➢ Variable Initialization

Initializing a variable means assigning a value to the variable. The following example demonstrates this:

```
Dim x As Integer
    x = 10
```

Here is another example:

```
Dim name As String
   name = "John"
Dim checker As Boolean
   checker = True
Static Dim flag As Boolean = False
Public Dim ch1 As Char = “A”
```

➢ Declaring Enumerations

The **Enum** statement declares an enumeration and defines the values of its members. The Enum statement can be used at the module, class, structure, procedure, or block level. The syntax for the Enum statement is as follows:

```
Enum enumerationname [ As datatype ]
   memberlist
End Enum
```

where

- ***Enumerationname***: Name of the enumeration. **Required**
- ***Datatype***: Data type of the enumeration and all its members. **Optional**
- ***Memberlist***: List of member constants being declared in this statement. **Required**.

Each member in the memberlist has the following syntax and parts:

```
MemberName [ = initializer ]
```

- ***Initializer*** − value assigned to the enumeration member. **Optional**.

Example:

```
Enum  Colors
   Red = 1
   Orange = 2
   Yellow = 3
   Azure = 4
   Blue = 5
   Violet = 6
End Enum
```

➢ Type of Conversion Functions

There are functions that we can use to convert from one data type to another. The following table includes some of them.

Function Name	Functions Description
CBool	Converts the expression to Boolean data type.
CByte	Converts the expression to Byte data type.
CChar	Converts the expression to Char data type.
CDate	Converts the expression to Date data type
CDbl	Converts the expression to Double data type.
CDec	Converts the expression to Decimal data type.
CInt	Converts the expression to Integer data type.
CLng	Converts the expression to Long data type.
CShort	Converts the expression to Short data type.
CSng	Converts the expression to Single data type.
CStr	Converts the expression to String data type.

➢ Print and Display Constant

Constant	Description
vbCrLf	Carriage return / linefeed character combination.
vbCr	Carriage return character.
vbLf	Line feed character.
vbNewLine	New line character.
vbTab	Tab character.
vbBack	Backspace character.

➢ The Operators

An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations. VB.Net is rich in built-in operators and provides following types of commonly used operators:

- Arithmetic Operators
- Comparison Operators
- Logical Operators
- Assignment Operators

Arithmetic Operators

Following table shows all the arithmetic operators supported by VB.Net. Assume variable **A** holds 2 and variable **B** holds 7, then −

Show Examples

Operator	Description	Example
^	Raises one operand to the power of another	B^A will give 49
+	Adds two operands	A + B will give 9
-	Subtracts second operand from the first	A - B will give -5
*	Multiplies both operands	A * B will give 14
/	Divides one operand by another and returns a floating point result	B / A will give 3.5
\	Divides one operand by another and returns an integer result	B \ A will give 3
MOD	Modulus Operator and remainder of after an integer division	B MOD A will give 1

Comparison Operators

Following table shows all the comparison operators supported by VB.Net. Assume variable **A** holds 10 and variable **B** holds 20, then −

Show Examples

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

Logical Operators

Following table shows all the logical operators supported by VB.Net. Assume variable A holds Boolean value True and variable B holds Boolean value False, then −

Show Examples

Operator	Description	Example
And	It is the logical as well as bitwise AND operator. If both the operands are true, then condition becomes true. This operator does not perform short-circuiting, i.e., it evaluates both the expressions.	(A And B) is False.
Or	It is the logical as well as bitwise OR operator. If any of the two operands is true, then condition becomes true. This operator does not perform short-circuiting, i.e., it evaluates both the expressions.	(A Or B) is True.
Not	It is the logical as well as bitwise NOT operator. Use to reverses the logical state of its operand. If a condition is true, then Logical NOT operator will make false.	Not(A And B) is True.
IsFalse	It determines whether an expression is False.	
IsTrue	It determines whether an expression is True.	

Assignment Operators

There are following assignment operators supported by VB.Net −

Show Examples

Operator	Description	Example
=	Simple assignment operator, Assigns values from right side operands to left side operand	C = A + B will assign 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 left operand	C -= A is equivalent to C = C - A
*=	Multiply AND assignment operator, It multiplies 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 (floating point division)	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 (Integer division)	C \= A is equivalent to C = C \A
^=	Exponentiation and assignment operator. It raises the left operand to the power of the right operand and assigns the result to left operand.	C^=A is equivalent to C = C ^ A
<<=	Left shift AND assignment operator	C <<= 2 is same as C = C << 2
>>=	Right shift AND assignment operator	C >>= 2 is same as C = C >> 2
&=	Concatenates a String expression to a String variable or property and assigns the result to the variable or property.	Str1 &= Str2 is same as Str1 = Str1 & Str2