

BÀI GIẢNG

CƠ SỞ LẬP TRÌNH

CHƯƠNG 1.

**TỔNG QUAN VỀ LẬP TRÌNH
VÀ NGÔN NGỮ C#**

NGUYỄN THÀNH THỦY

BỘ MÔN TIN HỌC QUẢN LÝ

TRƯỜNG ĐẠI HỌC KINH TẾ, ĐẠI HỌC ĐÀ NẴNG

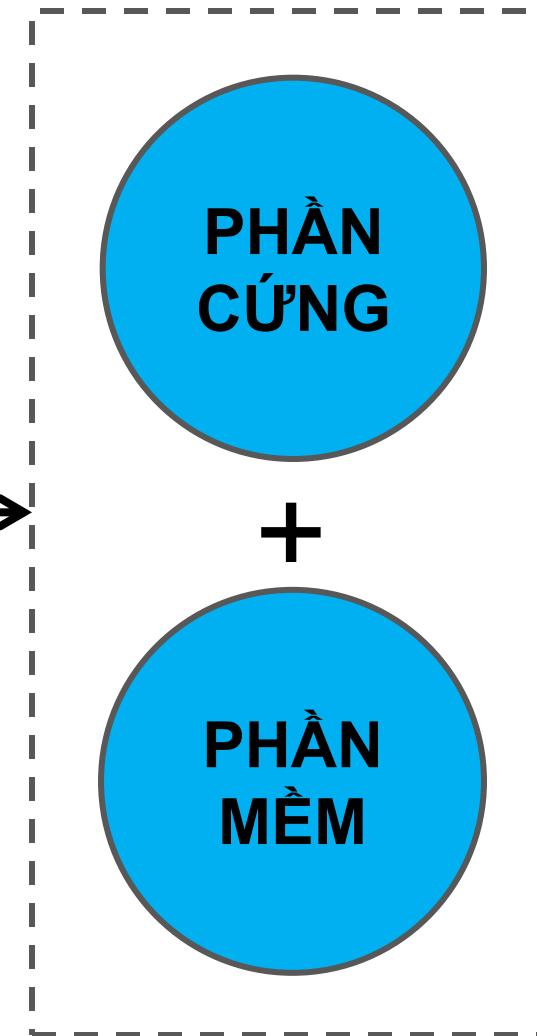
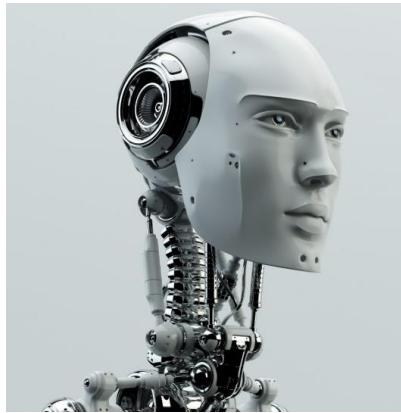
THUYNT@DUE.EDU.VN

NỘI DUNG

- Các khái niệm
- Các bước xây dựng chương trình
- Ngôn ngữ lập trình C#
- Môi trường làm việc MS Visual C#

CÁC KHÁI NIỆM

- ❑ Vì sao máy tính có năng lực mạnh mẽ ...?

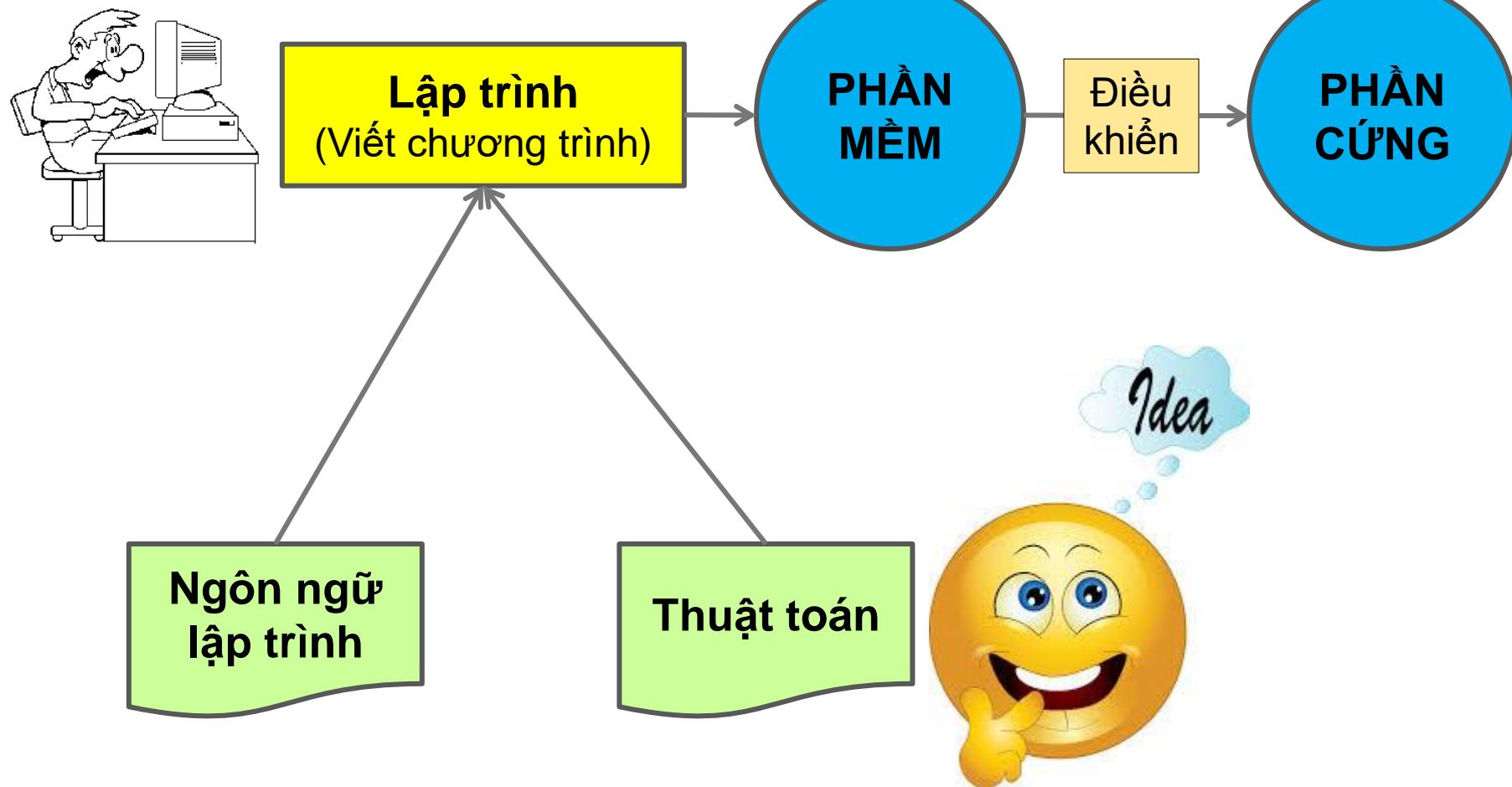


CÁC KHÁI NIỆM

□ Các ứng dụng minh họa trí thông minh của máy tính:

- Phần mềm **Calculator**
- Robot đánh bóng bàn 
- Cuộc thi MicroMouse 
- ...

CÁC KHÁI NIỆM



CÁC KHÁI NIỆM

□ Thuật toán (*Giải thuật*)

- Là một **dãy hữu hạn** các thao tác được **sắp xếp** theo một trình tự xác định, sao cho từ **dữ liệu đầu vào (Input)** của bài toán, ta nhận được **kết quả đầu ra (Output)** cần tìm.

CÁC KHÁI NIỆM

□ Thuật toán (*Giải thuật*)

- **Ví dụ 1:** xây dựng thuật toán giải và biện luận phương trình bậc nhất $ax + b = 0$

Input: a và b

Output: nghiệm của phương trình

Thuật toán:

- Nếu $a \neq 0$: pt có nghiệm $x = -b/a$
- Ngược lại, $a = 0$:
 - Nếu $b = 0$: pt có vô số nghiệm
 - Nếu $b \neq 0$: pt vô nghiệm

CÁC KHÁI NIỆM

□ Thuật toán (*Giải thuật*)

- **Ví dụ 2:** xây dựng thuật toán tìm số có giá trị lớn nhất trong 3 số a, b, c bất kỳ

Input: a, b và c

Output: giá trị lớn nhất

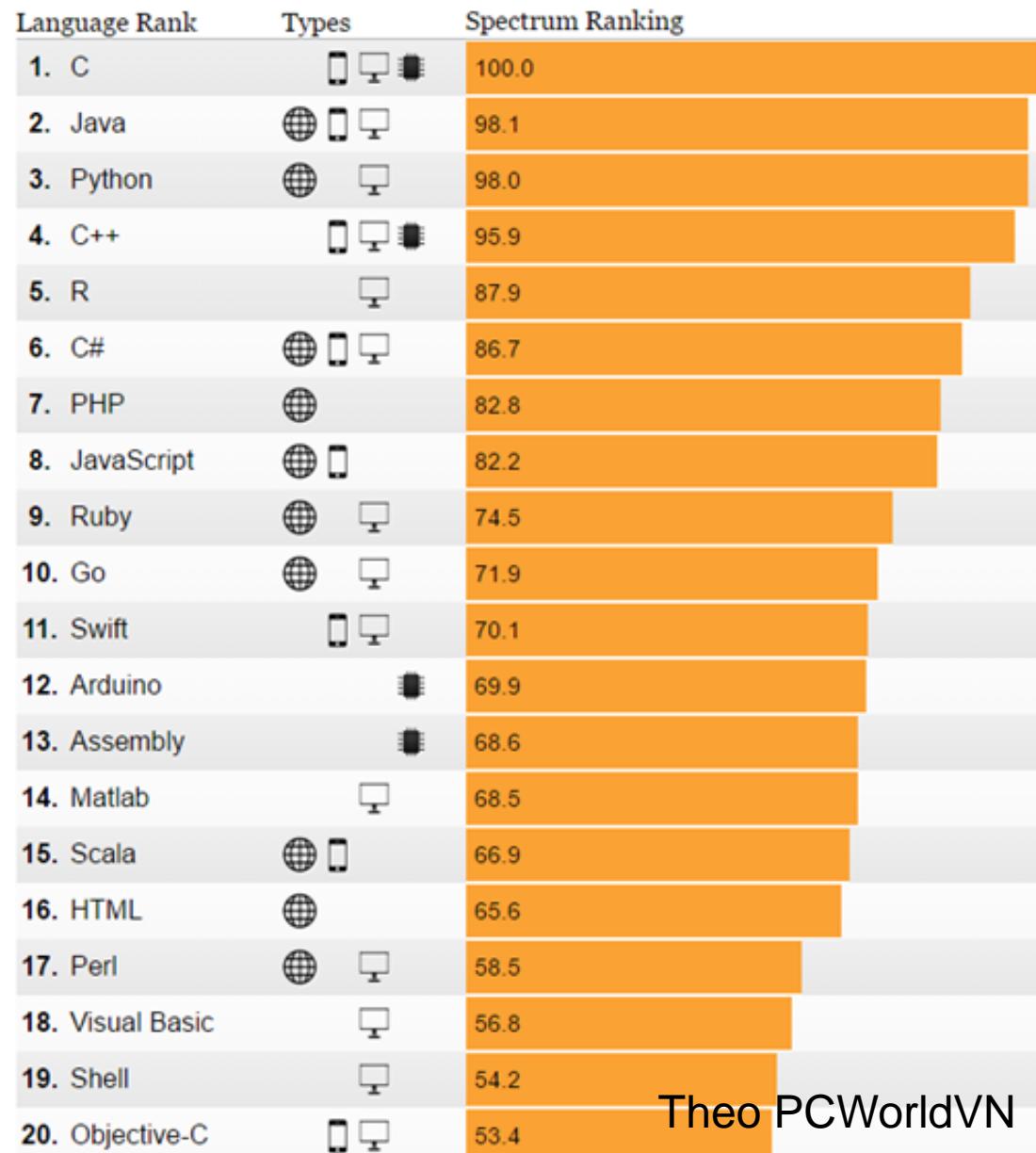
Thuật toán: tìm số lớn nhất

- **Bước 1:** $\text{max}=a$
- **Bước 2:** nếu $\text{max} < b$ thì đặt $\text{max}=b$
- **Bước 3:** nếu $\text{max} < c$ thì đặt $\text{max}=c$
- **Bước 4:** kết luận **max** là giá trị lớn nhất

CÁC KHÁI NIỆM

□ Ngôn ngữ lập trình

- Là **một tập các chỉ thị** được sắp xếp theo một trật tự nhất định, nhằm **hướng dẫn máy tính thực hiện các hành động cần thiết** để đáp ứng một mục tiêu đã định trước của con người như **truy xuất dữ liệu, tìm kiếm, giải bài toán**,...



Theo PCWorldVN

Các ngôn ngữ lập trình Phổ biến nhất 2016

CÁC KHÁI NIỆM

□ Lập trình

- Là việc **cài đặt** một hoặc nhiều **thuật toán** có liên quan với nhau bằng một **ngôn ngữ lập trình** để tạo ra một **chương trình** trên **máy tính**

CÁC KHÁI NIỆM

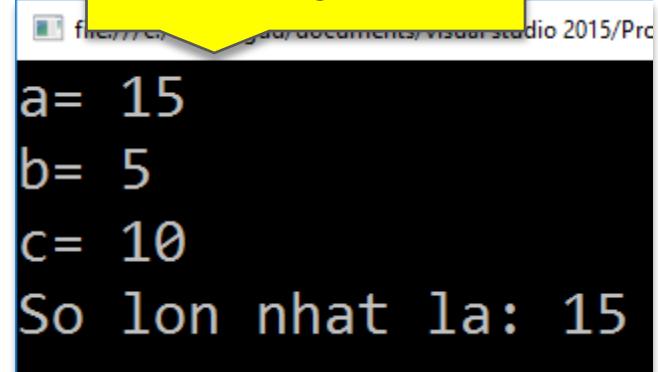
□ Lập trình

- Ví dụ: Chương trình tìm số lớn nhất trong 3 số

```
static void Main(string[] args)
{
    int a,b,c,max;
    Console.Write("a= ");
    a=Convert.ToInt32(Console.ReadLine());
    Console.Write("b= ");
    b = Convert.ToInt32(Console.ReadLine());
    Console.Write("c= ");
    c = Convert.ToInt32(Console.ReadLine());

    max = a;
    if (max < b) max = b;
    if (max < c) max = c;
    Console.WriteLine("So lon nhat la: " + max);
    Console.ReadLine();
}
```

Kết quả khi chạy
chương trình



```
a= 15
b= 5
c= 10
So lon nhat la: 15
```

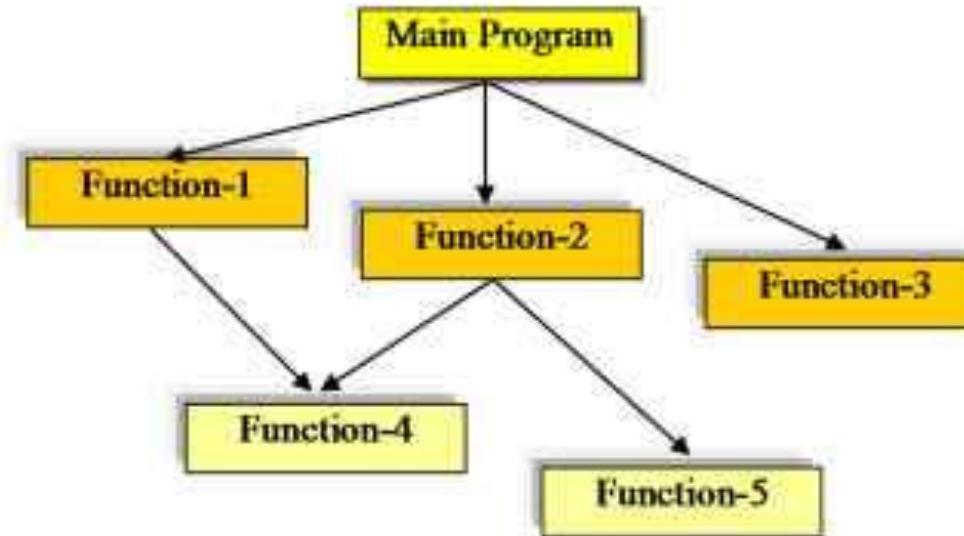
CÁC KHÁI NIỆM

□ Các kỹ thuật lập trình

■ Lập trình hướng cấu trúc (thủ tục/chức năng)

- Là phương pháp chia một chương trình lớn thành các khối chức năng (Hàm/Thủ tục) nhỏ, để dễ lập trình và kiểm tra.

Procedure-oriented Programming

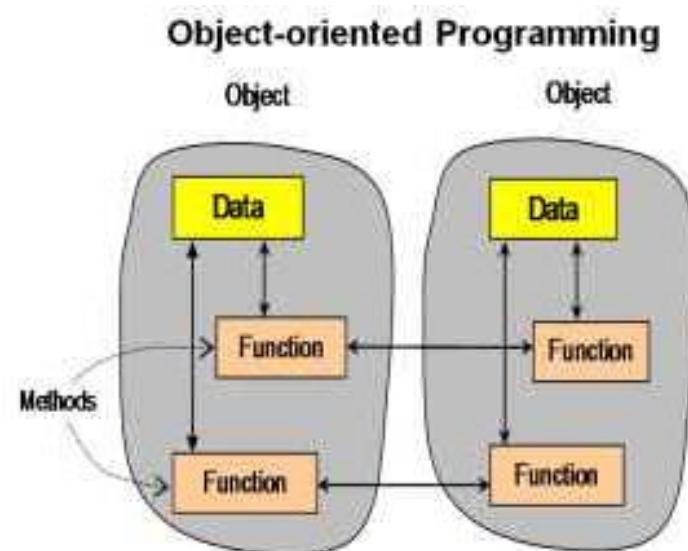


CÁC KHÁI NIỆM

□ Các kỹ thuật lập trình

■ Lập trình hướng đối tượng

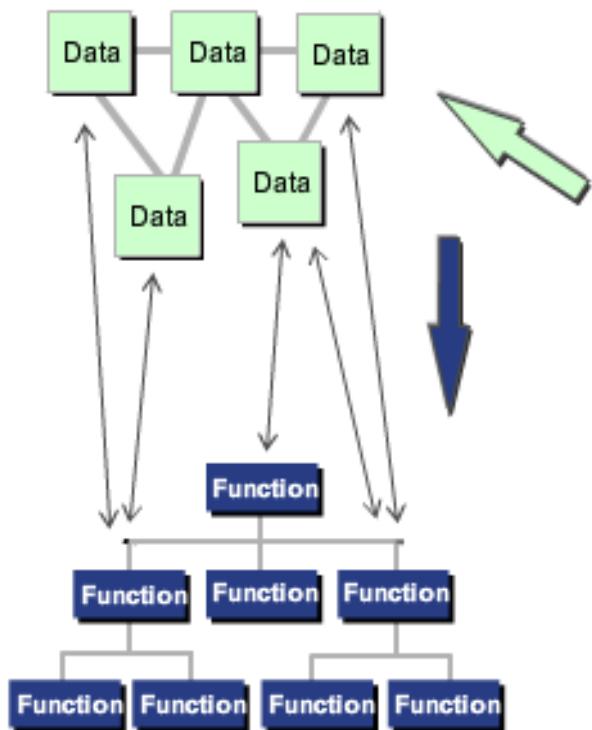
- Chương trình được chia thành các **Đối tượng** (Object) độc lập;
- Mỗi **Đối tượng** sẽ quản lý riêng **Phương thức** (Chức năng) và **Thuộc tính** (Dữ liệu) của riêng nó;
- Các **Đối tượng** sẽ trao đổi thông tin với nhau thông qua các **Phương thức**



CÁC KHÁI NIỆM

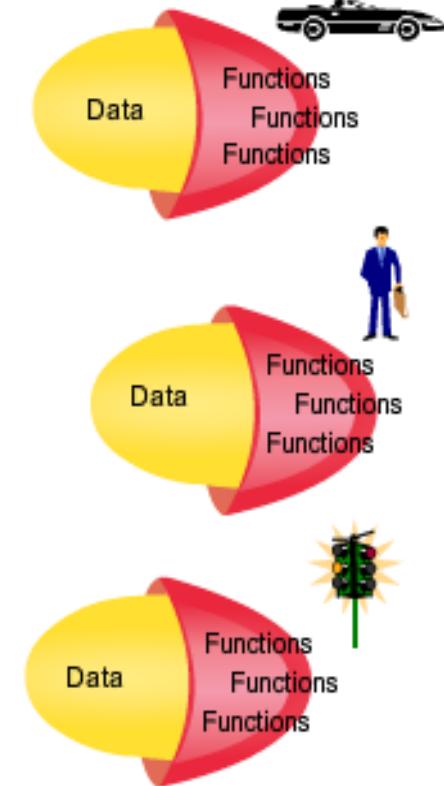
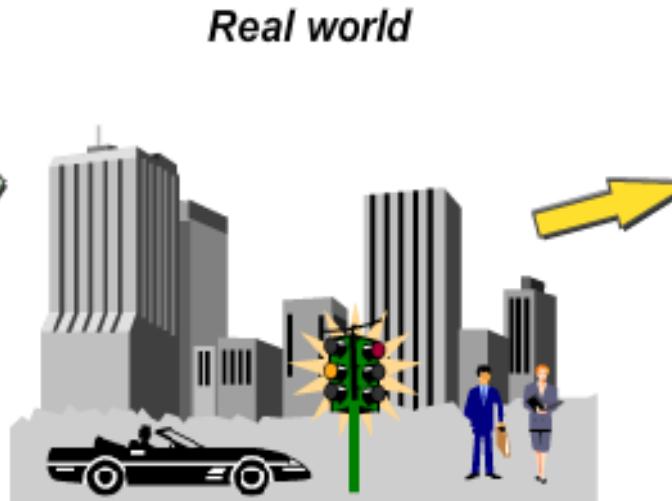
Procedural:

Separation of data and functions



Object-oriented:

Encapsulation of data and functions



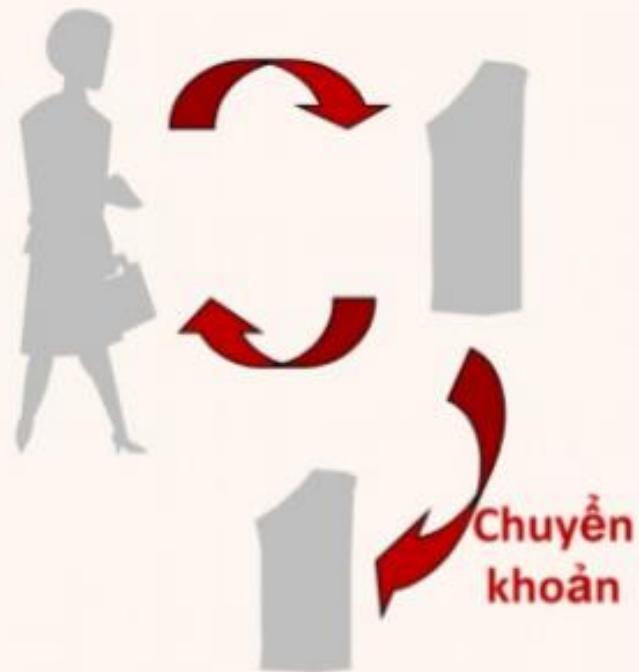
CÁC KHÁI NIỆM

- **Bài toán quản lý giao dịch Ngân hàng**
 - Khách hàng có các giao dịch với Ngân hàng
 - Nạp tiền vào Tài khoản
 - Rút tiền từ Tài khoản
 - Chuyển khoản giữa các Tài khoản

CÁC KHÁI NIỆM

Lập trình hướng CHỨC NĂNG

Quản lý Ngân hàng



Các chức năng: **Nạp tiền, Rút tiền, Chuyển khoản**

Lập trình hướng ĐỐI TƯỢNG

Quản lý Ngân hàng



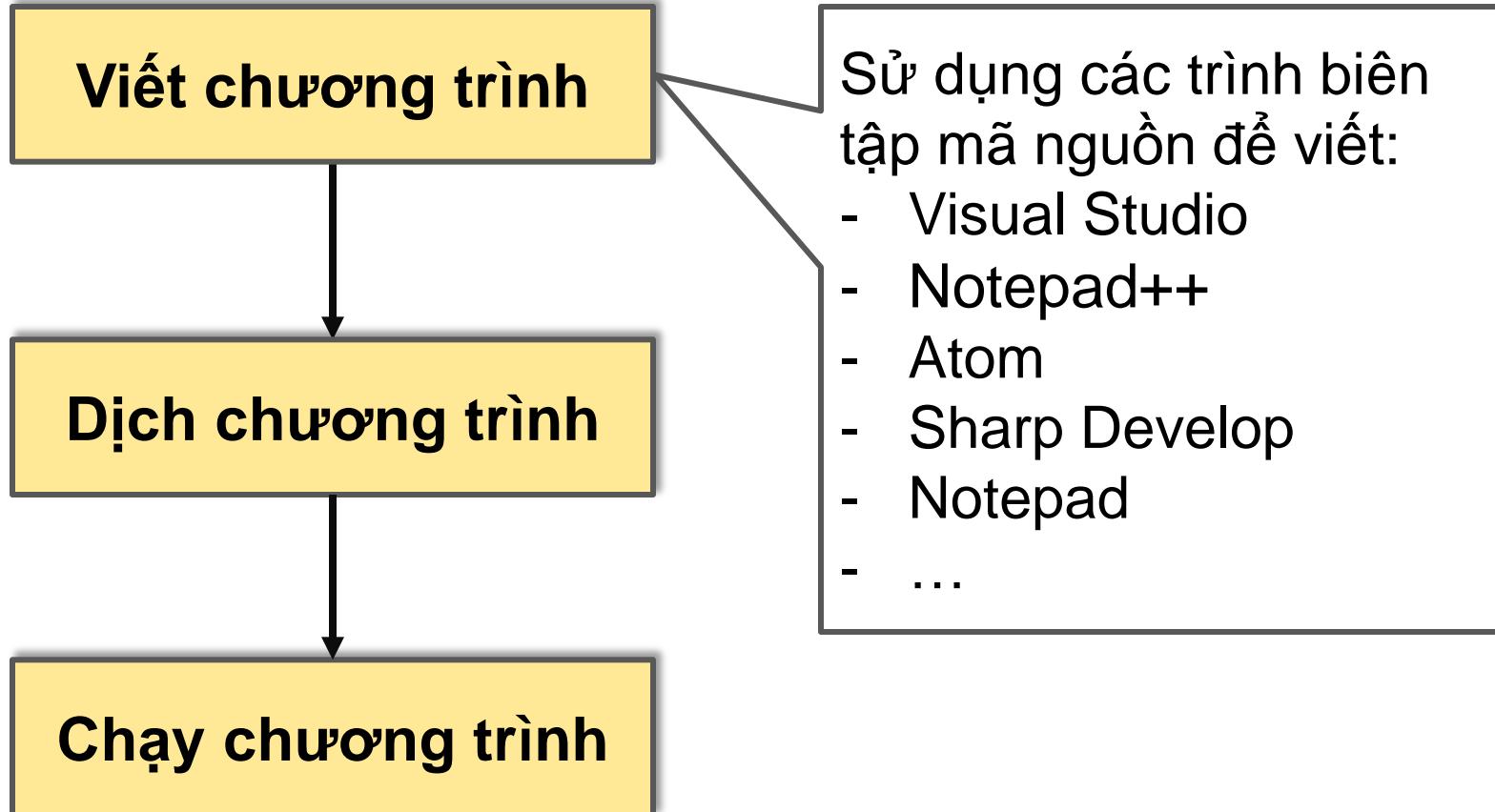
Các đối tượng: **Khách hàng, Tiền, Tài khoản**

CÁC BƯỚC XÂY DỰNG CHƯƠNG TRÌNH



NGÔN NGỮ LẬP TRÌNH C#

□ Các bước để xây dựng chương trình C#



NGÔN NGỮ LẬP TRÌNH C#

□ Tổng quan về C#

- Thuộc bộ ngôn ngữ .NET của Microsoft (*C++, Visual Basic, C#, J#*);
- Là ngôn ngữ đơn giản, hiện đại, hướng đối tượng, mạnh mẽ và mềm dẻo, có ít từ khóa (gần 80 từ), hướng module và phổ biến hiện nay;

NGÔN NGỮ LẬP TRÌNH C#

□ Danh mục từ khóa trong ngôn ngữ C#

| | | | | | | |
|----------|------------|----------|--------------|------------|----------|-----------|
| 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 |

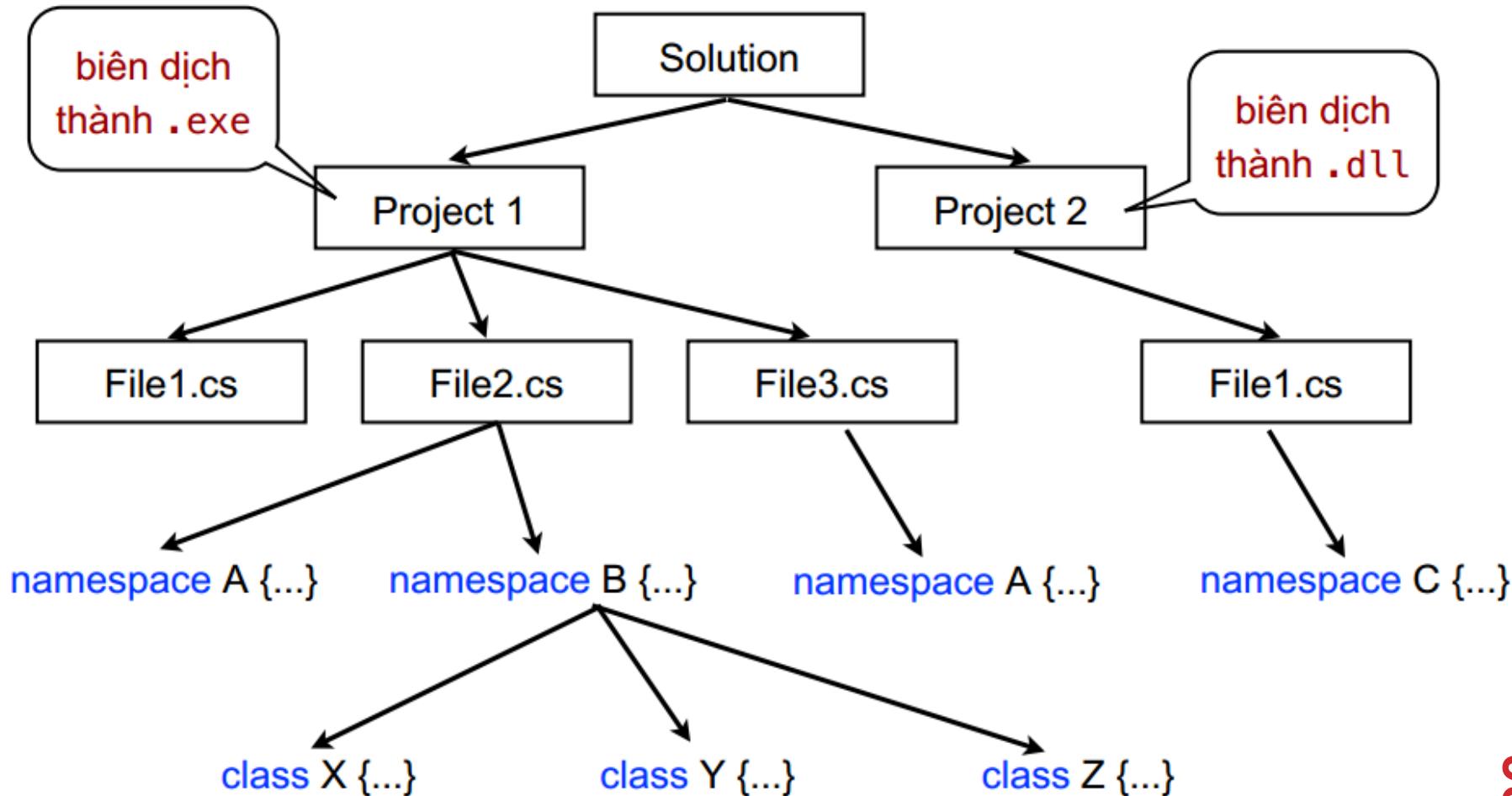
NGÔN NGỮ LẬP TRÌNH C#

□ Cấu trúc của một chương trình C#

- Mỗi Project được đặt trong một Solution
 - Mỗi Solution có thể chứa nhiều Project
- Một Project bao gồm một hoặc nhiều đơn vị biên dịch
 - Mỗi đơn vị là một file mã nguồn riêng biệt (.cs)
 - Program.cs là tên mặc định của đơn vị biên dịch đầu tiên của project
- Mỗi Project được biên dịch thành một tập tin assembly (.exe hoặc .dll)

NGÔN NGỮ LẬP TRÌNH C#

□ Cấu trúc của một chương trình C#



NGÔN NGỮ LẬP TRÌNH C#

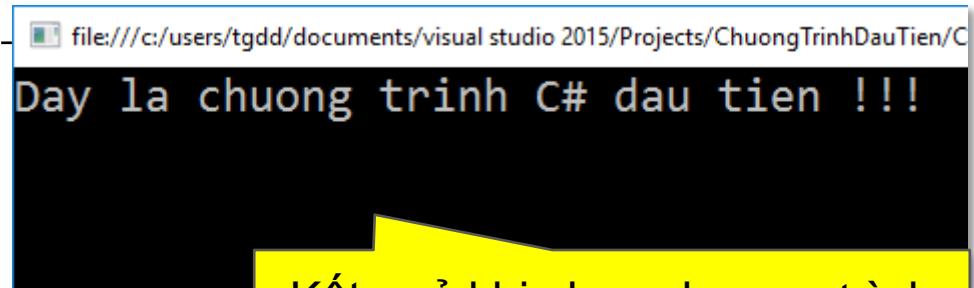
□ Cấu trúc của một chương trình C#

- Chương trình C# đầu tiên

Program.cs

```
using System;

namespace ChuongTrinhDauTien
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            //Viết nội dung lên màn hình
            Console.WriteLine("Day la chuong trinh C# dau tien !!!");
            Console.Read(); //Dừng màn hình
        }
    }
}
```



NGÔN NGỮ LẬP TRÌNH C#

□ Các thành phần trong chương trình C#

Program.cs

```
using System;
```

```
namespace ChuongTrinhDauTien
```

```
{
```

```
    0 references
```

```
    class Program
```

```
{
```

```
    0 references
```

```
        static void Main(string[] args)
```

```
        { //Viết nội dung lên màn hình
```

```
            Console.WriteLine("Day la chuong trinh C# dau tien !!!");
```

```
            Console.Read(); //Dừng màn hình
```

```
        }
```

```
}
```

```
}
```

Các tập tin mã nguồn của C# có phần mở rộng .cs

- **Program.cs** là tập tin mặc định để bắt đầu chương trình trong C#.
- Hàm **Main(...)** là hàm bắt đầu để gọi thực thi chương trình.

NGÔN NGỮ LẬP TRÌNH C#

□ Các thành phần trong chương trình C#

Program.cs

```
using System;
namespace ChuongTrinhDauTien
{
    class Program
    {
        static void Main(string[] args)
        {
            //Viết nội dung lên màn hình
            Console.WriteLine("Day la chuong trinh C# dau tien !!!");
            Console.Read(); //Dừng màn hình
        }
    }
}
```

using: từ khóa khai báo thư viện;
namespace: từ khóa khai báo một không gian tên, có thể có nhiều namespace lồng nhau;
class: từ khóa khai báo một lớp

NGÔN NGỮ LẬP TRÌNH C#

□ Các thành phần trong chương trình C#

Program.cs

```
using System;  
  
namespace ChuongTrinhDauTien  
{  
    class Program  
    {  
        static void Main(string[] args)  
        {  
            //Viết nội dung lên màn hình  
            Console.WriteLine("Day la chuong trinh C# dau tien !!!");  
            Console.Read(); //Dừng màn hình  
        }  
    }  
}
```

Mỗi câu lệnh thường kết thúc bằng dấu chấm phẩy (;)
Khối lệnh được đặt trong cặp dấu { và }

NGÔN NGỮ LẬP TRÌNH C#

□ Các thành phần trong chương trình C#

Program.cs

```
using System;  
  
namespace ChuongTrinhDauTien  
{  
    0 references  
    class Program  
    {  
        0 references  
        static void Main(string[] args)  
        {  
            //Viết nội dung lên màn hình  
            Console.WriteLine("Day la chuong trinh C# dau tien !!!");  
            Console.Read(); //Dừng màn hình  
        }  
    }  
}
```

Để thêm ghi chú vào chương trình,
có 2 cách:

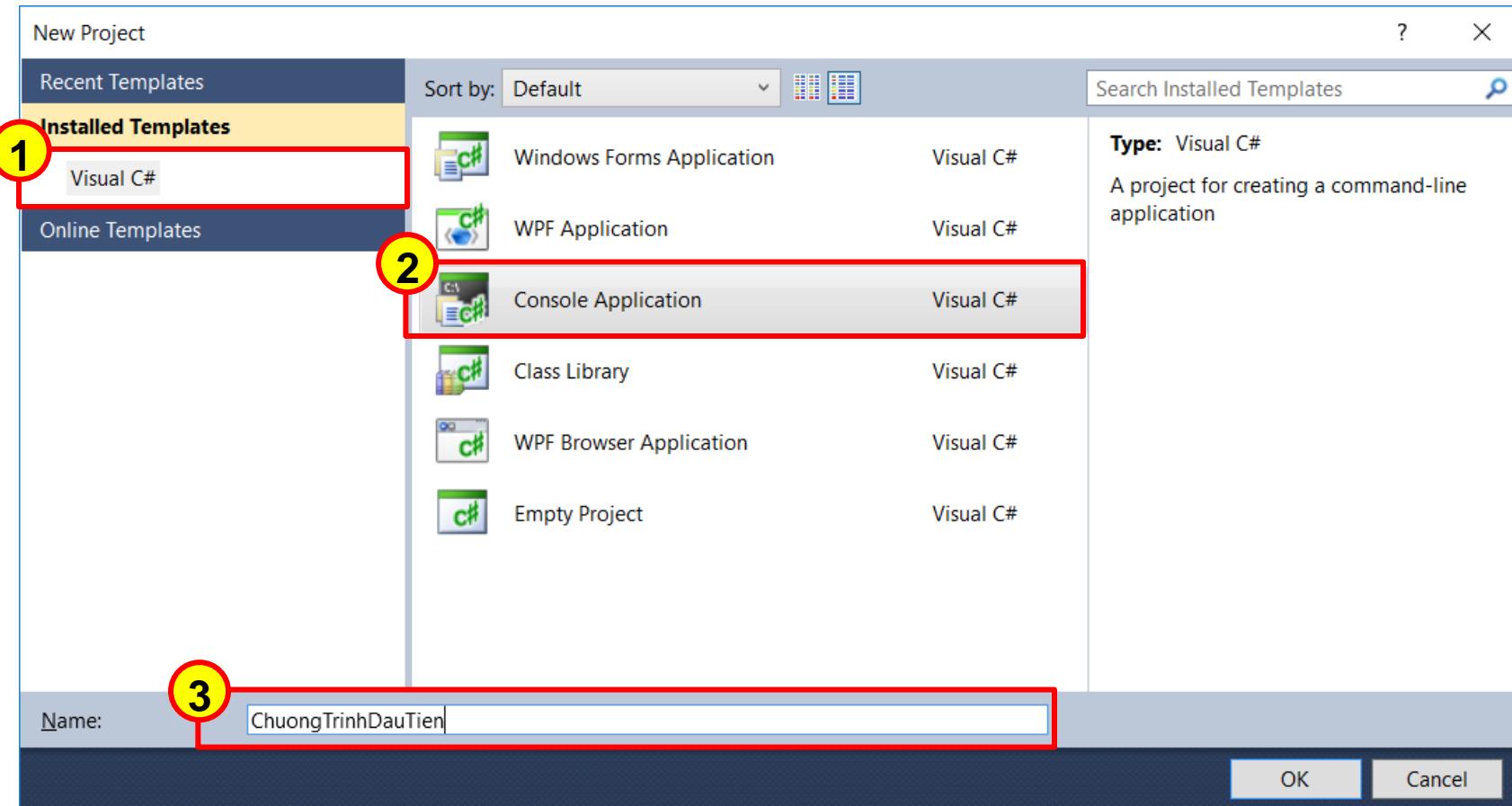
Cách 1: // Một dòng ghi chú

Cách 2: /* Nhiều dòng ghi chú */

MÔI TRƯỜNG LÀM VIỆC CỦA VISUAL STUDIO

☐ Tạo một dự án (Solution) mới

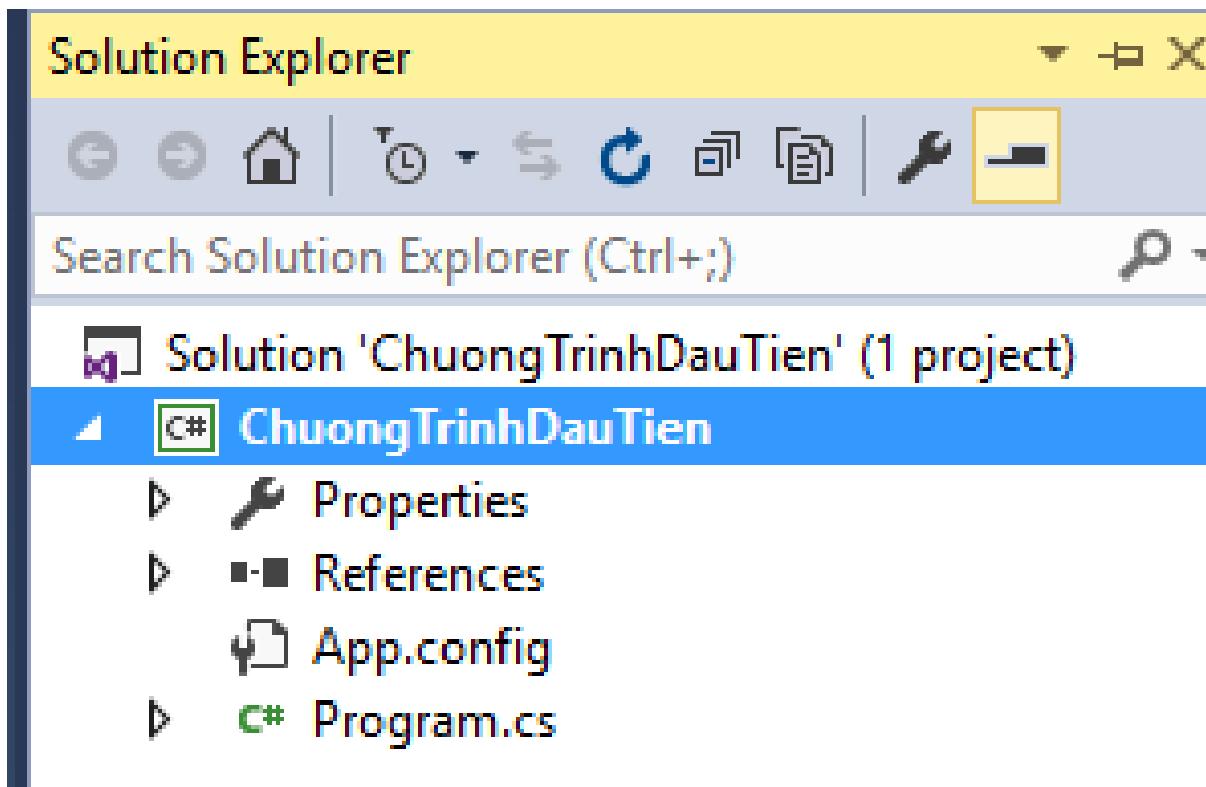
- File → New Project → Visual C# → Console Application → Nhập tên Project → OK



MÔI TRƯỜNG LÀM VIỆC CỦA VISUAL STUDIO

□ Solution Explorer

- Cửa sổ quản lý các tập tin và thư mục trong dự án



MÔI TRƯỜNG LÀM VIỆC CỦA VISUAL STUDIO

□ Code Editer

The screenshot shows the Visual Studio Code editor interface. The title bar displays "Program.cs*" and "ChuongTrinhDauTien". The code editor window contains the following C# code:

```
4  {
5      0 references
6      class Program
7      {
8          0 references
9          static void Main(string[] args)
10         {
11             //Viết nội dung lên màn hình
12             Console.WriteLine("Day la chuong trinh C# da");
13             Console.Wri
14         }
15     }
```

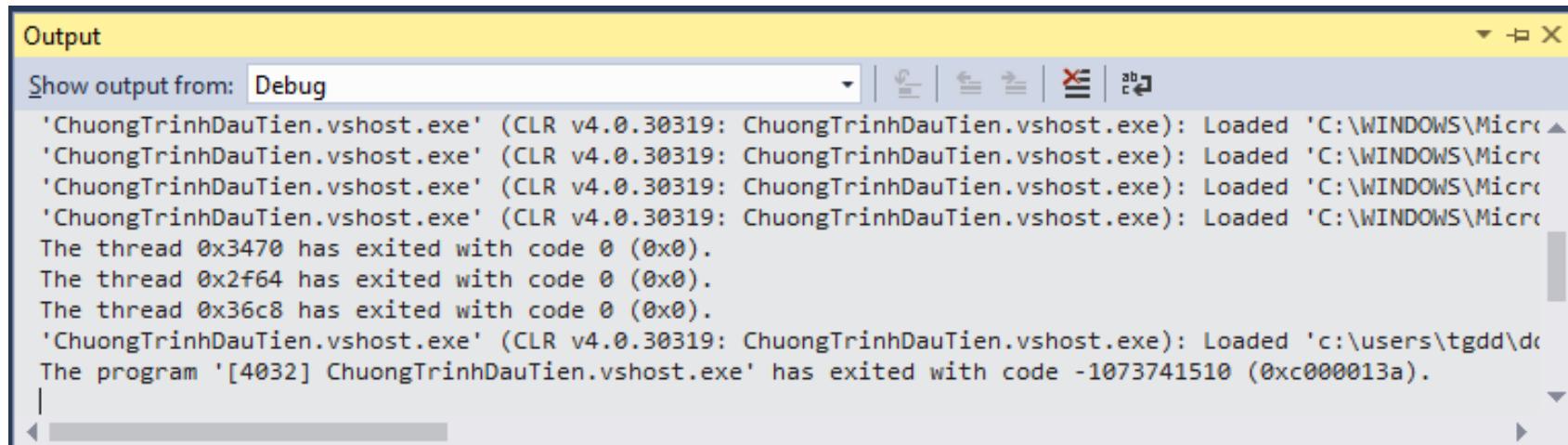
A code completion dropdown menu is open at line 10, showing two options: "Write" and "WriteLine". The "Write" option is highlighted. A yellow callout box points to this menu with the text: "Trình biên tập: cho phép soạn thảo mã nguồn C#, trình VS hỗ trợ nhiều chức năng thông minh trong viết code".

Trình biên tập: cho phép soạn thảo mã nguồn C#, trình VS hỗ trợ nhiều chức năng thông minh trong viết code

MÔI TRƯỜNG LÀM VIỆC CỦA VISUAL STUDIO

□ Output Window

- Cửa sổ hiển thị trạng thái làm việc của VS



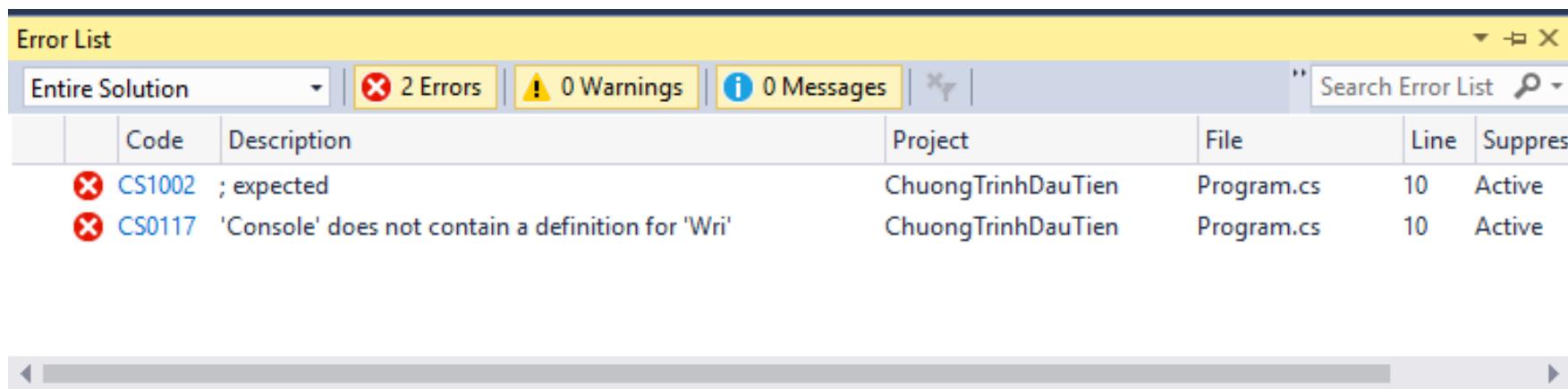
The screenshot shows the 'Output' window in Visual Studio. The title bar says 'Output'. The dropdown menu says 'Show output from: Debug'. The window contains the following text:

```
'ChuongTrinhDauTien.vshost.exe' (CLR v4.0.30319: ChuongTrinhDauTien.vshost.exe): Loaded 'C:\WINDOWS\Micro...  
The thread 0x3470 has exited with code 0 (0x0).  
The thread 0x2f64 has exited with code 0 (0x0).  
The thread 0x36c8 has exited with code 0 (0x0).  
'ChuongTrinhDauTien.vshost.exe' (CLR v4.0.30319: ChuongTrinhDauTien.vshost.exe): Loaded 'c:\users\tgdd\de...  
The program '[4032] ChuongTrinhDauTien.vshost.exe' has exited with code -1073741510 (0xc000013a).
```

MÔI TRƯỜNG LÀM VIỆC CỦA VISUAL STUDIO

□ Error List

- Cửa sổ hiển thị các thông báo lỗi, cảnh báo hoặc các thông báo kết quả xử lý trong quá trình biên dịch chương trình



The screenshot shows the 'Error List' window in Visual Studio. The title bar says 'Error List'. Below it, there's a toolbar with buttons for 'Entire Solution' (selected), '2 Errors', '0 Warnings', '0 Messages', and a search bar labeled 'Search Error List'. The main area is a table with columns: Code, Description, Project, File, Line, and Suppres. There are two rows of errors:

| | Code | Description | Project | File | Line | Suppres |
|---|--------|---|--------------------|------------|------|---------|
| × | CS1002 | ; expected | ChuongTrinhDauTien | Program.cs | 10 | Active |
| × | CS0117 | 'Console' does not contain a definition for 'Wri' | ChuongTrinhDauTien | Program.cs | 10 | Active |

MÔI TRƯỜNG LÀM VIỆC CỦA VISUAL STUDIO

□ Tổ chức thư mục của dự án

- Trên ổ cứng máy tính, tại nơi lưu chương trình:



bin



obj



Properties



App



ChuongTrinhDauTien



Program.cs

bin: thư mục lưu tập tin được biên dịch

obj: lưu các thư viện khai báo thêm trong dự án

Properties: lưu tập tin cấu hình của dự án

MÔI TRƯỜNG LÀM VIỆC CỦA VISUAL STUDIO

☐ Một số thao tác chính

- **Mở dự án đã có:** File → Open → Project Solution
- **Đóng dự án đang mở:** File → Close Solution
- **Tạo mới một tập tin .cs:** Trong cửa sổ Solution Explorer → Click phải lên tên dự án → Add → New Item → Class
- **Biên dịch chương trình:** Ctrl + Shift + B
- **Biên dịch và thực thi chương trình:** F5

BÀI TẬP KẾT THÚC CHƯƠNG

1. Cài đặt trên máy tính phần mềm Visual Studio C# 2010 Express

- Link download: <http://tinyurl.com/VSCSharp2010Express>

2. Tìm hiểu môi trường làm việc của Visual Studio

3. Tạo project, nhập code đã cho ở hai ví dụ tại slide 12 và 24, thử chạy và kiểm tra kết quả.

BÀI GIẢNG

CƠ SỞ LẬP TRÌNH

CHƯƠNG 2.

BIẾN VÀ KIỂU DỮ LIỆU

NGUYỄN THÀNH THỦY

BỘ MÔN TIN HỌC QUẢN LÝ

TRƯỜNG ĐẠI HỌC KINH TẾ, ĐẠI HỌC ĐÀ NẴNG

THUYNT@DUE.EDU.VN

NỘI DUNG

- Kiểu dữ liệu (Data Types)**
- Hằng (Constants)**
- Biến (Variable)**
- Quy tắc đặt tên cho các định danh**
- Toán tử**
- Các hàm nhập xuất dữ liệu**

KIỂU DỮ LIỆU

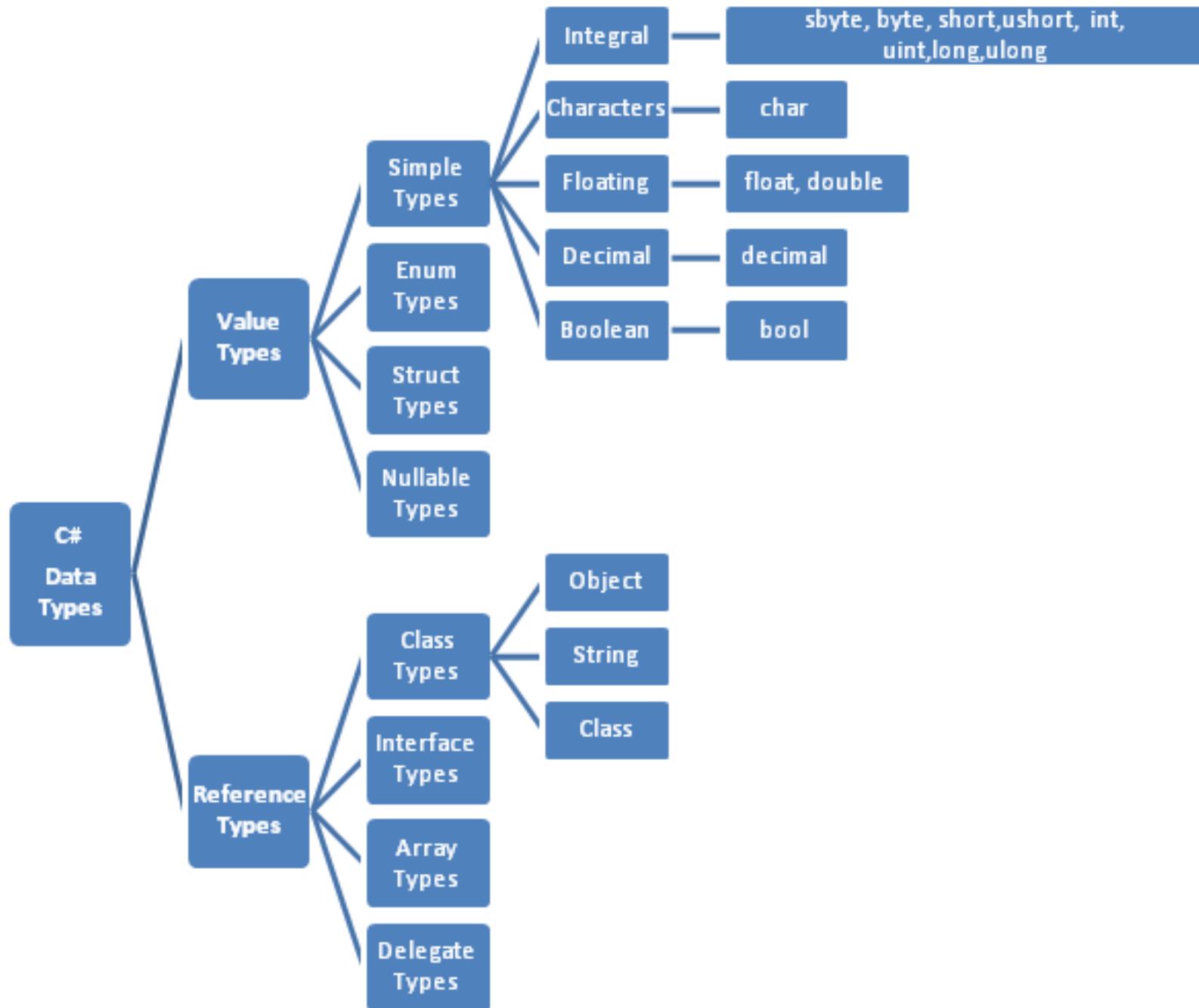
□ Bài toán:

- Nhập từ bàn phím: Họ tên, Ngày sinh và Điểm thi 3 môn Toán – Lý – Hóa của một học sinh. In lên màn hình: Họ tên, Ngày sinh và ĐTB của học sinh đó.
 - Các dữ liệu sẽ được nhập vào:
 - **Họ tên**: kiểu chuỗi
 - **Ngày sinh**: kiểu ngày tháng năm
 - **Điểm thi**: kiểu số thực
 - Các biến sẽ được sử dụng trong chương trình:
 - **HoTen**: kiểu chuỗi
 - **NgaySinh**: kiểu ngày tháng năm
 - **DiemToan**: kiểu số thực
 - **DiemLy**: kiểu số thực
 - **DiemHoa**: kiểu số thực
 - **DTB**: kiểu số thực

KIỂU DỮ LIỆU

- **Khái niệm về kiểu dữ liệu:** là kiểu của dữ liệu được đưa từ bên ngoài vào máy tính, hoặc kiểu (*phương pháp*) tổ chức các tập dữ liệu trên máy tính;

- **C# có hai kiểu dữ liệu:**
 - **Dữ liệu kiểu giá trị** (Value Types)
 - **Dữ liệu kiểu tham chiếu** (Reference Types)



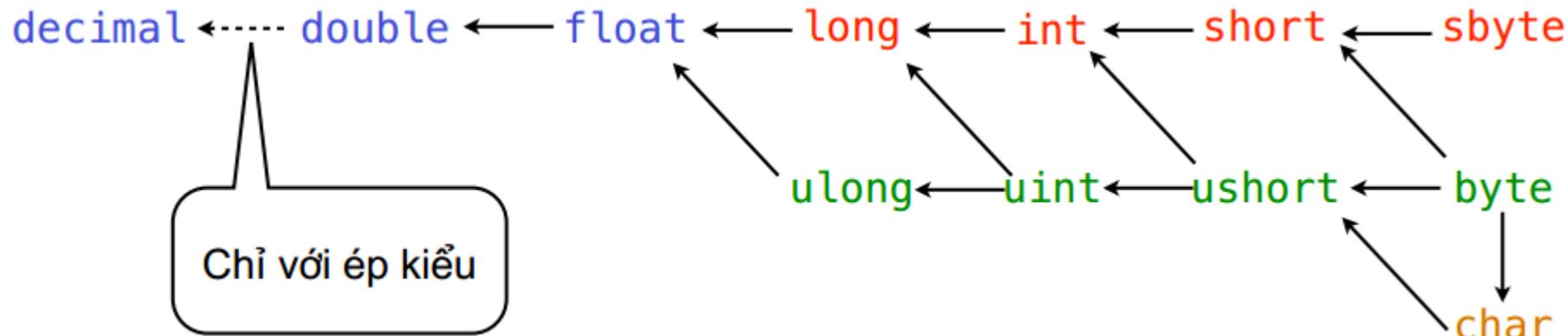
KIỂU DỮ LIỆU

- ❑ Các kiểu dữ liệu cơ bản: đã được xây dựng sẵn trong ngôn ngữ lập trình C#

| Type | Size in Bytes | Range | Meaning |
|---------|---------------|--|---|
| bool | 1/2 | True or False | Represents true/false values |
| byte | 1 | 0 to 255 | 8-bit unsigned integer |
| char | 2 | Any Unicode Character | Character |
| decimal | 12 | (-7.9 x 10 ²⁸ to 7.9 x 10 ²⁸) / 100 to 28 | Numeric type for financial calculations |
| double | 8 | (+/-)5.0 x 10 ⁻³²⁴ to (+/-)1.7 x 10 ³⁰⁸ | Double-precision floating point |
| float | 4 | -3.4 x 10 ³⁸ to + 3.4 x 10 ³⁸ | Single-precision floating point |
| int | 4 | -2,147,483,648 to 2,147,483,647 | Integer |
| long | 8 | 923,372,036,854,775,808 to 9,223,372,036,854,775,807 | Long integer |
| sbyte | 1 | -128 to 127 | 8-bit signed integer |
| short | 2 | -32,768 to 32,767 | Short integer |
| uint | 4 | 0 to 4,294,967,295 | Unsigned integer |

KIỂU DỮ LIỆU

□ SỰ TƯƠNG THÍCH GIỮA CÁC KIỂU



- Dùng khi gán dữ liệu giữa các biến có kiểu khác nhau;
- Chuyển đổi ngầm định theo chiều mũi tên
 - `int DiemThi = 5; //→ DiemThi = 5`
 - `float Mon1 = DiemThi; //→ Mon1 = 5.0`

KIỂU DỮ LIỆU

☐ Sự tương thích giữa các kiểu

- Chuyển đổi theo chiều ngược lại có thể xảy ra mất dữ liệu hoặc có kết quả không mong muốn

```
int i = 3.1416;
```

//→ lỗi vì về trái có kiểu thấp hơn về phải

// Được viết lại như sau:

```
int i = (int)3.1416;
```

// tuy nhiên sẽ bị mất dữ liệu

// → i có giá trị là: 3

HẰNG, KHAI BÁO VÀ SỬ DỤNG (CONSTANT)

□ Khái niệm

- Hằng là một đại lượng mang giá trị không thay đổi trong quá trình xử lý của chương trình

□ Một số trường hợp sử dụng Hằng

- Giá trị của số PI để tính diện tích hình tròn ($\text{PI}=3.14159265358979$)
- Tính tiền BHXH, biết rằng: $\text{BHXH}=30\% \text{ Tiền lương}$; Trong đó 30% chính là đại lượng Hằng
- Số SV tối đa để xét học bổng ở mỗi Khoa là 10 SV;

HẰNG, KHAI BÁO VÀ SỬ DỤNG (CONSTANT)

□ Cú pháp định nghĩa Hằng

```
const <Tên_Kiểu> <Tên_Hằng> = <Giá_trị>;
```

Tùy khóa khai báo

Giá trị của Hằng cần định nghĩa

■ Ví dụ:

```
const double PI = 3.14159265358979;
```

```
const float TyLeBHXB = 0.3;
```

```
const int Max = 10;
```

BIẾN, KHAI BÁO VÀ SỬ DỤNG (VARIABLE)

□ Vị trí khai báo HẰNG

```
3  namespace ViDui
4  {
5      class Program
6      {
7          //Vị trí khai báo HẰNG --> TOÀN CỤC
8          //...
9          static void Main(string[] args)
10         {
11             //Vị trí khai báo HẰNG --> CỤC BỘ
12             //...
13         }
14     }
15 }
```

HẰNG, KHAI BÁO VÀ SỬ DỤNG (CONSTANT)

□ Lưu ý khi sử dụng Hằng

- Hằng cần được khai báo và khởi gán giá trị trước khi sử dụng;
- Khai báo Hằng trong phạm vi một lớp và bên ngoài một Hàm;
- Khởi gán cho Hằng là một giá trị được xác định hoặc một biểu thức, không gán cho Hằng giá trị từ một Biến;

//Gán giá trị cho Hằng

```
const int MAX = 1000;  
const char newline = '\n';  
const float Pi = 3.1416;
```

//Gán biểu thức cho Hằng

```
const float SquarePi = Pi * Pi;
```

BIẾN, KHAI BÁO VÀ SỬ DỤNG (VARIABLE)

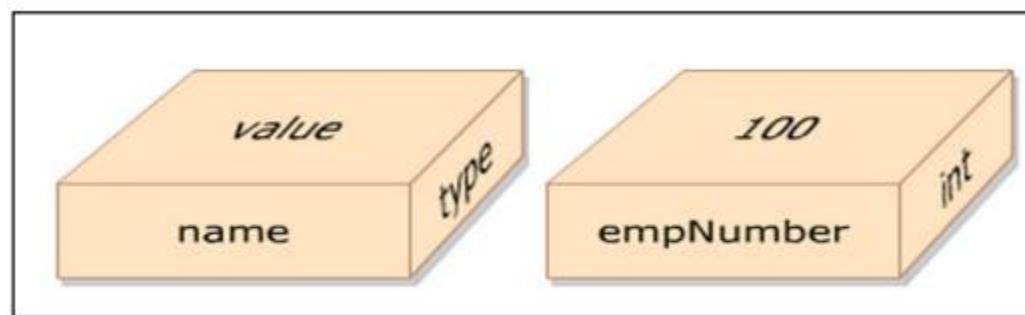
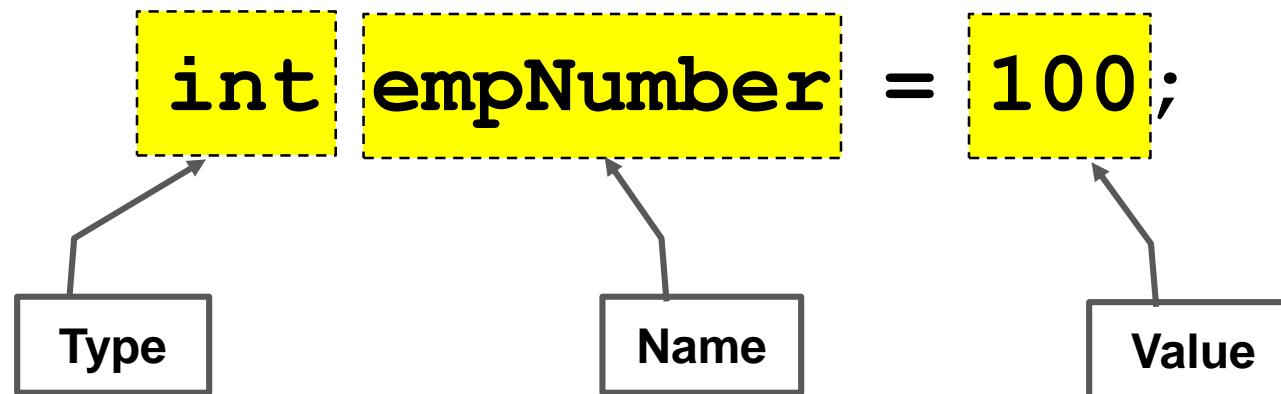
□ Khái niệm về biến:

- Biến là một đại lượng mang giá trị, và có thể thay đổi trong quá trình xử lý của chương trình;
- Một biến được sử dụng để lưu trữ dữ liệu trong một chương trình, và được khai báo với một kiểu dữ liệu nhất định;
- **Ví dụ:**
 - **Tuoi** (tuổi) của một sinh viên
 - **DiaChi** (địa chỉ) của một khách hàng
 - **TienLuong** (tiền lương) của một nhân viên, ...

BIẾN, KHAI BÁO VÀ SỬ DỤNG (VARIABLE)

□ Khái niệm về biến:

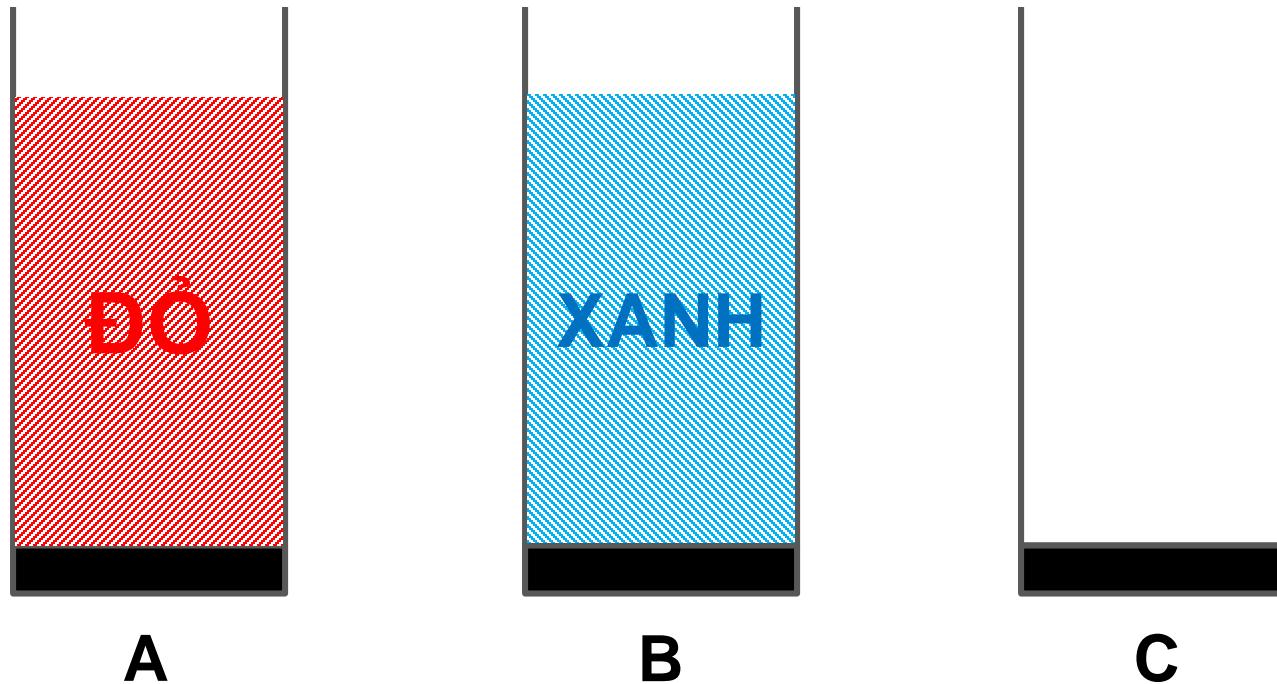
- Một biến sẽ có một cái tên và có thể chứa một giá trị;



BIẾN, KHAI BÁO VÀ SỬ DỤNG (VARIABLE)

□ Khái niệm về biến:

- **Ví dụ:** sử dụng biến trong bài toán hoán vị.
 - Trình bày phương pháp hoán vị nước giữa hai ly A và B, sử dụng ly trung gian C



BIẾN, KHAI BÁO VÀ SỬ DỤNG (VARIABLE)

□ Khái niệm về biến:

- Ví dụ: sử dụng biến trong bài toán hoán vị 3 số nguyên.

```
int a,b,c; //Khai báo biến  
a = 5;      //Gán a bằng 5  
b = 10;     //Gán b bằng 10  
c = a;      //Gán c bằng a  
a = b;      //Gán a bằng b  
b = c;      //Gán b bằng c
```

BIẾN, KHAI BÁO VÀ SỬ DỤNG (VARIABLE)

□ Khai báo biến:

- Một biến được sử dụng trong chương trình C# cần phải được khai báo trước;
- Cú pháp khai báo một biến:

```
<Tên_Kiểu> <Tên_Biến> ;
```

- Ví dụ:

```
float HeSoLuong; //Biến kiểu số thực  
int Tuoi_NV1; //Biến kiểu số nguyên  
string HoTen_NV1; //Biến kiểu chuỗi  
boolean GioiTinh; //Biến kiểu logic
```

BIẾN, KHAI BÁO VÀ SỬ DỤNG (VARIABLE)

□ Khai báo biến:

- Vừa khai báo vừa khởi tạo giá trị cho biến:

```
<Tên_Kiểu> <Tên_Biến> = <Giá_Trị> ;
```

- Ví dụ:

```
float HeSoLuong = 1.92;
```

```
int Tuoi_NV1 = 23;
```

```
string MônHoc = "CSLT";
```

```
char XepLoai = 'A'; //Biến kiểu ký tự
```

```
Boolean GioiTinh = True;
```

BIẾN, KHAI BÁO VÀ SỬ DỤNG (VARIABLE)

□ Khai báo biến:

- Khai báo đồng thời nhiều biến có cùng kiểu:

```
<Tên_Kiểu> <Tên_Biến1>, <Tên_Biến2>, ...;
```

- Ví dụ:

```
float HSL1, HSL2, HSL3;
```

```
int TuoiNV1, TuoiNV2, TuoiNV3;
```

BIẾN, KHAI BÁO VÀ SỬ DỤNG (VARIABLE)

□ Sử dụng biến:

- Cú pháp gán giá trị cho biến:

```
<Tên_Biến> = <Giá_Tri> ;
```

- Ví dụ:

```
float HeSoLuong;  
HeSoLuong = 1.92;
```

```
int Tuoi_NV1;  
Tuoi_NV1 = 23;
```

```
string HoTen_NV1;  
HoTen_NV1 = "Nguyễn Thành Thủy";
```

BIẾN, KHAI BÁO VÀ SỬ DỤNG (VARIABLE)

☐ Một số lưu ý khi sử dụng biến

- Biến phải được khai báo và khởi tạo giá trị trước khi sử dụng;
- Lựa chọn kiểu phù hợp với phạm vi của giá trị mà biến đó sẽ lưu trữ;
 - Điểm môn học: nếu điểm không có số lẻ thì kiểu int, còn lại thì phải dùng kiểu float
- Không thể khai báo nhiều biến trùng tên trong cùng một phạm vi (chương trình chính hoặc hàm)
- Nên đặt tên biến ngắn gọn, có ý nghĩa, viết hoa chữ cái đầu của từ

**SonguyenTo
DiemMonToan**

BIẾN, KHAI BÁO VÀ SỬ DỤNG (VARIABLE)

□ Các loại biến

■ Biến cục bộ

- Là biến được khai báo trong một HÀM và chỉ ảnh hưởng trong phạm vi của HÀM đó;
- Khi ra khỏi HÀM, biến sẽ không còn giá trị sử dụng;

■ Biến toàn cục

- Biến toàn cục được khai báo ngay sau dòng lệnh khai báo lớp class;
- Biến toàn cục có giá trị sử dụng trong toàn bộ chương trình;

CÁC LOẠI BIẾN VÀ PHẠM VI SỬ DỤNG

Phạm vi TOÀN CỤC

```
class Program
{
    const float PI = 3.1416F;
    static float ChuVi;
    0 references
    static void Main(string[] args)
    {
        int BanKinh=5;
        ChuVi = TinhChuVi(BanKinh);
        Console.WriteLine(ChuVi);
        Console.Read();
    }
    1 reference
    static float TinhChuVi(int bk)
    {
        float ChuVi;
        ChuVi = bk * PI;
        return ChuVi;
    }
}
```

Vị trí khai báo **BIẾN TOÀN CỤC** và **HẰNG**

Vị trí khai báo **BIẾN CỤC BỘ** trong hàm **Main()**

Vị trí khai báo **BIẾN CỤC BỘ** trong chương trình con

ĐỊNH DANH

☐ Quy tắc đặt tên các định danh (identifier)

- Tên của một định danh (*biến, hằng, phương thức, lớp,...*) là chuỗi gồm các ký tự: **chữ cái** (*chữ hoa hoặc chữ thường*), **chữ số**, **dấu gạch chân** (_) và **dấu @** (*chỉ được đặt trước tên trùng với từ khóa*)
- Tên chỉ được bắt đầu bằng **chữ cái** hoặc **dấu gạch chân**;
- Tên trong C# có phân biệt **chữ HOA** và **chữ thường**;
- Không được sử dụng **từ khóa** của C# để đặt tên
 - Trường hợp bắt buộc thì phải thêm dấu @ trước từ khóa để trở thành một tên hợp thức;
- Thường dùng những tên có ý nghĩa và dễ phân biệt;

ĐỊNH DANH

☐ Quy tắc đặt tên các định danh (identifier)

- Ví dụ: một số tên hợp lệ

- SoLuong
- soluong1
- So_Luong2
- _SL
- SDT1_NV1
- SDT2_NV1
- @int

→ int là một từ khóa trong C#

ĐỊNH DANH

☐ Quy tắc đặt tên cho các định danh

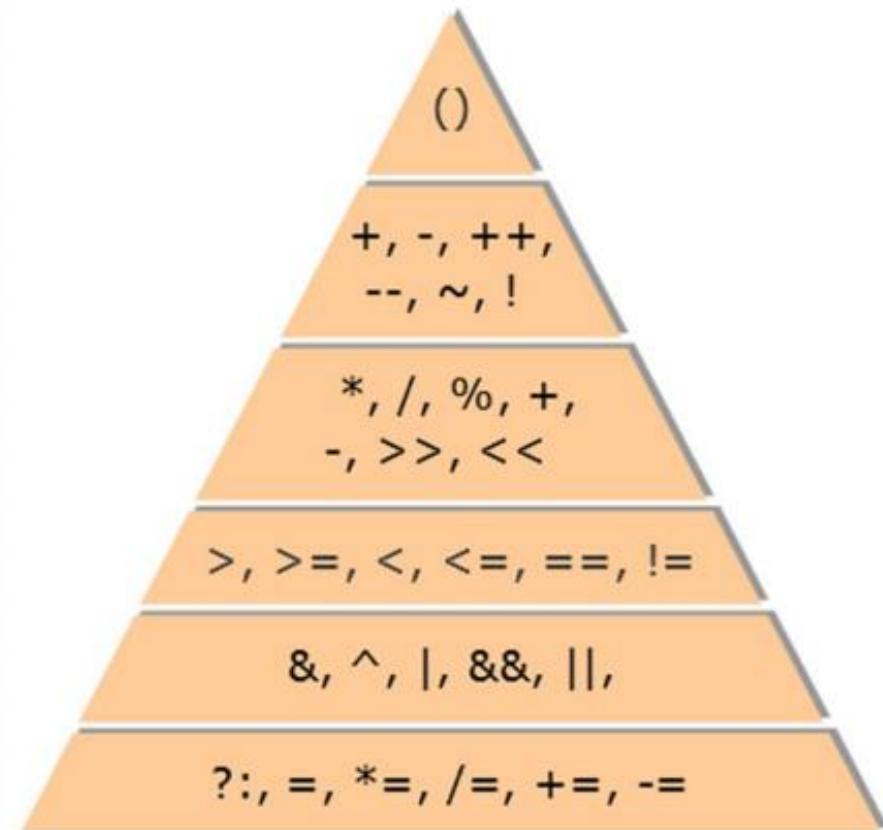
- **Ví dụ:** xác định tên đúng/ sai trong các trường hợp

| TT | Tên biến | Đúng/Sai ? |
|----|---------------|------------|
| 1 | Employee | |
| 2 | Student | |
| 3 | _Name | |
| 4 | Emp_Name | |
| 5 | @goto | |
| 6 | Static | |
| 7 | 4myclass | |
| 8 | Student&Class | |

TOÁN TỬ

□ Có 6 loại toán tử được sử dụng trong C#

- Toán tử toán học
- Toán tử tăng và giải
- Toán tử quan hệ
- Toán tử logic
- Toán tử điều kiện
- Toán tử gán



TOÁN TỬ

☐ Toán tử toán học

- Gán $a=10$; $b=5$;

| Toán tử | Mô tả | Biểu thức | Kết quả |
|---------|------------------|-----------|---------|
| + | Phép cộng | $a + b$ | 15 |
| - | Phép trừ | $a - b$ | 5 |
| * | Phép nhân | $a * b$ | 50 |
| / | Phép chia hết | a / b | 2 |
| % | Phép chia lấy dư | $a \% b$ | 0 |

TOÁN TỬ

☐ Toán tử quan hệ

- Gán $a=10$; $b=5$;

| Toán tử | Mô tả | Biểu thức | Kết quả |
|---------|--------------------------------|------------|---------|
| $==$ | Phép so sánh bằng | $a == b$ | False |
| $!=$ | Phép so sánh khác | $a != b$ | True |
| $>$ | Phép so sánh lớn hơn | $a > b$ | True |
| $<$ | Phép so sánh bé hơn | $a < b$ | False |
| \geq | Phép so sánh lớn hơn hoặc bằng | $a \geq b$ | True |
| \leq | Phép so sánh bé hơn hoặc bằng | $a \leq b$ | False |

TOÁN TỬ

□ Toán tử logic

- Gán $a=10$; $b=5$;

| Toán tử | Mô tả | Biểu thức | Kết quả |
|---------|---|---------------------|---------|
| & | Phép và (AND) | $(a==10) \& (b>10)$ | False |
| | Phép hoặc (OR) | $(a==10) (b>10)$ | True |
| | Phép hoặc (OR) | $(a==10) (b>10)$ | True |
| ^ | Phép duy nhất một, trả về TRUE nếu tồn tại đúng một biểu thức có giá trị TRUE | $(a==10) ^ (b>10)$ | False |

TOÁN TỬ

□ Toán tử tăng và giảm

- Gán $a=10$;

| Toán tử | Mô tả | Biểu thức | Kết quả |
|---------|-----------------|-----------|------------------|
| $++a$ | Phép tăng trước | $b = ++a$ | $b = 11; a = 11$ |
| $a++$ | Phép tăng sau | $b = a++$ | $b = 10; a = 11$ |
| $--a$ | Phép giảm trước | $b = --a$ | $b = 9; a = 9$ |
| $a--$ | Phép giảm sau | $b = a--$ | $b = 10; a = 9$ |

TOÁN TỬ

□ Toán tử gán

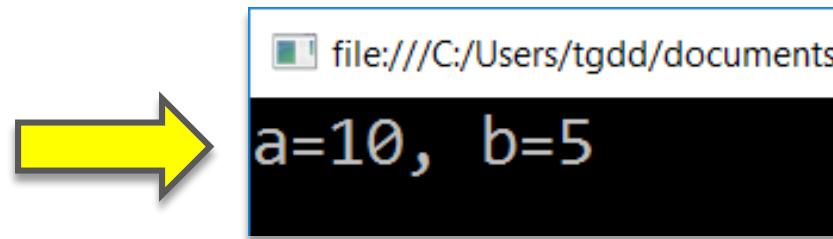
- Gán $a=10;$

| Biểu thức | Biểu thức tương đương | Kết quả |
|-----------|-----------------------|----------|
| $a += 5;$ | $a = a + 5;$ | $a = 15$ |
| $a -= 5;$ | $a = a - 5;$ | $a = 5$ |
| $a *= 5;$ | $a = a * 5;$ | $a = 25$ |
| $a /= 5;$ | $a = a / 5;$ | $a = 2$ |
| $a %= 5;$ | $a = a \% 5;$ | $a = 0$ |

TOÁN TỬ

- Phép cộng (+) dùng để nối chuỗi và biểu thức

```
static void Main(string[] args)
{
    int a = 10, b=5;
    Console.WriteLine("a=" + a + ", b=" + b);
    Console.ReadKey();
}
```



TOÁN TỬ

□ Toán tử điều kiện với dấu chấm hỏi (?)

- Cú pháp:

<Biểu thức Logic>?<Biểu thức 1>:<Biểu thức 2>;

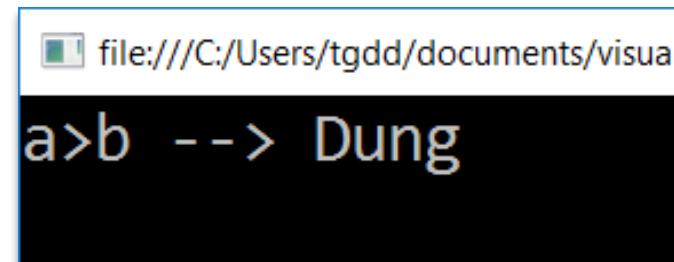
- Ý nghĩa:
 - Nếu <Biểu thức Logic> có giá trị ĐÚNG thì trả về <Biểu thức 1>
 - Còn lại trả về <Biểu thức 2>

TOÁN TỬ

□ Toán tử điều kiện với dấu chấm hỏi (?)

- Ví dụ:

```
static void Main(string[] args)
{
    int a = 10, b=5;
    Console.WriteLine((a>b)?"a>b --> Dũng":"a>b --> Sai");
    Console.ReadKey();
}
```



The screenshot shows a Windows Command Prompt window with a black background and white text. At the top, it displays the file path: "file:///C:/Users/tgdd/documents/visual". Below that, the text "a>b --> Dũng" is displayed in white. A yellow arrow points from the word "Dũng" in the output to the question mark in the code above.

CÁC HÀM NHẬP XUẤT DỮ LIỆU

- Hàm in dữ liệu lên màn hình
 - Cú pháp

```
Console.WriteLine("Chuỗi ký tự");
Console.WriteLine("Chuỗi ký tự" + Biến);
```

- In nội dung rồi xuống dòng:

```
Console.WriteLine("Chuỗi ký tự");
Console.WriteLine("Chuỗi ký tự" + Biến);
```

CÁC HÀM NHẬP XUẤT DỮ LIỆU

□ Hàm in dữ liệu lên màn hình

- Ví dụ: phân biệt giữa hàm **Write()** và **WriteLine()**

```
static void Main(string[] args)
{
    Console.WriteLine("C# la mot ngon ngu lap trinh pho bien");
    Console.WriteLine("C# la mot ngon ngu lap trinh");
    Console.WriteLine("Ngon ngu lap trinh");
    Console.Write("C# la mot ngon ngu lap trinh");
    Console.WriteLine(" Ngon ngu lap trinh");
    Console.ReadKey();
}
```



```
file:///C:/Users/tgdd/documents/visual studio 2010/Projects/ChuongTrinhDauTien/ChuongTrinhDauTien/bin/Debug/Ch
C# la mot ngon ngu lap trinh pho bien
C# la mot ngon ngu lap trinh
Ngon ngu lap trinh
C# la mot ngon ngu lap trinh Ngon ngu lap trinh
```

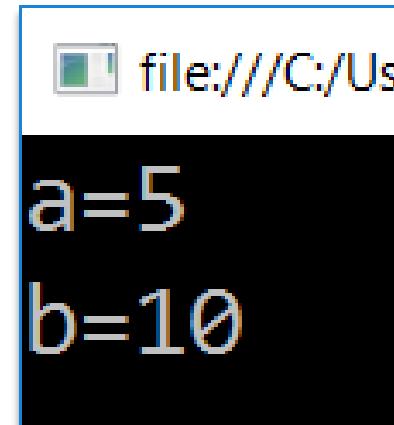
A screenshot of a terminal window showing the execution results of a C# program. The window title is partially visible as 'file:///C:/Users/tgdd/documents/visual studio 2010/Projects/ChuongTrinhDauTien/ChuongTrinhDauTien/bin/Debug/Ch'. The text output is:
C# la mot ngon ngu lap trinh pho bien
C# la mot ngon ngu lap trinh
Ngon ngu lap trinh
C# la mot ngon ngu lap trinh Ngon ngu lap trinh

CÁC HÀM NHẬP XUẤT DỮ LIỆU

□ Hàm in dữ liệu lên màn hình

- Ví dụ: sử dụng mã \n để xuống dòng

```
static void Main(string[] args)
{
    int a = 5, b = 10;
    Console.WriteLine("a=" + a);
    Console.WriteLine("\nb=" + b);
    Console.ReadKey();
}
```



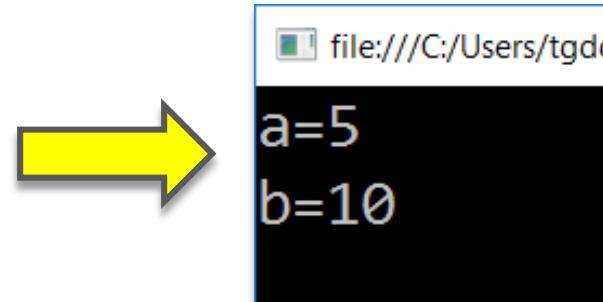
```
a=5
b=10
```

CÁC HÀM NHẬP XUẤT DỮ LIỆU

□ Hàm in dữ liệu lên màn hình

- Ví dụ: in giá trị của biến lên màn hình

```
static void Main(string[] args)
{
    int a = 5, b = 10;
    Console.WriteLine("a=" + a);
    Console.Write("b=" + b);
    Console.ReadKey();
}
```

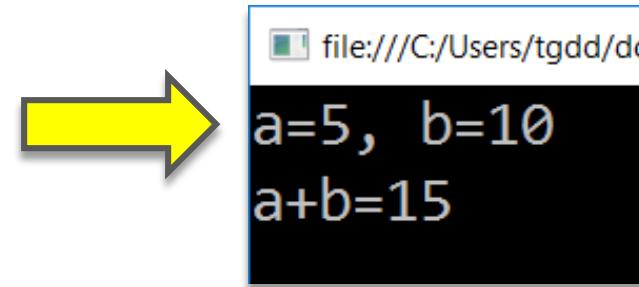


CÁC HÀM NHẬP XUẤT DỮ LIỆU

□ Hàm in dữ liệu lên màn hình

- Ví dụ: in giá trị của **biểu thức** lên màn hình

```
static void Main(string[] args)
{
    int a = 5, b = 10;
    Console.WriteLine("a=" + a + ", b=" + b);
    Console.Write("a+b=" + (a+b));
    Console.ReadKey();
}
```

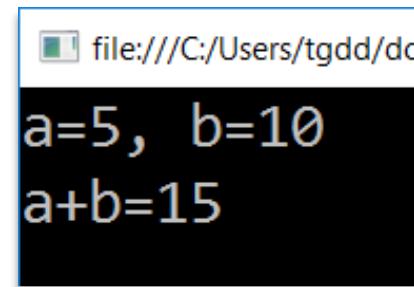


CÁC HÀM NHẬP XUẤT DỮ LIỆU

□ Hàm in dữ liệu lên màn hình

- Ví dụ: sử dụng tham số để in dữ liệu

```
static void Main(string[] args)
{
    int a = 5, b = 10;
    Console.WriteLine("a={0}, b={1}", a, b);
    Console.Write("a+b={0}", (a+b));
    Console.ReadKey();
}
```



```
file:///C:/Users/tgdd/dc
a=5, b=10
a+b=15
```

CÁC HÀM NHẬP XUẤT DỮ LIỆU

□ Hàm nhập dữ liệu từ bàn phím

- Nhập một ký tự: (ít sử dụng)

```
<Biến> = Console.ReadLine();
```

- Nhập một ký tự:

```
Console.ReadKey();
```

- Nhập một ký tự hoặc chuỗi ký tự:

```
<Biến> = Console.ReadLine();
```

- Nhập một số:

```
<Biến>=<Ép kiểu>Console.ReadLine();
```

CÁC HÀM NHẬP XUẤT DỮ LIỆU

□ Hàm nhập dữ liệu từ bàn phím

■ Ví dụ:

```
static void Main(string[] args)
{
    int Tuoi; string HoTen;
    Console.WriteLine("Ho va ten: ");
    HoTen = Console.ReadLine(); //Nhập mot XAU KY TU

    Console.WriteLine("Tuoi: ");
    Tuoi = Convert.ToInt32(Console.ReadLine()); //Nhập mot SO

    Console.WriteLine("\nXin chao ban: {0}, Chuc mung sinh nhat lan thu: {1}", HoTen, Tuoi);
    Console.ReadKey();
}
```

Ép sang kiểu số nguyên
(được sử dụng trong trường hợp nhập SỐ)



```
file:///C:/Users/tgdd/documents/visual studio 2010/Projects/ChuongTrinhDauTien/ChuongTrinhDauTien/bin/Debug/ChuongTrinh
Ho va ten: Bao Nam
Tuoi: 10

Xin chao ban: Bao Nam, Chuc mung sinh nhat lan thu: 10
```

CÁC HÀM NHẬP XUẤT DỮ LIỆU

- **Ví dụ:** Một số hàm ép kiểu thông dụng

```
float x;  
int y;  
char ch;  
double z;  
  
x = float.Parse(Console.ReadLine());  
ch = char.Parse(Console.ReadLine());  
  
y = Convert.ToInt32(Console.ReadLine());  
//Hoặc  
y = int.Parse(Console.ReadLine());  
  
z = Convert.ToDouble(Console.ReadLine());  
//Hoặc  
z = double.Parse(Console.ReadLine());
```

CÁC HÀM NHẬP XUẤT DỮ LIỆU

□ Hàm nhập dữ liệu từ bàn phím

```
string HoTen; char GioiTinh; double HSL; int Tuoi;  
Console.WriteLine("Ho va ten: ");  
HoTen = Console.ReadLine(); //Nhập một xâu KÝ TỰ --> Không ép kiểu  
  
Console.WriteLine("Gioi tinh (M/F): ");  
GioiTinh = Convert.ToChar(Console.ReadLine()); //Nhập một KÝ TỰ --> Ép kiểu  
  
Console.WriteLine("Tuoi: ");  
Tuoi = Convert.ToInt32(Console.ReadLine()); //Nhập một SỐ NGUYÊN --> Ép kiểu  
  
Console.WriteLine("He so luong: ");  
HSL = Convert.ToDouble(Console.ReadLine()); //Nhập một SỐ THỰC --> Ép kiểu  
  
Console.WriteLine("--> {0},{1},{2},{3}",HoTen,GioiTinh,Tuoi,HSL);
```



```
file:///C:/Users/tgdd/documents/visual studio 201  
Ho va ten: Bao Nam  
Gioi tinh (M/F): M  
Tuoi: 20  
He so luong: 2.34  
--> Bao Nam,M,20,2.34
```

BÀI TẬP ÔN TẬP

□ Câu 1. Viết chương trình thực hiện yêu cầu sau:

Nhập từ bàn phím:

- Họ tên của bạn

- Và 3 số nguyên bất kỳ

Tìm số lớn nhất trong 3 số trên, In lên màn hình
câu thông báo theo mẫu:

```
Ho ten: An
```

```
a=10
```

```
b=5
```

```
c=7
```

```
Chao ban An, So lon nhat la: 10
```

BÀI TẬP ÔN TẬP

- Câu 2. Viết chương trình tính và in lên màn hình **giá bán** của các mặt hàng với công thức sau:

$$\text{Giá bán} = \text{Giá niêm yết} - \text{Chiết khấu} + \text{VAT}$$

Trong đó:

- Nhập từ bàn phím **Giá niêm yết, Chiết khấu**
- **VAT= (Giá niêm yết – Chiết khấu)*0.01**

```
Nhap Gia niem yet: 100
```

```
Nhap Chiet khau: 10
```

```
Gia ban: 90.9
```

BÀI TẬP ÔN TẬP

- Câu 3. Viết chương trình để tính và in lên màn hình Tiền lanh cuối kỳ tiết kiệm tại ngân hàng, như sau:

- Tiền vốn đầu tư ban đầu là P đồng;
- Lãi suất tiền gửi mỗi tháng là r, được lãnh vào cuối kỳ;
- Sau n tháng, số tiền thu về cả vốn và lãi là:

$$\text{Tiền lanh cuối kỳ} = P * (1 + r * n)$$

Trong đó:

- P, r và n được nhập từ bàn phím

Tien dau tu ban dau: 1000

So thang gui: 10

Lai suat moi thang: 0.05

Tien lanh cuoi ky: 1500

BÀI TẬP ÔN TẬP

- Câu 4. Viết chương trình tính và in lên màn hình diện tích của một tam giác theo độ dài của các cạnh, theo công thức:

$$\text{Diện tích} = \sqrt{s(s - a)(s - b)(s - c)}$$

Trong đó:

- **a, b** và **c** là độ dài của 3 cạnh tam giác, được nhập từ bàn phím;
- **S=(a+b+c)/2**

Gợi ý: sử dụng hàm **Math.Sqrt(X)** để tính căn bậc hai của một số **X** bất kỳ;

BÀI GIẢNG

CƠ SỞ LẬP TRÌNH

CHƯƠNG 3.

CẤU TRÚC ĐIỀU KHIỂN

NGUYỄN THÀNH THỦY

BỘ MÔN TIN HỌC QUẢN LÝ

TRƯỜNG ĐẠI HỌC KINH TẾ, ĐẠI HỌC ĐÀ NẴNG

THUYNT@DUE.EDU.VN

NỘI DUNG

- Cấu trúc điều kiện if
- Cấu trúc rẽ nhánh switch ... case
- Cấu trúc lặp while
- Cấu trúc lặp do ... while
- Cấu trúc lặp for ... loop
- Câu lệnh nhảy break, continue, return

CẤU TRÚC ĐIỀU KIỆN IF – DẠNG 1

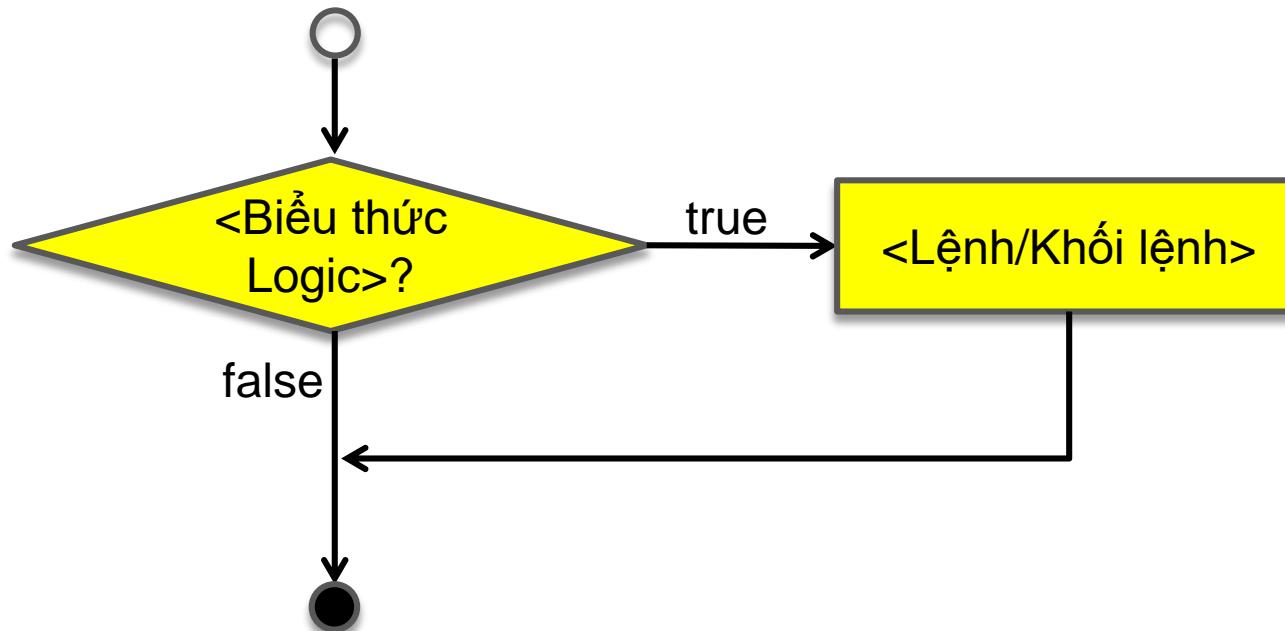
□ Cú pháp

if (<Biểu thức Logic>)

{

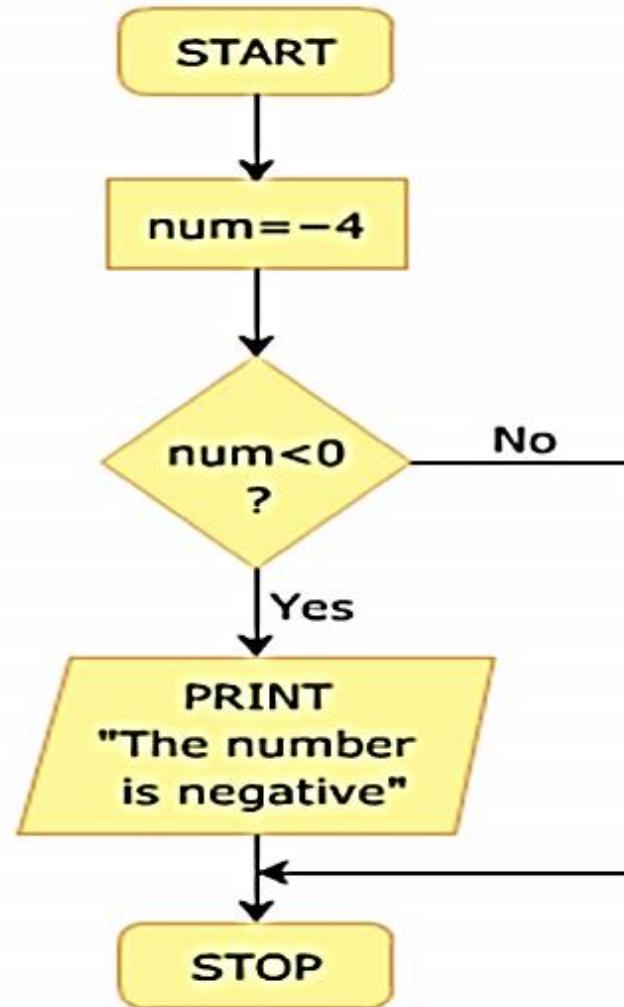
//Lệnh hoặc Khối lệnh

}



CẤU TRÚC ĐIỀU KIỆN IF – DẠNG 1

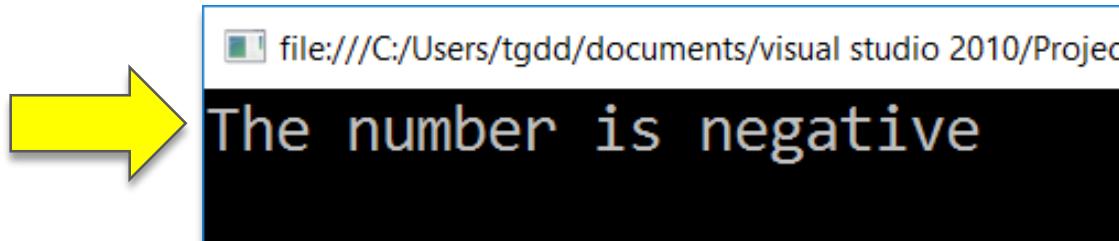
- Ví dụ: viết chương trình giải bài toán theo sơ đồ khối sau



CẤU TRÚC ĐIỀU KIỆN IF – DẠNG 1

- Ví dụ: viết chương trình giải bài toán theo sơ đồ khối sau

```
int num = -4;  
if (num < 0)  
{  
    Console.WriteLine("The number is negative");  
}
```



CẤU TRÚC ĐIỀU KIỆN IF – DẠNG 2

□ Cú pháp

```
if (<Biểu thức Logic>)
```

```
{
```

```
    //Lệnh hoặc Khối lệnh 1
```

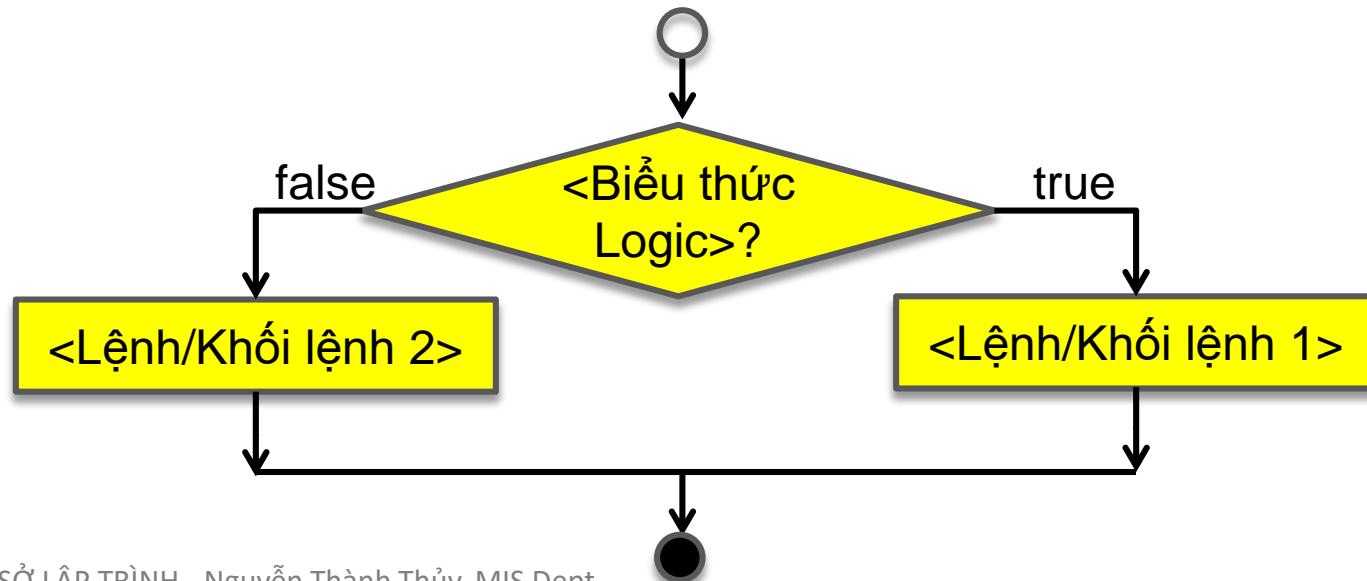
```
}
```

```
else
```

```
{
```

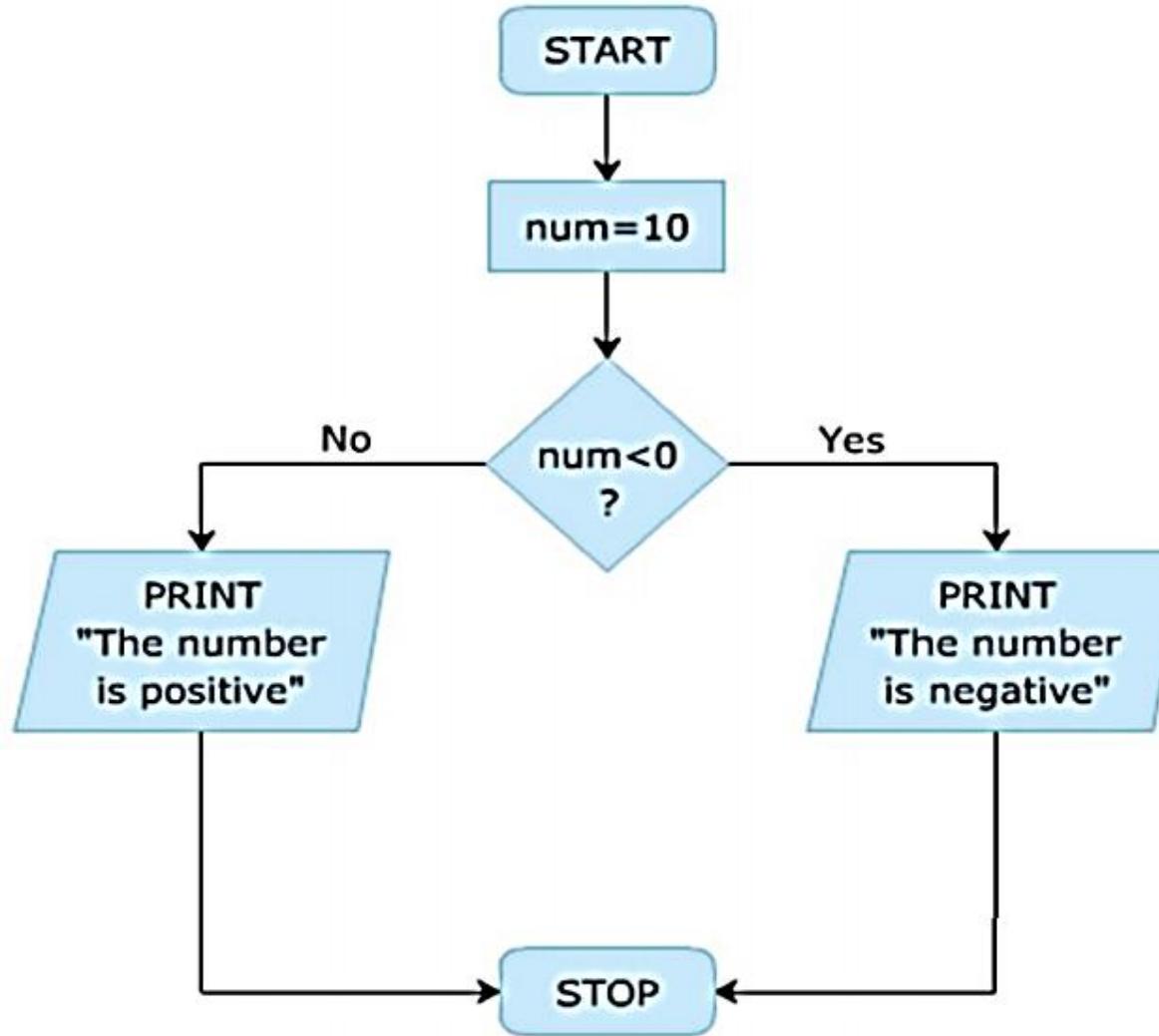
```
    //Lệnh hoặc Khối lệnh 2
```

```
}
```



CẤU TRÚC ĐIỀU KIỆN IF – DẠNG 2

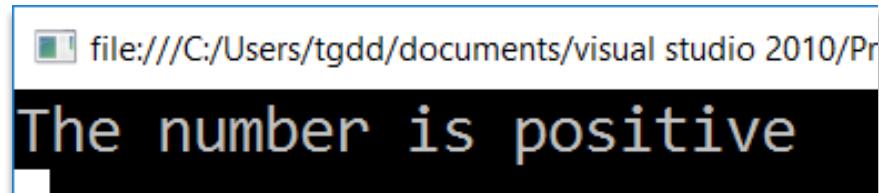
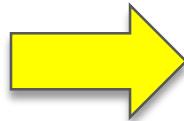
- Ví dụ: viết chương trình giải bài toán theo sơ đồ khối sau



CẤU TRÚC ĐIỀU KIỆN IF – DẠNG 2

- Ví dụ: viết chương trình giải bài toán theo sơ đồ khối sau

```
int num = 10;  
if (num < 0)  
{  
    Console.WriteLine("The number is negative");  
}  
else  
{  
    Console.WriteLine("The number is positive");  
}
```



The number is positive

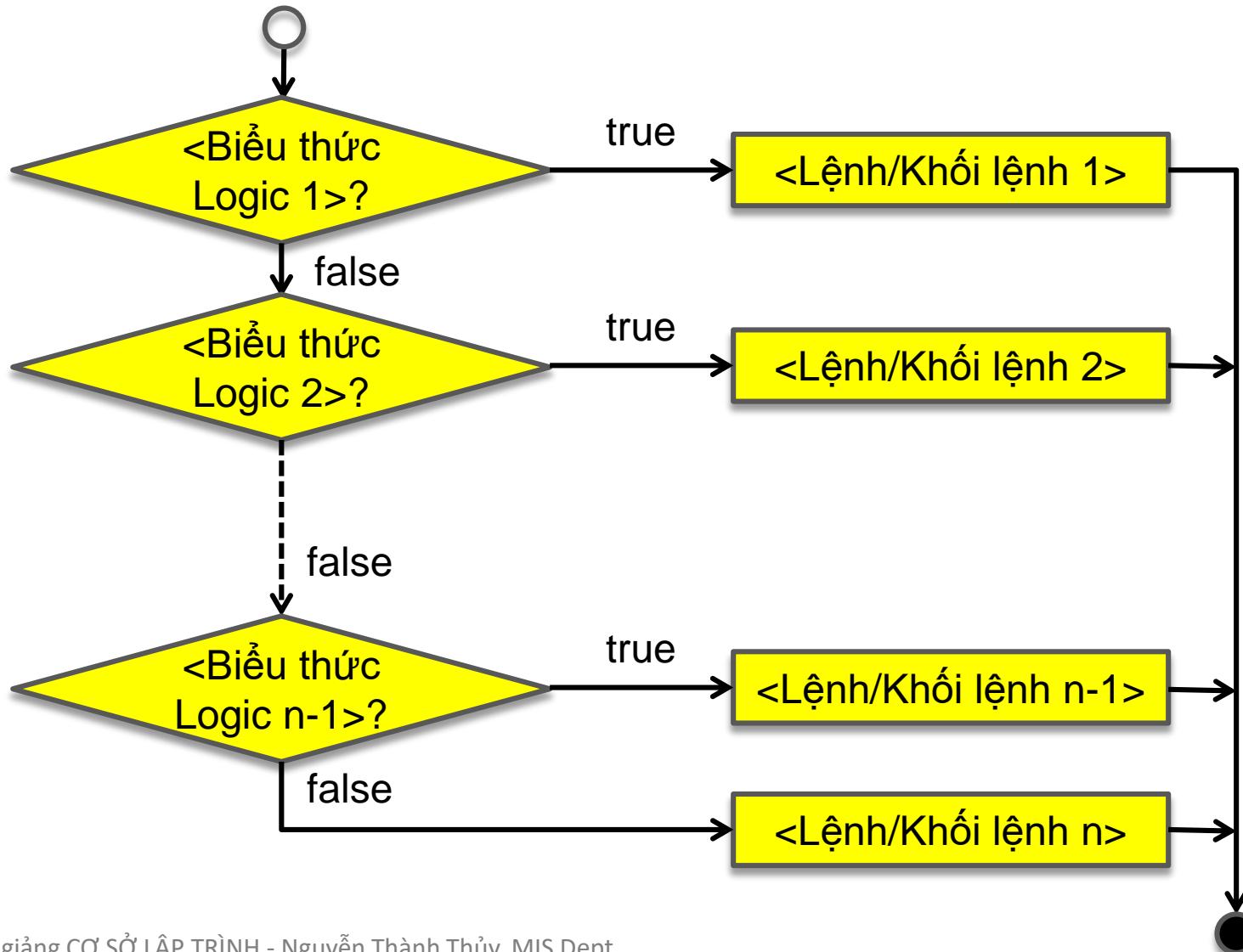
CẤU TRÚC ĐIỀU KIỆN IF – DẠNG 3

□ Cú pháp

```
if (<Biểu thức Logic 1>)
{
    //Lệnh/ khối lệnh 1
}
else if (<Biểu thức Logic 2>)
{
    //Lệnh/ khối lệnh 2
}
...
else if (<Biểu thức Logic n-1>)
{
    //Lệnh/ khối lệnh n-1
}
[ else
{
    //Lệnh/ khối lệnh n
} ]
```

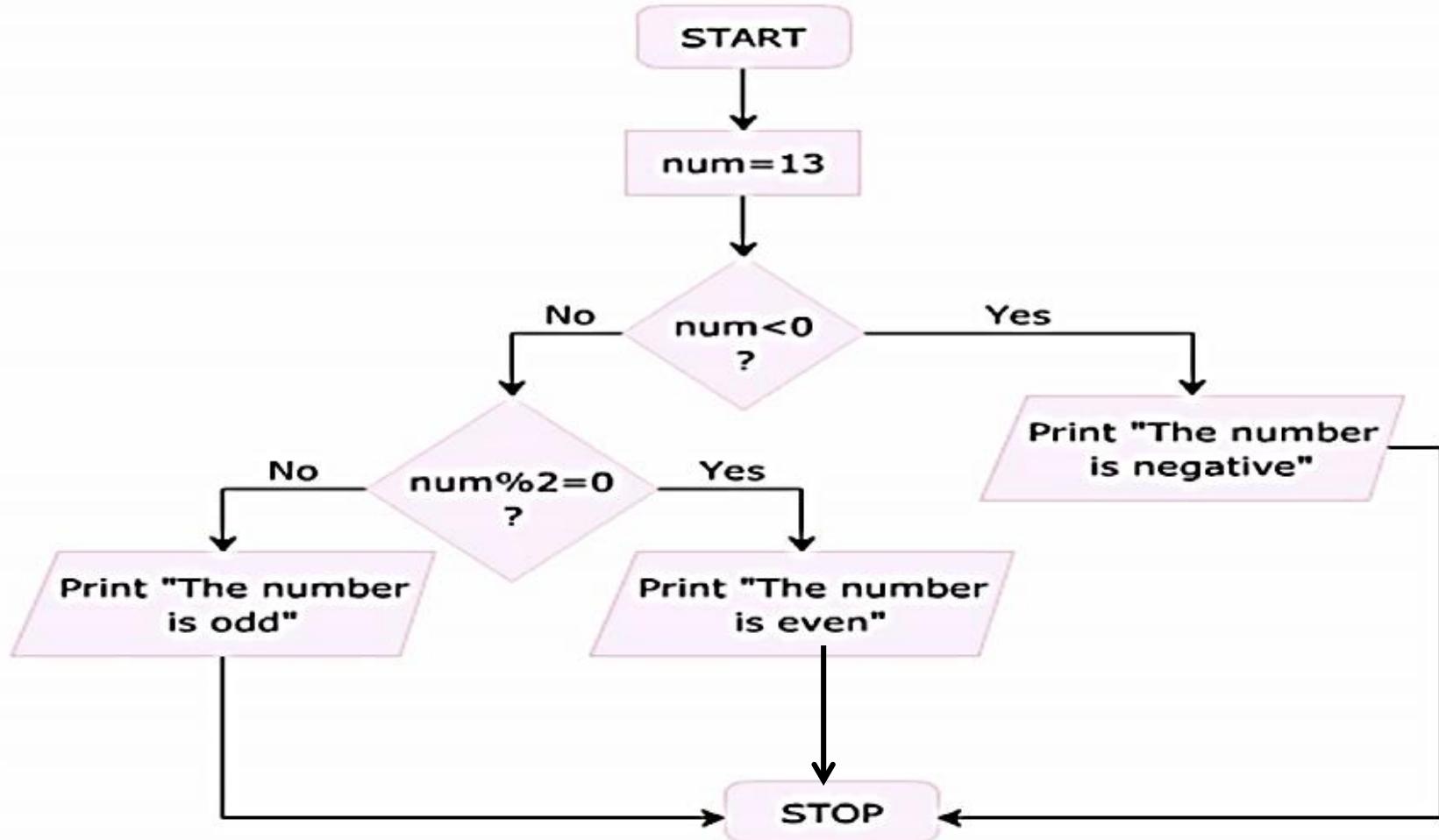
CẤU TRÚC ĐIỀU KIỆN IF – DẠNG 3

□ Cú pháp



CẤU TRÚC ĐIỀU KIỆN IF – DẠNG 3

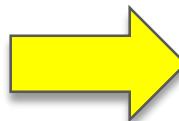
- Ví dụ: viết chương trình giải bài toán theo sơ đồ khối sau



CẤU TRÚC ĐIỀU KIỆN IF – DẠNG 3

- Ví dụ: viết chương trình giải bài toán theo sơ đồ khối sau

```
int num = 13;
if (num < 0)
{
    Console.WriteLine("The number is negative");
}
else if ((num % 2) == 0)
{
    Console.WriteLine("The number is even");
}
else
    Console.WriteLine("The number is odd");
```



```
file:///C:/Users/tgdd/documents/visual sti
The number is odd
```

SỰ LỒNG NHAU CỦA CÁC CẤU TRÚC ĐIỀU KIỆN

□ Cú pháp

```
if (<Biểu thức Logic 1>)
```

```
{
```

```
...
```

```
if (<Biểu thức Logic 2>)
```

```
{
```

```
...
```

```
if (<Biểu thức Logic 3>)
```

```
{
```

```
...
```

```
}
```

```
...
```

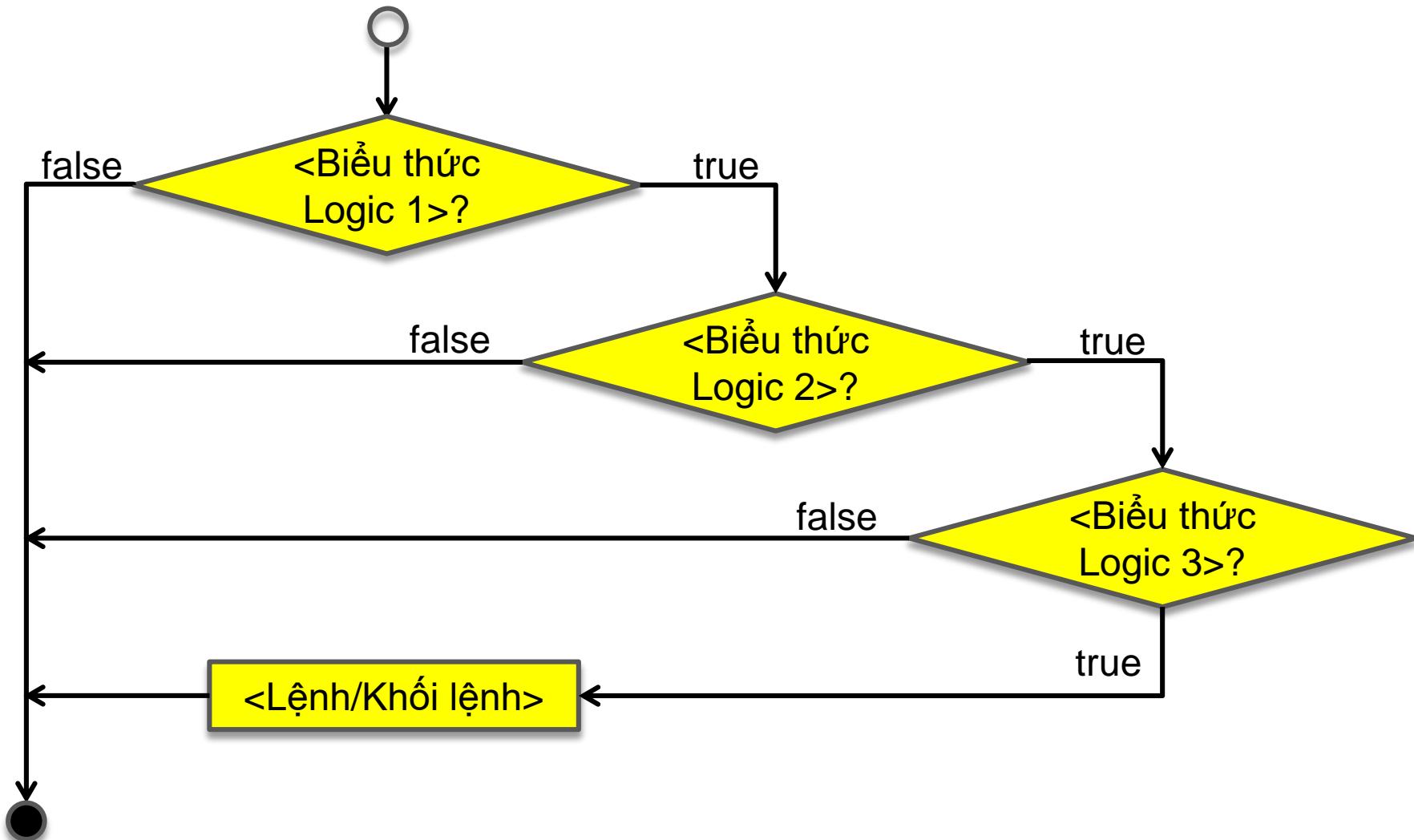
```
}
```

```
...
```

```
}
```

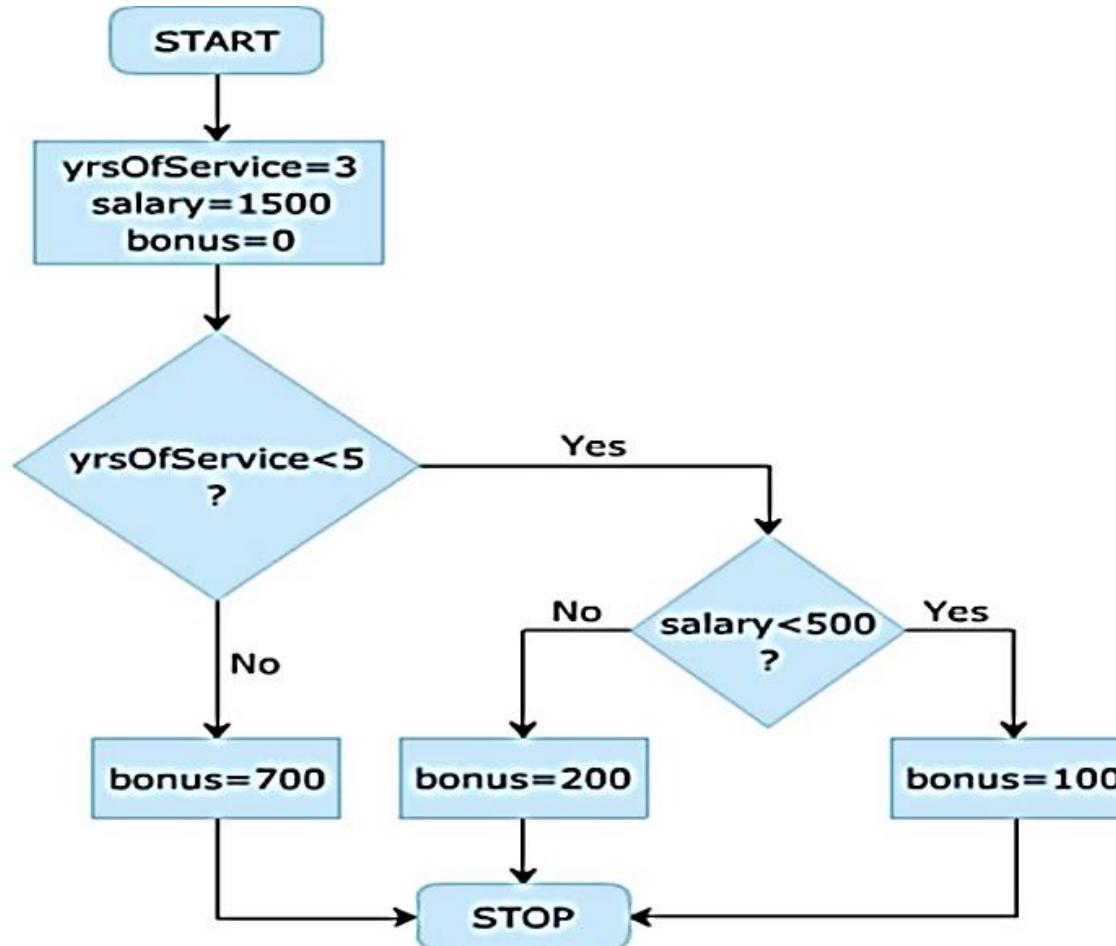
SỰ LỒNG NHAU CỦA CÁC CẤU TRÚC ĐIỀU KIỆN

□ Cú pháp



SỰ LỒNG NHAU CỦA CÁC CẤU TRÚC ĐIỀU KIỆN

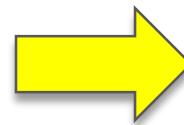
- Ví dụ: viết chương trình giải bài toán theo sơ đồ khối sau



SỰ LỒNG NHAU CỦA CÁC CẤU TRÚC ĐIỀU KIỆN

- Ví dụ: viết chương trình giải bài toán theo sơ đồ khối sau

```
int yrsOfService= 3;  
double salary = 1500;  
int bonus = 0;  
if (yrsOfService< 5)  
{  
    if (salary < 500)  
        bonus = 100;  
    else  
        bonus = 200;  
}  
else  
    bonus = 700;  
  
Console.WriteLine("Bonus amount: " + bonus);
```

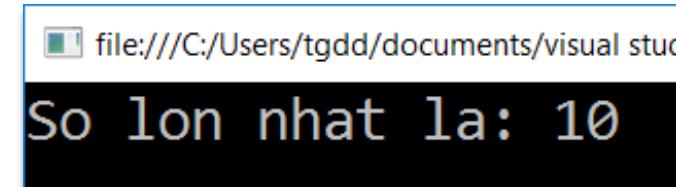
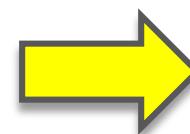


```
file:///C:/Users/tgdd/documents/visual stu  
Bonus amount: 200
```

CẤU TRÚC ĐIỀU KIỆN IF – BÀI TẬP VÍ DỤ

- **Ví dụ 1:** Tìm số lớn nhất giữa hai số a và b
 - Cách 1:

```
int a=5, b=10;  
if (a > b)  
    Console.WriteLine("So lon nhat la: " + a);  
else  
    Console.WriteLine("So lon nhat la: " + b);
```

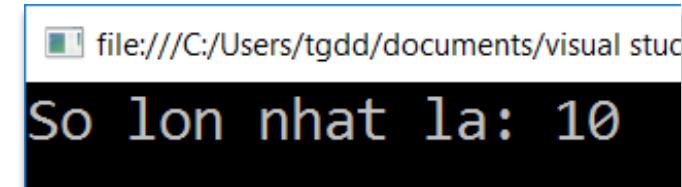
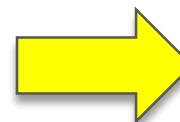


```
file:///C:/Users/tgdd/documents/visual studio 2010  
So lon nhat la: 10
```

CẤU TRÚC ĐIỀU KIỆN IF – BÀI TẬP VÍ DỤ

- Ví dụ 1: Tìm số lớn nhất giữa hai số a và b
 - Cách 2:

```
int max, a=5, b=10;  
if (a > b)  
    max = a;  
else  
    max = b;  
Console.WriteLine("So lon nhat la: " + max);
```

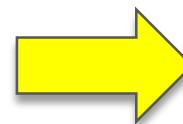


```
file:///C:/Users/tgdd/documents/visual studio  
So lon nhat la: 10
```

CẤU TRÚC ĐIỀU KIỆN IF – BÀI TẬP VÍ DỤ

- Ví dụ 1: Tìm số lớn nhất giữa hai số a và b
 - Cách 3:

```
int max, a=5, b=10;  
max=a;  
if (max < b)  
    max=b;  
Console.WriteLine("So lon nhat la: " + max);
```

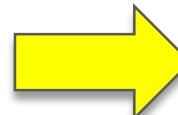


```
file:///C:/Users/tgdd/documents/visual studio  
So lon nhat la: 10
```

CẤU TRÚC ĐIỀU KIỆN IF – BÀI TẬP VÍ DỤ

- Ví dụ 2: Giải phương trình bậc nhất: $ax + b = 0$

```
int a, b;  
Console.WriteLine("Nhập a,b: ");  
a=Convert.ToInt32(Console.ReadLine());  
b=Convert.ToInt32(Console.ReadLine());  
if (a==0)  
{  
    if (b==0)  
        Console.WriteLine("Phương trình có vô số nghiệm!!!");  
    else //b<>0  
        Console.WriteLine("Phương trình vô nghiệm!!!");  
}  
else //a<>0  
    Console.WriteLine("Phương trình có nghiệm x= " + (-b/a));
```



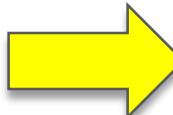
```
file:///C:/Users/tgdd/documents/visual studio 2010/Projects/Chu  
Nhập a,b: 5  
10  
Phương trình có nghiệm x= -2
```

CẤU TRÚC ĐIỀU KIỆN IF – BÀI TẬP VÍ DỤ

- ❑ **Ví dụ 3:** Giải và biện luận phương trình bậc hai: $ax^2 + bx + c = 0$

```
double a,b,c,d;
Console.WriteLine("Nhập a,b,c: ");
a = Convert.ToDouble(Console.ReadLine());
b = Convert.ToDouble(Console.ReadLine());
c = Convert.ToDouble(Console.ReadLine());

d=b*b-4*a*c; //Delta
if (d<0)
    Console.WriteLine("Phương trình vô nghiệm");
else if (d==0)
    Console.WriteLine("Phương trình có nghiệm kép x1,x2=" + (-b/(2*a)));
else
{
    Console.WriteLine("Phương trình có hai nghiệm:");
    Console.WriteLine(" x1= " + (-b + Math.Sqrt(d)) / (2 * a));
    Console.WriteLine(" x2= " + (-b - Math.Sqrt(d)) / (2 * a));
}
```



file:///C:/Users/tgdd/documents/visual studio 2010/Projects/ChuongTrinhBacHai/nhanh.exe

Nhập a,b,c: 4
5
1
Phương trình có hai nghiệm:
x1= -0.25
x2= -1

CẤU TRÚC ĐIỀU KIỆN IF – BÀI TẬP ÔN TẬP

Viết chương trình sử dụng cấu trúc IF:

- Câu 1.** Nhập 3 số thực từ bàn phím, in lên màn hình số lớn nhất và bé nhất trong 3 số trên.

Ví dụ:

```
a=5  
b=4.5  
c=7  
SLN=7  
SBN=4.5
```

CẤU TRÚC ĐIỀU KIỆN IF – BÀI TẬP ÔN TẬP

Viết chương trình sử dụng cấu trúc IF:

- **Câu 2.** Một hãng máy tính có chính sách khuyến mại, cứ mua từ 5 máy trở lên thì giá một máy sẽ là **450\$** còn không thì giá một máy sẽ là **500\$**. Viết chương trình yêu cầu người dùng nhập vào số máy muốn mua, sau đó in ra màn hình số tiền người đó phải trả cho hãng.

Ví dụ:

```
so may=10  
so tien=4500
```

CẤU TRÚC ĐIỀU KIỆN IF – BÀI TẬP ÔN TẬP

Viết chương trình sử dụng cấu trúc IF:

- Câu 3. Nhập vào số KW điện tiêu thụ của một hộ gia đình, sau đó in lên màn hình số tiền mà hộ gia đình đó phải trả biết rằng cách tính tiền điện như sau:

- Từ KW **1** đến KW thứ **100**: giá **550** đ/1KW
- Từ KW **101** đến KW thứ **150**: giá **750** đ/1KW
- Từ KW **151** đến KW thứ **200**: giá **950** đ/1KW
- Từ KW **201** trở đi: giá **1350** đ/1KW
- Thuế VAT là **10%**.

Thành tiền = Số KW tiêu thụ * Đơn giá + VAT

Ví dụ:

Tieu thu=110
Phai tra=68750

CẤU TRÚC RẺ NHÁNH SWITCH ... CASE

□ Cú pháp

switch (<Biểu thức nguyên>)

{

case <n₁>:

<Lệnh/Khối lệnh 1>; [break;]

case <n₂>:

<Lệnh/Khối lệnh 2>; [break;]

...

case <n_k>:

<Lệnh/Khối lệnh k>; [break;]

[default :

<Lệnh/Khối lệnh k+1>; break;]

}

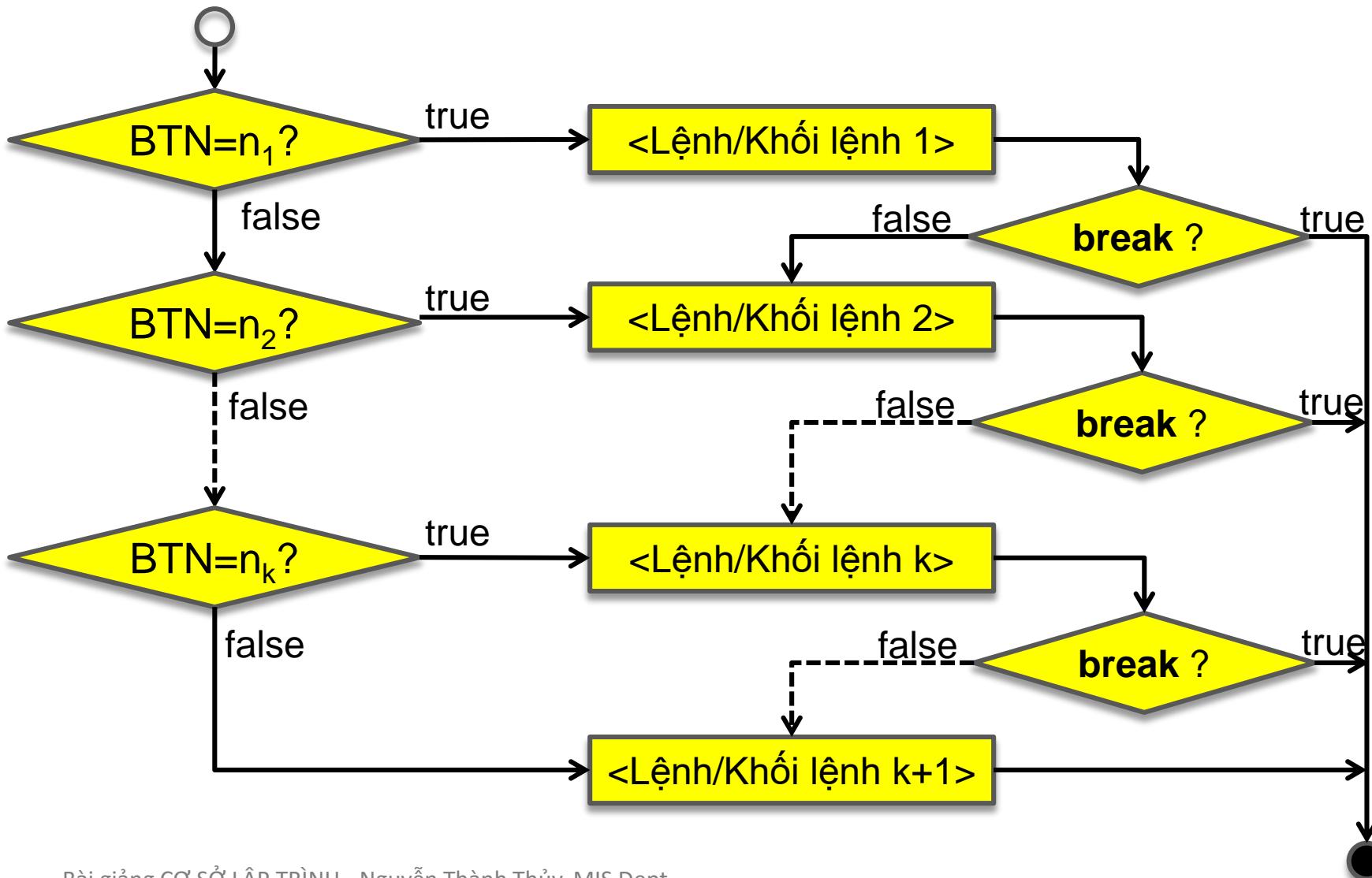
CẤU TRÚC RẺ NHÁNH SWITCH ... CASE

- Trong đó:

- <Biểu thức nguyên>: là một biến hoặc biểu thức có giá trị nguyên;
- < n_1 >, < n_2 >, ... < n_k >: là một giá trị hằng số, có kiểu số nguyên, ký tự hoặc kiểu logic (true/false);
- Từ khóa **break**: có thể được sử dụng hoặc không; được sử dụng để thoát khỏi thân **switch**;
- Từ khóa **default**: có thể được sử dụng hoặc không; có chức năng tương tự **else** trong cấu trúc **if**;
- Những bài toán được giải bằng **if ... else if ... else** thì có thể giải được bằng cấu trúc **switch ... case**;

CẤU TRÚC RẺ NHÁNH SWITCH ... CASE

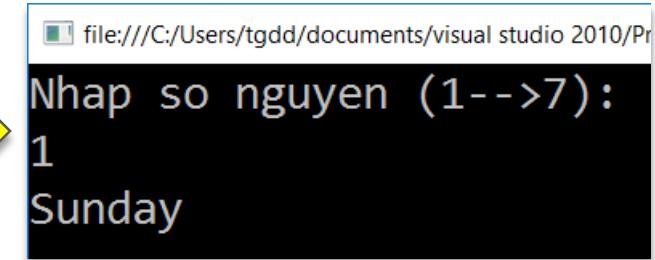
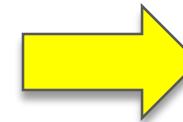
□ Cú pháp



CẤU TRÚC RẺ NHÁNH SWITCH ... CASE

- ❑ **Ví dụ 1:** nhập từ bàn phím một số nguyên từ 1 → 7, in lên màn hình tên của ngày trong tuần. Nhập sai thì thông báo lỗi.

```
int day;  
Console.WriteLine("Nhập số nguyên (1-->7): ");  
day=Convert.ToInt32(Console.ReadLine());  
switch (day)  
{  
    case 1:  
        Console.WriteLine("Sunday"); break;  
    case 2:  
        Console.WriteLine("Monday"); break;  
    case 3:  
        Console.WriteLine("Tuesday"); break;  
    case 4:  
        Console.WriteLine("Wednesday"); break;  
    case 5:  
        Console.WriteLine("Thursday"); break;  
    case 6:  
        Console.WriteLine("Friday"); break;  
    case 7:  
        Console.WriteLine("Saturday"); break;  
    default:  
        Console.WriteLine("Vui lòng nhập số từ 1-->7 !!!"); break;  
}
```



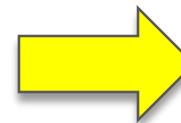
```
Nhap so nguyen (1-->7):  
1  
Sunday
```

Luôn có ít nhất một từ khóa **break** trong khối lệnh sau **default**

CẤU TRÚC RẺ NHÁNH SWITCH ... CASE

- ❑ **Ví dụ 2:** Nhập điểm thi từ 0 → 10, in lên màn hình xếp loại kết quả học tập. Nhập sai thì bỏ qua.

```
int diem; Console.WriteLine("Nhập điểm thi (0-->10): ");
diem=Convert.ToInt32(Console.ReadLine());
switch (diem)
{
    case 0:
    case 1:
    case 2:
    case 3: Console.WriteLine("Kém\n");break;
    case 4: Console.WriteLine("Yếu\n");break;
    case 5:
    case 6: Console.WriteLine("Trung bình\n");break;
    case 7:
    case 8: Console.WriteLine("Khá\n");break;
    case 9:
    case 10: Console.WriteLine("Gioi\n");break;
}
```



```
file:///C:/Users/tgdd/documents/visual studio 2010/Project1/Nhap diem thi (0-->10):
9
Gioi
```

CẤU TRÚC RẺ NHÁNH – BÀI TẬP ÔN TẬP

Viết chương trình sử dụng cấu trúc switch:

- **Câu 1.** In lên màn hình menu như sau, khi nhập vào một số từ 1 → 5 thì hiển thị tên màu tương ứng, còn lại thông báo “Khong hop le!!!”.

```
Hay cho biet mau sac ma ban ua thich
Nhap <1>: Do
Nhap <2>: Xanh la
Nhap <3>: Xanh duong
Nhap <4>: Trang
Nhap <5>: Den
Ban chon so: 1
-----
Ban thich mau: Do
-----
```

CẤU TRÚC RẺ NHÁNH – BÀI TẬP ÔN TẬP

Viết chương trình sử dụng cấu trúc switch:

- **Câu 2.** Nhập từ bàn phím một số nguyên X, in lên màn hình thông báo X là số chẵn hay lẻ.
- **Câu 3.** Nhập từ bàn phím một xâu ký tự là tên của một trong 4 màu sau: red, green, yellow, black. In lên màn câu thông báo “Ban da chon mau ...” tương ứng với màu đã nhập vào. Nếu nhập ngoài 4 màu trên thì thông báo “Khong hop le!!!”.

BÀI GIẢNG

CƠ SỞ LẬP TRÌNH

CHƯƠNG 3.

CẤU TRÚC ĐIỀU KHIỂN

NGUYỄN THÀNH THỦY

BỘ MÔN TIN HỌC QUẢN LÝ

TRƯỜNG ĐẠI HỌC KINH TẾ, ĐẠI HỌC ĐÀ NẴNG

THUYNT@DUE.EDU.VN

NỘI DUNG

- Cấu trúc điều kiện if
- Cấu trúc rẽ nhánh switch ... case
 - Cấu trúc lặp while
 - Cấu trúc lặp do-while
- Cấu trúc lặp for
- Câu lệnh nhảy break, continue, return

CẤU TRÚC LẶP

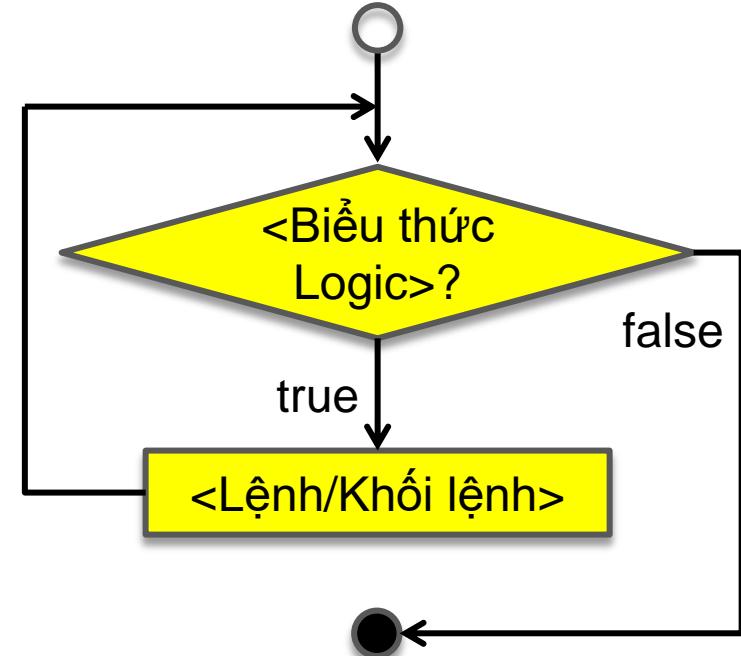
□ Đặc điểm chung

- Được sử dụng để giải các bài toán có tính chất lặp lại nhiều lần một hoặc một khối công việc nào đó;
- Số lần lặp có thể biết trước hoặc cho đến khi thỏa mãn một biểu thức Logic nào đó;
- Trong C#, có 4 loại cấu trúc lặp:
 - **while**
 - **do ... while**
 - **for ... loop**
 - **foreach ... loop**

CẤU TRÚC WHILE

□ Cú pháp

**while (<Biểu thức Logic>)
<Lệnh/Khối lệnh>**



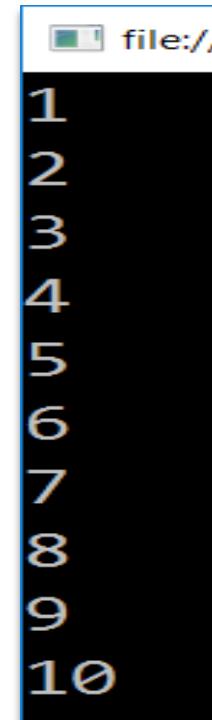
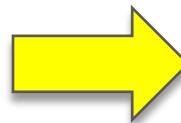
□ Hoạt động của toán tử while

- **Bước 1:** tính giá trị của <Biểu thức Logic>
- **Bước 2:**
 - Nếu có giá trị **True** thì thực hiện <Lệnh/Khối lệnh> rồi quay lại **Bước 1**;
 - Nếu có giá trị **False** thì kết thúc;

CẤU TRÚC WHILE

- ❑ **Ví dụ 1:** in các số từ 1 đến 10, mỗi số nằm trên một dòng.
Yêu cầu sử dụng cấu trúc while.

```
int i=1;
while (i <= 10) //i<=10 : là biểu thức Logic
{
    Console.WriteLine(i); //Công việc cần lặp lại
    i++; //i: được gọi là biến chạy, xác định số lần lặp lại
}
```



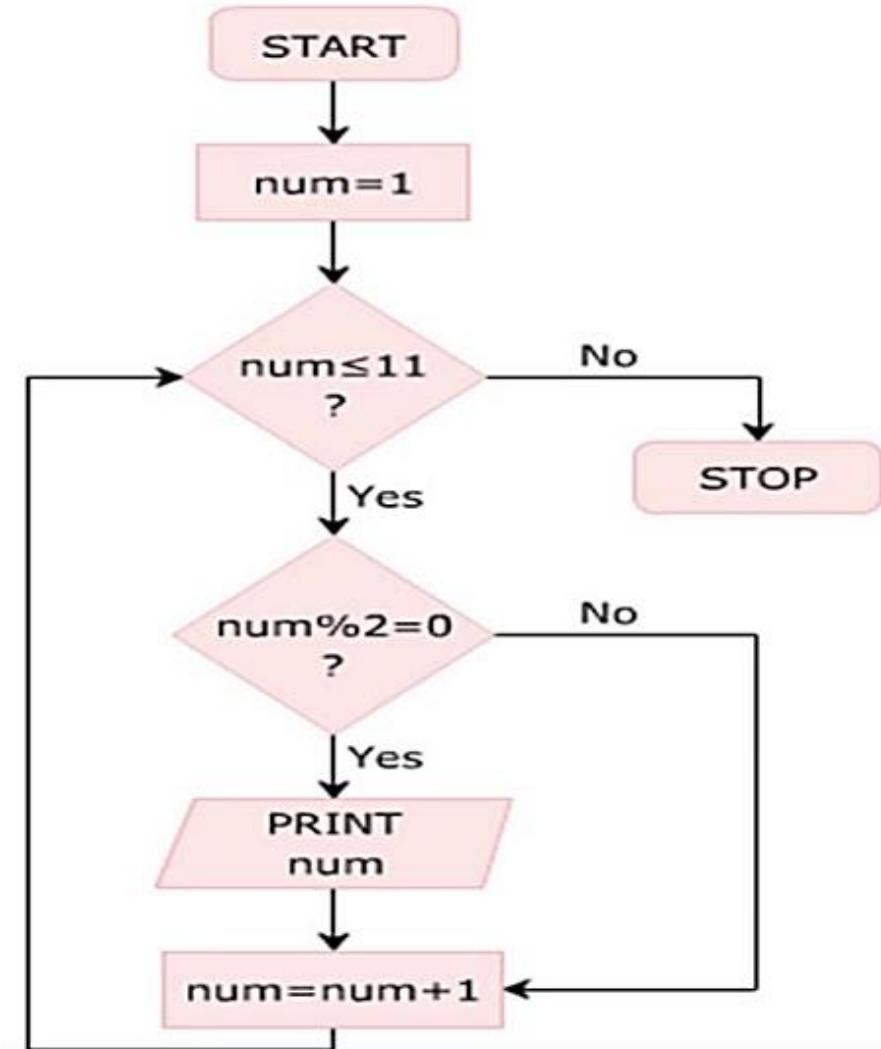
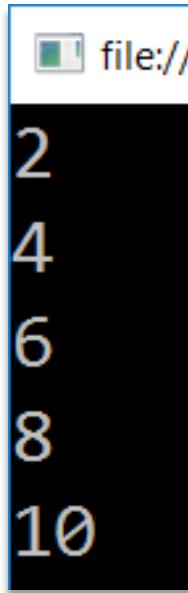
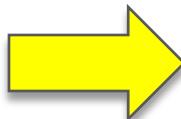
The screenshot shows a terminal window with a black background and white text. It displays the numbers 1 through 10, each on a new line. The window title bar says "file:///". A yellow arrow points from the explanatory text above to this terminal window.

```
1
2
3
4
5
6
7
8
9
10
```

CẤU TRÚC WHILE

- **Ví dụ 2:** in các số chẵn trong dãy các số từ 1 đến 10, mỗi số nằm trên một dòng. Yêu cầu sử dụng cấu trúc **while**.

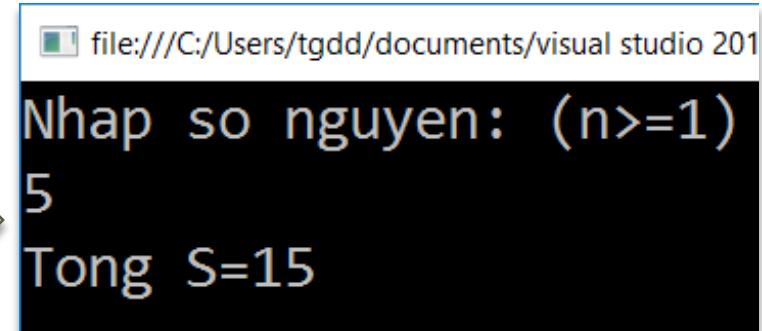
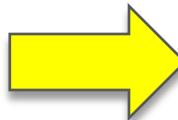
```
int num=1;  
while (num<=11)  
{  
    if (num % 2 == 0)  
        Console.WriteLine(num);  
    num++;  
}
```



CẤU TRÚC WHILE

- **Ví dụ 3:** Nhập từ bàn phím một số nguyên n ($n \geq 1$); in lên màn hình tổng của n số nguyên dương đầu tiên (các số liên tục từ 1 đến n): $S=1+2+3+\dots+n$
Yêu cầu sử dụng cấu trúc while.

```
int n, i=1, S=0;
Console.WriteLine("Nhập số nguyên: (n>=1)");
n = Convert.ToInt32(Console.ReadLine());
while (i <= n) //i<=n : là biểu thức Logic
{
    S += i; //tương đương với biểu thức: S = S + i
    i++; //i: được gọi là biến chạy, xác định số lần lặp lại
}
Console.WriteLine("Tổng S=" + S);
```

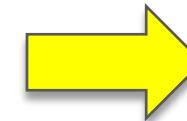


```
file:///C:/Users/tgdd/documents/visual studio 201
Nhập số nguyên: (n>=1)
5
Tổng S=15
```

CẤU TRÚC WHILE

- **Ví dụ 4:** in lên màn hình 6 dòng, có nội dung như sau. Yêu cầu sử dụng cấu trúc while.

```
int i=1,j;
while (i <= 6)
{
    j = 1;
    while (j <= i)
    {
        Console.WriteLine("*");
        j++;
    }
    Console.WriteLine();
    i++;
}
Console.ReadKey();
```



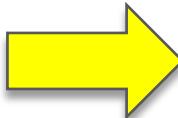
```
*  
* *  
* * *  
* * * *  
* * * * *  
* * * * * *
```

CẤU TRÚC WHILE

- ❑ **Ví dụ 5:** Nhập từ bàn phím một số nguyên n. Nếu $n \leq 0$ thì thông báo lỗi và yêu cầu nhập lại, còn lại thì dừng. Yêu cầu sử dụng cấu trúc while.

- **Cách 1:**

```
int n;
Console.WriteLine("Nhập số nguyên (n>0): n= ");
n = Convert.ToInt32(Console.ReadLine());
while (n<=0)
{
    Console.WriteLine("Không hợp lệ!!!\nMời nhập lại (n>0): n= ");
    n = Convert.ToInt32(Console.ReadLine());
}
```



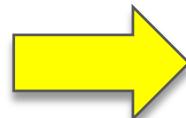
```
file:///C:/Users/tgdd/documents/visual studio 2010/Projects/C
Nhập số nguyên (n>0): n= -5
Không hợp lệ!!!
Mời nhập lại (n>0): n= 9
```

CẤU TRÚC WHILE

□ Ví dụ 5:

■ Cách 2:

```
int n;
while (true)
{
    Console.WriteLine("Nhập số nguyên (n>0): n= ");
    n = Convert.ToInt32(Console.ReadLine());
    if (n <= 0)
        Console.WriteLine("Không hợp lệ!!!");
    else break; //Thoát khỏi cấu trúc while
}
```



```
Nhập số nguyên (n>0): n= -4
Không hợp lệ!!!
Nhập số nguyên (n>0): n= -6
Không hợp lệ!!!
Nhập số nguyên (n>0): n= 9
```

CẤU TRÚC WHILE – BÀI TẬP ÔN TẬP

Sử dụng cấu trúc **while** để thực hiện các yêu cầu sau:

Câu 1. Nhập từ bàn phím một số nguyên n, trong đó $n \geq 1$ và $n \leq 50$. Nếu n không thuộc miền trên thì yêu cầu nhập lại.

Câu 2. Nhập số nguyên n ($1 \leq n \leq 50$) từ bàn phím. In lên màn hình bảng chữ số theo cấu trúc sau.

Ví dụ, nếu $n=45$, kết quả màn hình như sau:

1 2 3 4 5 6 7 8 9 10

11 12 13 14 15 16 17 18 19 20

21 22 23 24 25 26 27 28 29 30

...

41 42 43 44 45

CẤU TRÚC WHILE – BÀI TẬP ÔN TẬP

Câu 3. In lên màn hình bảng cửu chương 9x9

| | | | | | | | | |
|---|----|----|----|----|----|----|----|----|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 |
| 3 | 6 | 9 | 12 | 15 | 18 | 21 | 24 | 27 |
| 4 | 8 | 12 | 16 | 20 | 24 | 28 | 32 | 36 |
| 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 |
| 6 | 12 | 18 | 24 | 30 | 36 | 42 | 48 | 54 |
| 7 | 14 | 21 | 28 | 35 | 42 | 49 | 56 | 63 |
| 8 | 16 | 24 | 32 | 40 | 48 | 56 | 64 | 72 |
| 9 | 18 | 27 | 36 | 45 | 54 | 63 | 72 | 81 |

CẤU TRÚC WHILE – BÀI TẬP ÔN TẬP

Câu 4. In lên màn hình theo cấu trúc sau:

| | |
|--------------------|-------|
| \$\$\$\$\$\$\$\$\$ | * |
| \$\$\$\$\$\$\$ | *** |
| \$\$\$\$\$\$ | ***** |
| \$\$\$\$\$ | ***** |
| \$\$\$\$ | ***** |
| \$\$\$ | ***** |
| \$\$ | ***** |
| \$ | ***** |

a.

b.

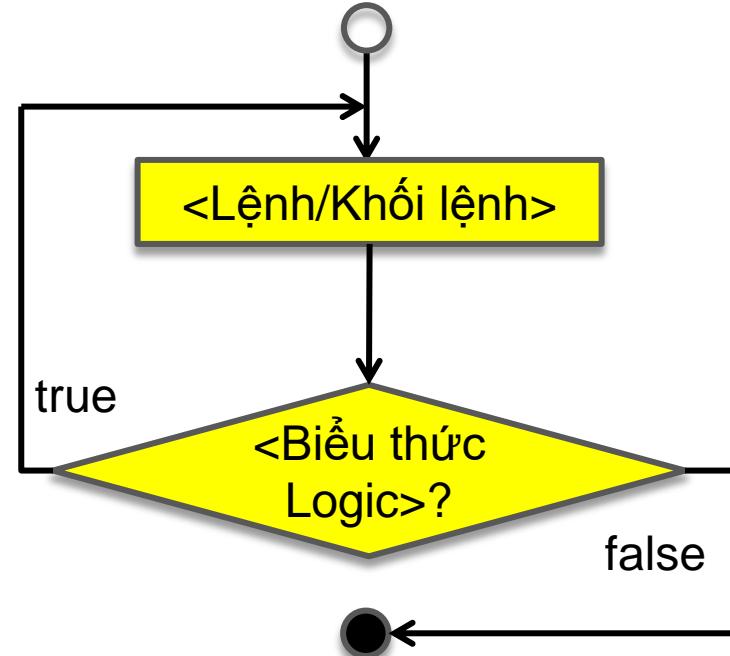
CẤU TRÚC DO-WHILE

□ Cú pháp

do

<Lệnh/Khối lệnh>

while (<Biểu thức Logic>);



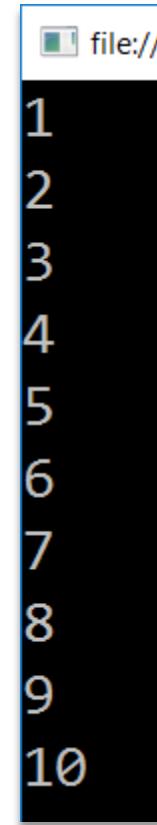
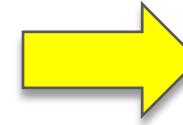
□ Hoạt động của toán tử do-while

- **Bước 1:** thực hiện <Lệnh/Khối lệnh>
- **Bước 2:** tính giá trị của <Biểu thức Logic>
 - Nếu có giá trị **đúng** thì quay lại **Bước 1**;
 - Nếu có giá trị **sai** thì kết thúc;

CẤU TRÚC DO-WHILE

- ❑ **Ví dụ 1:** in các số từ 1 đến 10, mỗi số trên một dòng. Yêu cầu sử dụng cấu trúc **do-while**.

```
int i = 1;  
do  
{  
    Console.WriteLine(i);  
    i++;  
} while (i <= 10);
```

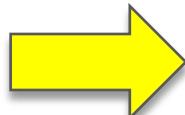


```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10
```

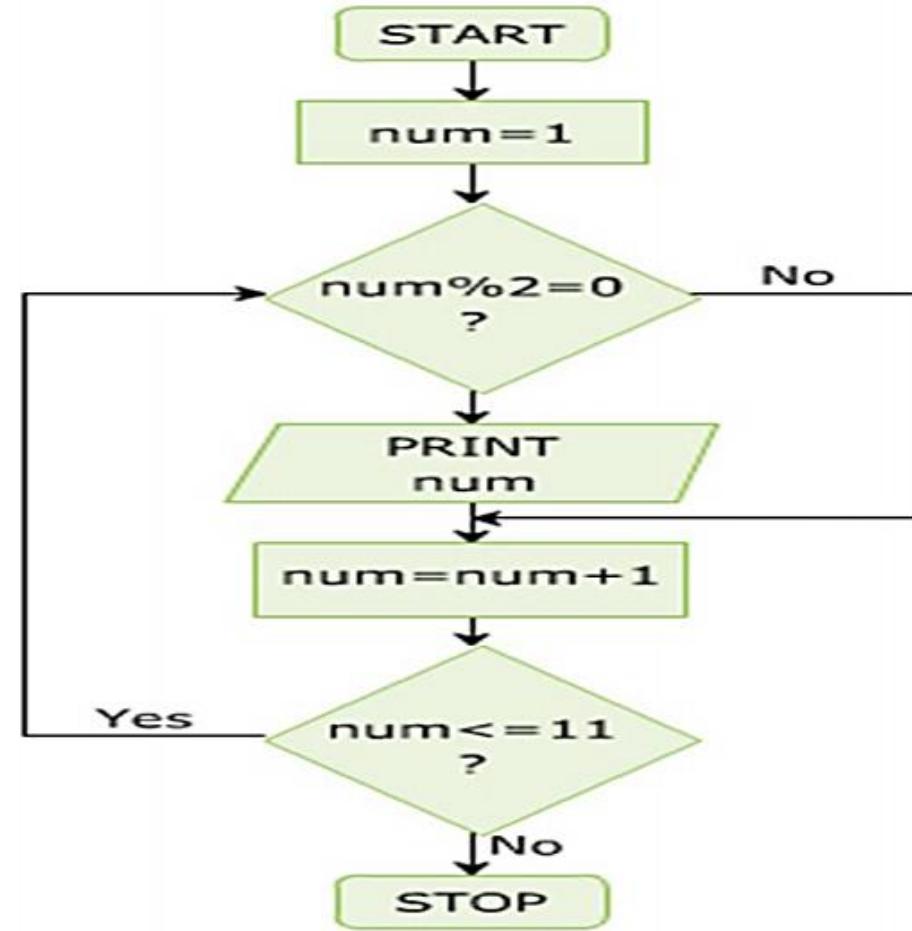
CẤU TRÚC DO-WHILE

- ❑ **Ví dụ 2:** in các số chẵn trong dãy các số từ 1 đến 10, mỗi số trên một dòng. Yêu cầu sử dụng cấu trúc **do-while**.

```
int num = 1;  
do  
{  
    if ((num % 2) == 0)  
    {  
        Console.WriteLine(num);  
    }  
    num = num + 1;  
} while (num <= 11);
```



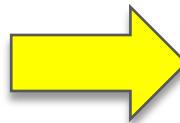
```
2  
4  
6  
8  
10
```



CẤU TRÚC DO-WHILE

- **Ví dụ 3:** Nhập từ bàn phím một số nguyên n ($n \geq 1$); in lên màn hình tổng của n số nguyên dương đầu tiên (các số liên tục từ 1 đến n): $S=1+2+3+\dots+n$
Yêu cầu sử dụng cấu trúc do-while.

```
int n, S=0, i=1;
Console.WriteLine("Nhập số nguyên: (n>=1)");
n=Convert.ToInt32(Console.ReadLine());
do
{
    S += i;
    i++;
} while (i <= n);
Console.WriteLine("Tổng S=" + S);
```

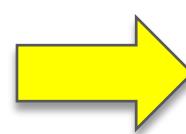


A screenshot of a Windows command prompt window titled "file:///C:/Users/tgdd/documents/visual studio 201". The window shows the following text:
Nhập số nguyên: (n>=1)
5
Tổng S=15

CẤU TRÚC DO-WHILE

- ❑ **Ví dụ 4:** in lên màn hình 6 dòng, có nội dung như sau. Yêu cầu sử dụng cấu trúc **do-while**.

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

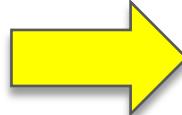


```
*  
* *  
* * *  
* * * *  
* * * * *
```

CẤU TRÚC DO-WHILE

- ❑ **Ví dụ 5:** Nhập từ bàn phím một số nguyên n. Nếu $n \leq 0$ thì thông báo lỗi và yêu cầu nhập lại, còn lại thì dừng. Yêu cầu sử dụng cấu trúc do-while.

```
int n;  
Console.WriteLine("Nhập số nguyên (n>0): ");  
do  
{  
    Console.Write("n= ");  
    n = Convert.ToInt32(Console.ReadLine());  
  
    if (n <= 0)  
        Console.WriteLine("Không hợp lệ!!!\nMoi nhập lại (n>0): ");  
} while (n<=0);
```



```
file:///C:/Users/tgdd/documents/visual studio 2010/Projects/C  
Nhập số nguyên (n>0): n= -5  
Không hợp lệ!!!  
Moi nhập lại (n>0): n= 9
```

CẤU TRÚC DO-WHILE – BÀI TẬP ÔN TẬP

Giải lại các bài tập ở slide 11 và 12, bằng cấu trúc do-while

BÀI GIẢNG

CƠ SỞ LẬP TRÌNH

CHƯƠNG 3.

CẤU TRÚC ĐIỀU KHIỂN

NGUYỄN THÀNH THỦY

BỘ MÔN TIN HỌC QUẢN LÝ

TRƯỜNG ĐẠI HỌC KINH TẾ, ĐẠI HỌC ĐÀ NẴNG

THUYNT@DUE.EDU.VN

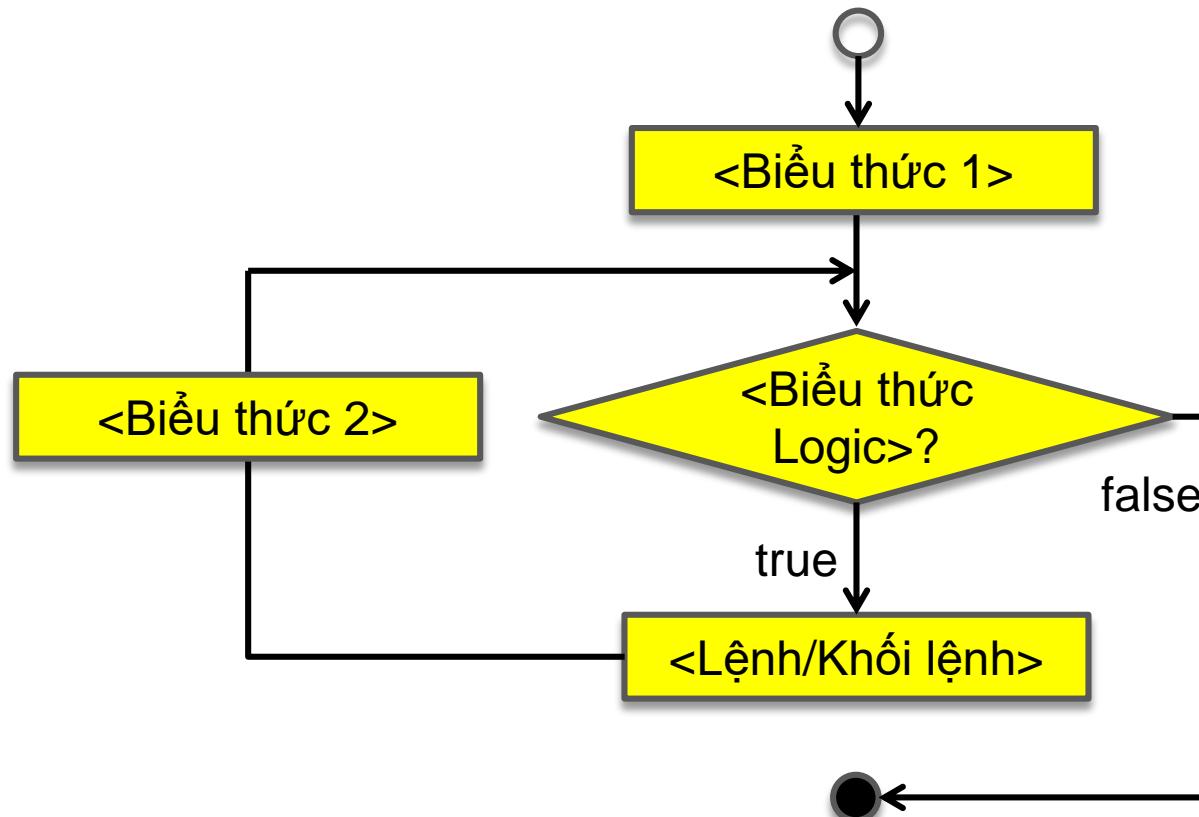
NỘI DUNG

- Cấu trúc điều kiện if
- Cấu trúc rẽ nhánh switch ... case
- Cấu trúc lặp while
- Cấu trúc lặp do-while
- Cấu trúc lặp for
- Câu lệnh nhảy break, continue

CẤU TRÚC FOR

□ Cú pháp

**for([<B.Thức 1>];<B.Thức Logic>;[< B.Thức 2 >])
 <Lệnh/Khối lệnh>**



CẤU TRÚC FOR

□ Hoạt động của toán tử while

- **Bước 1:** Thực hiện <Biểu thức 1>
- **Bước 2:** Tính giá trị của <Biểu thức Logic>
 - Nếu có giá trị **False** thì kết thúc;
 - Nếu có giá trị **True** thì thực hiện <Lệnh/Khối lệnh> rồi xuống **Bước 3**;
- **Bước 3:**
 - Thực hiện <Biểu thức 2>
 - Quay về **Bước 2**

CẤU TRÚC FOR

□ **Ví dụ 1:** in các số từ 1 đến 10, mỗi số nằm trên một dòng. Yêu cầu sử dụng cấu trúc **for**.

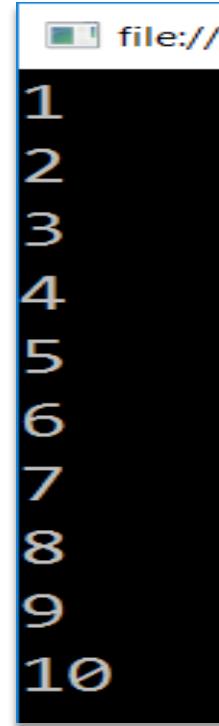
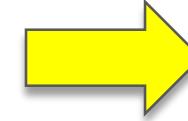
- **Thuật toán:**

- Với mỗi giá trị của $i = [1..10]$, thực hiện:
 - In i lên màn hình
 - Xuống dòng

CẤU TRÚC FOR

- Ví dụ 1: in các số từ 1 đến 10, mỗi số nằm trên một dòng. Yêu cầu sử dụng cấu trúc **for**.

```
int i; //i: biến chạy
for (i = 1; i <= 10; i++)
    Console.WriteLine(i);
```



The screenshot shows a terminal window with a black background and white text. The title bar says "file:///". The window contains the following text:
1
2
3
4
5
6
7
8
9
10

CẤU TRÚC FOR

- **Ví dụ 2:** in các số chẵn trong dãy các số từ 1 đến 10, mỗi số nằm trên một dòng. Yêu cầu sử dụng cấu trúc **for**.

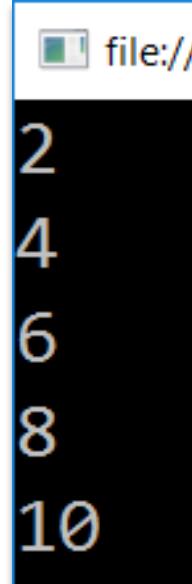
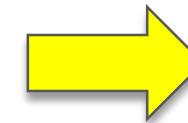
- **Thuật toán:**

- Với mỗi giá trị của $i = [1..10]$, thực hiện:
 - Nếu $i \% 2 = 0$ thì: In i lên màn hình
 - Xuống dòng

CẤU TRÚC FOR

- **Ví dụ 2:** in các số chẵn trong dãy các số từ 1 đến 10, mỗi số nằm trên một dòng. Yêu cầu sử dụng cấu trúc **for**.

```
int i; //i: biến chạy
for (i = 1; i <= 10; i++)
    if (i%2==0)
        Console.WriteLine(i);
```



```
2
4
6
8
10
```

CẤU TRÚC FOR

- **Ví dụ 3:** Nhập từ bàn phím một số nguyên n ($n \geq 1$); in lên màn hình tổng của n số nguyên dương đầu tiên (*các số liên tục từ 1 đến n*): $S=1+2+3+\dots+n$.
Yêu cầu sử dụng cấu trúc **for**.

- **Thuật toán:**

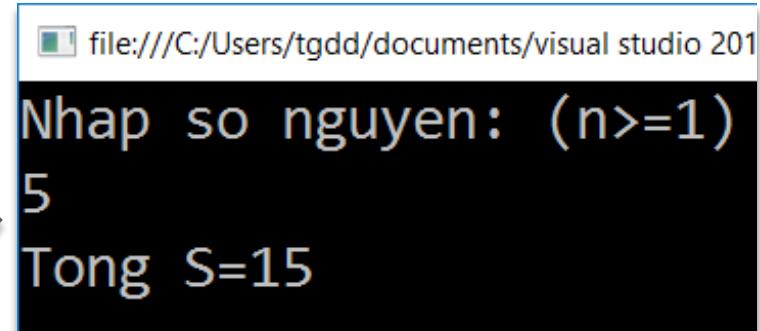
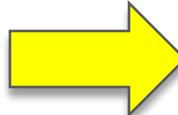
- Nhập n
- $S=0$
- Với mỗi giá trị của $i = [1..n]$, thực hiện:
 - $S = S + i$
- In S lên màn hình.

CẤU TRÚC FOR

- **Ví dụ 3:** Nhập từ bàn phím một số nguyên n ($n \geq 1$); in lên màn hình tổng của n số nguyên dương đầu tiên (*các số liên tục từ 1 đến n*): $S=1+2+3+\dots+n$.
Yêu cầu sử dụng cấu trúc **for**.

```
int i,n,S=0;
Console.WriteLine("Nhập số nguyên: (n>=1)");
n = int.Parse(Console.ReadLine());

for (i = 1; i <= n; i++)
    S += i;
Console.Write("Tổng S=" + S);
```



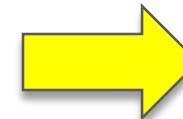
Nhập số nguyên: (n>=1)
5
Tổng S=15

CẤU TRÚC FOR

- **Ví dụ 4:** in lên màn hình 6 dòng, có nội dung như sau. Yêu cầu sử dụng cấu trúc **for**.

- **Thuật toán:**

- Với mỗi giá trị của $i = [1..6]$, thực hiện:
 - Với mỗi giá trị của $j = [1..i]$, thực hiện:
In lên màn hình dấu “*”
 - Xuống dòng



| |
|-------------|
| * |
| * * |
| * * * |
| * * * * |
| * * * * * |
| * * * * * * |

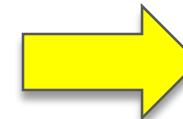
CẤU TRÚC FOR

- **Ví dụ 4:** in lên màn hình 6 dòng, có nội dung như sau. Yêu cầu sử dụng cấu trúc **for**.

```
int i, j;
for (i = 1; i <= 6; i++)
{
    //Tại dòng thứ i: In lên màn hình i dấu hoa thị
    for (j = 1; j <= i; j++)
        Console.WriteLine("* ");

    Console.WriteLine("//Xuống dòng"
}

```



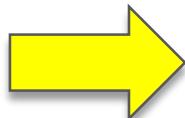
*
* *
* * *
* * * *
* * * * *
* * * * *

CẤU TRÚC FOR

- **Ví dụ 5:** Nhập từ bàn phím một số nguyên n. Nếu $n \leq 0$ thì thông báo lỗi và yêu cầu nhập lại, còn lại thì dừng. Yêu cầu sử dụng cấu trúc **for**.

- **Thuật toán:**

- **Bước 1:** Nhập số nguyên n
- **Bước 2:**
 - Nếu $n \leq 0$ thì:
 - In lên màn hình “Khong hop le!!!”
 - Quay về **Bước 1**
 - Còn lại thì đến **Bước 3**
- **Bước 3:** Kết thúc

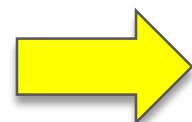


```
file:///E:/BaiTapCSLT/Chuong3/ViDu2b/bin/  
Nhap so nguyen (n>0): n= -10  
Khong hop le!!!  
Nhap so nguyen (n>0): n= 5
```

CẤU TRÚC FOR

- **Ví dụ 5:** Nhập từ bàn phím một số nguyên n. Nếu $n \leq 0$ thì thông báo lỗi và yêu cầu nhập lại, còn lại thì dừng. Yêu cầu sử dụng cấu trúc **for**.

```
int n;
for (;true;)
{   //Biểu thức 1 và Biểu thức 2 rỗng
    Console.WriteLine("Nhập số nguyên (n>0): n= ");
    n = int.Parse(Console.ReadLine());
    if (n <= 0)
        Console.WriteLine("Không hợp lệ!!!");
    else break; //Thoát khỏi vòng lặp for
}
```



```
file:///E:/BaiTapCSLT/Chuong3/ViDu2b/bin/
Nhập số nguyên (n>0): n= -10
Không hợp lệ!!!
Nhập số nguyên (n>0): n= 5
```

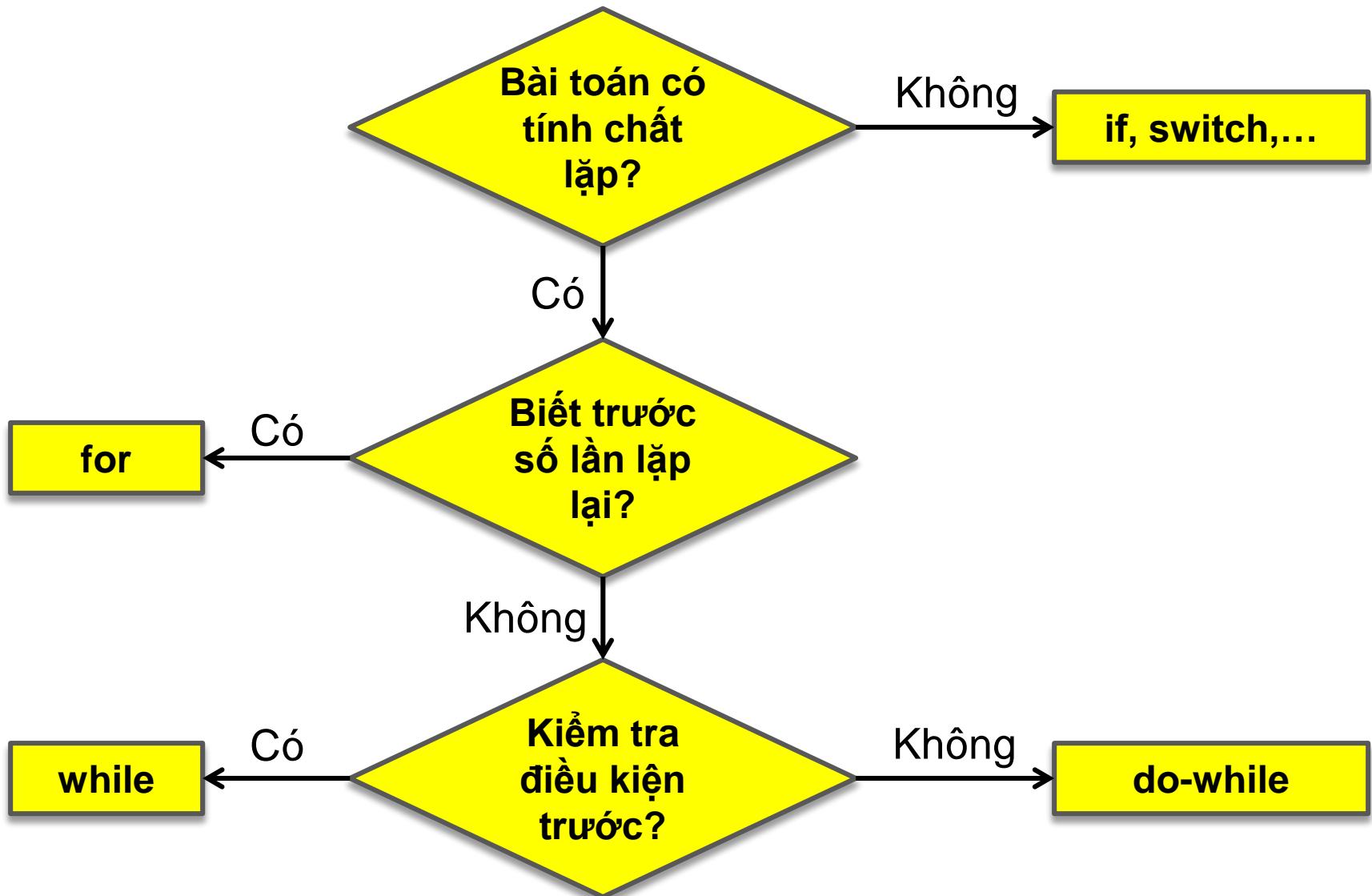
CẤU TRÚC FOR – BÀI TẬP ÔN TẬP

Giải lại các bài tập: **Câu 2, Câu 3, Câu 4** ở slide 11 và 12 (**Chương 3-Part2**), bằng cấu trúc **for**

SO SÁNH GIỮA CÁC CẤU TRÚC LẶP

| Đặc trưng | while | do-while | for |
|---|-------|----------|-------|
| Được dùng để giải các bài toán có tính chất lặp | X | X | X |
| Kiểm tra điều kiện ... | Trước | Sau | Trước |
| Biết trước số lần lặp lại? | Không | Không | Có |
| Số lần lặp tối thiểu? | 0 | 1 | 0 |

SO SÁNH GIỮA CÁC CẤU TRÚC LẶP



CÂU LỆNH NHẢY BREAK, CONTINUE

□ Lệnh break

- Được sử dụng để thoát khỏi cấu trúc lặp gần nhất

while (<Điều kiện lặp>)

{

...

if (<Biểu thức logic>)

break;

}

...

Thoát khỏi vòng lặp

...

CÂU LỆNH NHẢY BREAK, CONTINUE

□ Lệnh break

```
do  
{  
    ...  
    if (<Biểu thức logic>)  
        break; •  
    ...  
}  
while (<Điều kiện lặp>);  
... ←
```

Thoát khỏi vòng lặp

CÂU LỆNH NHẢY BREAK, CONTINUE

□ Lệnh break

```
for (<Lệnh 1>; <Điều kiện lặp>; <Lệnh 2>)
```

```
{
```

```
...
```

```
if (<Biểu thức logic>)
```

```
break;
```

Thoát khỏi vòng lặp

```
...
```

```
}
```

```
...
```

CÂU LỆNH NHẢY BREAK, CONTINUE

□ Lệnh continue

- Được sử dụng để bỏ qua các lệnh còn lại của vòng lặp và bắt đầu chu trình lặp tiếp theo.

```
while (<Điều kiện lặp>)
```

```
{
```

```
...
```

```
if (<Biểu thức logic>)  
    continue;
```

```
}
```

```
...
```



Bắt đầu chu kỳ lặp mới

CÂU LỆNH NHẢY BREAK, CONTINUE

□ Lệnh continue

- Được sử dụng để bỏ qua các lệnh còn lại của vòng lặp và bắt đầu chu trình lặp tiếp theo.

do

{

...

if (<Biểu thức logic>)

continue;

...

}

while (<Điều kiện lặp>);

...



CÂU LỆNH NHẢY BREAK, CONTINUE

□ Lệnh continue

- Được sử dụng để bỏ qua các lệnh còn lại của vòng lặp và bắt đầu chu trình lặp tiếp theo.

```
for (<Lệnh 1>; <Điều kiện lặp>; <Lệnh 2>)
{
```

```
...
if (<Biểu thức logic>)
    continue;
```

```
}
```

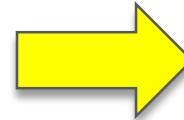
```
...
```

Bắt đầu chu kỳ lặp mới

CÂU LỆNH NHẢY BREAK, CONTINUE

- **Ví dụ 1:** In lên màn hình một dãy các số chẵn liên tục từ 1 đến 10.

```
int i;  
for (i = 1; i <= 10; i++)  
{  
    if (i % 2 != 0) continue;  
    Console.WriteLine(i + " ");  
}  
Console.ReadKey();
```

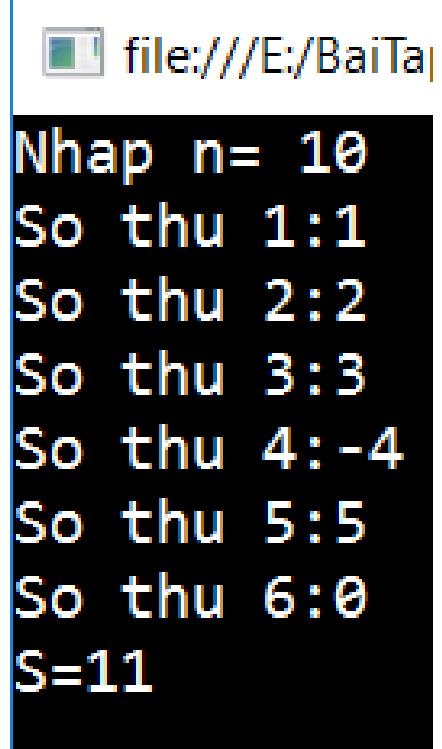


```
file:///E:/Bai1  
2 4 6 8 10
```

CÂU LỆNH NHẢY BREAK, CONTINUE

- **Ví dụ 2:** Nhập từ bàn phím một số nguyên n và n số nguyên; Việc nhập sẽ dừng lại cho đến khi đủ n số nguyên hoặc số 0 được nhập vào. In lên màn hình tổng của các số nguyên dương đã được nhập.

```
int i,n,x,S=0;
Console.WriteLine("Nhập n= ");
n=int.Parse(Console.ReadLine());
for (i = 1; i <= n; i++)
{
    Console.WriteLine("So thu " + i + ":" );
    x=int.Parse(Console.ReadLine());
    if (x<0) continue;
    if (x == 0)
        break;
    else S += x;
}
Console.WriteLine("S=" + S);
```



```
Nhập n= 10
So thu 1:1
So thu 2:2
So thu 3:3
So thu 4:-4
So thu 5:5
So thu 6:0
S=11
```

BÀI TẬP ÔN TẬP - CHƯƠNG 3

Bài 1. Viết chương trình nhập từ bàn phím một số nguyên n ($0 \leq n \leq 100$).

In lên màn hình n!

$$n! = \begin{cases} 1 & n = 0 \\ 1 * 2 * .. * n & n \geq 1 \end{cases}$$

Ví dụ:

```
n=5  
5!=120
```

BÀI TẬP ÔN TẬP - CHƯƠNG 3

Bài 2. Viết chương trình nhập từ bàn phím một số nguyên n ($1 \leq n \leq 100$). Cho biết n có phải là số nguyên tố hay không. Biết rằng, n là số nguyên tố nếu n ***chỉ chia hết cho 1 và chính nó.***

Ví dụ 1:

```
n=10  
10 khong la SNT
```

Ví dụ 2:

```
n=7  
7 la SNT
```

BÀI TẬP ÔN TẬP - CHƯƠNG 3

Bài 3. Viết chương trình nhập từ bàn phím một số nguyên dương n ($n>0$). Cho biết n có bao nhiêu chữ số. (*yêu cầu sử dụng cấu trúc lặp*)

Ví dụ 1:

```
n=123  
123 co 3 chu so
```

Ví dụ 2:

```
n=4500  
4500 co 4 chu so
```

BÀI TẬP ÔN TẬP - CHƯƠNG 3

Bài 4.

- Nhập từ bàn phím hai số thực: **a** và **b**;
- Nhập từ bàn phím một toán tử (+, -, *, /);
- In lên màn hình kết quả của biểu thức tương ứng;
- Chương trình sẽ lặp lại việc tính trên cho đến khi bấm phím **T** hoặc **t** thì kết thúc.

Ví dụ:

```
a=2.5  
b=10  
Toan tu:+  
2.5+10=12.5  
Tiep tuc:t
```

```
a=2.5  
b=10  
Toan tu:+  
2.5+10=12.5  
Tiep tuc:x  
a=2.5  
b=10  
Toan tu:*
```

$2.5 * 10 = 25$
Tiep tuc:t

BÀI GIẢNG

CƠ SỞ LẬP TRÌNH

CHƯƠNG 4. MẢNG

NGUYỄN THÀNH THỦY
BỘ MÔN TIN HỌC QUẢN LÝ
TRƯỜNG ĐẠI HỌC KINH TẾ, ĐẠI HỌC ĐÀ NẴNG
THUYNT@DUE.EDU.VN

MẢNG

☐ Tổng quan về mảng

- **Ví dụ 4.1:** Nhập từ bàn phím và in lên màn hình trung bình cộng của n số nguyên.

- Trường hợp n=10, ta có thể khai báo 10 biến: **a, b, c, d, e, f, g, h, i, j** có kiểu **int** và lập thao tác nhập cho 10 biến này như sau:

```
Console.WriteLine("a= ");
a=int.Parse(Console.ReadLine());
```

- Với 10 biến ta sẽ thực hiện 2 lệnh trên 10 lần, sau đó tính trung bình:

$$S=(a + b + c + d + e + f + g + h + i + j)/10$$

- Phương pháp này không phù hợp với n có giá trị lớn.

MẢNG

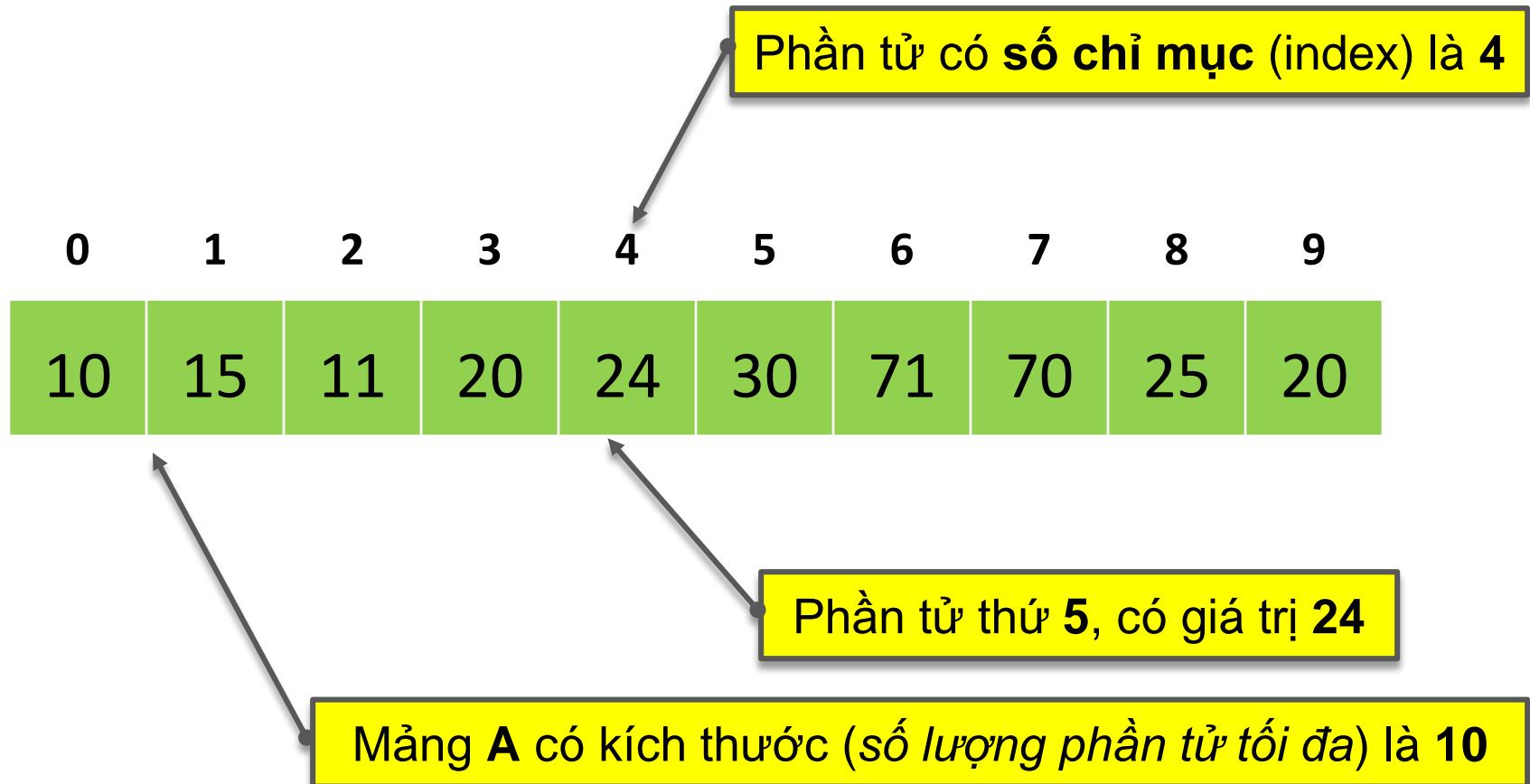
□ Tổng quan về mảng

- Mảng là một tập hợp gồm **nhiều phần tử** có **cùng một kiểu** và **chung một tên**.
- Có thể **truy xuất** đến các **phần tử** của mảng **qua số chỉ mục** (index).
- Số chỉ mục trong mảng bắt đầu từ **0 → size-1**; trong đó **size** là số lượng phần tử của mảng
- Mỗi phần tử của mảng biểu diễn được một giá trị.
- Có 2 loại mảng:
 - **Mảng 1 chiều**
 - **Mảng nhiều chiều**

MẢNG – MẢNG MỘT CHIỀU

□ Cấu trúc mảng một chiều

- Ví dụ 4.2: mảng một chiều có tên là A, lưu trữ 10 số nguyên.



MẢNG – MẢNG MỘT CHIỀU

□ Khai báo mảng:

■ Cách 1:

<Tên kiểu>[] <Tên mảng>;

<Tên mảng> = new <Tên kiểu> [size];

size là số lượng phần tử tối đa của mảng, hay còn gọi là kích thước của mảng.

Ví dụ 4.3a:

```
int[] A;
```

```
A = new int[100];
```

Khai báo cố định số phần tử

//Khai báo mảng

//Khởi tạo mảng

MẢNG – MẢNG MỘT CHIỀU

□ Khai báo mảng:

- Cách 2: vừa khai báo – vừa khởi tạo mảng

<Tên kiểu>[] <Tên mảng> = new <Tên kiểu>[size];

Ví dụ 4.3b:

```
int[] A = new int[100];
```

Khai báo cố định số phần tử

- Cách 3: cấp phát động số phần tử cho mảng

Ví dụ 4.3c:

```
int n;  
int[] A; //Khai báo mảng
```

```
n = int.Parse(Console.ReadLine());  
A= new int[n]; //Cấp phát động số phần tử
```

MẢNG – MẢNG MỘT CHIỀU

□ Khai báo mảng:

- **Ví dụ 4.4:** Khai báo một mảng có tên là **Diem**, dùng để lưu trữ điểm thi của tối đa **10** sinh viên:

- **Cách 1:**

```
float[] Diem;           //Khai báo mảng  
Diem = new float[10];   //Khởi tạo mảng
```

- **Cách 2:**

```
//Vừa khai báo vừa khởi tạo mảng  
float[] Diem = new float[10];
```

MẢNG – MẢNG MỘT CHIỀU

□ Khai báo mảng:

- **Ví dụ 4.4:** Khai báo một mảng có tên là **Diem**, dùng để lưu trữ điểm thi của tối đa **10** sinh viên:
 - **Cách 3:**

```
int n;  
float[] Diem; //Khai báo mảng
```

```
Console.WriteLine("So luong SV: ");  
n = int.Parse(Console.ReadLine());  
Diem= new float[n]; //Cấp phát động số phần tử
```

MẢNG – MẢNG MỘT CHIỀU

□ Gán giá trị cho mảng:

■ Cách 1: Khởi gán giá trị cho mảng

<Tên kiểu>[] <Tên mảng> = {gtri1, gtri2, ..., gtriN};

```
int[] A = {10, 15, 11, 20, 24, 30, 71, 70, 25, 20};
```



| [0] | [1] | [2] | [3] | [4] | [5] | [6] | [7] | [8] | [9] |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 10 | 15 | 11 | 20 | 24 | 30 | 71 | 70 | 25 | 20 |

MẢNG – MẢNG MỘT CHIỀU

□ Gán giá trị cho mảng:

- Cách 2: Gán giá trị cho từng phần tử trong mảng

<Tên mảng>[<Số chỉ mục>] = <Giá trị>;

```
int[] A = new int[10];
A[0] = 10;
A[1] = 15;
A[2] = 11;
A[3] = 20;
A[4] = 24;
A[5] = 30;
A[6] = 71;
A[7] = 70;
A[8] = 25;
A[9] = 20;
```

Phần tử **đầu tiên** có
số chỉ mục là **0**



| | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| [0] | [1] | [2] | [3] | [4] | [5] | [6] | [7] | [8] | [9] |
| 10 | 15 | 11 | 20 | 24 | 30 | 71 | 70 | 25 | 20 |

MẢNG – MẢNG MỘT CHIỀU

□ Gán giá trị cho mảng:

Ví dụ 4.5: khai báo mảng Diem, và gán giá trị điểm thi của 5 sinh viên.

//Cách 1:

```
float[] Diem = {(float)5.5,8,(float)6.5,10,7};
```

//Cách 2:

```
Diem[0] = (float)5.5;  
Diem[1] = 8;  
Diem[2] = (float)6.5;  
Diem[3] = 10;  
Diem[4] = 7;
```

MẢNG – MẢNG MỘT CHIỀU

□ Gán giá trị cho mảng:

Ví dụ 4.6: khai báo mảng HoTen, và gán giá trị họ tên của 5 sinh viên.

//Cách 1

```
string[] HoTen = {"Nguyen An", "Le Binh", "Tran Ngoc",
                  "Pham Thanh", "Nguyen Hanh"};
```

//Cách 2

```
HoTen[0] = "Nguyen An";
HoTen[1] = "Le Binh";
HoTen[2] = "Tran Ngoc";
HoTen[3] = "Pham Thanh";
HoTen[4] = "Nguyen Hanh";
```

MẢNG – MẢNG MỘT CHIỀU

- ☐ Nhập giá trị từ bàn phím cho các phần tử trong mảng
 - Ví dụ 4.7: Nhập từ bàn phím một số nguyên n và điểm thi của n sinh viên.

```
int n, i;  
float[] Diem; //Khai báo mảng
```

Sử dụng pp khai báo động

```
Console.WriteLine("So luong SV: ");  
n = int.Parse(Console.ReadLine());  
Diem= new float[n]; //Cấp phát động số phần tử
```

```
Console.WriteLine("Nhập điểm của {0} SV: ", n);  
for (i = 0; i < n;i++ )  
{  
    Console.Write(" -SV {0}: ", i + 1);  
    Diem[i] = float.Parse(Console.ReadLine());  
}
```

Nhập giá trị cho pt có số chỉ mục i

MẢNG – MẢNG MỘT CHIỀU

- Nhập giá trị từ bàn phím cho các phần tử trong mảng
 - Ví dụ 4.8: Nhập từ bàn phím Họ tên và Điểm thi của 10 sinh viên.

```
string[] HoTen = new string[10];
float[] Diem = new float[10];

for (int i = 0; i < 10; i++)
{
    Console.WriteLine("SV {0}: ", i + 1);
    Console.Write(" - Ho ten: ");
    HoTen[i] = Console.ReadLine();
    Console.Write(" - Diem: ");
    Diem[i] = float.Parse(Console.ReadLine());
}
```



```
+ SV 1:
- Ho ten: Nguyen An
- Diem: 5.5
+ SV 2:
- Ho ten: Pham Binh
- Diem: 9
+ SV 3:
- Ho ten: Le Anh
- Diem: 7.5
+ SV 4:
- Ho ten: Tran Lap
- Diem: 8
+ SV 5:
- Ho ten: ■
```

MẢNG – MẢNG MỘT CHIỀU

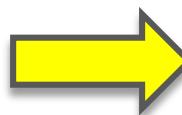
□ Truy xuất mảng:

Ví dụ 4.9: In lên màn hình điểm thi của 5 SV trong mảng **Diem**

```
int i = 0;  
float[] Diem = {(float)5.5,8,(float)6.5,10,7};  
while (i < Diem.Length)  
{  
    Console.WriteLine("Diem {0} = {1}",i+1,Diem[i]);  
    i++;  
}
```

<Tên mảng>.Length → cho
biết chiều dài của mảng

Truy xuất phần tử thứ i



```
Diem 1 = 5.5  
Diem 2 = 8  
Diem 3 = 6.5  
Diem 4 = 10  
Diem 5 = 7
```

MẢNG – MẢNG MỘT CHIỀU

□ Truy xuất mảng:

Ví dụ 4.10: In lên màn hình Họ tên và Điểm thi của 5 SV trong mảng HoTen và Diem

```
int i = 0;  
string[] HoTen = {"Nguyen An", "Le Binh", "Tran Ngoc",  
                  "Pham Thanh", "Nguyen Hanh"};  
float[] Diem = { (float)5.5, 8, (float)6.5, 10, 7 };  
  
while (i < HoTen.Length)  
{  
    Console.WriteLine("{0}. Ho ten: {1} - Diem = {2}",  
                      i + 1, HoTen[i], Diem[i]);  
    i++;  
}
```



```
1. Ho ten: Nguyen An - Diem = 5.5  
2. Ho ten: Le Binh - Diem = 8  
3. Ho ten: Tran Ngoc - Diem = 6.5  
4. Ho ten: Pham Thanh - Diem = 10  
5. Ho ten: Nguyen Hanh - Diem = 7
```

MẢNG – MẢNG MỘT CHIỀU

□ Cấu trúc lặp **foreach**

- Cú pháp:

**foreach (<Tên kiểu> <Tên biến> in <Tên mảng>)
<Lệnh>/<Khối lệnh>**

- **Chức năng:** duyệt qua từng phần tử trong mảng

MẢNG – MẢNG MỘT CHIỀU

□ Cấu trúc lặp **foreach**

■ Ví dụ 4.11:

```
int[] A = {1,2,3,4,5};  
//Cach 1:  
Console.WriteLine("Cach 1:");  
for (int i = 0; i < 5; i++)//A.Length hoặc A.GetLength(0)  
    Console.WriteLine(A[i]);  
  
//Cach 2:  
Console.WriteLine("Cach 2:");  
foreach (int So in A)  
    Console.WriteLine(So);
```

Cach 1:

1
2
3
4
5

Cach 2:

1
2
3
4
5



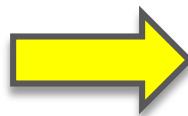
MẢNG – MẢNG MỘT CHIỀU

□ Cấu trúc lặp **foreach**

- Ví dụ 4.11:

```
string[] HoTen = {"Nguyen An", "Le Binh", "Tran Ngoc",
                  "Pham Thanh", "Nguyen Hanh"};
```

```
foreach (string ten in HoTen)
{
    Console.WriteLine("Ho ten: " + ten);
}
```



```
Ho ten: Nguyen An
Ho ten: Le Binh
Ho ten: Tran Ngoc
Ho ten: Pham Thanh
Ho ten: Nguyen Hanh
```

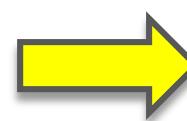
MẢNG – MẢNG MỘT CHIỀU

□ Cấu trúc lặp **foreach**

▪ Ví dụ 4.12:

```
int i = 0;  
float[] Diem = { (float)5.5, 8, (float)6.5, 10, 7 };  
foreach (float d in Diem)  
{  
    Console.WriteLine("Diem {0} = {1}", i+1, d);  
    i++;  
}
```

Biến **d** có kiểu **float**, sẽ được gán bằng giá trị của các phần tử trong mảng **Diem**



```
Diem 1 = 5.5  
Diem 2 = 8  
Diem 3 = 6.5  
Diem 4 = 10  
Diem 5 = 7
```

MẢNG – MẢNG MỘT CHIỀU

□ Sao chép mảng

- Hàm Array.Copy()

Array.Copy(<Mảng nguồn>, <Mảng đích>, <Số phần tử>)

- **<Mảng nguồn>**: mảng cần copy
- **<Mảng đích>**: mảng mới được tạo ra
- **<Số phần tử>**: số phần tử cần sao chép từ **<Mảng nguồn>** sang **<Mảng đích>**, lớn nhất là bằng kích thước của **<Mảng nguồn>**

MẢNG – MẢNG MỘT CHIỀU

□ Sao chép mảng

- Cách 1: Sử dụng hàm Array.Copy()

- Ví dụ 4.13:

```
int[] A = { 5, 10, 15, 20, 25 };  
int[] B = new int[5];
```

```
Array.Copy(A, B, 5); //Sao chép toàn bộ Mảng A --> B
```

```
Console.WriteLine("Mang A: ");  
foreach (int x in A)  
    Console.Write(x + " ");
```



```
Console.WriteLine("Mang B: ");  
foreach (int x in B)  
    Console.Write(x + " ");
```

```
Mang A: 5 10 15 20 25  
Mang B: 5 10 15 20 25
```

MẢNG – MẢNG MỘT CHIỀU

□ Sao chép mảng

- Cách 2: Sao chép từng phần tử trong mảng

- Ví dụ 4.14:

```
int[] A = { 5, 10, 15, 20, 25 };
```

```
int[] B = new int[5];
```

```
for (int i = 0; i < A.Length; i++)  
    B[i] = A[i]; //Sao chép từng phần tử A --> B
```



MẢNG – MẢNG MỘT CHIỀU

□ Sao chép mảng

- Phép tham chiếu mảng bằng phép gán

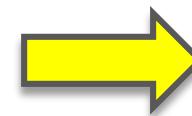
- Ví dụ 4.14:

```
int[] A = { 5, 10, 15, 20, 25 };
```

```
int[] B;
```

```
B = A;
```

```
foreach (int x in B)  
    Console.WriteLine(x + " ");
```



```
5 10 15 20 25
```

Lưu ý:

- Tham chiếu là tạo ra một biến mảng thứ 2, nhưng cùng trỏ đến cùng không gian lưu trữ của mảng ban đầu;
- Mọi tác động trên mảng B, đều ảnh hưởng đến mảng A.

MẢNG – MẢNG MỘT CHIỀU

□ Sao chép mảng

- Phép tham chiếu mảng bằng phép gán

- Ví dụ 4.15:

```
int[] A = { 5, 10, 15, 20, 25 };  
int[] B = new int[5];
```

```
B = A;  
B[0] = 100;  
Console.WriteLine("Mang A: ");  
foreach (int x in A)  
    Console.Write(x + " ");
```

```
Console.WriteLine();  
Console.WriteLine("Mang B: ");  
foreach (int x in B)  
    Console.Write(x + " ");
```

| |
|-------------------------|
| Mang A: 100 10 15 20 25 |
| Mang B: 100 10 15 20 25 |

MẢNG – MẢNG MỘT CHIỀU – BÀI TẬP

Bài 1. Nhập vào một số nguyên n , nhập liên tục từ bàn phím n số nguyên và lưu trữ vào một mảng A. In lên màn hình tập hợp các số nguyên trong mảng A theo thứ tự ngược lại.

MẢNG – MẢNG MỘT CHIỀU – BÀI TẬP

Bài 2. Nhập liên tục một dãy các số nguyên, việc dừng lại khi số nhập vào là 0. Lưu vào mảng A những số nguyên dương, mảng B những số nguyên âm. In lên màn hình hai danh sách của các tập hợp được lưu trữ trong A và B.

MẢNG – MẢNG MỘT CHIỀU – BÀI TẬP

Bài 3. Nhập vào ba số nguyên n, X và Y, nhập liên tục từ bàn phím n số nguyên và lưu trữ vào một mảng A.

- Cho biết **có bao nhiêu lần xuất hiện số nguyên X** trong tập hợp A
- **Tìm và thay thế** toàn bộ những phần tử bằng X trong tập hợp A bằng Y
- **In lên màn hình** tập hợp mới sau khi được thay thế.

MẢNG – MẢNG HAI CHIỀU

□ Cấu trúc mảng hai chiều

| | | j cột | | | | | | | A[i , j] |
|--------|---|-------|---|----|---|----|---|----|----------|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | |
| i dòng | 0 | 5 | 7 | 2 | 1 | 10 | 4 | 3 | |
| | 1 | 1 | 2 | 3 | 5 | 9 | 2 | 6 | |
| | 2 | 6 | 4 | 5 | 2 | 3 | 7 | 5 | |
| | 3 | 1 | 9 | 10 | 7 | 8 | 8 | 2 | |
| | 4 | 2 | 6 | 5 | 3 | 2 | 1 | 10 | |

MẢNG – MẢNG HAI CHIỀU

□ Khai báo mảng hai chiều

■ Cách 1:

`<Tên kiểu>[,] <Tên mảng>;`

`<Tên mảng> = new <Tên kiểu> [size1, size2];`

■ Cách 2:

`<Tên kiểu>[,] <Tên mảng>=new<Tên kiểu>[size1, size2];`

size1: số dòng

size2: số cột

MẢNG – MẢNG HAI CHIỀU

□ Khai báo mảng hai chiều

- Cách 3: Cấp phát động kích thước mảng

```
int size1, size2;  
int[,] A; //Khai báo mảng
```

```
size1=int.Parse(Console.ReadLine()); //Chiều 1 - Dòng  
size2=int.Parse(Console.ReadLine()); //Chiều 2 - cột  
A = new int[size1, size2];
```

MẢNG – MẢNG HAI CHIỀU

□ Khởi gán giá trị cho mảng hai chiều

■ Ví dụ 4.16:

<Tên mảng>.GetLength(<Số chiều>) →
Trả về số phần tử ở Chiều tương ứng

```
int[,] A = { { 1, 2, 3, 4 }  
            , { 4, 3, 2, 1 }  
            , { 1, 2, 3, 4 } };
```

```
for (int i = 0; i < A.GetLength(0); i++) //Duyệt qua từng Dòng  
{  
    for (int j = 0; j < A.GetLength(1); j++) //Duyệt qua từng Cột  
        Console.WriteLine(A[i, j] + " ");  
    Console.WriteLine();  
}
```



| | | | |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| 4 | 3 | 2 | 1 |
| 1 | 2 | 3 | 4 |

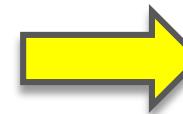
MẢNG – MẢNG HAI CHIỀU

□ Nhập giá trị cho từng phần tử trong mảng

- Ví dụ 4.17: nhập từ bàn phím giá trị của các phần tử trong ma trận 3x4

```
int[,] A = new int[3, 4];  
  
for (int i = 0; i < 3; i++) //Duyệt qua từng Dòng  
    for (int j = 0; j < 4; j++) //Duyệt qua từng Cột  
    {  
        Console.WriteLine("Nhập A[{0},{1}] = ", i, j);  
        A[i, j] = int.Parse(Console.ReadLine());  
    }
```

Nhập giá trị cho phần tử **A[i , j]**



```
Nhập A[0,0] = 1  
Nhập A[0,1] = 2  
Nhập A[0,2] = 3  
Nhập A[0,3] = 4  
Nhập A[1,0] = 4  
Nhập A[1,1] = 3  
Nhập A[1,2] = 2  
Nhập A[1,3] = 1  
Nhập A[2,0] = 1  
Nhập A[2,1] = 2  
Nhập A[2,2] = 3  
Nhập A[2,3] = 4
```

MẢNG – CÁC HÀM THAO TÁC TRÊN MẢNG

☐ Các hàm thuộc class: **System.Array**

| | |
|----------------------|---|
| Clear() | Hàm tĩnh này gán giá trị mặc định (theo kiểu dữ liệu của mảng) cho tất cả các thành phần của mảng |
| Copy() | Hàm cho phép sao chép một phần của mảng này lên mảng khác |
| CopyTo() | Hàm này được sử dụng để sao chép các thành phần của mảng nguồn sang mảng đích |
| GetLength() | Hàm này trả về số lượng thành phần trên một chiều chỉ định của mảng |
| IndexOf() | Hàm này trả về chỉ số của giá trị xuất hiện đầu tiên trong mảng một chiều |
| LastIndexOf() | Hàm trả về chỉ số của giá trị xuất hiện cuối cùng trên mảng một chiều |
| Length | Thuộc tính này trả ra số lượng thành phần của mảng |
| Rank | Thuộc tính này trả ra số chiều của mảng |
| Reverse() | Hàm tĩnh này đảo ngược thứ tự các thành phần của một mảng một chiều |
| Sort() | Hàm tĩnh này sắp xếp một mảng một chiều kiểu cơ bản |

MẢNG – MẢNG HAI CHIỀU – BÀI TẬP

Bài 1. Nhập từ bàn phím hai số nguyên n và m ; nhập giá trị cho hai ma trận **A** và **B** có hai chiều $n \times m$. In lên giá trị tổng của hai ma trận **A** và **B**.

Bài 2. Nhập điểm thi hai môn **Toán** và **Lý** của n học sinh ($n \leq 50$). Tính và in ra điểm trung bình học tập của các học sinh trên. Trong đó n được nhập vào từ bàn phím.

BÀI GIẢNG

CƠ SỞ LẬP TRÌNH

CHƯƠNG 6. HÀM

NGUYỄN THÀNH THỦY
BỘ MÔN TIN HỌC QUẢN LÝ
TRƯỜNG ĐẠI HỌC KINH TẾ, ĐẠI HỌC ĐÀ NẴNG
THUYNT@DUE.EDU.VN

NỘI DUNG

- Khái niệm về Hàm
- Khai báo Hàm
- Lời gọi Hàm
- Truyền giá trị từ ngoài vào trong Hàm
- Truyền giá trị từ trong ra ngoài Hàm

KHÁI NIỆM VỀ HÀM

□ Bài toán:

Tính tổng của n số nguyên dương đầu tiên:

- Nhập từ bàn phím một số nguyên **n**
- Tính tổng **S** của **n** số nguyên đầu tiên
$$S=1+2+\dots+n$$
- In lên màn hình tổng **S**

KHÁI NIỆM VỀ HÀM

■ Lời giải không sử dụng hàm

```
static void Main(string[] args)
{
    int n, S;

    Nhap(n) { //Nhập liệu
        Console.WriteLine("Nhập n= ");
        n = int.Parse(Console.ReadLine());
    }

    Tinh(n) { //Tính tổng
        S = 0;
        for (int i = 1; i <= n; i++)
            S += i;
    }

    InKQ(S) { //In kết quả
        Console.WriteLine("Tổng S= " + S);
        Console.ReadKey();
    }
}
```

KHÁI NIỆM VỀ HÀM

□ Phân tích bài toán để chuyển vào hàm

- Hàm Nhập(n):
 - **Input:** không có
 - **Output:** số nguyên n
 - **Process:** cho phép người dùng nhập từ bàn phím một số nguyên và lưu vào biến n;
- Hàm Tính(n):
 - **Input:** số nguyên n
 - **Output:** trả kết quả về qua tên hàm
 - **Process:** thực hiện tính tổng của n số nguyên dương đầu tiên;
- Hàm InKQ(S):
 - **Input:** số nguyên S
 - **Output:** kết quả được in lên màn hình
 - **Process:** In lên màn hình giá trị của biến S;

KHÁI NIỆM VỀ HÀM

- Chương trình hoàn chỉnh sử dụng hàm

```
class Program
{
    static void Main(string[] args)
    {
        int n, S;

        Nhap(out n);
        S=Tinh(n);
        InKQ(S);
    }

    private static void Nhap(out int n)
    {
        //Nhập liệu
        Console.WriteLine("Nhap n= ");
        n = int.Parse(Console.ReadLine());
    }
}
```

Còn tiếp >>

KHÁI NIỆM VỀ HÀM

- Chương trình hoàn chỉnh sử dụng hàm

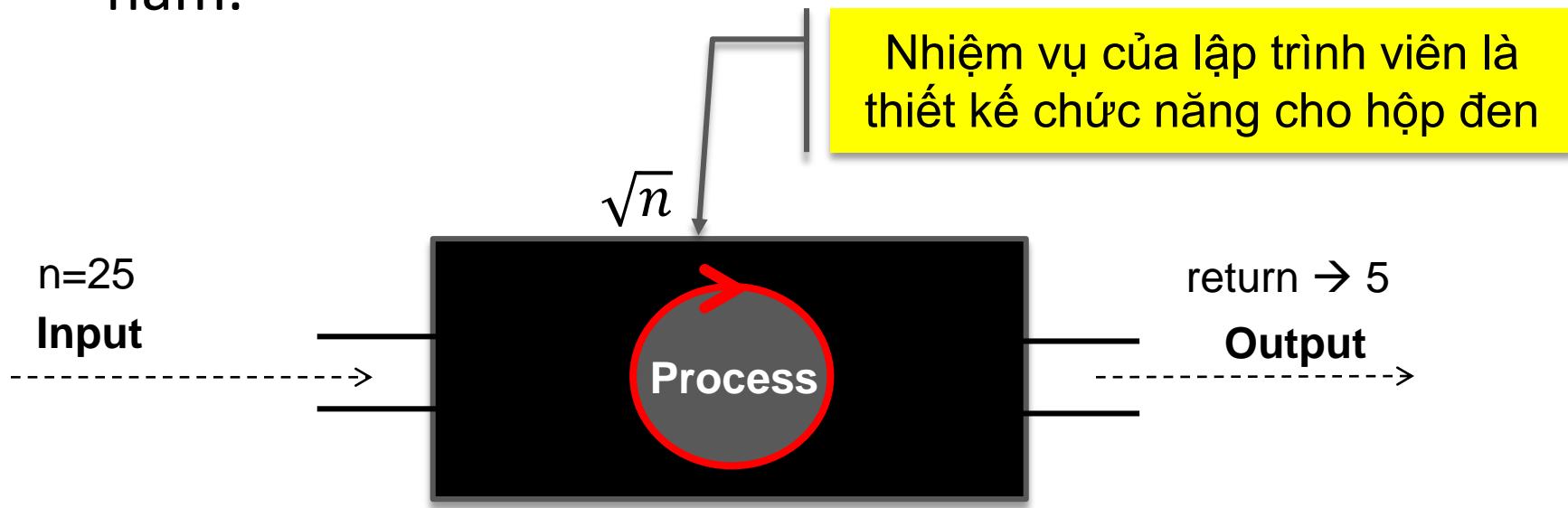
```
{ private static int Tinh(int n)
{
    //Tính tổng
    int S = 0;
    for (int i = 1; i <= n; i++)
        S += i;
    return S;
}
private static void InKQ(int s)
{
    //In kết quả
    Console.WriteLine("Tong S= " + s);
    Console.ReadKey();
}
```

KHÁI NIỆM VỀ HÀM

- Ngoài các hàm chuẩn (do C# định nghĩa sẵn), hệ thống còn cho phép người dùng tự thiết kế ra những hàm theo mục đích riêng;
- Hàm chia các bài toán lớn thành các công việc nhỏ hơn, giúp thực hiện những công việc lặp lại nào đó một cách nhanh chóng mà không phải viết lại đoạn chương trình;
- Trong chương trình C#, có một hàm đặc biệt là hàm main();
- Thứ tự các hàm trong chương trình là bất kỳ, song chương trình bao giờ cũng đi thực hiện từ hàm main().

KHÁI NIỆM VỀ HÀM

- **Ví dụ:** hàm **Math.Sqrt(n)**, thực hiện tính căn bậc hai của số nguyên n. Trong đó n là tham số của hàm.



Hàm được ví như một cái hộp đen (**Black Box**) đối với người sử dụng:

- Có hoặc không có dữ liệu đầu vào (**input**)
- Thực hiện (**process**) một yêu cầu cụ thể gì đó khi được gọi (Call)
- Kết quả xử lý được trả về gọi là đầu ra (**output**)

KHAI BÁO HÀM

□ Cú pháp khai báo hàm:

[<Phạm_vi>][static]<Kiểu><Tên_Hàm>([<DS/Tham số>])

{

Lệnh/Khối lệnh

}

■ Trong đó:

- <Phạm_vi>: Cho biết phạm vi có thể tiếp cận đến hàm (*xem thêm trang sau*)
- <Kiểu>: Kiểu giá trị của kết quả trả về qua tên hàm
- <Tên_Hàm>: Do người lập trình tự đặt
- <DS/Tham số>: Các tham số sẽ được truyền qua hàm (*xem thêm ở trang sau*)

KHAI BÁO HÀM

□ Các loại <Phạm_vi> : (Access Modifier)

- **private**: Truy cập bị giới hạn trong phạm vi của class chứa nó. Nếu để trống thì mặc định là **private**.
- **protected**: Truy cập bị giới hạn trong phạm vi của class chứa nó và bất kỳ class con **thừa kế** từ class này.
- **internal**: Truy cập bị giới hạn trong phạm vi của **Solution** hiện tại.
- **protected internal**: Truy cập bị giới hạn trong phạm vi của Solution hiện tại và trong class định nghĩa hoặc các class con.
- **public**: Không có bất kỳ giới hạn nào khi truy cập vào các thành viên công khai (public).

Đọc thêm >> <http://o7planning.org/vi/10439/access-modifier-trong-csharp#a1291528>

KHAI BÁO HÀM

□ <DS/Tham số> (*danh sách tham số*)

- Cú pháp:

[ref/out]<Kiểu> <Biến>, [ref/out] <Kiểu> <Biến>,...

Ví dụ:

ref int b, out float c, int a

- Ý nghĩa:

- Là các biến: mang giá trị từ bên ngoài VÀO bên trong hàm (input), hoặc mang giá trị từ bên trong RA bên ngoài hàm (output).
- **ref**: mang dữ liệu VÀO và RA
- **out**: chỉ mang dữ liệu RA
- **Để trống**: chỉ mang dữ liệu VÀO

(sẽ được trao đổi thêm trong phần truyền giá trị cho hàm)

KHAI BÁO HÀM

□ Ví dụ 6.1a: Khai báo hàm Nhap(n)

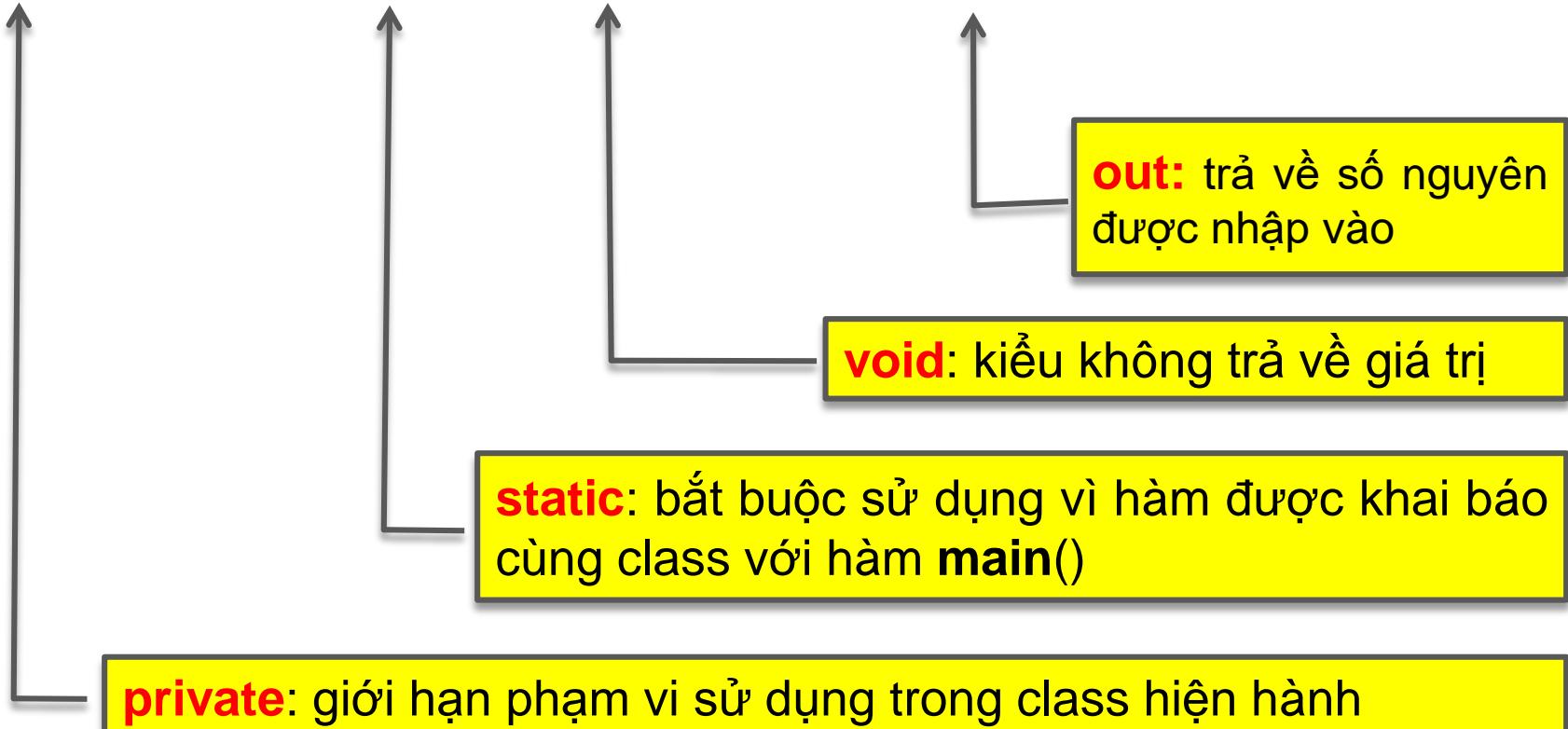
- **Input:** không
- **Output:** số nguyên n
- **Process:** cho phép người dùng nhập từ bàn phím một số nguyên và lưu vào tham số n;

KHAI BÁO HÀM

□ Ví dụ 6.1a: Khai báo hàm Nhap(n)

- Khai báo phần **header** của hàm như sau:

```
private static void Nhap(out int n)
```



KHAI BÁO HÀM

□ Ví dụ 6.1a: Khai báo hàm Nhap(n)

- Hàm được viết đầy đủ như sau:

```
private static void Nhap(out int n)
{
    {
        //Nhập liệu
        Console.WriteLine("Nhập n= ");
        n = int.Parse(Console.ReadLine());
    }
}
```

Phần **thân** (body) của hàm

Phần **đầu** (header) của hàm

KHAI BÁO HÀM

□ Ví dụ 6.1b: Khai báo hàm Tinh(n)

- **Input:** số nguyên n
- **Output:** kết quả được trả về qua tên hàm, tổng n số nguyên dương đầu tiên;
- **Process:** thực hiện tính tổng của n số nguyên đầu tiên;

KHAI BÁO HÀM

□ Ví dụ 6.1b: Khai báo hàm Tinh(n)

```
private static int Tinh(int n)
{
    //Tính tổng
    int S = 0;
    for (int i = 1; i <= n; i++)
        S += i;
    return S;
}
```

- *Hàm kiểu khác void, phải có ít nhất 1 từ khóa return;*

KHAI BÁO HÀM

□ Ví dụ 6.1c: Khai báo hàm InKQ(S)

- **Input:** số nguyên S
- **Output:** kết quả được xuất lên màn hình;
- **Process:** In lên màn hình giá trị của biến S;

KHAI BÁO HÀM

□ Ví dụ 6.1c: Khai báo hàm InKQ(s)

```
private static void InKQ(int s)
{
    //In kết quả
    Console.WriteLine("Tong S= " + s);
    Console.ReadKey();
}
```

S chỉ truyền tham số vào nên
không sử dụng ref hoặc out

KHAI BÁO HÀM

☐ Một số lưu ý quan trọng

- Muốn đưa vào/ lấy ra giá trị từ hàm thì: **sử dụng tham số hoặc qua tên hàm** (lấy ra/output);
- Sử dụng **ref** khi: **tham số** đồng thời **input** và **output** giá trị từ hàm;
- Sử dụng **out** khi: **tham số** chỉ **output** giá trị từ hàm;
- Để trống: **tham số** chỉ **input** giá trị từ hàm;
- Một hàm có thể không có tham số nào;
- Vị trí đặt code khai báo hàm:
 - Bên ngoài và dưới hàm main()
 - Hoặc trong một file class khác

KHAI BÁO HÀM

☐ Một số lưu ý quan trọng

- Kiểu của hàm phụ thuộc vào Output của hàm:
 - Hàm có kiểu **void**:
 - Kết quả được trả về qua tham số
 - Kết quả được in lên màn hình
 - Kết quả được ghi ra tệp tin
 - Hàm có kiểu **KHÁC void** (int, float, char,...):
 - Kết quả được trả về qua **tên hàm**

KHAI BÁO HÀM

□ Bài tập ôn tập

- **Bài toán:** Sử dụng hàm để thực hiện các yêu cầu sau: Nhập từ bàn phím một số nguyên n , nhập liên tục từ bàn phím n số nguyên. Đếm và in lên màn hình có bao nhiêu chữ số chẵn đã được nhập vào.
 - Xác định các hàm cần thiết phải xây dựng và các thông tin: **input**, **process** và **output** cho từng hàm;
 - Viết đầy đủ nội dung của từng hàm;

LỜI GỌI HÀM

□ Ý nghĩa:

- Khi cần sử dụng một hàm đã định nghĩa, ta cần thực hiện lời gọi hàm;
- Vị trí đặt code để gọi hàm là bên trong hàm main() hoặc trong một hàm khác (hàm gọi hàm);
- Hàm được gọi thông qua tên hàm, theo sau là danh sách tham số (nếu có);
- Sử dụng cú pháp hai ngôi (có phép gán =) khi hàm trả kết quả qua tên hàm (khác kiểu void).

LỜI GỌI HÀM

- **Ví dụ 6.2:** gọi hàm trong hàm Main()

```
static void Main(string[] args)
{
    int n, s;
    Nhap(out n);
    S=Tinh(n);
    InKQ(S);
}
```

private static void **Nhap(out int n)**

private static int **Tinh(int n)**

private static void **InKQ(int S)**

LỜI GỌI HÀM

- **Ví dụ 6.3:** gọi hàm trong một hàm khác hàm

```
static void Main(string[] args)
{
    BaiToanSo1(); ←
}

private static void BaiToanSo1()
{
    int n, s; →
    Nhap(out n);
    s = Tinh(n);
    InKQ(s);
}
```

Cần phải gọi hàm **BaiToanSo1()** trong hàm **Main()** để kích hoạt chương trình

Các hàm **Nhap()**, **Tinh()**, **InKQ()** được gọi trong hàm **BaiToanSo1()**

LỜI GỌI HÀM

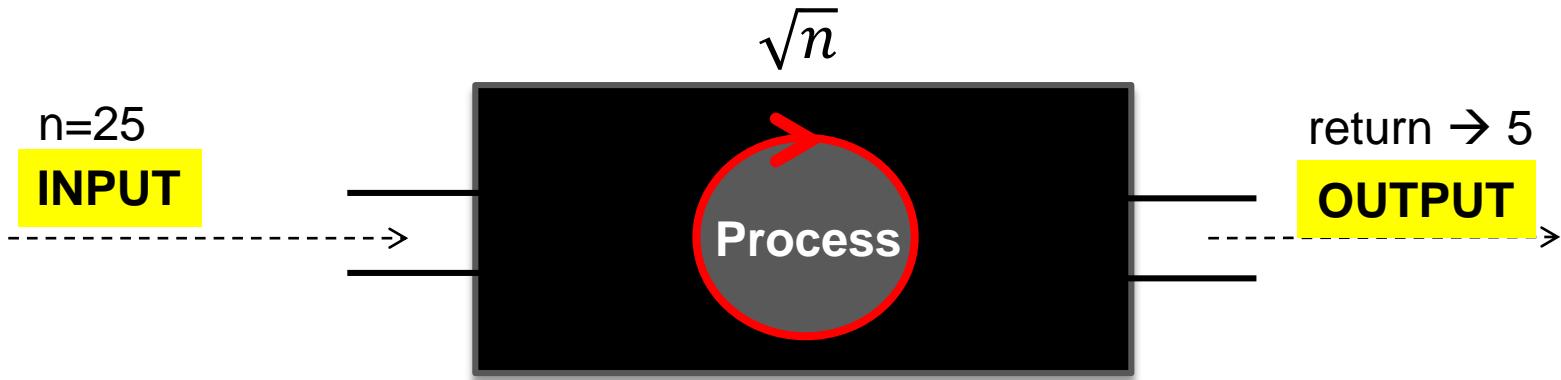
□ Bài tập ôn tập

- Thực hiện lời gọi hàm và để hoàn thiện Bài tập tại Slide 22

TRUYỀN GIÁ TRỊ CHO HÀM

□ Truyền giá trị cho hàm là gì?

- Là phương pháp đưa giá trị từ bên ngoài vào bên trong hàm, gọi tắt là **INPUT**
- Lấy giá trị từ bên trong truyền ra bên hàm, gọi tắt là **OUTPUT**

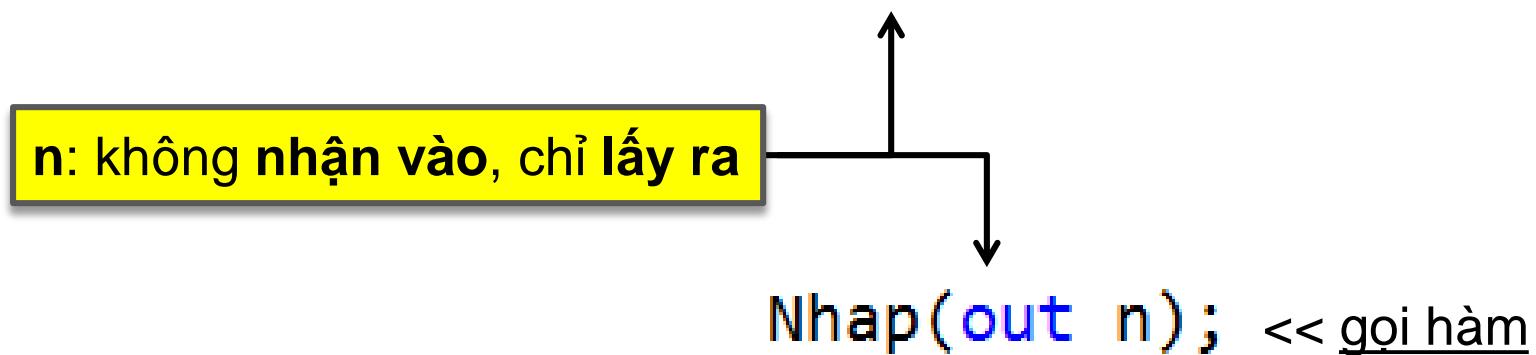


TRUYỀN GIÁ TRỊ CHO HÀM

□ Phương pháp 1: Sử dụng tham số của hàm

- Bổ sung từ khóa khi khai báo hàm
 - **out**: không nhận vào, chỉ lấy ra
 - **ref**: nhận vào và lấy ra
 - **Để trống**: chỉ nhận vào, không lấy ra

```
private static void Nhap(out int n) << khai báo
```



TRUYỀN GIÁ TRỊ CHO HÀM

□ Phương pháp 1: Sử dụng tham số của hàm

```
private static void InKQ(int s) << khai báo
```

S: chỉ nhận vào, không lấy ra

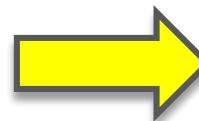
```
InKQ(S); << gọi hàm
```

TRUYỀN GIÁ TRỊ CHO HÀM

□ Phương pháp 1: Sử dụng tham số của hàm

- Bài toán hoán vị, sử dụng **ref**

```
static void Main(string[] args)
{
    int x = 5, y = 10;
    HoanVi(ref x, ref y);
    InKQ(x, y);
}
static void HoanVi(ref int x, ref int y)
{
    int z;
    z = x;
    x = y;
    y = z;
}
```



x=10, y=5

TRUYỀN GIÁ TRỊ CHO HÀM

□ Phương pháp 2: Trả giá trị qua tên hàm

- Hàm kiểu khác **void**
- Trong thân hàm có ít nhất 1 từ khóa **return** để trả kết quả

```
private static int Tinh(int n)
{
    //Tính tổng
    int S = 0;
    for (int i = 1; i <= n; i++)
        S += i;
    return S;
}
```

Kiểu khác **void**, kết quả sẽ được trả về qua tên hàm

return <x>; kết thúc hàm và trả giá trị **<x>** về qua tên hàm

TRUYỀN GIÁ TRỊ CHO HÀM

□ Phương pháp 2: Trả giá trị qua tên hàm

- Lời gọi hàm phải ở dạng biểu thức hai ngôi
- Vết phải: là tên hàm cần gọi, kèm tham số (nếu có)
- Vết trái: là 1 biến, sẽ lưu trữ giá trị là kết quả của hàm ở vết phải;

S = Tinh(n);

TRUYỀN GIÁ TRỊ CHO HÀM

□ Phương pháp 3: Sử dụng biến toàn cục

```
static int x = 5, y = 10;  
static void Main(string[] args)  
{  
    HoanVi();  
    InKQ(x, y);  
}  
static void HoanVi()  
{  
    int z;  
    z = x;  
    x = y;  
    y = z;  
}
```

Khai báo x, y là biến toàn cục



x=10, y=5

TRUYỀN GIÁ TRỊ CHO HÀM

□ Phạm vi của biến trong và ngoài hàm:

- .NET cấp phát cho mỗi hàm một vùng nhớ có kích thước cố định;
- Các biến được tạo ra trong hàm nào có phạm vi hoạt động trong hàm đó;
- Khi hết phạm vi của hàm thì biến đó bị huỷ (*không sử dụng được*);
- Hai hàm khác nhau, không thể sử dụng biến lẫn nhau;

TRUYỀN GIÁ TRỊ CHO HÀM

Bài 1. Viết hàm thực hiện tính tổng của n số nguyên dương đầu tiên. Trong đó kết quả tính toán được trả ra ngoài bằng 5 cách:

- a. Cách 1: Qua tham số (**TongA()**)
- b. Cách 2: Qua tên hàm (**TongB()**)
- c. Cách 3: Biến toàn cục (**TongC()**)
- d. Cách 4: Kết quả được in lên màn hình (**TongD()**)
- e. Cách 5: Kết quả được ghi ra một tệp tin (**TongE()**)

TRUYỀN GIÁ TRỊ CHO HÀM

Bài 2. Viết hàm thực hiện tính tổng của hai số nguyên a và b bất kỳ. Trong đó kết quả tính toán được trả ra ngoài bằng 5 cách:

- a. **Cách 1:** Qua tham số (**TongA()**)
- b. **Cách 2:** Qua tên hàm (**TongB()**)
- c. **Cách 3:** Biến toàn cục (**TongC()**)
- d. **Cách 4:** Kết quả được in lên màn hình (**TongD()**)
- e. **Cách 5:** Kết quả được ghi ra một tệp tin (**TongE()**)

TRUYỀN GIÁ TRỊ CHO HÀM

Bài 3. Viết hàm thực hiện kiểm tra một số nguyên X có phải là **số chẵn** hay không. Trả về 0 nếu X là số chẵn, trả về 1 nếu X là số lẻ.

Trong đó kết quả tính toán được trả ra ngoài bằng 5 cách:

- a. **Cách 1:** Qua tham số (**LaSoChanA(x)**)
- b. **Cách 2:** Qua tên hàm (**LaSoChanB(x)**)
- c. **Cách 3:** Biến toàn cục (**LaSoChanC(x)**)
- d. **Cách 4:** Kết quả được in lên màn hình (**LaSoChanD(x)**)
- e. **Cách 5:** Kết quả được ghi ra một tệp tin

HÀM CHỒNG

- ☐ Hàm chồng (Operator overloading) là những hàm trùng tên nhau, nhưng khác nhau về tham số (số lượng, kiểu,...)

```
static void Nhap(out int a)
{
    Console.Write("Nhập số nguyên: ");
    a = int.Parse(Console.ReadLine());
}

static void Nhap(out float a)
{
    Console.Write("Nhập số thực: ");
    a = float.Parse(Console.ReadLine());
}

static void Nhap(out int a, out float b)
{
    Console.Write("Nhập số nguyên: ");
    a = int.Parse(Console.ReadLine());
    Console.Write("Nhập số thực: ");
    b = float.Parse(Console.ReadLine());
}
```

BÀI TẬP ÔN TẬP CHƯƠNG 6

Bài 1. Viết hàm thực hiện tính bình phương của số nguyên n . Hàm có tên và tham số như sau:

a. **BinhPhuong(n)**

- Kết quả được in lên màn hình

b. **BinhPhuong(n)**

- Kết quả được trả về qua tên của hàm

c. **BinhPhuong(S,n)**

- Kết quả sẽ được trả về qua tham số S

BÀI TẬP ÔN TẬP CHƯƠNG 6

Bài 2. Viết chương trình thực hiện các yêu cầu sau:

- Nhập vào từ bàn phím hai số nguyên **a** và **b**
 - Tính tổng bình phương của **a** và **b**
- a. Hàm **TBP(a,b)**: kết quả được in lên màn hình
- b. Hàm **TBP(a,b,kq)**: tham số kq sẽ mang kết quả
- c. Hàm **TBP(a,b)**: kết quả được trả về qua tên hàm.

BÀI GIẢNG

CƠ SỞ LẬP TRÌNH

CHƯƠNG 7.

LÀM VIỆC VỚI TẬP TIN

NGUYỄN THÀNH THỦY

BỘ MÔN TIN HỌC QUẢN LÝ

TRƯỜNG ĐẠI HỌC KINH TẾ, ĐẠI HỌC ĐÀ NẴNG

THUYNT@DUE.EDU.VN

NỘI DUNG

- **Làm việc với tập tin**
 - **Ghi dữ liệu ra tập tin**
 - **Đọc dữ liệu từ tập tin**

LÀM VIỆC VỚI TẬP TIN

□ Ghi dữ liệu ra tập tin

- Các bước thực hiện
 - **Bước 1: Tạo mới tập tin**
 - **Bước 2: Ghi dữ liệu**
 - **Bước 3: Đóng tập tin**

- Khai báo thư viện:
`using System.IO;`

LÀM VIỆC VỚI TẬP TIN

□ Tạo mới tập tin

- Cú pháp

StreamWriter <Biến>= new StreamWriter(<Tên tập tin>);

- Ví dụ

//Mở tập tin

```
StreamWriter file = new StreamWriter("data.txt");
```

- Lưu ý: chương trình sẽ ghi đè dữ liệu vào tập tin, nếu tập tin này đã tồn tại.

LÀM VIỆC VỚI TẬP TIN

□ Ghi dữ liệu vào tập tin

■ Cú pháp

- Ghi dữ liệu vào tập tin và không xuống dòng

<Biến>.Write(<Nội dung cần ghi>);

- Ghi dữ liệu vào tập tin và sau đó xuống dòng

<Biến>.WriteLine(<Nội dung cần ghi>);

- Thuộc tính chỉ định xuống dòng và về đầu dòng

<Biến>.NewLine

LÀM VIỆC VỚI TẬP TIN

□ Ghi dữ liệu vào tập tin

```
// Ghi dữ liệu vào tập tin
file.WriteLine("Dãy 10 chữ số, mỗi số nằm trên một dòng:");
for(int i = 1; i <= 10; i++)
    file.Write(i + file.NewLine);
```

LÀM VIỆC VỚI TẬP TIN

□ Đóng tập tin

- Cú pháp

<Biến>.Close();

- Ví dụ

```
// Đóng tập tin  
file.Close();
```

LÀM VIỆC VỚI TẬP TIN

- **Ví dụ:** Tạo một dãy số từ 1 đến 10, mỗi chữ số nằm trên 1 dòng. Ghi kết quả vào tập tin có tên data.txt.

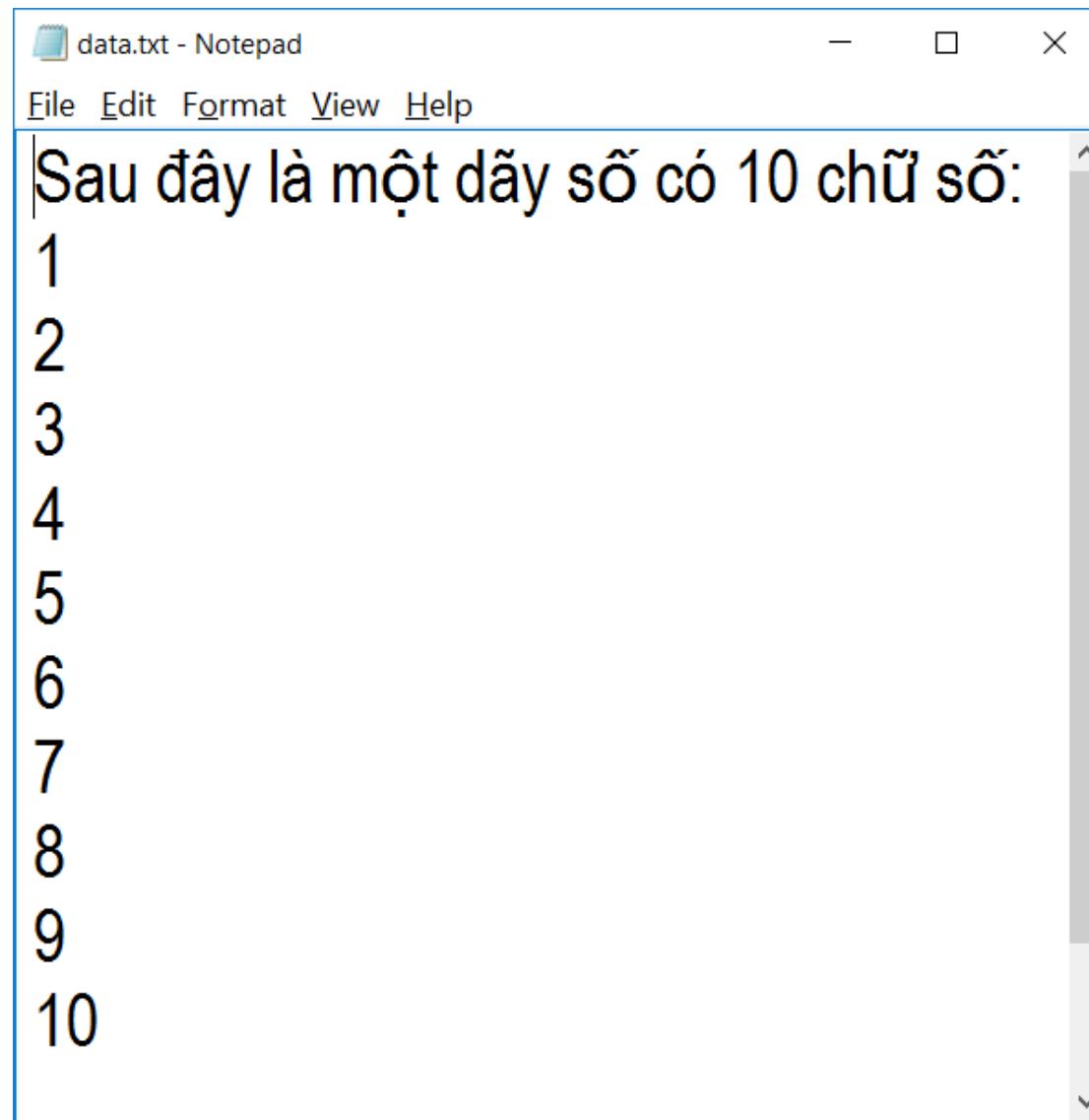
```
StreamWriter file = new StreamWriter("data.txt");
if(file == null) //Lỗi khi tạo mới tập tin
    return;

// Ghi dữ liệu vào tập tin
file.WriteLine("Dãy 10 chữ số, mỗi số nằm trên một dòng:");
for(int i = 1; i <= 10; i++)
    file.Write(i + file.NewLine);

// Đóng tập tin
file.Close();
```

LÀM VIỆC VỚI TẬP TIN

☐ Ví dụ:



LÀM VIỆC VỚI TẬP TIN

- **Đọc dữ liệu từ tập tin**
 - Các bước thực hiện
 - **Bước 1: Mở tập tin**
 - **Bước 2: Đọc dữ liệu**
 - **Bước 3: Đóng tập tin**

LÀM VIỆC VỚI TẬP TIN

□ Mở tập tin

- Cú pháp

```
StreamReader <Biến>= new StreamReader(<Tên tập tin>);  
if (<Biến> == null) return; //Dừng nếu có lỗi
```

- Ví dụ

```
// Mở tập tin  
StreamReader file = new StreamReader("data.txt");
```

LÀM VIỆC VỚI TẬP TIN

□ Đọc dữ liệu

- Đọc từng dòng trong file

<Biến_Kiểu_Chuỗi> = <Biến>.ReadLine();

- Đọc toàn bộ file

<Biến_Kiểu_Chuỗi> = <Biến>.ReadToEnd();

- Ví dụ

```
//Đọc từng dòng văn bản
string str;
while ((str = file.ReadLine()) != null)
{
    Console.WriteLine(str);
}
```

LÀM VIỆC VỚI TẬP TIN

□ Đọc dữ liệu

- Ví dụ

```
//Đọc toàn bộ file, lưu vào biến str  
string str;  
str=file.ReadToEnd();
```

LÀM VIỆC VỚI TẬP TIN

- **Ví dụ:** Đọc dữ liệu từ tập tin data.txt, in kết quả đọc được ra màn hình
 - **Cách 1:** đọc từng dòng trong file

```
//Mở tập tin
StreamReader file = new StreamReader("data.txt");
if (file == null) return;

//Đọc từng dòng văn bản
string str;
while ((str = file.ReadLine()) != null)
{
    Console.WriteLine(str);
}

// Đóng tập tin
file.Close();
Console.ReadKey();
```

LÀM VIỆC VỚI TẬP TIN

- **Ví dụ:** Đọc dữ liệu từ tập tin data.txt, in kết quả đọc được ra màn hình
 - **Cách 2:** đọc toàn bộ file

```
//Mở tập tin
StreamReader file = new StreamReader("data.txt");
if (file == null) return;

//Đọc toàn bộ file, lưu vào biến str
string str;
str=file.ReadToEnd();

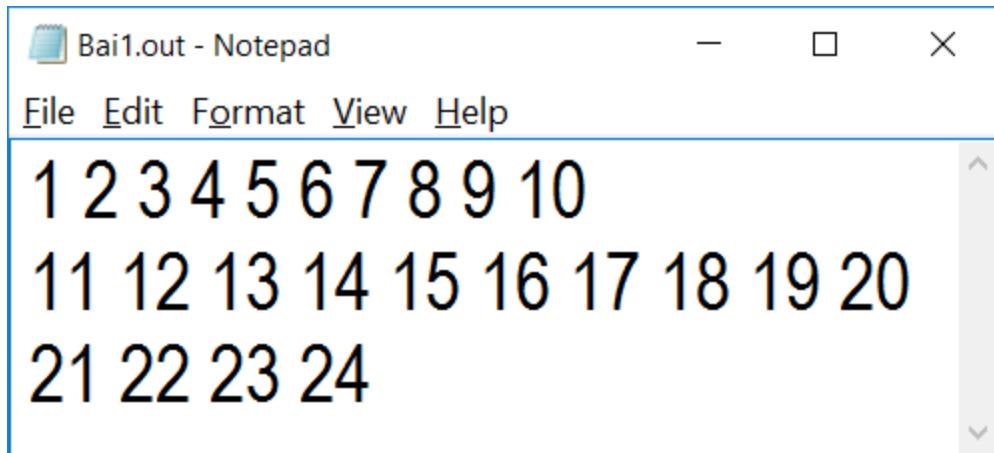
//In kết quả lên màn hình
Console.WriteLine(str);

// Đóng tập tin
file.Close();
Console.ReadKey();
```

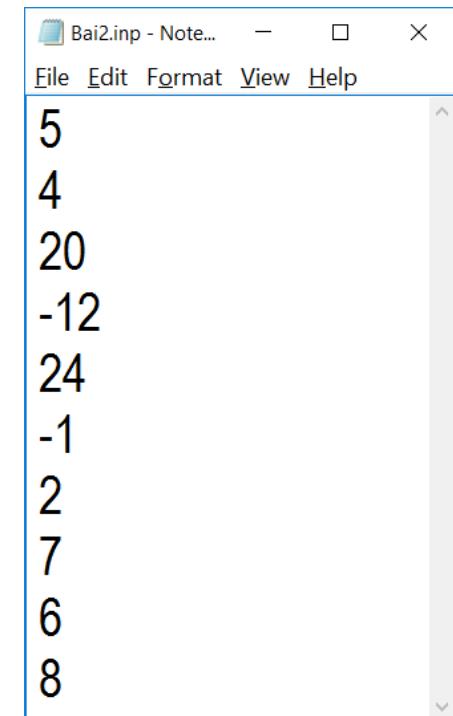
BÀI TẬP ÔN TẬP

Bài 1. Nhập từ bàn phím một số nguyên dương n , ghi ra tập tin dãy số từ 1 đến n , mỗi dòng có tối đa 10 chữ số. Tập tin được đặt tên là **Bai1.out**

Bài 2. Tập tin **Bai2.inp**, có 10 dòng, mỗi dòng có 1 chữ số nguyên. Hãy đọc dãy số nguyên trên và in lên màn hình **tổng của các số nguyên chẵn**.



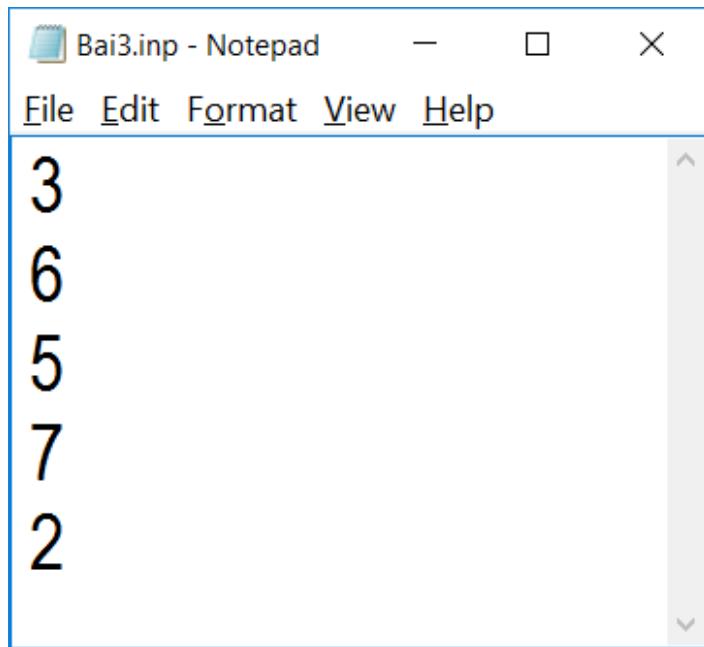
```
12345678910
11121314151617181920
21222324
```



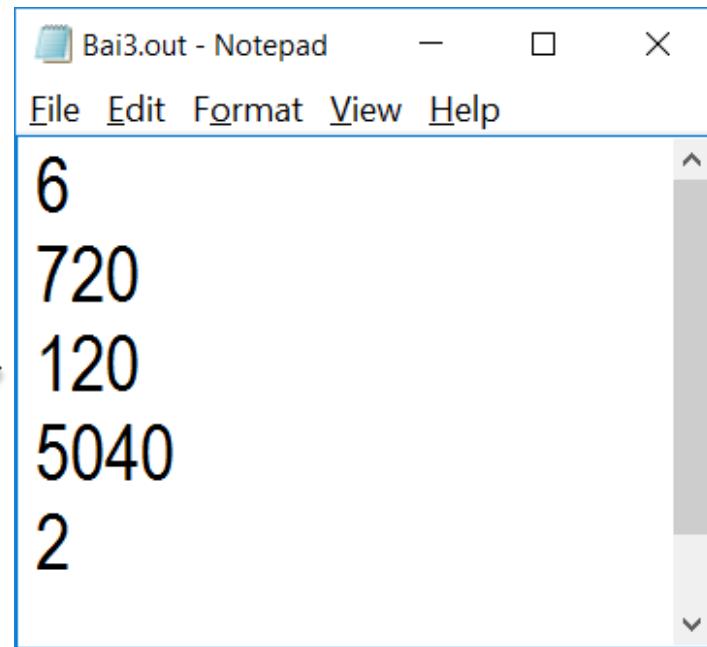
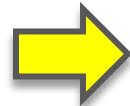
```
5
4
20
-12
24
-1
2
7
6
8
```

BÀI TẬP ÔN TẬP

Bài 3. Tập tin **Bai3.inp**, 5 dòng, mỗi dòng là một số nguyên dương. Đọc các số nguyên trong file, tính giai thừa và ghi kết quả vào tập tin **Bai3.out**, mỗi kết quả được ghi trên 1 dòng.
(Yêu cầu sử dụng hàm)



```
3
6
5
7
2
```



```
6
720
120
5040
2
```