Smit Vora

Learnings and Conclusion

Module-2

Topics

8. **Operators and Expressions**
   - **Types Of Operators**

     8..1. Arithmetic Operators - +, -, /, *, %,++(Incremental), --(Decremental) etc.

     8..2. Relational Operators - ==, !=, <, >, >=, <=, etc.

     8..3. Logical Operators - &&, ||, !

     8..4. Bitwise Operators - &, |, ^, >>, <<, etc.

     8..5. Assignment Operators - =, +=, -=, *=, /=, etc.

     8..6. Misc Operators – sizeof(), typeof(), ?:(Ternary), etc.
   - **Arithmetic Operators** – They are used to perform arithmetic operations with the operands.
   - **Relational Operators** – They are used to compare operands to obtain a result.
   - **Logical Operators** – They are used for to perform logical operations with Boolean operands.
   - **Bitwise Operators** – They are used to perform bit operations and perform bitwise shift operations.
   - **Assignment Operators** – They are used for assigning a value to a variable.
   - **Misc Operators** – They perform miscellaneous tasks like giving type of operand, size of operand etc.
9. **Loop Iteration**
   - **While –** It has a condition in the bracket after while keyword. If the statement is true it keeps on going in the loop or the code block**.**
   - **Syntax –**

     while(condition)

```
{
        // code to be executed
}
```

- **Do-While Statement -** The do while loop is same as while loop except that it will be running at least one time if condition is matched or not. It is because it does not check the condition first time. So, it is guaranteed to execute the code of program at least one iteration.
- **Syntax –**
- do
  {
  //code to be executed
  }
  while(condition)

- **For Statement** –For loop is used if you know the start point and end point. You can run a statement or a block of statements repeatedly until a specified expression evaluates to false. It is useful where you know in advance how many times program should iterate.
- **Syntax –**
- for(statement1; statement2;statement3;)
  {
  //code to be executed
  }
- **foreach Statement –** this is exclusively used to loop through element of array directly.
- **Syntax –**
- foreach(*type variablename* in *arrayname*)
  {
  //code to be executed
  }

- **break** – It is used to jump out of a loop when a particular condition is met.
- **continue** – It is used to continue the loop but breaking that particular iteration when continue statement is executed.

10. Understanding Arrays

- An array is a fixed sized sequential collection of same datatype starting with indexing 0.
- Declaration – [datatype] [][arrayname];
- Initialize an array ex – int [] arr = new int[10];
- It is used to store a no. of contiguous values in memory and manipulate data in it.
- Loop iterations can also be through them to manipulate data.
- They can be also initialized like - int [] arr = new int[5]{2,5,6,7,8};

11. Defining and Calling Methods

- **Define method and use –** A method is a codeblock used for performing a task in a program.
- **Syntax –**

```
<Access Specifier> <Return Type> <Method Name> (Parameter List) {
  //Method Body
}
```

- Access Specifier – describes the access of the method to classes in the program ex-public, private, protected, Internal.
- Return Type – can be a type of value which a method returns or it also can be void if a method returns nothing.
- Method name – method receives a user defined name.
- Parameter list – it is the list of parameters passed on as argument when it is called.
- **Calling Method –** It can be called by using name of the method.

- **Different type of parameters in method (Value type, Ref. type, optional)**
  - **Value type –** It is the normal C# value parameter which means value is directly passed.

- **Ref. Type –** References of a variable is passed as an argument which are assigned first. Argument passed consists of 'ref' keyword first. Any changes made in this argument in method will reflect on the variable in the calling method.
- **Optional or Default type –** These are the type of arguments which may be passed only if operations on them are required otherwise the function consists default values of these parameters. This parameter should only be passed after required parameters.
- **Named Parameters –** These are Parameters which are given the same name in arguments as well so the ordering doesn't matter and user doesn't have to remember the ordering, they can assign values according to argument names.
- **Out parameters –** It's just like ref parameter but the keyword here is 'out' and any argument passed need not to be initialized.

12.Working with strings – strings are reference predefined datatypes in c# and take and it is sequential collection of characters that's used to represent text.

- **String class study -** In C#, string is an object of **System.String** class that represent sequence of characters. We can perform many operations on strings such as concatenation, comparison, getting substring, search, trim, replacement etc. Its object is immutable.
- **Use of various string methods –**

12..1. **Compare() -** It is a method used for comparing two string objects.

12..2. **Concat() –** It concatenates one or more instances of the string.

12..3. **Copy(String) –** it creates a new object of specified string with same value as it.

12..4. **Remove() –** removes the specified number of characters from the string by deleting it and returns the new string with characters deleted.

12..5. **Replace()-** replaces all the instances of specified character in the given string and returns a replaced version of this string in a new string.

13. **Working with Datetime –** We use DateTime class to work with dates and time.
    - **Datetime class study –** A DateTime class object contains date, culture, time, localization, milliseconds.
        13..1. **Properties of DateTime –**
        13..2. It gives us date and time
        13..3. It gives us year, day of week, day of year, time of day.
        13..4. Now() property gives the current date and time.
        13..5. It also gives month, day, hour, minutes, seconds etc.