In C programming language, operators have different levels of precedence, which determine the order in which the operators are evaluated in an expression.

The precedence of operators determines which operators are evaluated first, and in what order, when an expression contains multiple operators.

The table lists the operators in order of their precedence, with operators of higher precedence listed first.

Precedence	Name	Operator	Associativity
1	Postfix	() [] -> . ++ —	Left-to-right
2	Unary	+ – ! ~ * & sizeof	Right-to-left
3	Multiplicative	* / %	Left-to-right
4	Additive	+ –	Left-to-right
5	Shift	<< >>	Left-to-right
6	Relational	< <= > >=	Left-to-right
7	Equality	== !=	Left-to-right
8	Bitwise AND	&	Left-to-right
9	Bitwise XOR	^	Left-to-right
10	Bitwise OR	|	Left-to-right
11	Logical AND	&&	Left-to-right
12	Logical OR	||	Left-to-right
13	Conditional	?:	Right-to-left

Precedence	Name	Operator	Associativity
14	Assignment	= += -= *= /= %= <<= >>= &= ^= |=	Right-to-left
15	Comma	,	Left-to-right

Arithmetic Operators:

- Addition: “+”
- Subtraction: “-“
- Multiplication: “*”
- Division: “/”
- Modulo (remainder): “%”
- Increment: “++”
- Decrement: “–“

Example:

```
int a = 10;
int b = 3;
int c = a + b;    // c = 13
int d = a % b;    // d = 1
a++;              // a = 11
b--;              // b = 2
```

Relational Operators:

- Equal to: “==”
- Not equal to: “!=”

- Greater than: “>”
- Less than: “<” Greater than or equal to: “>=”
- Less than or equal to: “<=”

Example:

```
int a = 5;
int b = 10;
int c = (a == b);  // c = 0 (false)
int d = (a != b);  // d = 1 (true)
int e = (a > b);   // e = 0 (false)
```

Logical Operators:

- Logical AND: “&&”
- Logical OR: “||”
- Logical NOT: “!”

Example:

```
int a = 5;
int b = 10;
int c = (a > 0) && (b < 20);  // c = 1 (true)
int d = !(a == b);           // d = 1 (true)
```

Assignment Operators:

- Assignment: “=”
- Addition assignment: “+=”
- Subtraction assignment: “-=”
- Multiplication assignment: “*=”
- Division assignment: “/=”
- Modulo assignment: “%=”

Example:

```
int a = 5;
a += 3;   // a = a + 3 = 8
a -= 2;   // a = a - 2 = 6
a *= 4;   // a = a * 4 = 24
```

Practice Problems on Operators:

Problem 1:

Which operator is used to access the value at a particular address in C?

a) * (Asterisk)

b) & (Ampersand)

c) -> (Arrow)

d) . (Dot)

Explanation: The correct answer is a) * (Asterisk). The asterisk is the dereference operator in C, which is used to access the value at a specific memory address. For example, if you have a pointer variable ptr that points to an integer, you can access the value at that address using *ptr.

Problem 2:

What is the result of the expression 5 + 3 * 2 in C?

a) 16
b) 11
c) 13
d) 19

Explanation: The correct answer is b) 11. In C, multiplication (*) has higher precedence than addition (+). Therefore, the expression is evaluated as 5 + (3 * 2), which simplifies to 5 + 6, resulting in 11.

Problem 3:

Which operator is used to perform explicit type casting in C?

a) !
b) &
c) #
d) (Parentheses)

Explanation: The correct answer is d) (Parentheses). In C, you can use parentheses to

explicitly cast a value from one type to another. For example, (int) 3.14 casts the floating-point value 3.14 to an integer.

Problem 4:

What is the size of the following structure in C?

```
struct MyStruct {
    int x;
    char c;
};
```

a) 4 bytes
b) 5 bytes
c) 6 bytes
d) 8 bytes

Explanation: The correct answer is c) 6 bytes. The size of a structure in C is determined by the sum of the sizes of its individual members. In this case, int occupies 4 bytes, and char occupies 1 byte. Therefore, the total size of the structure is 4 + 1 = 5 bytes. However, most compilers add padding between structure members for alignment purposes. In this case, the structure is likely to be padded to a multiple of the largest member's size, which is 4 bytes (for int). So, the actual size of the structure becomes 4 + 1 + 1 (padding) = 6 bytes.

Problem 5:

What does the sizeof operator return in C?

a) The value of the operand
b) The size of the variable in bytes
c) The memory address of the operand
d) The type of the operand

Explanation: The correct answer is b) The size of the variable in bytes. The sizeof operator in C returns the size of the operand or variable in bytes. For example, sizeof(int) would return the size of an int type, usually 4 bytes on most systems.

Problem 6:

Write a C program that takes two integer inputs from the user and swaps their values using a temporary variable.

```
#include <stdio.h>

int main() {
    int a, b, temp;
    printf("Enter the value of a: ");
    scanf("%d", &a);
    printf("Enter the value of b: ");
    scanf("%d", &b);
    printf("Before swapping: a = %d, b = %d\n", a, b);
    // Swap the values using a temporary variable
    temp = a;
    a = b;
    b = temp;
    printf("After swapping: a = %d, b = %d\n", a, b);
    return 0;
```

```
}
```

Explanation: This program demonstrates the use of the assignment (=) operator and a temporary variable to swap the values of two variables. The user is prompted to enter the values of a and b, which are stored in the respective variables. The values are then printed before swapping. The values of a and b are swapped by using a temporary variable temp. Finally, the swapped values are printed.

Output:

```
Enter the value of a: 10
Enter the value of b: 20
Before swapping: a = 10, b = 20
After swapping: a = 20, b = 10
```

Problem 7:

Write a C program to check whether a given number is even or odd using the modulus (%) operator.

```
#include <stdio.h>

int main() {
    int number;
    printf("Enter a number: ");
    scanf("%d", &number);
    if (number % 2 == 0) {
        printf("%d is an even number.\n", number);
```

```
    } else {
        printf("%d is an odd number.\n", number);
    }
    return 0;
}
```

Explanation: This program checks whether a given number is even or odd using the modulus (%) operator. The user is prompted to enter a number, which is stored in the number variable. The modulus operator % returns the remainder when the number is divided by 2. If the remainder is 0, it means the number is divisible by 2 and hence even. Otherwise, it is odd. The program then prints the appropriate message based on the result.

Output:

```
Enter a number: 10
10 is an even number.
```

Problem 8:

Write a C program to calculate the area of a circle using the constant value of PI (3.14159) and the arithmetic operators.

```
#include <stdio.h>

int main() {
    float radius, area;
    const float PI = 3.14159;
    printf("Enter the radius of the circle: ");
```

```
    scanf("%f", &radius);
    area = PI * radius * radius;
    printf("The area of the circle is: %.2f\n", area);
    return 0;
}
```

Explanation: This program calculates the area of a circle using the formula area = PI * radius * radius. The user is prompted to enter the radius of the circle, which is stored in the radius variable. The constant value of PI (3.14159) is stored in the PI variable. The area is calculated by multiplying PI with the square of the radius. The calculated area is then printed using the printf function.

Output:

```
Enter the radius of the circle: 5
The area of the circle is: 78.54
```

Problem 9:

What is the output of the code?

```
#include <stdio.h>

int main() {
    int a = 5;
    int b = 2;
    int c = a / b * 3;

    printf("%d\n", c);
```

```
    return 0;
}
```

Explanation:
The output of the code will be 6. In C, division (/) and multiplication (*) have the same precedence and associate from left to right. Therefore, the expression a / b * 3 is evaluated as (a / b) * 3. The division a / b is an integer division, which results in 2. Then, 2 * 3 is evaluated, resulting in 6.

Problem 10:

What is the output of the code?

```
#include <stdio.h>

int main() {
    int a = 5;
    int b = 2;
    int c = a % b + 1;

    printf("%d\n", c);

    return 0;
}
```

Explanation:
The output of the code will be 2. In C, the modulus operator (%) returns the remainder of the division operation. The expression a % b calculates the remainder of 5 / 2, which is 1. Then, 1 + 1 is evaluated, resulting in 2.

Problem 11:

What is the output of the code?

```
#include <stdio.h>

int main() {
    int a = 5;
    int b = 2;
    int c = a > b ? a : b;

    printf("%d\n", c);

    return 0;
}
```

Explanation:
The output of the code will be 5. This code uses the conditional operator (?:) to assign the greater value between a and b to c. The expression a > b evaluates to true, so the value of a is assigned to c. Therefore, c becomes 5.

Problem 12:

What is the output of the code?

```
#include <stdio.h>

int main() {
    int a = 5;
    int b = 2;
```

```
    int c = a & b;

    printf("%d\n", c);

    return 0;
}
```

Explanation:
The output of the code will be 0. The & operator in C is the bitwise AND operator. It performs a bitwise AND operation between the individual bits of a and b. In this case, a (binary: 101) and b (binary: 010) are bitwise ANDed, resulting in 000 (binary). Converting 000 back to decimal gives 0, which is the value of c.

Related Posts:

1. C prgoram to convert inch to feet
2. C program to convert KM to CM
3. C program to convert meter to centimeter
4. C program to calculate remainder, difference, division, product
5. C program to use printf() without semicolon ” ; “
6. C program to swap two numbers using 2 variables
7. C program to find nth term using Arithmetic progrssion
8. C program to find sum of first n even positive numbers
9. C program to calculate sum of first n even numbers
10. C program to find nth odd number
11. C program to find sum of first n odd positive numbers
12. C program to calculate perimeter and area of a rectangle
13. C program to calculate perimeter and area of a square
14. C program to calculate Perimeter and Area of Circle

15. Function in C Programming
16. C Programming Q & A
17. Main function in C Programming Q and A
18. Void main in C Programming
19. Variables Q and A in C Programming
20. Write a C Program to find the percentage of marks ?
21. Write a c program to find age of a person ?
22. Write a c program to get table of a number
23. What is Break statement in C Programming ?
24. Write a c program to generate all combinations of 1, 2 and 3 using for loop.
25. Write a C program to print all the prime numbers between 1 to 50.
26. Write a C program to get factorial of a number ?
27. What is user defined function in C programming ?
28. Difference between C and C++ Programming ?
29. Difference between C, C++ and Java Programming
30. C program addition of numbers using pointer
31. C Syntax
32. Comments in C
33. Variables in C
34. Data types in C
35. Format specifiers in C
36. Type Conversion in C
37. Constants in C
38. Pre and Post Increament Practice Problems
39. Pre and Post Increament
40. Array in C
41. C Introduction

42. C Get Started
43. C Pointers
44. C History
45. C Program Compiling and running
46. C While loop
47. C Do While Loop
48. C For loop
49. break and continue statement
50. Control Statements in C
51. C if-else ladder
52. C if statements
53. C 2-Dimensional array
54. C String library functions
55. C Functions
56. C Functions Categories
57. C Actual Arguments
58. Write a program that prints the message “Hello, World!”
59. Write a program that asks the user to enter two numbers, and then prints the sum of those two numbers.
60. Write a program that asks the user to enter a number and then determines whether the number is even or odd.
61. Write a program that swaps the values of two variables.
62. Write a program that asks the user to enter a number and then calculates and prints its factorial.
63. Write a program that asks the user to enter a number N and then prints the first N numbers in the Fibonacci sequence
64. Write a program that swaps the values of two variables without using a temporary

variable

65. Converts a number into integer, float, and string
66. Program to find the length of the string
67. Program to convert string to uppercase or lowercase
68. Program to prints the numbers from 1 to 10.
69. What is identifier expected error
70. Difference between static and non static methods in Java
71. C String Input
72. C Character input
73. C Programming Variables MCQ
74. Object & Classes
75. C Programming find the output MCQs