Operators:
operators are symbols that perform operations on variables and values.

1. **Arithmetic Operators**
Used to perform basic arithmetic operations:`+` (Addition): Adds two operands. `a + b`
`-` (Subtraction): Subtracts the second operand from the first. `a - b`
`*` (Multiplication): Multiplies two operands. `a * b`

`/` (Division): Divides the numerator by the denominator. `a / b`
`%` (Modulus): Returns the remainder of a division operation. `a % b`

2. **Relational Operators**
Used to compare two values:
`==` (Equal to): Checks if two values are equal. `a == b`
`!=` (Not equal to): Checks if two values are not equal. `a != b`
`>` (Greater than): Checks if the left value is greater than the right. `a > b`
`<` (Less than): Checks if the left value is less than the right. `a < b`
`>=` (Greater than or equal to): Checks if the left value is greater than or equal to the right. `a >= b`
`<=` (Less than or equal to): Checks if the left value is less than or equal to the right. `a <= b`

3. **Logical Operators**
Used to perform logical operations:
`&&` (Logical AND): Returns true if both operands are true. `(a && b)`
`||` (Logical OR): Returns true if at least one operand is true. `(a || b)`
`!` (Logical NOT): Inverts the boolean value of the operand. `!a`

4. **Bitwise Operators**
Operate on bits and perform bit-level operations:
`&` (Bitwise AND): Performs AND operation on each bit. `a & b`
`|` (Bitwise OR): Performs OR operation on each bit. `a | b`
`^` (Bitwise XOR): Performs XOR operation on each bit. `a ^ b`
`~` (Bitwise NOT): Inverts all the bits. `~a`
`<<` (Left shift): Shifts bits of the left operand to the left. `a << b`
`>>` (Right shift): Shifts bits of the left operand to the right. `a >> b`

5. **Assignment Operators**
Used to assign values to variables:
`=` (Simple assignment): Assigns a value to a variable. `a = b`
`+=` (Addition assignment): Adds and assigns. `a += b` (equivalent to `a = a + b`)
`-=` (Subtraction assignment): Subtracts and assigns. `a -= b` (equivalent to `a = a - b`)
`*=` (Multiplication assignment): Multiplies and assigns. `a *= b` (equivalent to `a = a * b`)
`/=` (Division assignment): Divides and assigns. `a /= b` (equivalent to `a = a / b`)
`%=` (Modulus assignment): Computes modulus and assigns. `a %= b` (equivalent to `a = a % b`)

6. **Increment and Decrement** Operators**
Used to increase or decrease a variable's value by one:
- `++` (Increment): Increases the value by one. `a++` (post-increment) or `++a` (pre-increment)
- `--` (Decrement): Decreases the value by one. `a--` (post-decrement) or `--a` (pre-decrement)

7. **Conditional (Ternary) Operator**
A shorthand for `if-else` statements:

- `? :` (Ternary operator): `condition ? expr1 : expr2` evaluates `expr1` if `condition` is true, otherwise evaluates `expr2`.

8. **Sizeof Operator**

Returns the size of a data type or variable in bytes:

`sizeof`: `sizeof(data_type)` or `sizeof(variable)`

1. **Arithmetic Operators**
Used for basic mathematical operations.

```
#include <stdio.h>
int main() {
    int a = 10, b = 5;
    printf("Addition: %d\n", a + b);      // Addition
    printf("Subtraction: %d\n", a - b);   // Subtraction
    printf("Multiplication: %d\n", a * b);// Multiplication
    printf("Division: %d\n", a / b);      // Division
    printf("Modulus: %d\n", a % b);       // Modulus
    return 0;
}
```

2. **Relational Operators**
Used to compare two values.

```
#include <stdio.h>
int main() {
    int a = 10, b = 5;
    printf("a > b: %d\n", a > b);   // Greater than
    printf("a < b: %d\n", a < b);   // Less than
    printf("a == b: %d\n", a == b); // Equal to
    printf("a != b: %d\n", a != b); // Not equal to
    printf("a >= b: %d\n", a >= b); // Greater than or equal to
    printf("a <= b: %d\n", a <= b); // Less than or equal to
    return 0;
}
```

3. **Logical Operators**
Used to combine multiple conditions.

```
#include <stdio.h>
int main() {
    int a = 10, b = 5, c = 0;
    printf("a > b && b > c: %d\n", (a > b && b > c)); // Logical AND
    printf("a > b || b < c: %d\n", (a > b || b < c)); // Logical OR
    printf("!(a == b): %d\n", !(a == b));            // Logical NOT
    return 0;
}
```

4. **Bitwise Operators**
Operate on bits and perform bit-by-bit operations.

```
#include <stdio.h>
int main() {
    int a = 5, b = 9;
    printf("a & b: %d\n", a & b);   // AND
    printf("a | b: %d\n", a | b);   // OR
    printf("a ^ b: %d\n", a ^ b);   // XOR
    printf("~a: %d\n", ~a);         // NOT
```

```
    printf("b << 1: %d\n", b << 1); // Left shift
    printf("b >> 1: %d\n", b >> 1); // Right shift
    return 0;
}
```

5. **Assignment Operators**
Used to assign values to variables.

```
#include <stdio.h>
int main() {
    int a = 10;
    a += 5; // a = a + 5
    printf("a += 5: %d\n", a);
    a -= 3; // a = a - 3
    printf("a -= 3: %d\n", a);
    a *= 2; // a = a * 2
    printf("a *= 2: %d\n", a);
    a /= 2; // a = a / 2
    printf("a /= 2: %d\n", a);
    a %= 3; // a = a % 3
    printf("a %%= 3: %d\n", a);
    return 0;
}
```

6. **Increment and Decrement Operators**
Used to increase or decrease the value of a variable by 1.

```
#include <stdio.h>
int main() {
    int a = 10;
    printf("a++: %d\n", a++);  // Post-increment
    printf("++a: %d\n", ++a);  // Pre-increment
    printf("a--: %d\n", a--);  // Post-decrement
    printf("--a: %d\n", --a);  // Pre-decrement
    return 0;
}
```

7. **Conditional (Ternary) Operator**
Used for simple decision-making.

```
#include <stdio.h>
int main() {
    int a = 10, b = 5;
    int max = (a > b) ? a : b;
    printf("Max: %d\n", max);
    return 0;
}
```

8. **Sizeof Operator**
Returns the size of a variable or data type.

```
#include <stdio.h>
int main() {
    int a;
    printf("Size of int: %zu\n", sizeof(a)); // Output: 4 (on most systems)
    printf("Size of char: %zu\n", sizeof(char)); // Output: 1
    printf("Size of float: %zu\n", sizeof(float)); // Output: 4
    return 0;
}
```

9. **Comma Operator**
Used to separate multiple expressions where only one is expected.

```
#include <stdio.h>
int main() {
    int a, b;
    a = (b = 10, b + 5); // b is assigned 10, and a is assigned b + 5
    printf("a: %d, b: %d\n", a, b);
    return 0;
}
```

Precedence

- **Definition**: Precedence dictates which operators are evaluated first in an expression.
- **Example**: In the expression `3 + 4 * 2`, the multiplication `*` has higher precedence than addition `+`, so `4 * 2` is evaluated first.

Associativity

- **Definition**: Associativity determines the order of operations when operators of the same precedence appear in an expression.
- **Two Types**:
  1. **Left-to-Right (Left Associative)**: Most operators, like `+`, `-`, `*`, and `/`, are left associative. This means if two operators of the same precedence are present, the one on the left is evaluated first.
     - **Example**: In `5 - 2 - 1`, the expression is evaluated as `(5 - 2) - 1`.
  2. **Right-to-Left (Right Associative)**: Operators like the assignment operator `=` and the exponentiation operator `**` (if present in certain extensions or libraries) are right associative. This means if two operators of the same precedence are present, the one on the right is evaluated first.
     - **Example**: In `a = b = 5`, the expression is evaluated as `a = (b = 5)`.

OPERATOR	TYPE	ASSOCIAVITY
( ) [ ] . ->		left-to-right
++ -- + - ! ~ (*type*) * & sizeof	Unary Operator	right-to-left
* / %	Arithmetic Operator	left-to-right
+ -	Arithmetic Operator	left-to-right
<< >>	Shift Operator	left-to-right
< <= > >=	Relational Operator	left-to-right
== !=	Relational Operator	left-to-right
&	Bitwise AND Operator	left-to-right
^	Bitwise EX-OR Operator	left-to-right
|	Bitwise OR Operator	left-to-right
&&	Logical AND Operator	left-to-right
||	Logical OR Operator	left-to-right
? :	Ternary Conditional Operator	right-to-left
= += -= *= /= %= &= ^= |= <<= >>=	Assignment Operator	right-to-left
,	Comma	left-to-right

20 numerical questions on precedence and associativity in C

1. **Question:
Evaluate the expression `5 + 3 * 2`.
Answer: `11`
(Multiplication `*` has higher precedence than addition `+`, so `3 * 2 = 6`, then `5 + 6 = 11`)

2. **Question:
Evaluate `10 / 2 + 3 * 4`.
Answer: `17`
(`10 / 2 = 5`, then `3 * 4 = 12`, finally `5 + 12 = 17`)

3. **Question:
Evaluate `2 + 3 * 4 / 2 - 1`.
Answer: `7`
(`3 * 4 = 12`, `12 / 2 = 6`, `2 + 6 = 8`, `8 - 1 = 7`)

4. **Question:
What is the result of `8 + 12 / 4 * 2 - 3`?
Answer: `11`
(`12 / 4 = 3`, `3 * 2 = 6`, `8 + 6 = 14`, `14 - 3 = 11`)

5. **Question:
Calculate the value of `3 + 6 % 4 * 5 - 2`.
Answer: `11`
(`6 % 4 = 2`, `2 * 5 = 10`, `3 + 10 = 13`, `13 - 2 = 11`)

6. **Question:
Determine the value of `4 * 2 / 8 + 5`.
Answer: `6`

(`4 * 2 = 8`, `8 / 8 = 1`, `1 + 5 = 6`)

7. **Question:
Evaluate `5 + 10 > 8 && 6 - 2 < 5`.
Answer: `1`
(true) (`5 + 10 = 15`, `15 > 8` is `true`, `6 - 2 = 4`, `4 < 5` is `true`, `true && true` is `true`)

8. **Question:
What is the output of `7 + 3 * 6 / 2 - 1`?
Answer: `15`
(`3 * 6 = 18`, `18 / 2 = 9`, `7 + 9 = 16`, `16 - 1 = 15`)

9. **Question:
Calculate the result of `(5 + 3) * (8 - 4) / 2`.
Answer: `16`
(`5 + 3 = 8`, `8 - 4 = 4`, `8 * 4 = 32`, `32 / 2 = 16`)

10. **Question:
What is the value of `8 / 2 * (2 + 2)`?
Answer: `16`
(`8 / 2 = 4`, `2 + 2 = 4`, `4 * 4 = 16`)

11. **Question:
Evaluate `6 - 3 + 2 * 4`.
Answer: `11`
(`2 * 4 = 8`, `6 - 3 = 3`, `3 + 8 = 11`)

12. **Question:
Calculate the value of `10 + 2 * 5 / 2`.
Answer: `15`
(`2 * 5 = 10`, `10 / 2 = 5`, `10 + 5 = 15`)

13. **Question:
Evaluate the expression `3 + 4 * 2 / (1 - 5)`.
Answer: `1`
(`1 - 5 = -4`, `4 * 2 = 8`, `8 / -4 = -2`, `3 + (-2) = 1`)

14. **Question:
Find the result of `5 + 6 * 3 % 4`.
Answer: `8` (`6 * 3 = 18`, `18 % 4 = 2`, `5 + 2 = 8`)

15. **Question:
Evaluate the expression `2 * 3 + 4 * 5`.
Answer: `26`
(`2 * 3 = 6`, `4 * 5 = 20`, `6 + 20 = 26`)

16. **Question:
What is the value of `10 - 2 + 3 * 4`?

16. **Question:**
What is the value of `10 - 2 + 3 * 4`?

Answer: `20`
(`3 * 4 = 12`, `10 - 2 = 8`, `8 + 12 = 20`)

17. **Question:**
Determine the result of `8 + 3 * (4 - 2)`.
Answer: `14`
(`4 - 2 = 2`, `3 * 2 = 6`, `8 + 6 = 14`)

18. **Question:**
Evaluate `5 * 4 / 2 + 3`.
Answer: `13`
(`5 * 4 = 20`, `20 / 2 = 10`, `10 + 3 = 13`)

19. **Question:**
Calculate the value of `4 + 8 / 2 * 3`.
Answer: `16` (`8 / 2 = 4`, `4 * 3 = 12`, `4 + 12 = 16`)

20. **Question:**
What is the result of `12 / 4 * 3 + 2`?
Answer: `11`
(`12 / 4 = 3`, `3 * 3 = 9`, `9 + 2 = 11`)