C# Programming - Fundamentals

1. Introduction to C# Programming

- **C# (C-Sharp)** is a modern, object-oriented programming language developed by **Microsoft**.
- Runs on the **.NET Framework** or **.NET Core**.
- Used for:
  - Desktop Applications
  - Web Applications (ASP.NET)
  - Games (Unity Engine)
  - Cloud Services

Features of C#:

- Simple and easy to learn
- Type-safe
- Object-Oriented
- Rich Library Support
- Automatic Garbage Collection

---

2. Basic Structure of a C# Program

```
using System;

class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Hello, World!");
    }
}
```

Explanation:

- `using System;`: Imports the System namespace
- `class Program`: Declares a class named Program
- `static void Main()`: Entry point of every C# application
- `Console.WriteLine()`: Outputs text to the console

---

3. Variables and Data Types

Variables:

- Containers to store data values.

Syntax:

```
datatype variableName = value;
```

Common Data Types:

Data Type	Description	Example
`int`	Integer numbers	`int x = 10;`
`double`	Decimal numbers	`double pi = 3.14;`
`char`	Single character	`char grade = 'A';`
`string`	Sequence of characters	`string name = "John";`
`bool`	Boolean (true/false)	`bool isActive = true;`

---

4. Constants

- Value cannot change once assigned.

```
const double PI = 3.14159;
```

---

5. Operators

Type	Operators	Example
Arithmetic	+, -, *, /, %	`int sum = a + b;`
Comparison	==, !=, >, <, >=, <=	`if(a > b)`
Logical	&&,	
Assignment	=, +=, -=, *=, /=	`a += 5;`

---

6. Conditional Statements (Decision Making)

if Statement

```
if (condition)
{
    // Code block
}
```

if-else

```
if (age >= 18)
{
    Console.WriteLine("Adult");
}
else
{
    Console.WriteLine("Minor");
}
```

switch Statement

```
switch (day)
{
    case 1: Console.WriteLine("Monday"); break;
    case 2: Console.WriteLine("Tuesday"); break;
    default: Console.WriteLine("Invalid Day"); break;
}
```

---

7. Loops (Iteration)

for Loop

```
for (int i = 0; i < 5; i++)
{
    Console.WriteLine(i);
}
```

while Loop

```
int i = 0;
while (i < 5)
{
    Console.WriteLine(i);
    i++;
}
```

do-while Loop

```
int i = 0;
do
{
    Console.WriteLine(i);
    i++;
} while (i < 5);
```

---

8. Methods (Functions)

Purpose:

- Reusable block of code
- Makes programs modular and easier to manage

Syntax:

```
returnType MethodName(parameters)
{
    // Method Body
}
```

Example:

```
static int Add(int a, int b)
{
    return a + b;
}
```

Method Call:

```
int result = Add(5, 10);
Console.WriteLine(result);
```

---

9. Input / Output in C#

- **Output:** `Console.WriteLine("Hello");`
- **Input:**

```
Console.WriteLine("Enter your name:");
string name = Console.ReadLine();
```

10. Sample Program - Putting it All Together

```
using System;

class Program
{
    static void Main()
    {
        Console.WriteLine("Enter your age:");
        int age = Convert.ToInt32(Console.ReadLine());

        if (age >= 18)
        {
            Console.WriteLine("You are eligible to vote.");
        }
        else
        {
            Console.WriteLine("You are not eligible to vote.");
        }

        for (int i = 1; i <= 5; i++)
        {
            Console.WriteLine("Number: " + i);
        }

        int sum = Add(10, 20);
        Console.WriteLine("Sum is: " + sum);
    }

    static int Add(int a, int b)
    {
        return a + b;
    }
}
```

11. Key Points to Remember

- Always declare variables with correct data types.
- Loops help execute code repeatedly.
- Conditional statements make decisions based on logic.
- Methods avoid code repetition.
- C# is case-sensitive.

12. Homework / Practice Questions:

1. Write a program to check if a number is even or odd.
2. Create a calculator using switch-case.

3. Write a program to print the first 10 natural numbers using a loop.
4. Create a method to find the factorial of a number.
