

Công nghệ .NET

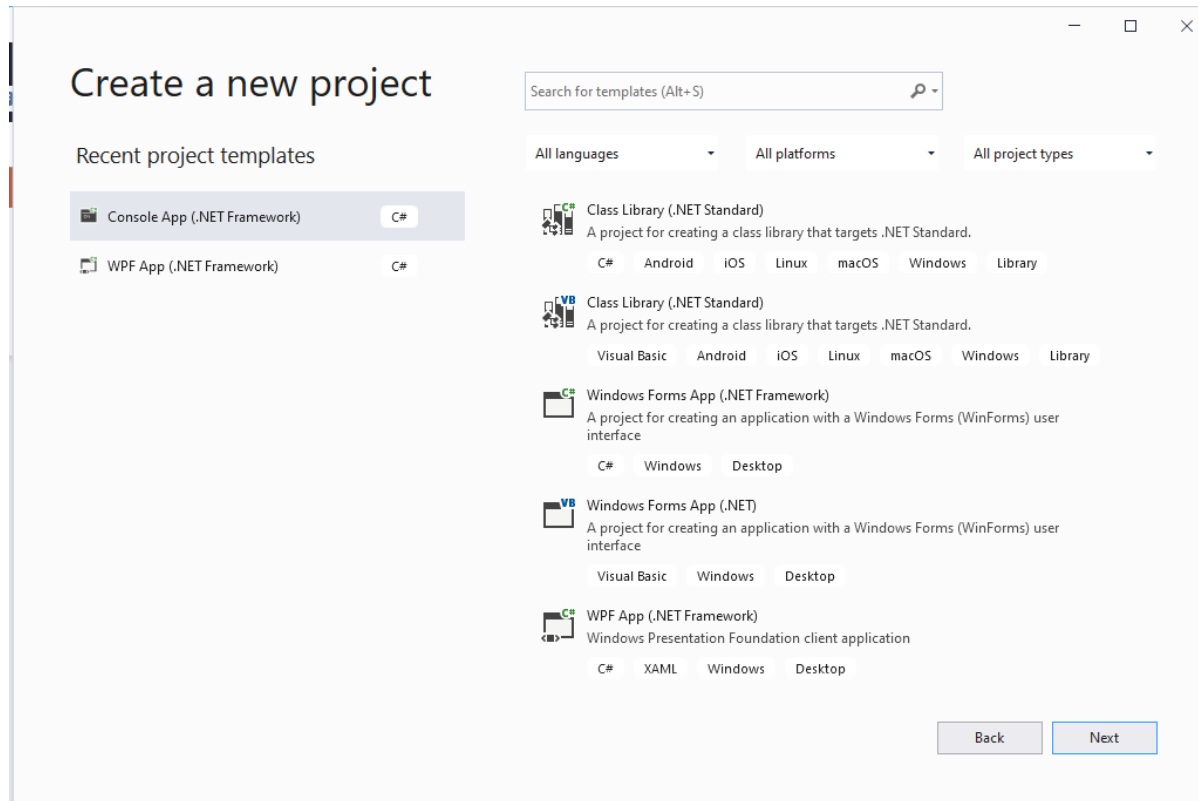
Bài 2 – Làm quen với C#

Nguyễn Thành Trung – Khoa CNTT

Email: trung.nguyenthanh1@phenikaa-uni.edu.vn

Hello World

- Khởi động Visual Studio 2019
- Chọn Create a new project => Console App (.NET Framework)



Hello World

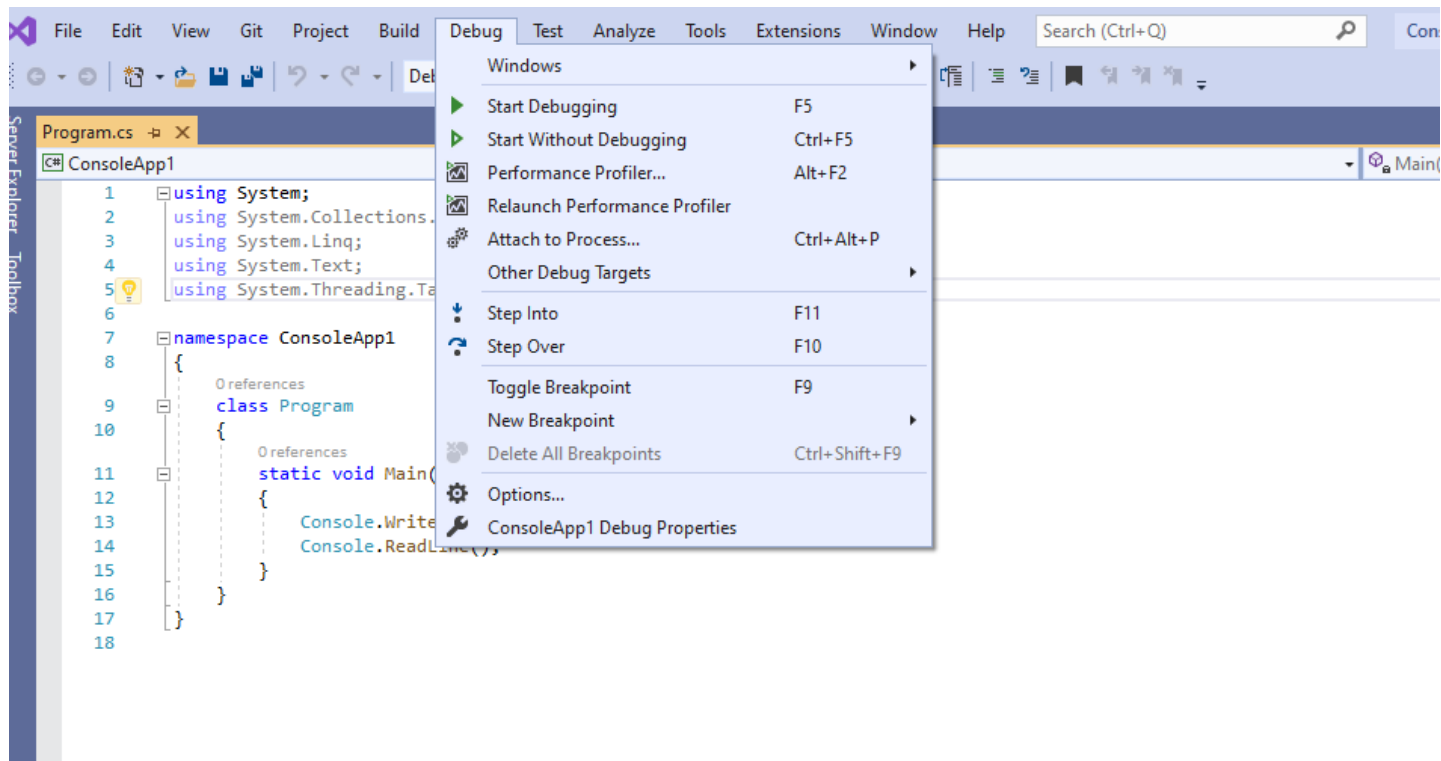
- Thiết lập các tham số
 - Project name: Tên dự án
 - Location: Thư mục chứa các tệp của dự án
 - Solution name: Tên solution (cho phép quản lý nhiều dự án có liên quan và sử dụng chung mã nguồn)
 - .Framework: lựa chọn phiên bản của .NET Framework
- Hello World

```
Console.WriteLine("Hello World");
Console.ReadKey();
```

=> Khi gõ một (vài) ký tự ban đầu, Visual Studio sẽ tự động gợi ý tên lệnh/hàm. (IntelliSense)

Hello World

- Biên dịch và chạy debug
 - Dịch và chạy chương trình ở chế độ debug: F5
 - Dịch và chạy chương trình: Ctrl + F5



Hello World

Welcome to C#

Cú pháp C# cơ bản

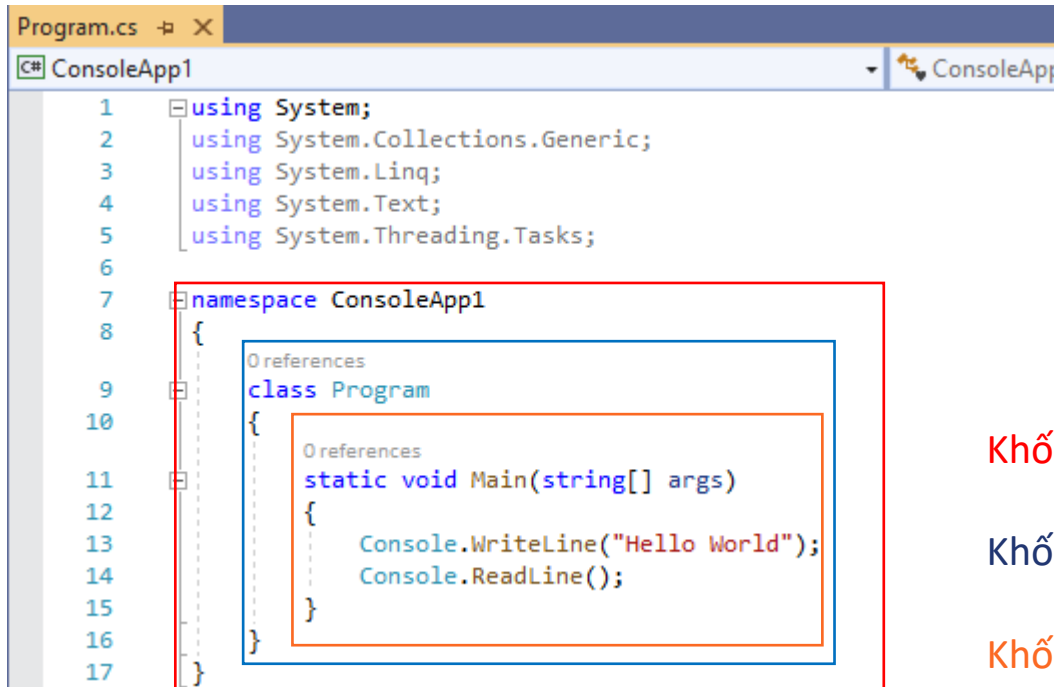
- Câu lệnh

- Bạn vừa được trải nghiệm viết **câu lệnh** (statement) trong phương thức main
- Vậy câu lệnh là gì ?
 - Là hành động chúng ta yêu cầu chương trình thực hiện. VD: truyền tham số, tính toán biểu thức,....
- Trình tự thực hiện các lệnh được gọi là luồng điều khiển (flow of control)
- Một câu lệnh trong C# kết thúc bằng **dấu chấm phẩy**

Cú pháp C# cơ bản

- Khối lệnh

- Một chuỗi các câu lệnh được nhóm lại thành một khối lệnh (code block), đặt chung trong một cặp dấu ngoặc nhọn {...}
- Các khối lệnh có thể lồng nhau



```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace ConsoleApp1
8  {
9      class Program
10     {
11         static void Main(string[] args)
12         {
13             Console.WriteLine("Hello World");
14             Console.ReadLine();
15         }
16     }
17 }
```

The image shows a C# code editor window titled 'Program.cs' with a 'ConsoleApp1' project. The code is as follows:

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace ConsoleApp1
8  {
9      class Program
10     {
11         static void Main(string[] args)
12         {
13             Console.WriteLine("Hello World");
14             Console.ReadLine();
15         }
16     }
17 }
```

Three nested blocks are highlighted with colored rectangles:

- A red rectangle highlights the `namespace ConsoleApp1` block (lines 7-17).
- A blue rectangle highlights the `class Program` block (lines 9-16).
- An orange rectangle highlights the `static void Main` method block (lines 11-15).

Khối lệnh namespace

Khối lệnh class

Khối lệnh phương thức main

Cú pháp C# cơ bản

- Ghi chú (comment) và khoảng trắng
 - Là những thông tin giải thích thêm cho chương trình và **không ảnh hưởng** đến quá trình dịch
 - Ghi chú trên 1 dòng: //
 - Ghi chú trên nhiều dòng: /* */
 - Ghi chú tài liệu: ///

```
/// <summary>  
/// class for students  
/// </summary>  
class Student {  
    // TODO: add more member  
}
```

Thường dùng để tạo ra một hướng dẫn cơ bản hoặc mô tả tổng quan cho một lớp (class), phương thức (method),...

Cú pháp C# cơ bản

- Từ khoá
 - Là những từ được quy định sẵn ý nghĩa trong C#
 - Danh sách các từ khoá trong C#:

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

Cú pháp C# cơ bản

- Định danh
 - Là chuỗi ký tự được dùng để đặt tên cho biến, hằng, tham số,...
 - Quy tắc định danh
 - Bao gồm các ký tự a-z, A-Z, _
 - Chữ số không đứng đầu
 - Có sự phân biệt giữa ký tự hoa và thường
 - Quy ước định danh: không bắt buộc nhưng được khuyến khích sử dụng để chương trình được “clear”
 - PascalCase: thường dùng để đặt tên kiểu (class, struct,...), tên phương thức (method),...
 - camelCase: thường dùng để đặt tên biến cục bộ, tham số, ...

Cú pháp C# cơ bản

- Lớp Program
 - Mỗi project bắt buộc phải có ít nhất một lớp (Class)
 - Program là lớp được tự động sinh ra khi khởi tạo một project mới.
- Phương thức Main()
 - Trong lớp Program có một phương thức (method) đặc biệt, được gọi đầu tiên khi chạy chương trình C#

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

Biến và hằng

- Biến
 - Biến cục bộ: lưu trữ thông tin trong phạm vi phương thức
 - Biến thành viên: lưu trữ thông tin trong phạm vi class hoặc struct
 - Tham số: lưu trữ tạm thời thông tin để truyền vào phương thức
- Khai báo biến

```
[datatype] [name]
```

```
int totalCom;
```

```
int totalCom = 5;
```

```
int totalCom = 5, viTech = 20;
```

Chú ý sử dụng quy ước camelCase khi đặt tên

Biến và hằng

- Phạm vi biến

```
static void Main()
{
    var i = 10;
    Console.WriteLine(i);

    { // bắt đầu một khối lệnh nào đó
        var j = 100; // j có phạm vi từ đây đến hết khối này
        Console.WriteLine(i); // phạm vi của i bao trùm khối này
        Console.WriteLine(j); // vẫn trong phạm vi của j
    } // kết thúc khối

    Console.WriteLine(i); // i vẫn còn tác dụng
    Console.WriteLine(j); // đã ra ngoài phạm vi của j => lỗi
}
```

Biến và hằng

- Hằng

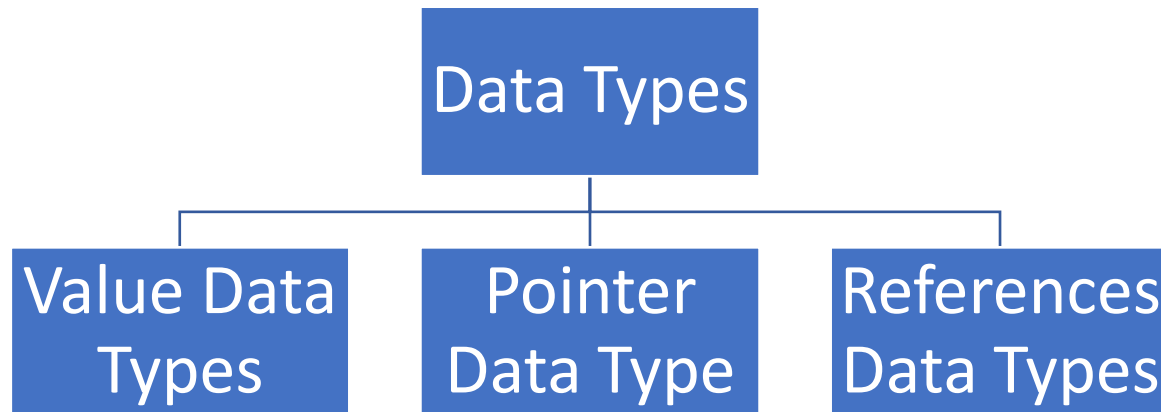
- Cú pháp

```
const [datatype] [name] = [value]
```

- Ví dụ: `const int x = 500;`
 - Bắt buộc phải được khởi tạo giá trị ngay lúc khai báo
 - Giá trị của hằng không thể thay đổi sau đó

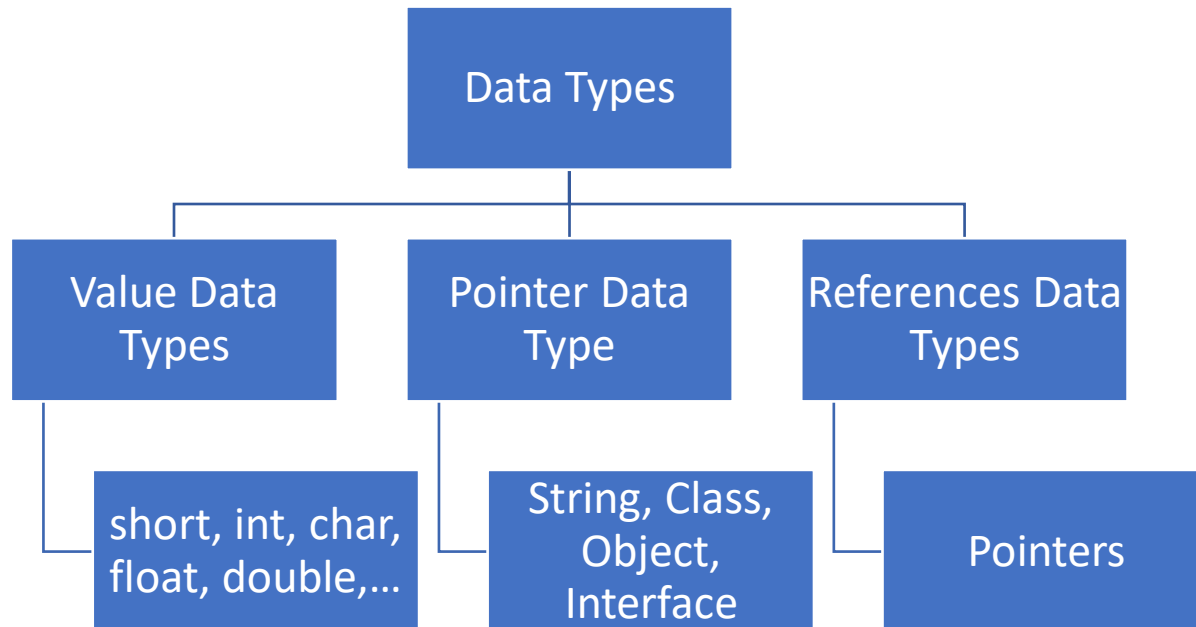
Kiểu dữ liệu

- C# là một ngôn ngữ gắn liền với .NET Framework. Do đó, .NET sẽ là nơi cung cấp các kiểu dữ liệu cho C#.
- Để đơn giản hoá lệnh thì C# định nghĩa tên riêng cho một số kiểu cơ bản.



Kiểu dữ liệu

- C# là một ngôn ngữ gắn liền với .NET Framework. Do đó, .NET sẽ là nơi cung cấp các kiểu dữ liệu cho C#.
- Để đơn giản hoá lệnh thì C# định nghĩa tên riêng cho một số kiểu cơ bản.



Kiểu dữ liệu

- Value Data Type

- Các kiểu dữ liệu được xác định trước - như Integer, Boolean, Float, v.v.
- Các kiểu dữ liệu do người dùng xác định - như Structure, ...
- Kích thước bộ nhớ của kiểu dữ liệu có thể thay đổi theo hệ điều hành 32 hoặc 64 bit.

Data Types	Memory Size	Range
char	1 byte	-128 to 127
signed char	1 byte	-128 to 127
unsigned char	1 byte	0 to 127
short	2 byte	-32,768 to 32,767
signed short	2 byte	-32,768 to 32,767
unsigned short	2 byte	0 to 65,535
int	4 byte	-2,147,483,648 to -2,147,483,647
signed int	4 byte	-2,147,483,648 to -2,147,483,647
unsigned int	4 byte	0 to 4,294,967,295

Kiểu dữ liệu

Data Types	Memory Size	Range
long	8 byte	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
signed long	8 byte	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
unsigned long	8 byte	0 - 18,446,744,073,709,551,615
float	4 byte	$1.5 * 10^{-45}$ - $3.4 * 10^{38}$, 7-digit precision
double	8 byte	$5.0 * 10^{-324}$ - $1.7 * 10^{308}$, 15-digit precision
decimal	16 byte	at least $-7.9 * 10^{28}$ - $7.9 * 10^{28}$, with at least 28-digit precision

Kiểu dữ liệu

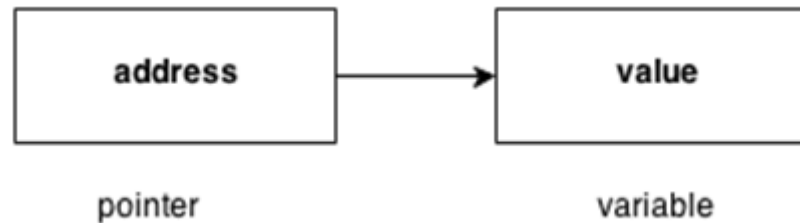
- Reference Data Type

- Kiểu dữ liệu tham chiếu không chứa dữ liệu thực tế được lưu trữ trong một biến, mà chứa một tham chiếu đến các biến.
- Có 2 loại kiểu dữ liệu tham chiếu trong ngôn ngữ C #.
 - Các loại được định nghĩa trước: Objects, String.
 - Các kiểu do người dùng định nghĩa: Classes, Interface.

Kiểu dữ liệu

- Pointer Data Type

- Con trỏ là là **một biến** được dùng để lưu trữ **địa chỉ** của biến khác.



- Các ký hiệu được sử dụng trong con trỏ

Ký hiệu	Ý nghĩa
&	Xác định địa chỉ của một biến.
*	Truy cập giá trị của một địa chỉ.

Toán tử

Nhóm	Toán tử
Phép toán số học	+ - * / %
Phép toán logic và nhị phân	& ^ ~ && !
Phép toán ghép chuỗi	+
Phép toán tăng giảm	++ --
Phép toán dịch bit	<< >>
Phép toán so sánh	== != < > <= >=
Phép gán	= += -= *= /= %=
Phép toán truy xuất thành viên (object và struct)	.
Phép toán indexer (cho mảng)	[]
Ép kiểu	()
Phép toán điều kiện	?:

Console trong C#

- Giao diện dòng lệnh (CUI hay CLI) là giao diện đơn giản nhất được sử dụng khi bắt đầu với một ngôn ngữ lập trình.
- Trong C#, lớp `System.Console` được sử dụng để thực hiện các công việc với giao diện dòng lệnh.
 - `Write()`, `WriteLine()`: in thông tin ra
 - `WriteLine()`: sau khi in thông tin thì sẽ xuống một dòng mới

Console trong C#

- Read(), ReadLine(), ReadKey(): đọc thông tin vào
 - ReadLine(): đọc một dòng và trả về một chuỗi ký tự
 - Read(): đọc một ký tự và trả về mã của ký tự đó.
 - ReadKey(): đọc một ký tự và trả về kiểu ConsoleKeyInfo. Phương thức này thường dùng để dừng màn hình chờ người dùng ấn một phím bất kỳ để tiếp tục. Kiểu ConsoleKeyInfo lưu trữ một số thông tin như sau:

- KeyChar: ký tự người dùng đã nhập
- Modifiers: các phím điều khiển bấm cùng như Ctrl, Shift, Alt
- Key: phím chuẩn trên console

```
Console.Title = "ReadKey";

Console.Write("Press any key: ");
var key = Console.ReadKey();

Console.WriteLine();
Console.WriteLine(key.KeyChar);
Console.WriteLine(key.Modifiers);

if (key.Key == ConsoleKey.H)
    Console.WriteLine("Hello!");

Console.ReadKey();
```

Console trong C#

- `BackgroundColor`: đặt màu nền cho văn bản
- `ForegroundColor`: đặt màu văn bản
- `Title`: đặt tiêu đề cho cửa sổ dòng lệnh
- `Clear()`: xoá nội dung

Console trong C#

- Ví dụ

```
Console.Title = "Colorful Console";

Console.ForegroundColor = ConsoleColor.Magenta;
Console.Write("Your name: ");
Console.ResetColor();
var name = Console.ReadLine();
Console.BackgroundColor = ConsoleColor.Blue;
Console.Write(name);

Console.ReadLine();
```

Bài tập

- Bài 2.1: Dùng biểu thức điều kiện ?: để đưa ra số lớn nhất trong 3 số thực nhập vào từ bàn phím
- Bài 2.2: Viết chương trình yêu cầu người dùng nhập vào bán kính của vòng tròn, sau đó tính và đưa ra màn hình diện tích và chu vi hình tròn đó. Sử dụng hằng số khai báo PI.
- Bài 2.3: Viết chương trình nhập vào các thông tin trên thẻ sinh viên, sau đó in ra màn hình các thông tin đó trên từng dòng với định dạng màu.