Basic C#

.Net

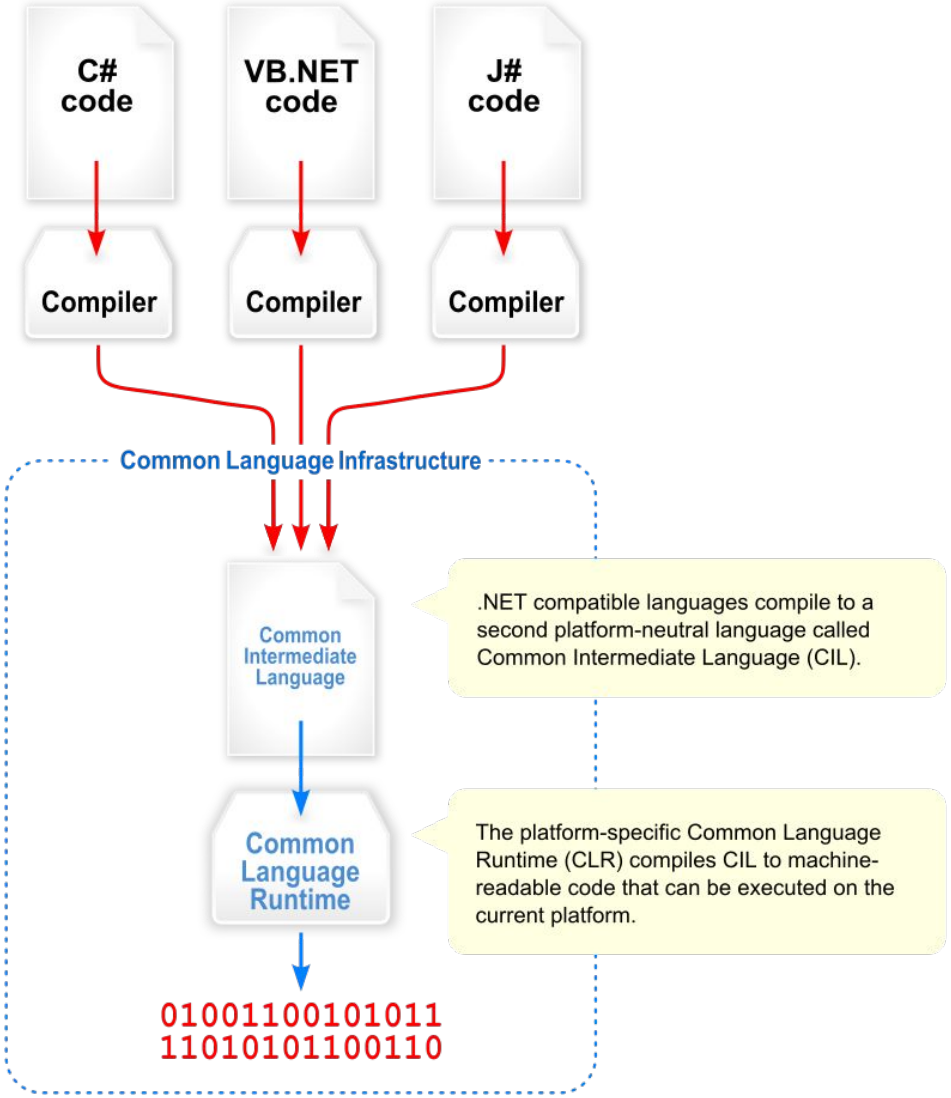
- Software framework by Microsoft

.NET 7	November 8, 2022[23]	Visual Studio 2022 Version 17.4	7.0.20	May 28, 2024	May 14, 2024	1.5 years
.NET 8	November 14, 2023[24]	Visual Studio 2022 Version 17.8	8.0.11 (LTS)	November 12, 2024	November 10, 2026	3 years
.NET 9	November 12, 2024[25]	Visual Studio 2022 Version 17.12	9.0.0	November 12, 2024	May 12, 2026	1.5 years
.NET 10	November 2025 (projected)		(will be LTS)		November 2028 (projected)	3 years (projected)
.NET 11	November 2026 (projected)				May 2028 (projected)	1.5 years (projected)

Legend: Old version, not maintained | Old version, still maintained | **Latest version** | Latest preview version | Future release

From Wikipedia, last update: Jan 2025

Mechanism


C# code
VB.NET code
J# code
Compiler
Compiler
Compiler
Common Language Infrastructure
Common Intermediate Language
.NET compatible languages compile to a second platform-neutral language called Common Intermediate Language (CIL).
Common Language Runtime
The platform-specific Common Language Runtime (CLR) compiles CIL to machine-readable code that can be executed on the current platform.
01001100101011
11010101100110

Hello world & artifacts

- Create a new console app

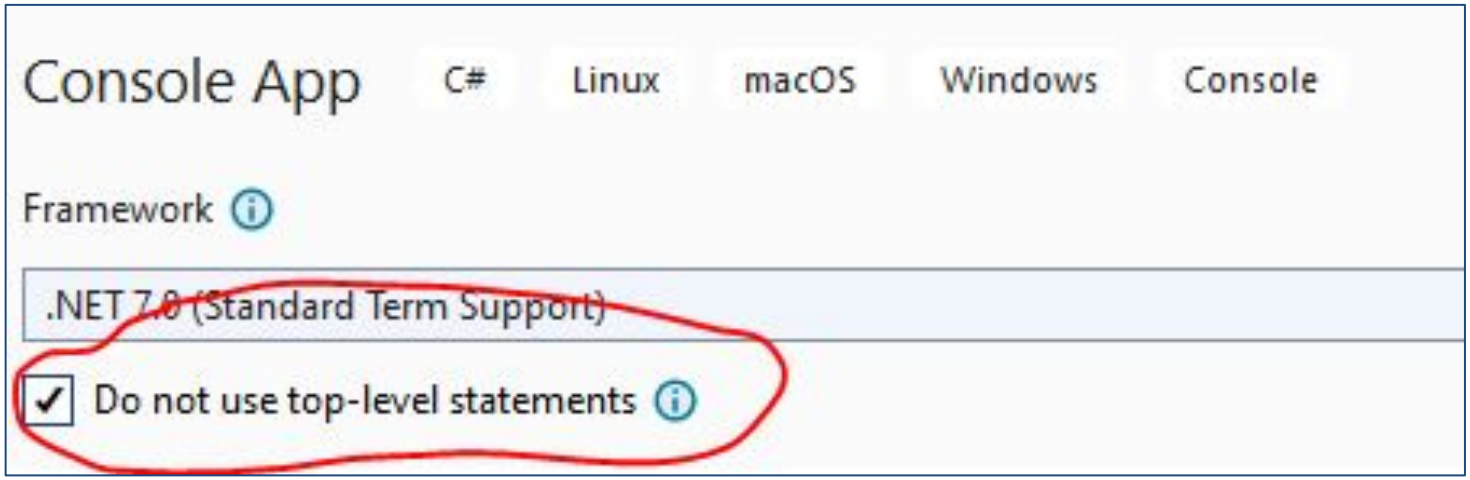
```
using System;

namespace ConsoleApp1
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```

```
Select C:\WINDOWS\system32\cmd.exe
Hello World!
Press any key to continue . . .
```

Top level statements

```
// See https://aka.ms/new-console-template for more information
Console.WriteLine("Hello, World!");
```



Constants & Variables

- Write a program that calculate sum from 1 to 10

```
static void Main(string[] args)
{
    const int start = 1;
    int end = 10;
    int sum = 0;

    for (int i = start; i <= end; i++)
    {
        sum += i;
    }

    Console.WriteLine("Sum from {0} to {1} is {2}", start, end, sum);
}
```

```
C:\WINDOWS\system32\cmd.exe
Sum from 1 to 10 is 55
Press any key to continue . . .
```

Messing around

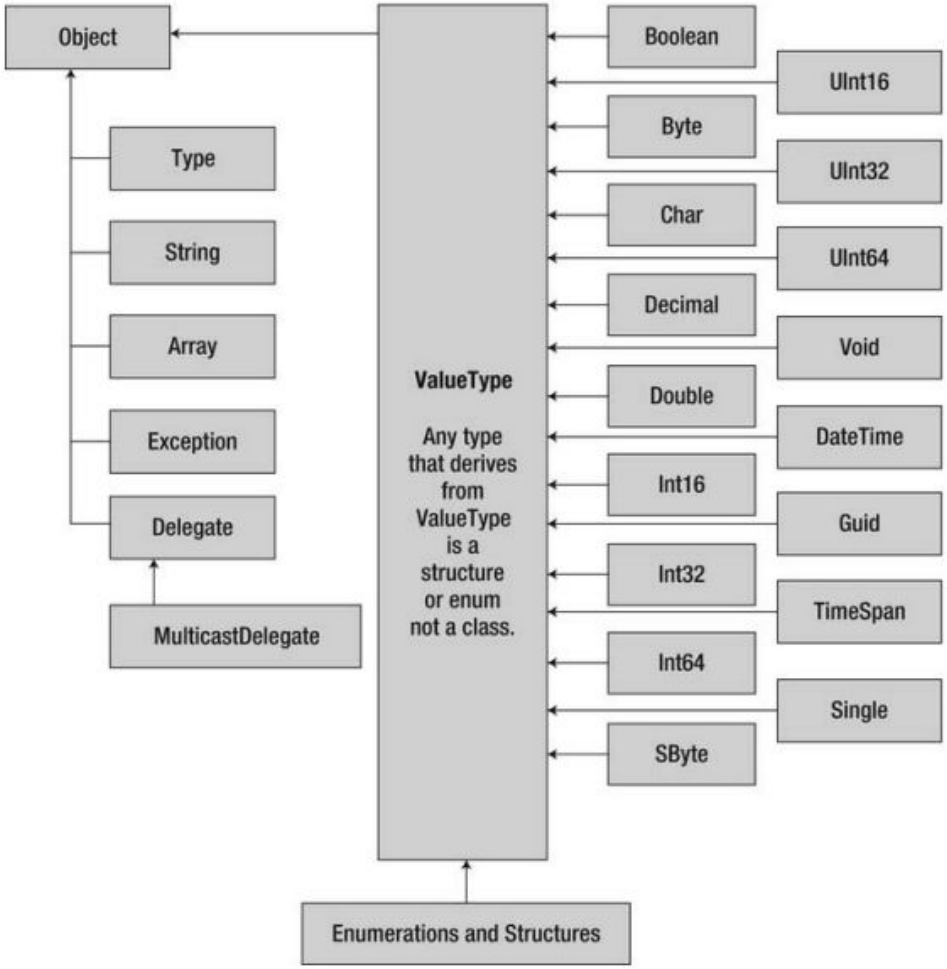
- Position of parameter for string format
- String interpolation
- Function examples
  - ref / out

Artifact

- Build => Error
- Run vs Run without debugging
- Release
- Debug
  - Breakpoint & watch
  - Step in & Step over & step out

- Set as startup project
- Open project location

Types


Object
Type
String
Array
Exception
Delegate
MulticastDelegate
ValueType
Any type that derives from ValueType is a structure or enum not a class.
Boolean
Byte
Char
Decimal
Double
Int16
Int32
Int64
SByte
UInt16
UInt32
UInt64
Void
DateTime
Guid
TimeSpan
Single
Enumerations and Structures

Exercise

Write a function to check if

- An integer is a prime number
- An integer has all odd digits

bool IsPrime(int value) / CheckPrime

bool HasAllOddDigits(int value) / CheckAllOddDigits

Answer

Tên hàm phải là động từ (to be)

bool IsPrime(int number)

bool CheckAllOddDigits(int number)

bool HasAllOddDigits(int number)

Array

```
int[] a = new int[3];
a[0] = 0;
a[1] = 1;
a[2] = 2;

int[] primes = new int[] { 2, 3, 5, 7};
int[] squares = { 4, 9, 16, 25};

string[] extensions = { "jpg", "png", "bmp" };
var colors = {"red", "green", "blue"};
```

Exercises

- Given a date stored in 3 variables: day, month, year (hard code, change everytime the app runs)
- Write a function to check if these variables create a valid date
  - Write a helper function to check for leap year
- Write a function that prints out the previous day
- Write a function that prints out the next day

Advanced

make all these above functions part of class DateTime

Loop through an array

```
static void Main(string[] args)
{
    var a = new int[] { 10, 12, 27, 1 };
    var sum = 0;

    for(var i = 0; i < a.Length; i++)
    {
        var number = a[i];
        sum += number;
    }

    Console.WriteLine($"Sum of all numbers is: {sum}");
}
```

```
foreach (var number in a)
{
    sum += number;
}
```

```
C:\WINDOWS\system32\cmd.exe
Sum of all numbers is: 50
Press any key to continue . . .
```

Random generator

```
static void Main(string[] args)
{
    var generator = new Random();

    Console.WriteLine($"A random number: {generator.Next()}");
    Console.WriteLine($"Random from 0-19: {generator.Next(20)}");
}
```

Exercise

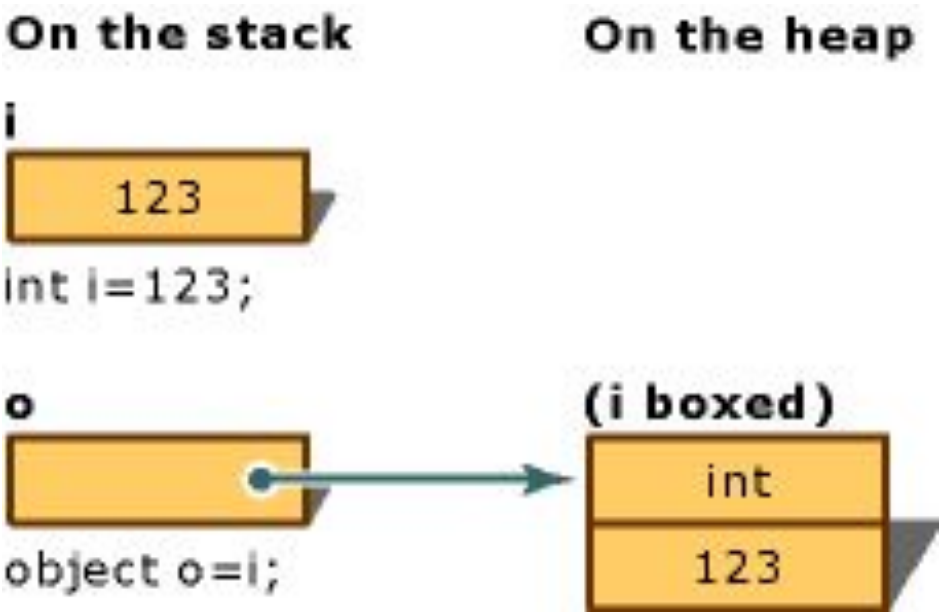
1. Generate 10 integers into an int array
2. Check if all integers are even numbers
3. Count the number of prime numbers
4. Create a sub array that contains all odd numbers

Number casting

```
// Small to big -> OK
byte b = 5; // implicit int to byte
int i = b;  // i = 5
// Big to small -> NOT OK
int i = 500;
byte j = i; // compile error
byte j = (byte)i; // i = 244
```

Boxing / Unboxing

- Boxing: convert value type to object
- Unboxing: extract value type from object



Exercise

Understanding the boxing/unboxing mechanism

```
int i = 7;
string name = i.ToString() + " $";
int result = i.CompareTo(100);
```

Minimum & maximum value

```
Console.WriteLine("=> Data type Functionality:");
Console.WriteLine("Max of int: {0}", int.MaxValue); //  2.147.483.647
Console.WriteLine("Min of int: {0}", int.MinValue); // -2.147.483.648
Console.WriteLine("Max of double: {0}", double.MaxValue); //  1.79769313486232E+300
Console.WriteLine("Min of double: {0}", double.MinValue); // -1.79769313486232E+300
Console.WriteLine("double.Epsilon: {0}", double.Epsilon); //  4.94065645841247E-324
Console.WriteLine("double.PositiveInfinity: {0}", double.PositiveInfinity); // ∞
Console.WriteLine("double.NegativeInfinity: {0}", double.NegativeInfinity); // -∞
Console.WriteLine();
```

```
int zero = 0;

double x = 10.0 / zero;
int y = 10 / zero;

Console.WriteLine("x={0} y={1}", x, y);
```

Result?

Operation 1/2

- References: . () [] new ->
- Arithmetic: + ++ - -- * / % sizeof
- Logical: & | ^ !
- Conditional: && (&) || ! == != > >= < <=
- Type verification: is as typeof
- Bitwise: ~ >> <<
- Assignment: = += -= *= /= %= &= |= ^= >>= <<=
- Selection: ?: ?? (not null)
- Lambda expression definition: =>

Operation 2/2

```
int i = 5;
// Selection ?:
string x = i == 5 ? "Yes": "No";

int? x = 5;      // nullable type
int y = x ?? 0; // Selection ?? operator
int z = x;       // Error: nullable type
     // cannot assign to non-nullable type

// Lambda expression - anonymous method
(int x) => x * 2; <=>
public int Double(int x){return x * 2;}
```

Use String format for concatenation

```
string firstName = "Tran";
string middleName = "Duy";
string lastName = "Quang";

string fullName = string.Format("{0} {1} {2}", firstName, middleName, lastName);
```

```
Console.WriteLine(string.Format("{0,15:C}", -125.34));
```

- Positive: right align, negative: left align

String concatenation

- Use “+” character

```
string times = "two times";
string hello = "Hello " + "world " + times;
Console.WriteLine(hello);
```

- Use StringBuilder

```
var sb = new StringBuilder();
sb.Append("Hello ");
sb.Append("world ");
sb.Append(times);
Console.WriteLine(sb.ToString());
```

- Use string interpolation with $“{variable}”
- Use string.Format

String Split

```
string SPACE = " ";
string fullName = "Tran Duy Quang";

string[] tokens = fullName.Split(new string[] {SPACE}, StringSplitOptions.None);
string firstName = tokens[0];
string middleName = tokens[1];
string lastName = tokens[2];
```

- Exercise
  - Calculate sum of string numbers = “5, 3, 8, 11, -12, 3”
  - Split String fraction = “3/4” into int numerator and denominator

    What if we meet 3//4?

Format string in VND

```
CultureInfo cul = CultureInfo.GetCultureInfo("vi-VN");  // en-US /en-UK
int money = 1250000;

string message = money.ToString("#,### đ", cul.NumberFormat);
string message = string.Format(info, "{0:c}", money);
```

n, n, $ n, n $

1.000.000,29 đ 10/07/2023

$ 1,000,000.29 07/10/2023

Cùng 1 dữ liệu => nhiều cách hiển thị khác nhau

String search

```
string s = "The quick brown fox jumps over the lazy dog and fox.";

string pattern = "fox";
int startIndex = 0;
int first = s.IndexOf(pattern, startIndex);
int last = s.LastIndexOf(pattern);
```

- Exercise
  - Given string s = “She sells seashells by the seashore. The shells she sells are seashells”
  - Calculate the number of occurrence of the word “sells” and “she”
- Further reading: replace and regular expression

String Exercises

1. Read a string and give statistics about the number of occurrence for each of the word in the string.

2. Normalize a string of full name and print out on the screen: no more than one spaces between words, the first letter is capitalized meanwhile the rest are in lower case, no space in the beginning and the end of the string.

3. Split String fullpath = “C:\Documents\Photos\Test.jpg” into a) Containing directory b) File name c) Extension

DateTime

```
using System;
DateTime a = new DateTime();

// Full construction
DateTime b = new DateTime(2013, 06, 15,
   15, 28, 31, 927);

// Current time
DateTime c = DateTime.Now;
```

Datetime string display

```
var cul = CultureInfo.GetCultureInfo("vi-VN");
var now = DateTime.Now;

now.ToString(cul);              // 10/07/2023 18:34:25
now.ToLongDateString();  // Monday, July 10, 2023
now.ToShortDateString(); // Mon 10 07 2023
```

How about 10-07-2023?

```
now.ToString("dd-MM-yyyy", CultureInfo.InvariantCulture)
```

Flow control

- Branching
  - Selection: ?: ??
  - if… else if … else
  - switch … case … default
- Iteration
  - for
  - foreach
  - do while
  - while
- Ignore & breaking
  - ✔ continue
  - ✔ break

switch case example

```
var day = "1";

switch (day)
{
    // Fall through
    case "3":
    case "7":
        Console.WriteLine("Good day to move out");
        break;
    default:
        Console.WriteLine("Stay at home is the best!");
        break;
}
```

Tuple

Creating a tuple

Accessing elements in a tuple

Destructuring a tuple into multiple elements

Basic coding convention

if for do while

- Add an empty line before and after
- Always use brackets even if there is one line of code

```
5     int start = 1;
6     int end = 10;
7     int sum = 0;
8
9     for(int i = start; i <= end; i++) {
10        sum += i;
11    }
12
13    Console.WriteLine($"Total sum from {start} to {end} is {sum}");
```

Comment

- Should provide purpose of a block of function
- XML comment for document generation (**doxygen**)

```
/// <summary>
/// Hàm tính tổng hai số
/// </summary>
/// <param name="a">Số nguyên thứ nhất</param>
/// <param name="b">Số nguyên thứ hai</param>
/// <returns>Tổng hai số</returns>
static int sum(int a, int b)
{
    return a + b;
}
```

Function name

- Start with a verb
- Private: _camelCase
- Public: PascalCase

- Good: isPrime, checkExists, called, calling
  - didCall, willChange
- Avoid: **do**Calculate => calculate

- Quiz: Check if a element exists in an array

Constants

- PascalCase

Exercises

- Check if a number is a prime number
  - bool IsPrime(int number) / LaSoNguyenTo()
- Check if a number is a square number
  - bool isSquare(int number) / LaSoChinhPhuong()
- Calculate x^n
  - double Pow(double x, int n) / LuyThua