = 270

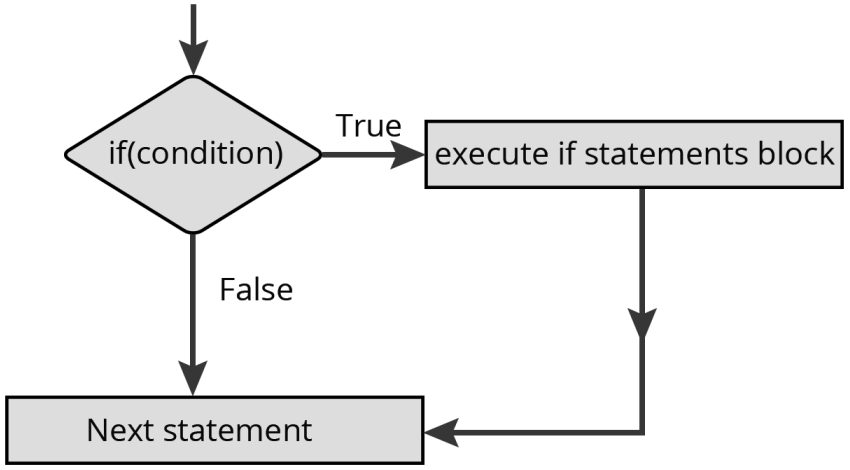
geeksforgeeks

```
#include <stdio.h>
int main(){
    int a = 54;
    int b = (a++,++a);
    a = 54,
    printf("%d\n",a),
    printf("%d\n",b),
    printf("%d\n",b++);
    return (0);
    }
```

variable declaration

Conditional Statements in C

- **if-else if-else statement**



Syntax:

```
if (condition1) {
  // block of code to be executed if condition1 is true
} else if (condition2) {
  // block of code to be executed if the condition1 is false and condition2 is true
} else {
  // block of code to be executed if the condition1 is false and condition2 is false
}
```

Short hand:

```
int k = 15;
char c = k>5 ? 'c' : 'd';
printf("%c", c);
```

Courtesy: w3schools.com

Operators Precedence

Q. What is the output of the following program? (% indicates modulo operation, which results the reminder of a division operation)

```
#include <stdio.h>
int main()
 {
        float i = -3.0;
        int k = i % 2;
        printf("%d", k);
        return 0;
}
```

-3.0 % 2 compilation error
% only on integer operands

(int) i % 2
= -3 % 2 = -1

What is the output?

a) 1
b) -1
c) 0
d) Compilation error

Courtesy: w3schools.com

Q. Find the output of the following C code.

```
#include<stdio.h>
#include<math.h>
 int main()
{
     int a=5, b=2, c=6;
     float  x1, x2;
if(b*b>4*a*c)
{
  x1= -b+sqrt(b*b-4*a*c)/2*a;
  x2= -b-sqrt(b*b-4*a*c)/2*a;
  printf("\n x1=%f, x2=%f",x1,x2);
}
     else
 printf("\n Roots are imaginary");
return 0;
}
```

4 > 120 false

a) x1=4, x2=3
b) x1=-5, x2=-4
c) x1=-2.5, x2=4.2
d) Roots are imaginary

Operators in C

Find the output of the following C program

```
#include<stdio.h>
  int main()
{
    int a, z, x=10, y=12;
      z=x*y++;
      a= x*y;
     printf("%d, %d", z, a);
     return 0;
}
```

a) 120, 120
b) 120, 130
c) 130, 120
d) 130, 130

postfix increment -> do the operation first, then the increment.

z = x * y++; xy = 120, y = 12+1 = 13, z = 120

a = x * y; xy = 130, a = 130

What is the output of the following C code?

```
#include <stdio.h>
int main()
{
int h = 8;
int b = 4 * 6 + 3 * 4 < h*5 ?4 : 3;
printf("%d\n", b);
return 0;

}
```

What is the output?
a) 0
b) 3
c) 4
d) Compilation error

true false

* + < ?:

24 + 3 x 4 < 8 x 5 ? 4 : 3

24 + 12 < 8 x 5 ? 4 : 3

24 + 12 < 40 ? 4 : 3

36 < 40 ? 4 : 3

true ? 4 : 3

4

Operators in C

Q. What will be the output of the following program?

```
#include<stdio.h>
 int main()
{
     int p=2;
     int m=10;
     int   k;
     k= !((p<2)&&(m>2));
     printf("\n%d", k);
     return 0;
}
```

a) 1
b) 0
c) -1
d) 2

Q. Find the output of the following code.

```
#include<stdio.h>
 int main()
{
  int p=6, q=4, r=10;
if(p>q)
{
   if(p>r)
    printf("%d", p);
   else
   printf("%d", r);
}
 else
{
 if(r>q)
     printf("%d", r+q);
    else
    printf("%d", q);
}
return 0;
}
```

a) 6
b) 4
c) 10
d) 14

Conditonal statements in C

Q. The output of the following program will be

```
#include<stdio.h>
 int main()
   {
   if(0)
   printf(" C programming\n");
   if(0.5)
   printf("Java \n");
   if(-0.5)
   printf("Python \n");
   return 0;
   }
```

a) C programming
b) Java
   Python
c) C programming
   Java
d) Compilation error

true false
1 0

0 → false

anything else → true

if (b*b<4*a*c)

Q. What will be the output?

```
#include<stdio.h>
int main()
 {
   int s=20;
   if(s=19)
   printf("Right\n");
   else
   printf("Wrong\n");
   return 0;
 }
```

a) Right
b) Wrong
c) 0
d) No output

Miscellaneous

Q. What will be the output of the program ?

```
#include<stdio.h>
 int main()
 {
    int a=2, b=3, c=5, d=8, ans;
    ans=a-b+c*a/d%b;
    printf(" The answer will be %d", ans);
    return 0;
 }
```

2 - 3 + 5 * 2 / 8 % 3
= 2 - 3 + 10 / 8 % 3
= 2 - 3 + 1 % 3
= 2 - 3 + 1 = -1 + 1 = 0

* / %
+ -

a) The answer will be 15
b) The answer will be 0
c) The answer will be 1
d) Compilation error

Q. What is the output of this C code?

```
#include <stdio.h>
int main()
{
int x = 10;
if (x > 0)
printf("inside if\n");
else if (x > 0)
printf("inside elseif\n");
return 0;
}
```

true

a) inside if
b) inside elseif
c) inside if
   inside elseif
d) Compile time error

compiler => high-level → binary code (machine level)

assembler => assembly level → machine level.

Conditional Statements in C

Write a program to enter 3 integers and print the smallest number among them.

```
int p, q, r;
if (p<q)
{
   if (p<r)   smallest = p;
   else       smallest = r;
}
else
{  if(q<r)    smallest = q;
   else       smallest = r;
}
```

nested if-else

```
if ( )
{  if ( )
   else
}
else
{  if ( )
   else
}
```

nested if

```
if ( )
else if ( )
else
```

if-else if-else

Course page

↓

Week 1

Week 2

⋮

Problem solving session ⟶ link to an excel sheet

programiz C examples

```
if (a>5 && a<10)
```

or

```
if(b>10 && b<20)
{
if(a>5)
{ if(a<10)
```

Week	video link	pdf link
week 2	..	..
.		
.		
.		
.		

↑ 1

↑ 2

↑ 3