Exclusive study materials developed by Anisoft Computer Academy

Operators in java: A detailed discussion

Arithmetic operators

Here int A = 10,B = 20; A and B are operands.

|  |  |  |
| --- | --- | --- |
| Operator | Description | Example |
| + | Addition- Adds values on either side of the operator | A + B will give 30 |
| - | Subtraction- Subtracts the right hand operand from the left hand operand | A – B will give -10 |
| \* | Multiplication- Multiplies values on either side of the operator | A \* B = 200 |
| / | Division – Divides left hand operand by right hand operand | B / A will give 2 |
| % | Modulus – Divides the left hand operand by the right hand operand and returns a remainder | B % A will give 0 |
| ++ | Increment – Increase the value of operand by 1 | B ++ gives 21 |
| -- | Decrement – Decrease the value of operand by 1 | B – gives 19 |

Assignment operators

|  |  |  |
| --- | --- | --- |
| Operator | Description | Example |
| = | Simple assignment operator, assigns values from the right side operands to left side operand | C = A + B will assign the value of A + B into C |
| + = | Add AND assignment operator, it adds right operand to the left operand and assigns the result to left operand | C += A is equivalent to  C = C + A |
| - = | Subtract AND assignment operator, it subtracts right operand from the left operand and assigns the result to the left operand | C -= A is equivalent to C = C - A |
| \* = | Multiply AND assignment operator, it multiplies the right operand with the left operand and assigns the result to left operand | C \*= A is equivalent to  C = C \* A |
| / = | Divide AND assignment operator, it divides left operand with the right operand and assigns the result to left operand | C /= A is equivalent to  C = C / A |
| % = | Modulus AND assignment operator, it takes modulus using two operands and assigns the result to the left operand | C %= A is equivalent to  C = C % A |

Relational operators

Here int A = 10,B = 20; A and B are operands.

|  |  |  |
| --- | --- | --- |
| Operator | Description | Example |
| = = | Checks if the value of two operands are equal or not, if yes then the condition becomes true | (A = = B) is not true |
| ! = | Checks if the value of two operands are equal or not, if values are not equal then the condition becomes true | (A! = B) is true |
| > | Checks if the value of the left operand is greater than the value of the right operand, if yes then the condition becomes true | (A > B ) is not true |
| < | Checks if the value of the left operand is less than the value of the right operand, if yes then the condition becomes true | (A < B ) is true |
| >= | Checks if the value of the left operand is greater than or equal to the value of the right operand , if yes then condition becomes true | (A >= B) is not true |
| <= | Checks if the value of the left operand is less than or equal to the value of the right operand , if yes then condition becomes true | (A <= B) is true |

Detailed discussions of some more types of operators will be given in future study materials……..

SOME CORE CONCEPTS OF OOP:

1. METHOD
2. ABSTRACTION
3. ENCAPSULATION
4. DATA HIDING
5. POLYMORPHISM
6. INHERITANCE

Method

Methods in java determine the messages an object can receive. The fundamental parts of a method are the name, the arguments, the return type and the body. Methods in java can be created only as a part of a class. A method can be called only for an object and that object must be able to perform that method call.

A java method is a collection of statements that are grouped to perform an operation. A class can contain many methods, it is in methods where the logics are written, data is manipulated and all the actions are executed.

Abstraction

Abstraction refers to the act of representing essential features without including the background details or explanation. It refers to the process of hiding the details and exposing only the essential features of a particular concept or object. Abstraction refers to the ability to make a class abstract in OOP. An abstract class is one that cannot be instantiated or in other words an object cannot be created of that class. All other functionalities of the class still exists and its field, methods are all accessed through a sub class.

Encapsulation

Encapsulation is the ability of an object to be a container for related properties (data members) and methods. Encapsulation is a fundamental feature of OOP. Encapsulation refers to the act of wrapping up data and methods in a single unit called class. The data is not accessible to the outside world. They can be accessed only by its methods, which are wrapped within the class.

Benefits of encapsulation:

1. The fields of a class can be made read-only or write only.
2. A class can have total control over what is stored in its fields and who can access what.
3. Encapsulated code is more flexible and easy to change with new requirements.