

C# 2 – Biến và Kiểu dữ liệu

Giảng viên: **ThS. Lê Thiện Nhật Quang**

Email: quangln.dotnet.vn@gmail.com

Website: <http://dotnet.edu.vn>

Điện thoại: **0868.917.786**



MỤC TIÊU

- Tìm hiểu biến và kiểu dữ liệu trong C#
- Giải thích chú thích và tài liệu XML
- Tìm hiểu constants và literals
- Danh sách các từ khóa và escape sequence(chuỗi thoát)
- Giải thích input và output

1.1. KHÁI NIỆM BIẾN – VARIABLE

Biến là một cái tên được đặt cho một vùng lưu trữ mà chương trình của chúng ta có thể thao tác.

Mỗi biến trong C# có một loại cụ thể, xác định kích thước và cách bố trí bộ nhớ của biến đó.

Cú pháp để định nghĩa biến trong C# là:

`<data_type> <variable_list>;`

Ở đây, **data_type** phải là một kiểu dữ liệu hợp lệ trong C# hoặc bất kỳ kiểu dữ liệu người dùng tự định nghĩa (user-defined) và **variable_list** là danh sách tên biến có thể chứa một hoặc nhiều tên định danh được cách nhau bởi dấu phẩy.

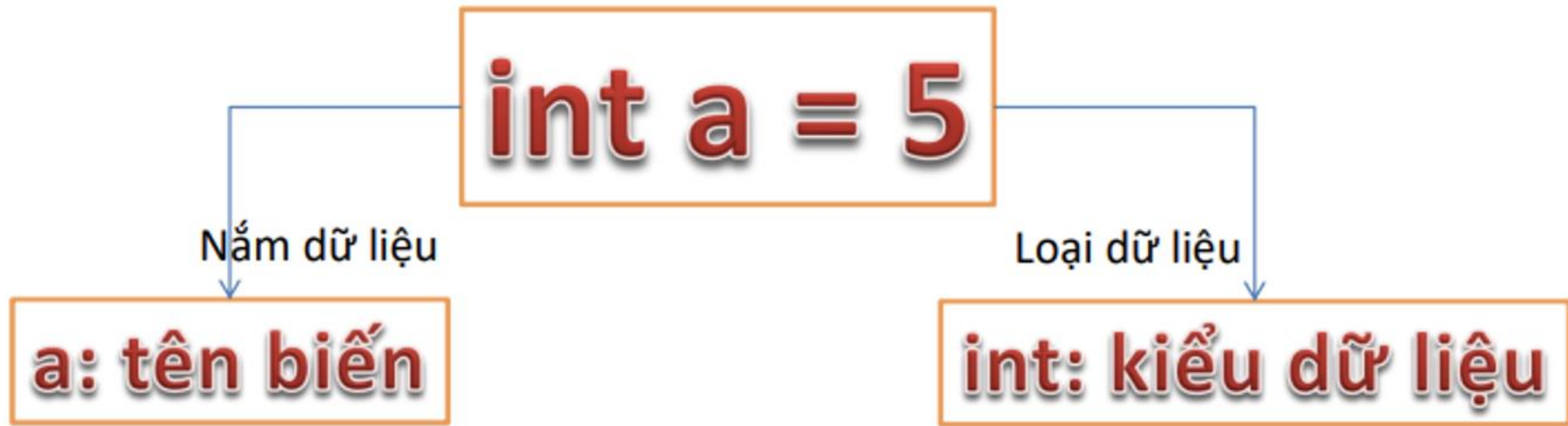
Ví dụ: `int a;`

1.1. KHÁI NIỆM BIẾN – VARIABLE (2)

```
namespace Session1
{
    0 references
    internal class TinhTong
    {
        0 references
        static void Main(string[] args)
        {
            int a = 10;
            int b = 21;
            int tong = a + b;
            Console.WriteLine("Tổng là: " + tong);
        }
    }
}
```

- Đoạn mã trên gán các giá trị 10 cho a, 21 cho b và tổng a + b cho tong sau đó xuất tổng ra màn hình.
- a, b và tong gọi là biến số nguyên
- Biến là thành phần nắm giữ dữ liệu được chương trình sử dụng trong các biểu thức tính toán
- Mỗi biến có kiểu dữ liệu riêng

1.1. KHÁI NIỆM BIẾN – VARIABLE (3)



- Biến là thành phần nắm giữ dữ liệu được chương trình sử dụng trong các biểu thức tính toán (biến a nắm giữ số 5)

- int: Số nguyên
- double : số thực
- String: Chuỗi
-

1.1. KHÁI NIỆM BIẾN – VARIABLE (4)

- Cú pháp

<kiểu dữ liệu> <tên biến>[=giá trị khởi đầu];

- Ví dụ:

int a; // *khai báo biến không khởi đầu giá trị*

double b = 5; // *khai báo biến có khởi đầu giá trị*

- Khai báo nhiều biến cùng kiểu

int a, b=5, c;

- Gán giá trị cho biến

c = 9;

a = 15;

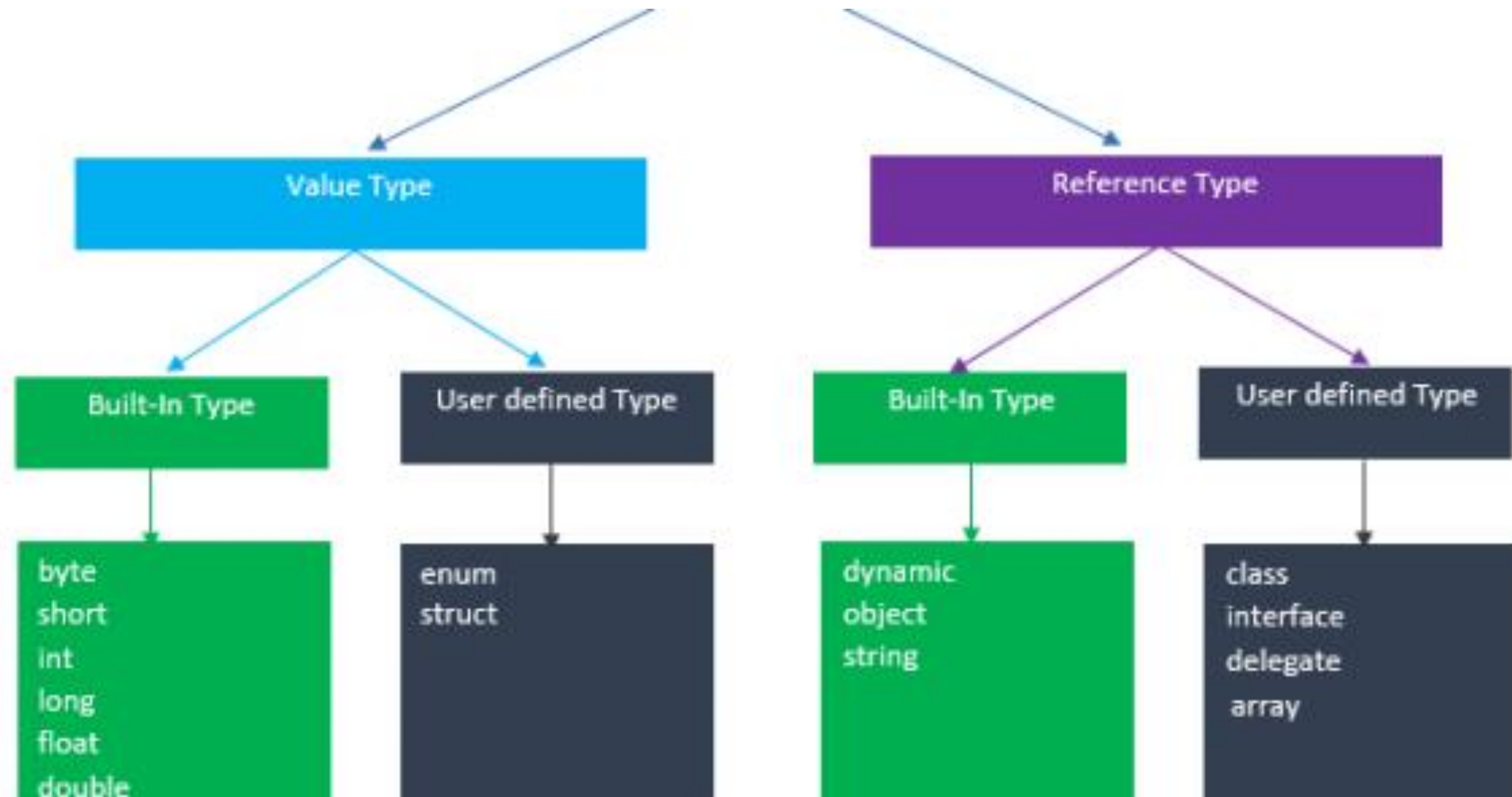
1.2. KIỂU DỮ LIỆU – DATA TYPE

Kiểu dữ liệu là tập hợp các nhóm dữ liệu có đặc điểm chung: đặc tính, cách lưu trữ và thao tác xử lý. Ví dụ một số kiểu dữ liệu cơ bản: kiểu đúng sai (bool), kiểu chuỗi (string), kiểu số nguyên (int), kiểu enum....

Có 2 loại kiểu dữ liệu:

- **Kiểu dữ liệu tham trị (value type)** thì vùng nhớ của biến và giá trị sẽ được lưu trữ ở cùng 1 nơi - bộ nhớ stack. Một số kiểu dữ liệu tham trị như: int, long, float, double, decimal, bool, char.
- **Kiểu dữ liệu tham chiếu (reference type)** thì vùng nhớ của biến sẽ được lưu ở 1 nơi - bộ nhớ stack. Giá trị sẽ được lưu ở 1 vùng nhớ khác - bộ nhớ heap. Một số kiểu dữ liệu tham chiếu như: object, string, dynamic.

1.2. KIỂU DỮ LIỆU (2)



1.2. KIỂU DỮ LIỆU (2)

NAME	.NET TYPE	DESCRIPTION	RANGE (MIN:MAX)
sbyte	System.SByte	8-bit signed integer	-128:127 ($-2^7:2^7-1$)
short	System.Int16	16-bit signed integer	-32,768:32,767 ($-2^{15}:2^{15}-1$)
int	System.Int32	32-bit signed integer	-2,147,483,648:2,147,483,647 ($-2^{31}:2^{31}-1$)
long	System.Int64	64-bit signed integer	-9,223,372,036,854,775,808: 9,223,372,036,854,775,807 ($-2^{63}:2^{63}-1$)
byte	System.Byte	8-bit unsigned integer	0:255 ($0:2^8-1$)
ushort	System.UInt16	16-bit unsigned integer	0:65,535 ($0:2^{16}-1$)
uint	System.UInt32	32-bit unsigned integer	0:4,294,967,295 ($0:2^{32}-1$)
ulong	System.UInt64	64-bit unsigned integer	0:18,446,744,073,709,551,615 ($0:2^{64}-1$)

Các kiểu số nguyên của c# và .net

1.2. KIỂU DỮ LIỆU (3)

NAME	.NET TYPE	DESCRIPTION	SIGNIFICANT FIGURES	RANGE (APPROXIMATE)
float	System.Single	32-bit, single-precision floating point	7	$\pm 1.5 \times 10^{245}$ to $\pm 3.4 \times 10^{38}$
double	System.Double	64-bit, double-precision floating point	15/16	$\pm 5.0 \times 10^{2324}$ to $\pm 1.7 \times 10^{308}$

Các kiểu số thực của C# và .NET

NAME	.NET TYPE	DESCRIPTION	SIGNIFICANT FIGURES	RANGE (APPROXIMATE)
decimal	System.Decimal	128-bit, high-precision decimal notation	28	$\pm 1.0 \times 10^{228}$ to $\pm 7.9 \times 10^{28}$

Kiểu decimal của C# và .NET

NAME	.NET TYPE	DESCRIPTION	SIGNIFICANT FIGURES	RANGE
bool	System.Boolean	Represents true or false	NA	true or false

Kiểu logic bool của C# và .NET

1.2. KIỂU DỮ LIỆU (4)

NAME	.NET TYPE	VALUES
char	System.Char	Represents a single 16-bit (Unicode) character

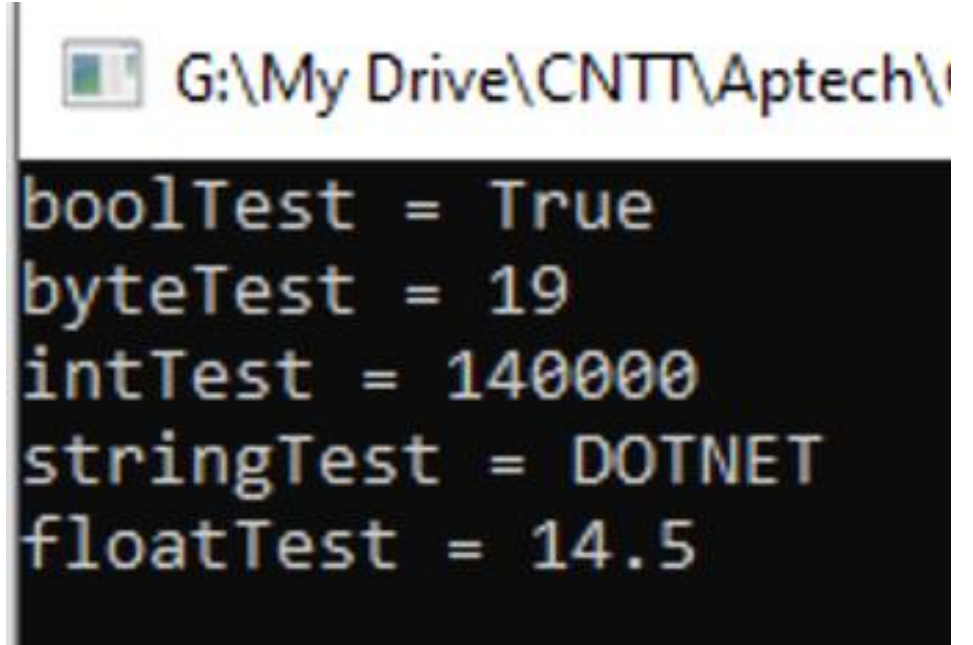
Kiểu ký tự của C# và .NET

NAME	.NET TYPE	DESCRIPTION
object	System.Object	The root type. All other types (including value types) are derived from object.
string	System.String	Unicode character string

Kiểu object và string của C# và .NET

1.2. KIỂU DỮ LIỆU (5)

```
bool boolTest = true;
short byteTest = 19;
int intTest;
string stringTest = "DOTNET";
float floatTest;
intTest = 140000;
floatTest = 14.5f;
Console.WriteLine("boolTest = {0} ", boolTest);
Console.WriteLine("byteTest = " + byteTest);
Console.WriteLine("intTest = " + intTest);
Console.WriteLine("stringTest = " + stringTest);
Console.WriteLine("floatTest = " + floatTest);
```



G:\My Drive\CNTT\Aptech\

```
boolTest = True
byteTest = 19
intTest = 140000
stringTest = DOTNET
floatTest = 14.5
```

1.3. PHÂN LOẠI

Kiểu dữ liệu tham chiếu lưu trữ bộ nhớ các biến khác của tham chiếu mà giá trị thực có thể được phân loại trong các kiểu sau:

- **Object**: Là kiểu dữ liệu tham chiếu built-in, là lớp cơ sở cho tất cả các kiểu dữ liệu được định nghĩa trước và do người dùng định nghĩa. Lớp là cấu trúc login đại diện cho thực thể trong thế giới thực. Các kiểu dữ liệu được định nghĩa trước và do người dùng định nghĩa được tạo dựa theo lớp Object
- **String**: Là một kiểu tham chiếu tích hợp, biểu thị các giá trị chuỗi ký tự Unicode. Nó cho phép gán và thao tác các giá trị chuỗi. Khi các chuỗi được tạo, chúng không thể được sửa đổi.
- **Class**: Là cấu trúc do người dùng định nghĩa có chức các biến và phương thức. Ví dụ: lớp Employee là cấu trúc do người dùng định nghĩa chứa các biến như empSalary, empName và empAddress. Nó cũng có thể chứa các phương thức như CalculateSalary(), trả về lương ròng của nhân viên.
- **Delegate**: Ủy quyền là kiểu tham chiếu xác định người dùng lưu trữ tham chiếu một hoặc nhiều phương thức
- **Interface**: Là giao diện do người dùng định nghĩa, nhóm các chức năng liên quan có thể thuộc về bất kỳ lớp - class hoặc cấu trúc - struct
- **Array**: Mảng là cấu trúc dữ liệu do người dùng định nghĩa có chức các giá trị của cùng một kiểu dữ liệu, ví dụ như điểm của sinh viên.

1.4. QUY TẮC ĐẶT TÊN

- Tên biến là một chuỗi ký tự liên kết (không có khoảng trắng) và không chứa ký tự đặc biệt.
- Tên biến không được đặt bằng tiếng việt có dấu.
- Tên không được bắt đầu bằng số. Tên biến phải bắt đầu bằng kí tự hoặc dấu _
- Tên biến không được trùng nhau.
- Tên biến phân biệt hoa thường
- Tên biến đặt ngắn gọn, dễ hiểu và mô tả được ý nghĩa của việc sử dụng
- Tên biến không được trùng với từ khóa.

2.1. CHÚ THÍCH - COMMENT

Chú thích là nội dung không được thực thi nhằm giải thích các đoạn mã giúp người khác có thể đọc code dễ dàng hơn hoặc có thể giúp ích trong quá trình sửa lỗi chương trình.

Chú thích trong C# dùng dấu `//` để chú thích một dòng và dùng `/*...*/` để chú thích nhiều dòng.

Chú thích XML có thể được thực hiện bằng cách thêm các phần tử XML. Chú thích XML bắt đầu bằng ba dấu `///`.

Ví dụ phương thức **Add** có thể chú thích **XML** như sau: đặt con trỏ chuột phía trên phương thức **Add** và gõ ba dấu `/` liên tiếp sẽ xuất hiện các thẻ **XML** như `<summary>` hay `<param>`. Các thẻ **XML** cũng như các thẻ **HTML** có thẻ bắt đầu, như `<summary>`, và thẻ kết thúc, như `</summary>`

Khi sử dụng phương thức **Add**, giả sử trong phương thức **Main**, các chú thích sẽ hiển thị như sau:

```
int sum = Add(  
    int MainWindow.Add(int a, int b)  
    Phương thức Add tính tổng hai số nguyên bất kỳ  
    a: Số nguyên thứ nhất
```

2.2. XML

3.1. HẲNG - CONSTANT

Hằng là một biến không thay đổi giá trị trong suốt chương trình và bắt buộc phải khởi tạo giá trị khi khai báo. Const là hàm số thực thi (compile) có giá trị bắt buộc phải gán khi khai báo. Const chỉ dùng cho kiểu dữ liệu nguyên thủy, không thể dùng cho kiểu tham chiếu.

```
const int myNum = 15;
```

```
myNum = 20; // error
```

Ngoài ra ta có thể dùng readonly để khai báo hằng số. Có thể gọi readonly là hằng số runtime có giá trị được gán giá trị lúc khai báo hoặc trong hàm khởi tạo. Có thể dùng readonly để khai báo hằng số kiểu tham chiếu.

3.2. LITERAL

Constant liên quan tới các giá trị cố định mà chương trình không thể thay đổi trong khi thực thi. Những giá trị cố định này cũng được gọi là literal.

Constant là một kiểu dữ liệu thay thế cho Literal, còn Literal thể hiện chính nó. Trong ví dụ: `const PI = 3.14` thì Constant ở đây là `PI`, còn Literal là `3.14`.

- Hằng Số Nguyên (Integer Literal)
- Hằng Boolean
- Hằng Số Thực (Floating Point Literal)
- Hằng Ký Tự (Character Literal)
- Hằng Chuỗi
- Hằng null

4.1. TỪ KHÓA – KEYWORD & KÝ TỰ LỆNH – ESCAPE SEQUENCE

Escape Sequence	Character
\'	Single quote
\"	Double quote
\\	Backslash
\0	Null
\a	Alert
\b	Backspace
\f	Form feed
\n	Newline
\r	Carriage return
\t	Tab character
\v	Vertical tab

```
string str = @"\u0048\u0065\u006C\u006C\u006F";  
Console.Write(@"\t" + str + "!\n");  
Console.WriteLine("David\u0020\"2007\" ");
```

4.1. TỪ KHÓA – KEYWORD & KÝ TỰ LỆNH – ESCAPE SEQUENCE (2)

```
using System;
class FileDemo {
    static void Main(string[] args) {
        string path = "C:\\Windows\\MyFile.txt";
        bool found = true;
        if (found){
            Console.WriteLine("File path : \" + path + "\");
        }
        else {
            Console.WriteLine("File Not Found!\a");
        }
    }
}
```

4.1. TỪ KHÓA – KEYWORD & KÝ TỰ LỆNH – ESCAPE SEQUENCE (3)

abstract	as	base	bool	break	byte	case
catch	char	checked	class	const	continue	
decimal	default	delegate	do	double	else	enum
event	explicit	Extern	false	finally	fixed	float
for	foreach	goto	if	implicitin	int	Interface
internal	is	lock	long	namespace	new	null
object	operator	out	overrid e	params	private	protected
public	readonly	ref	return	sbyte	sealed	short
sizeof	stackalloc	static	string	struct	switch	this
throw	true	try	typeof	uint	ulong	unchecked
unsafe	ushort	using	virtual	void	volatile	while

4.1.

add	alias	ascending	async	await	descending yield	dynamic
from	get	global	group	into	join	let
orderby	partial	remove	select	set	value	var
where						

5.1. INPUT VÀ OUTPUT

Để xuất - **output** nội dung trong C#, chúng ta có thể sử dụng:

- `System.Console.WriteLine()`
- `System.Console.Write()`

Ở đây, `System` là một namespace, `Console` là lớp bên trong namespace `System` và `WriteLine` và `Write` là phương thức của lớp `Console`.

Cách đơn giản nhất để lấy **input** từ user là sử dụng phương thức `ReadLine()` của lớp `Console`. Tuy nhiên, `Read()` và `ReadKey()` cũng có thể làm điều đó được. Hai phương thức này cùng thuộc lớp `Console`.

5.1.

```
int number, result;
number = 5;
result = 100 * number;
Console.WriteLine("Result is {0} when 100 is multiplied by {1}",
result, number);
result = 150 / number;
Console.WriteLine("Result is {0} when 150 is divided by {1}",
+result, number);

string name;
Console.Write("Enter your name: ");
name = Console.ReadLine();
Console.WriteLine("You are {0}", name);
```


5.1.

```
using System;
class Loan {
    static void Main(string[] args) {
        string custName;
        double loanAmount;
        float interest = 0.09F;
        double interestAmount = 0;
        double totalAmount = 0;
        Console.Write("Enter the name of the customer : ");
        custName = Console.ReadLine();
        Console.Write("Enter loan amount : ");
        loanAmount = Convert.ToDouble(Console.ReadLine());
        interestAmount = loanAmount * interest;
        totalAmount = loanAmount + interestAmount;
        Console.WriteLine("\nCustomer Name : {0}", custName);
        Console.WriteLine("Loan amount : ${0:#,###.##} \nInterest rate : 
{1:0#}% \n Interest Amount : ${2:#,###.##}", loanAmount, interest, interestAmount );
        Console.WriteLine("Total amount to be paid : ${0:#,###.##} ", totalAmount);
    }
}
```

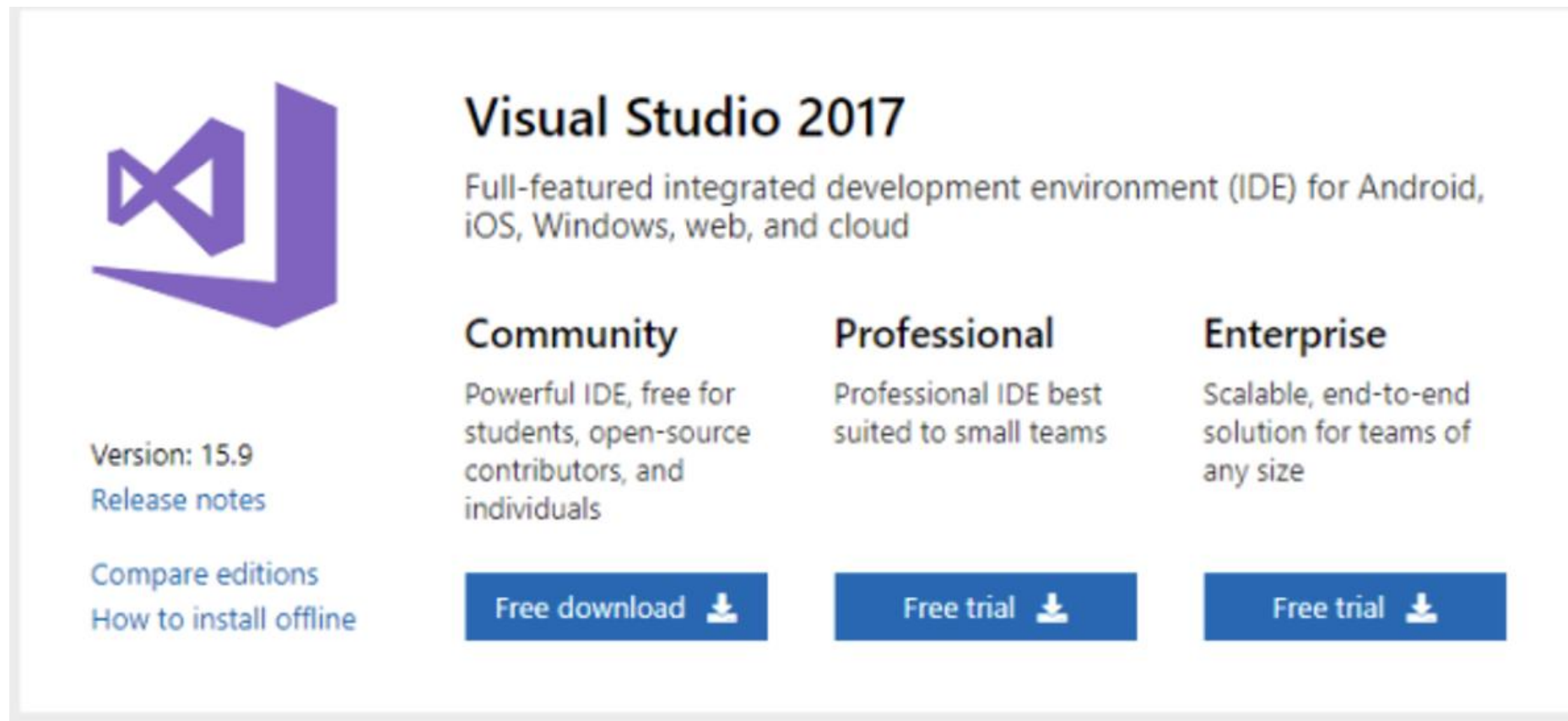
5.2. PLACEHOLDER (2)

BÀI TẬP

1. Chuẩn bị môi trường lập trình với ngôn ngữ lập trình C#. Sử dụng công cụ VS 2017 hoặc công cụ khác tùy vào hướng dẫn của giảng viên, chạy ứng dụng Hello Word.

➤ Để cài đặt VS 2017 cần đảm bảo một số yêu cầu hệ thống và phần mềm, học viên tham khảo: <https://docs.microsoft.com/en-us/visualstudio/productinfo/vs2017-systemrequirements-vs>

➤ Để tải Visual Studio 2017 vào web: <https://www.visualstudio.com/downloads>



The image shows the Visual Studio 2017 download page. On the left is the Visual Studio logo. To its right, the text 'Visual Studio 2017' is displayed, followed by a description: 'Full-featured integrated development environment (IDE) for Android, iOS, Windows, web, and cloud'. Below this, three editions are listed: 'Community' (described as 'Powerful IDE, free for students, open-source contributors, and individuals'), 'Professional' (described as 'Professional IDE best suited to small teams'), and 'Enterprise' (described as 'Scalable, end-to-end solution for teams of any size'). At the bottom left, there are links for 'Version: 15.9', 'Release notes', 'Compare editions', and 'How to install offline'. At the bottom right, there are three blue buttons: 'Free download' with a download icon, 'Free trial' with a download icon, and 'Free trial' with a download icon.

Visual Studio 2017
Full-featured integrated development environment (IDE) for Android, iOS, Windows, web, and cloud


Community
Powerful IDE, free for students, open-source contributors, and individuals


Professional
Professional IDE best suited to small teams


Enterprise
Scalable, end-to-end solution for teams of any size

Version: 15.9
[Release notes](#)

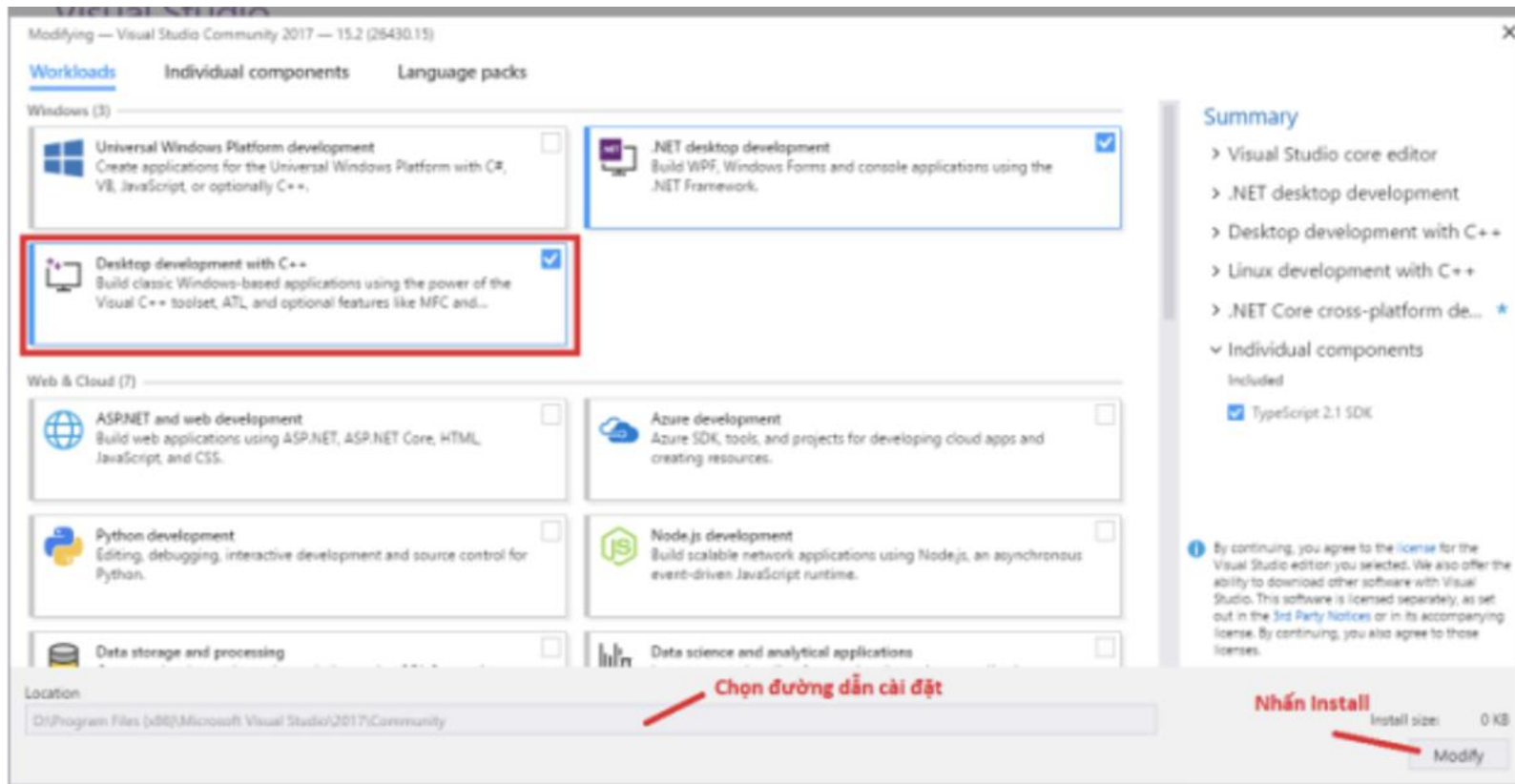
[Compare editions](#)
[How to install offline](#)

[Free download](#) 

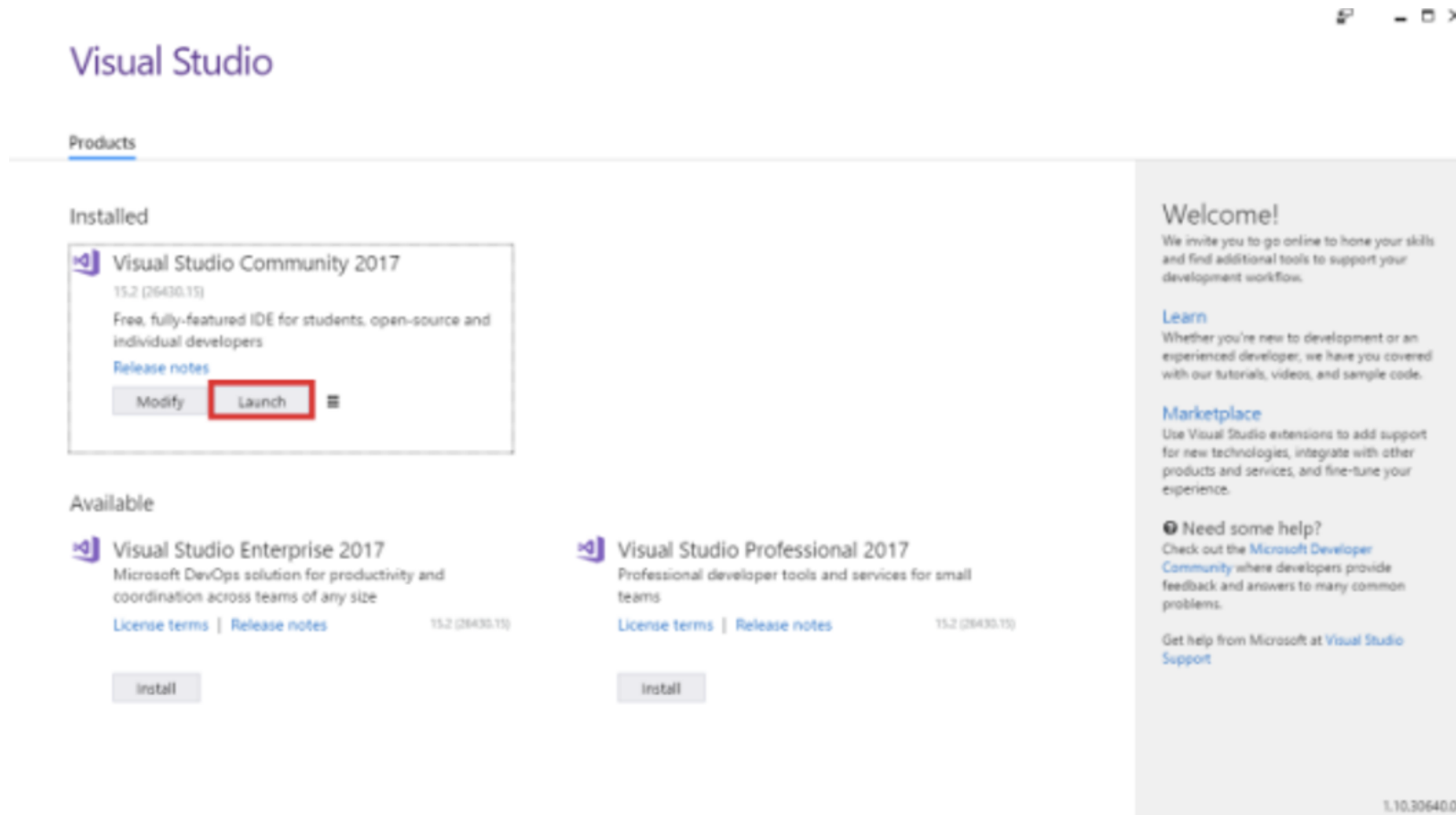
[Free trial](#) 

[Free trial](#) 

- Chọn bản Community rồi nhấn Free download, Download xong thì vào thư mục Download chạy file vs_Community.exe
- Chọn tính năng bạn muốn cài (.NET Framework, Visual C++, C++ for Linux, C#, .NET Core,...) sau đó nhấn **Install**



- Việc install nhanh chậm tùy vào cấu hình máy, sau khi cài xong thì khởi chạy

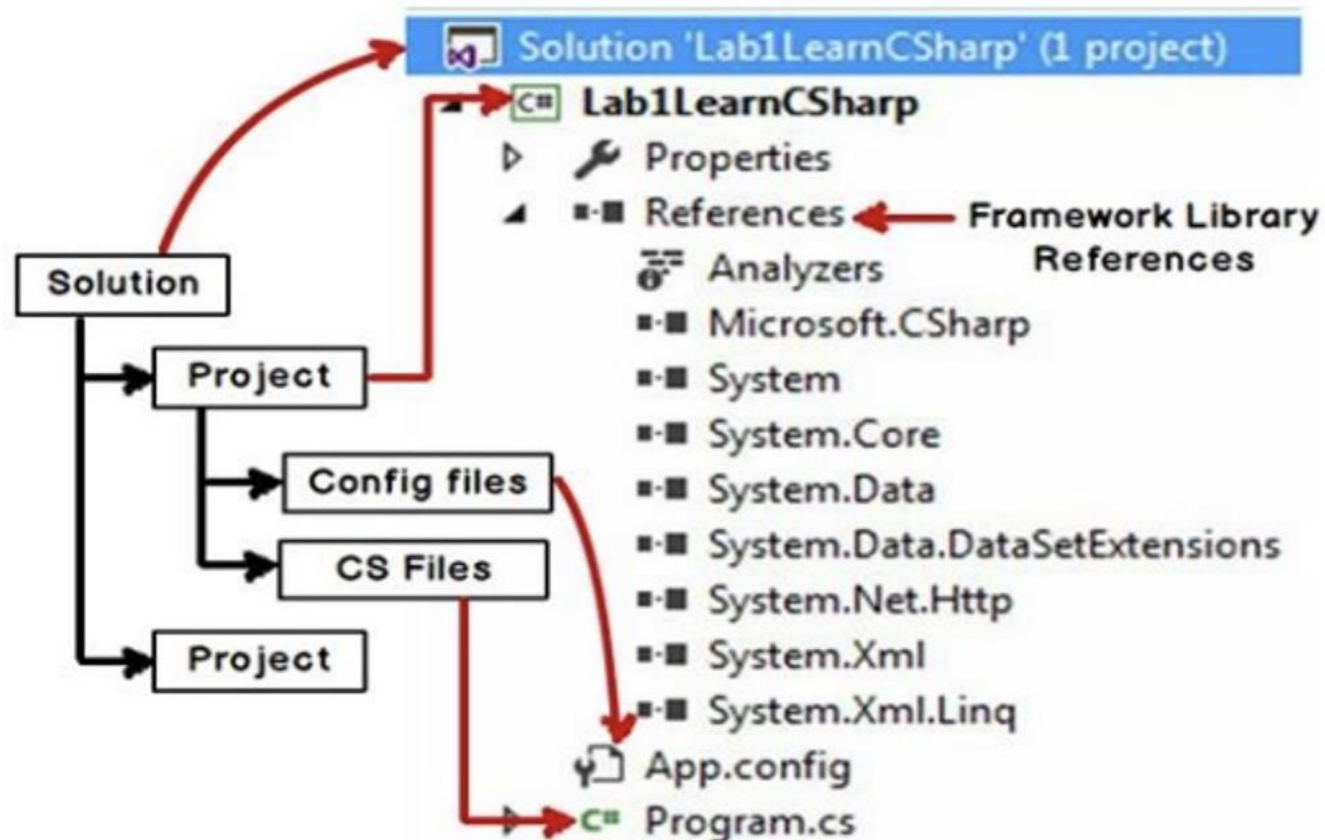


➤ Tạo ứng dụng Console HelloWorld

Chọn File->New->Project hoặc có thể sử dụng tổ hợp phím Ctrl+Shift+N

Chọn visual C# từ các mẫu cung cấp và chọn Windows, Chọn tiếp Console

Application => Đặt tên cho project => OK



Click **Start** hoặc **F5** để thực thi project. Một màn hình console sẽ xuất hiện hiện kết quả.

2. Viết chương trình nhập từ bàn phím 2 cạnh của hình chữ nhật. Tính và xuất chu vi, diện tích và cạnh nhỏ của hình chữ nhật.

HƯỚNG DẪN:

- Chu vi = $(\text{dai} + \text{rong}) * 2$
- Diện tích = $\text{dai} * \text{rong}$
- Cạnh nhỏ nhất = $\text{Math.min}(\text{dai}, \text{rong})$

3. Viết chương trình nhập từ bàn phím cạnh của một khối lập phương. Tính và xuất thể tích của khối chữ nhật.

HƯỚNG DẪN:

- Thể tích lập phương = $\text{canh} * \text{canh} * \text{canh}$
- Hoặc $\text{Math.pow}(\text{canh}, 3)$

4. Viết chương trình nhập các hệ số của phương trình bậc 2. Tính delta và xuất căn delta ra màn hình

HƯỚNG DẪN:

- $\text{Delta} = \text{Math.pow}(b, 2) - 4 * a * c$
- Sử dụng $\text{Math.sqrt}(\text{delta})$ để tính căn delta