OPERATORS

- An operator is a symbol that tells the compiler to perform specific mathematical or logical functions
- Types of operators:
  - Arithmetic Operators
  - Comparison Operators
  - Boolean Operators
  - Bitwise Operators
  - Compound Operators

Arithmetic Operators

OPERATOR NAME	OPERATOR SYMBOL	DESCRIPTION	EXAMPLE
Assignment operator	=	Stores the value of the right of the equal sing in the variable to the left of the equal sign	A = B
Addition	+	Adds two operands	A + B
Subtraction	-	Subtract second operand from the first	A - B
Multiplication	*	Multiply both operands	A * B
Division	/	Divide numerator by denominator	B / A
Modulo	%	Modulus operator and remainder of after an integer division	B % A

Comparison Operator

Assume variable A = 10 & variable B = 20

OPERATOR NAME	OPERATOR SYMBOL	DESCRIPTION	EXAMPLE
Equal to	==	Checks if the value of two operands is equal or not. If yes then, condition becomes true.	(A == B) iis false
Not equal to	!=	Checks if the value of two operands is equal or not, if values are not equal then condition becomes true	(A != B) is true
Less than	<	Checks if the value of the left operand is less than the value on the right operand. If yes then condition becomes true.	(A < B) is true

Greater than	>	Checks if the value of the left operand is greater than the value on the right operand. If yes then condition becomes true.	(A > B) is false
Less than or equal to	<=	Checks if the value of the left operand is less than or equal the value on the right operand. If yes then condition becomes true.	(A <= B) is true
Greater than or equal to	>=	Checks if the value of the left operand is greater than or equal the value on the right operand. If yes then condition becomes true.	(A >= B) is false

Boolean Operators

Assume variable A = 10 & variable B = 20

OPERATOR NAME	OPERATOR SYMBOL	DESCRIPTION	EXAMPLE
And	&&	Called logical AND operator, if both the operands are non-zero then the condition becomes true	(A && B) is true
Or	||	Called Logical OR operator. If any of the two operands is non-zero then the condition becomes true	(A || B) is true
Not	!	Called logical NOT operator. Use to reverses the logical state of its operand. If condition is true then logical NOT operator will make false	!(A && B) is false

Bitwise Operators

Assume variable A = 60 and variable B = 13

OPERATOR NAME	OPERATOR SYMBOL	DESCRIPTION	EXAMPLE
And	&	Binary AND operator copies a bit to the result if it exists in both operands	(A & B) will give 12, with is 0000 1100
Or	|	Binary OR operator copies a bit if it exists in either operand	(A | B) will give 61 which is 0011 1101
xor	^	Binary XOR operator copies the bit if it is set in one operand but not both	(A ^ B) will give 49 which is 0011 0001
Not	~	Binary ones complement operator is unary and has the effect of 'flipping' bits	(~ A) will give -60 which is 1100 0011
Shift left	<<	Binary Left Shift operator. The left operands value is moved left by the number of bits specified by the right operand.	A << 2 will give 240 which is 1111 0000
Shift right	>>	Binary Left Shift operator. The left operands value is moved right by the number of bits specified by the right operand.	A >> 2 will give 15 which is 0000 1111

Compound Operators

Assume variable A = 10 and variable B = 20

OPERATOR NAME	OPERATOR SYMBOL	DESCRIPTION	EXAMPLE
Increment	++	Increment operator, increases integers value by one	A++
Decrement	- -	Decrement operator, decreases integers value by one	A- -
Compound addition	+=	Add AND assignment operator. It adds the right operand to the left operand and assign the result to the left operand.	A += B
Compound Subtraction	-=	Subtract AND assignment operator. It subtracts the right operand from the left operand and assign the result to the left operand.	A -= B
Compound Multiplication	*=	Multiply AND assignment operator. It multiplies the right operand with the left operand and assign the result to the left operand.	A *= B
Compound Division	/=	Divide AND assignment operator. It divides the left operand with the right operand and assign the result to the left operand.	A /= B
Compound Modulo	%=	Modulus AND assignment operator. It takes modulus using two operands and aligns the result to left operand	A %= B
Compound Bitwise OR	|=	Bitwise inclusive OR and assignment operator	A |= B
Compound Bitwise AND	&=	Bitwise AND assignment operator	A &= B