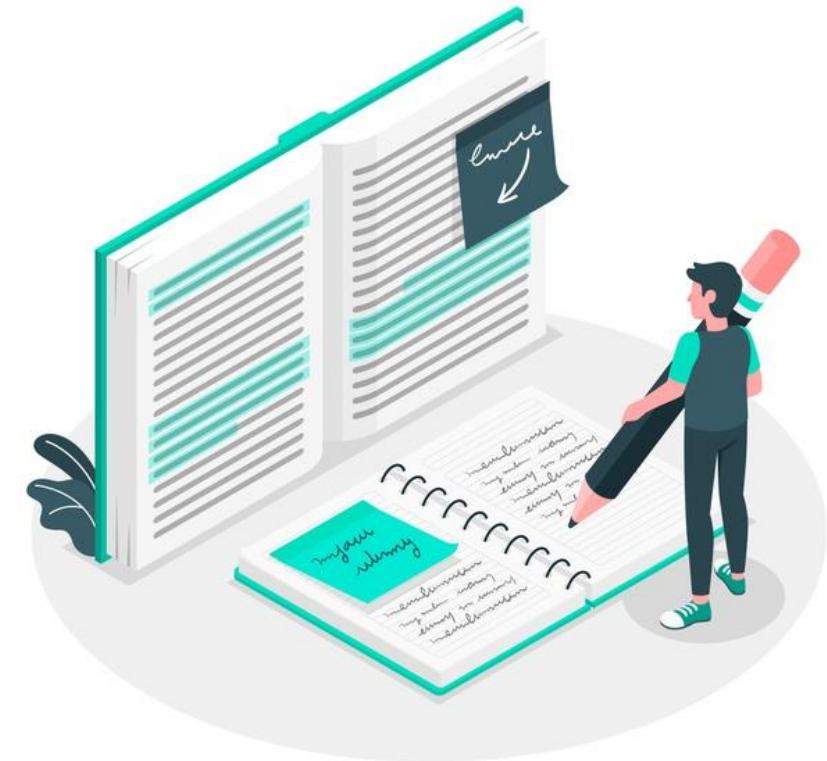
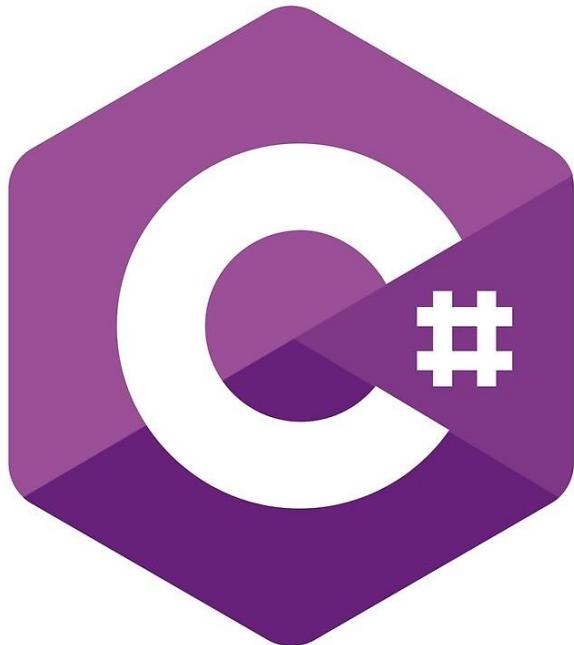


Khoa học: Game Developer - OOP

Bài 01: Căn bản ngôn ngữ lập trình **C#**





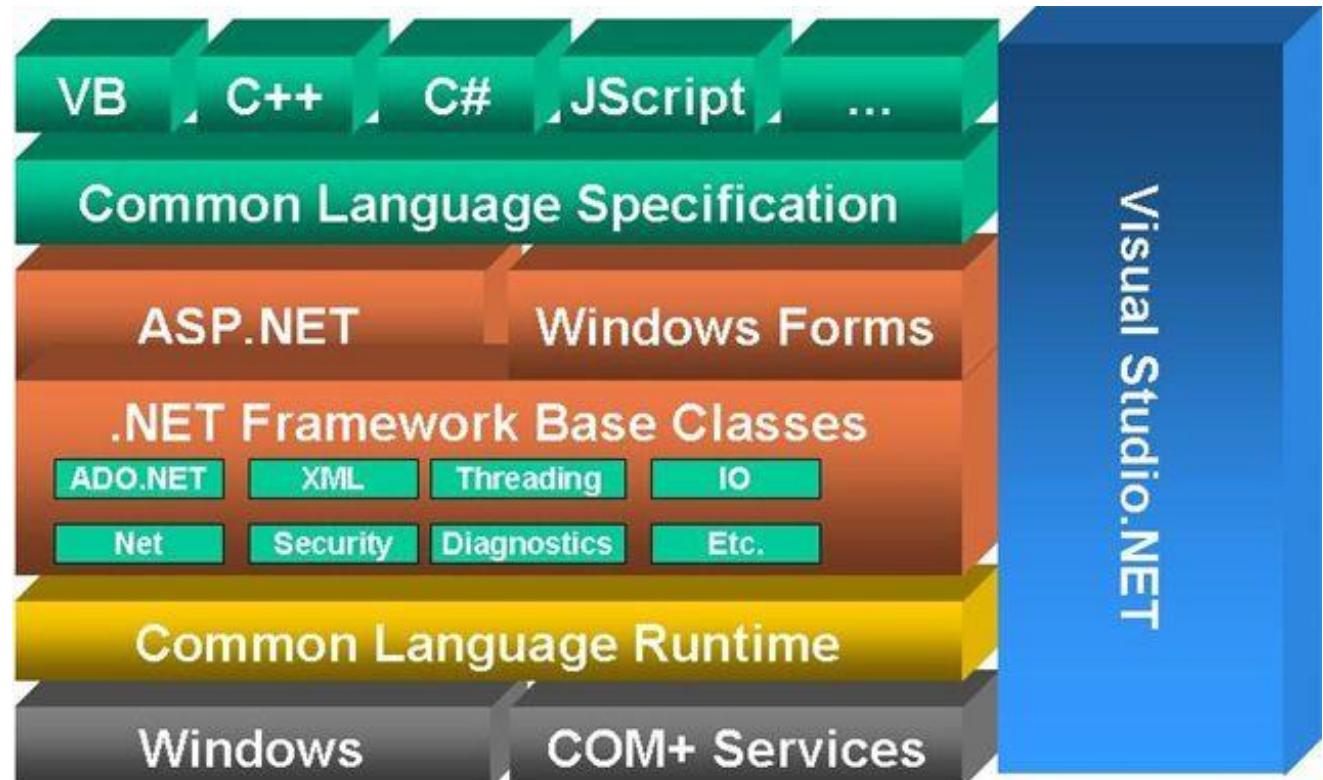
Nội dung

1. Nền tảng .NET
2. Tạo project C# với Visual Studio 2015
3. Cấu trúc chương trình C# cơ bản
4. Tạo, biên dịch và thực thi chương trình C#.

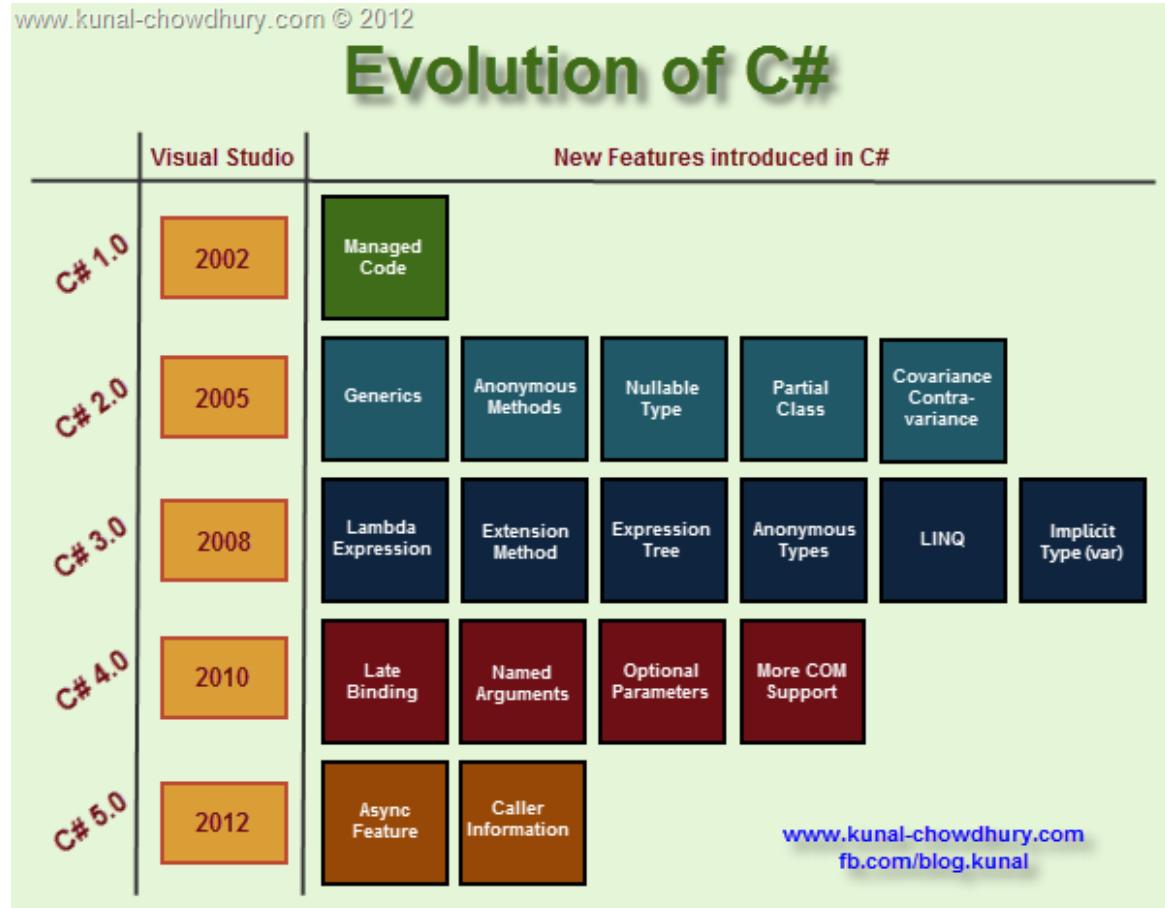
.NET Framework

Nền tảng .Net

- **Bốn ngôn ngữ chính thức:** C#, VB.Net, C++, và Jscript.NET
- **Common Language Runtime (CLR)**, nền tảng hướng đối tượng cho phát triển ứng dụng Windows và Web mà các ngôn ngữ có thể chia sẻ sử dụng.
- Bộ thư viện Framework Class Library - FCL.



Ngôn ngữ C#



- Chương trình viết bằng C# có 2 dạng:
 - Console
 - Windows Form
- C# là một ngôn ngữ rất đơn giản, với khoảng 80 từ khoá và hơn mươi kiểu dữ liệu dựng sẵn, nhưng C# có tính diển đạt cao.
- C# hỗ trợ lập trình có cấu trúc, hướng đối tượng, hướng thành phần (component oriented).

Intergrated Development Environment

- IDE là một bộ các công cụ phần mềm hỗ trợ lập trình viên soạn thảo, biên dịch, liên kết, gỡ rối, ... Ví dụ: Visual Studio.
- IDE giúp phát triển ứng dụng nhanh chóng và hiệu quả hơn. Đơn giản hóa quá trình phát triển phần mềm.
- IDE dành cho lập trình C#:
 - Visual Studio
 - NetBeans

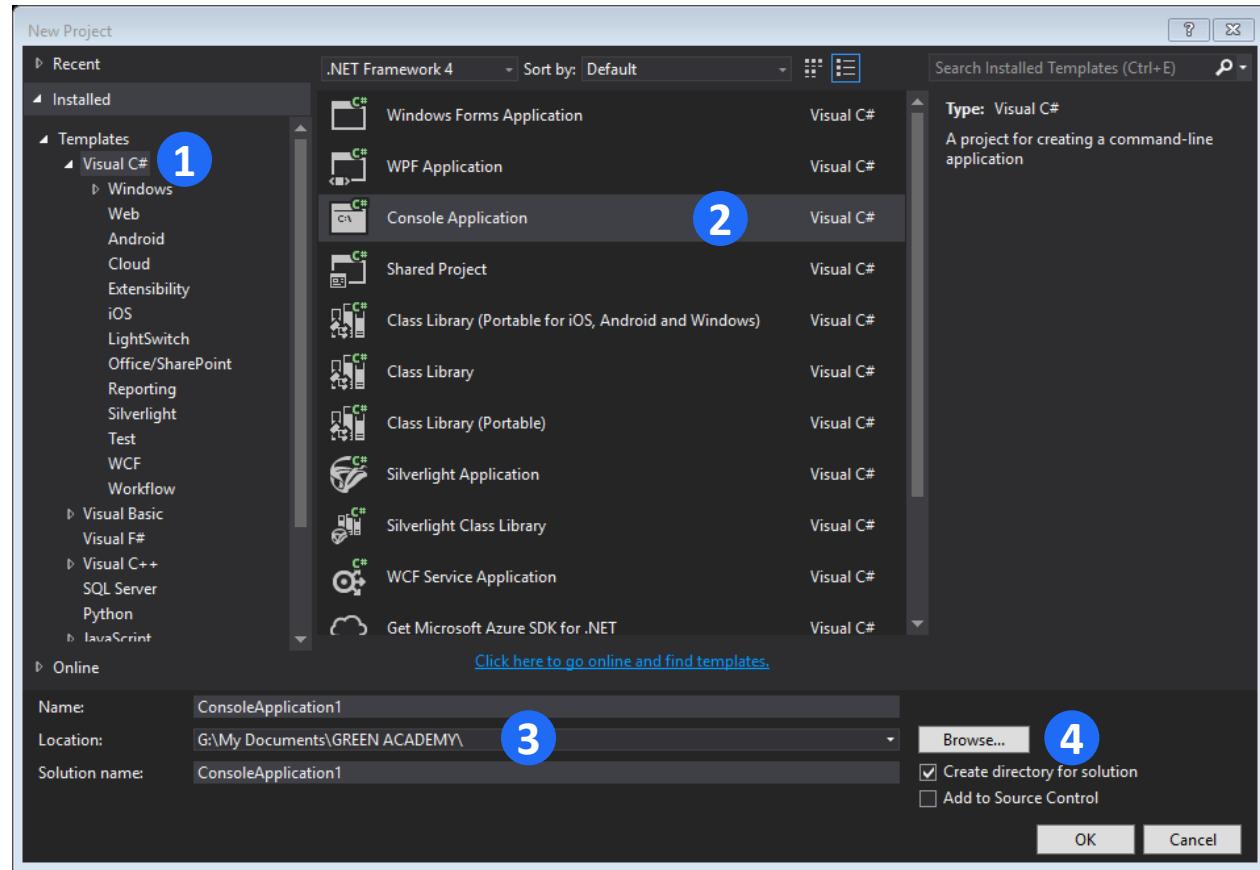


NetBeans

Visual
Studio

New Project

Tạo dự án mới với VS 2015



File -> New Project (Ctrl + Shift + N)

Số 1: Chọn ngôn ngữ C#

Số 2: Chọn Console Application

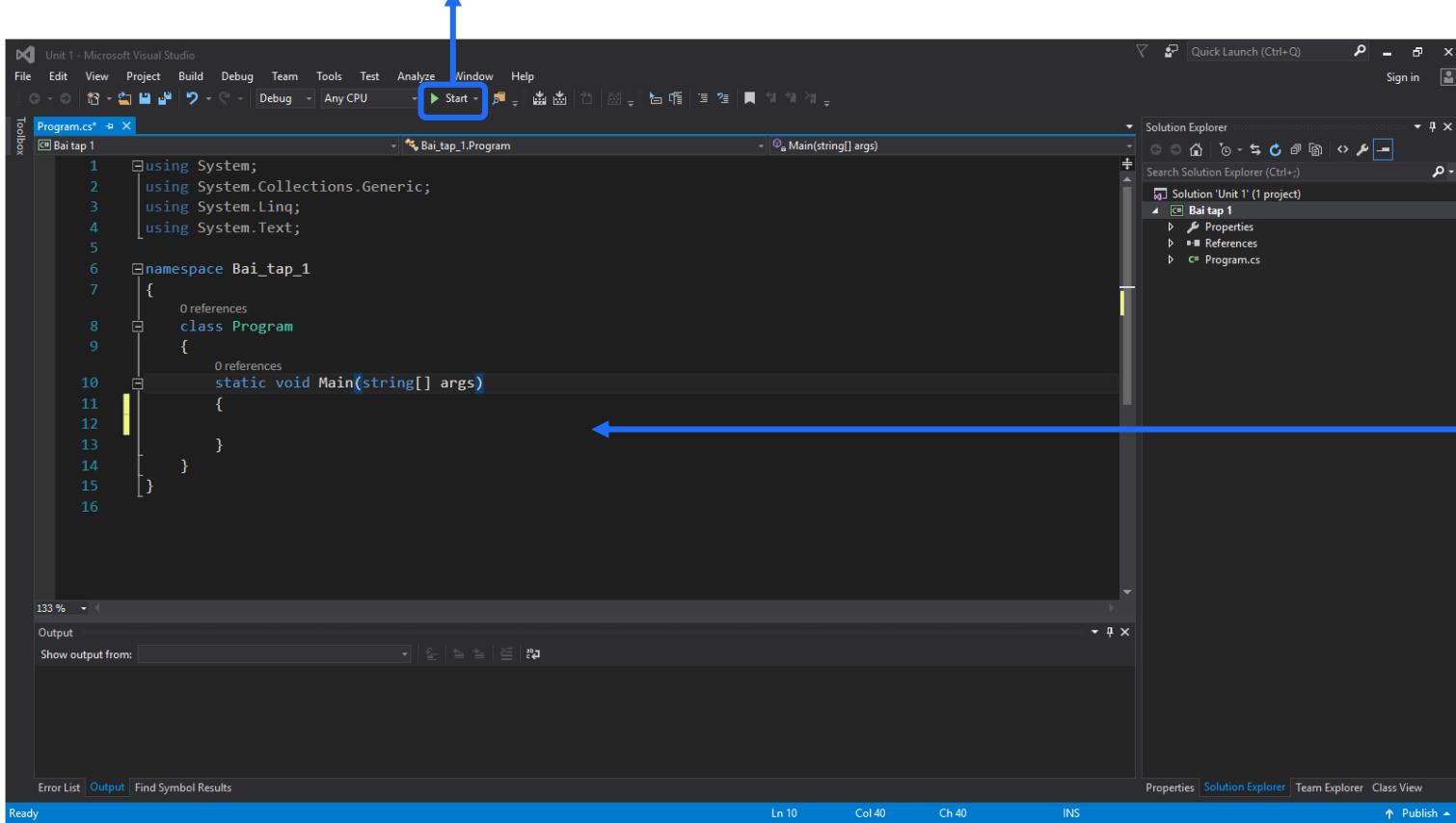
Số 3: Name: Nhập tên của dự án (Ví dụ: Baitap01)

Số 4: Location: Vị trí lưu dự án (Ví dụ: C:\)

New Project

Tạo dự án mới với VS 2015

Run project (F5) để chạy
chương trình đã biên dịch



Viết code tại đây

Cấu trúc chương trình C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Bai_tap_1
{
    class Program
    {
        static void Main(string[] args)
        {
        }
    }
}
```

Khai báo sử dụng thư viện
using: Chỉ thị tiền xử lý

Chương trình chính, bắt buộc là
hàm main()
Kết thúc 1 phát biểu đơn là dấu
chấm phẩy (;)

Output Data

Xuất dữ liệu

- **Console.WriteLine("text");**

Xuất nội dung **text** ra màn hình.

- **Console.WriteLine("text");**

Xuất nội dung **text** ra màn hình sau đó tự động xuống dòng.

- **Console.Write("format",...);**

Console.WriteLine("format",...);

Xuất nội dung theo định dạng cho trước. Các định dạng bắt đầu bằng dấu { số thứ tự : đặc tả}

```
Console.WriteLine("Green ");
```

```
Console.WriteLine("Academy");
```

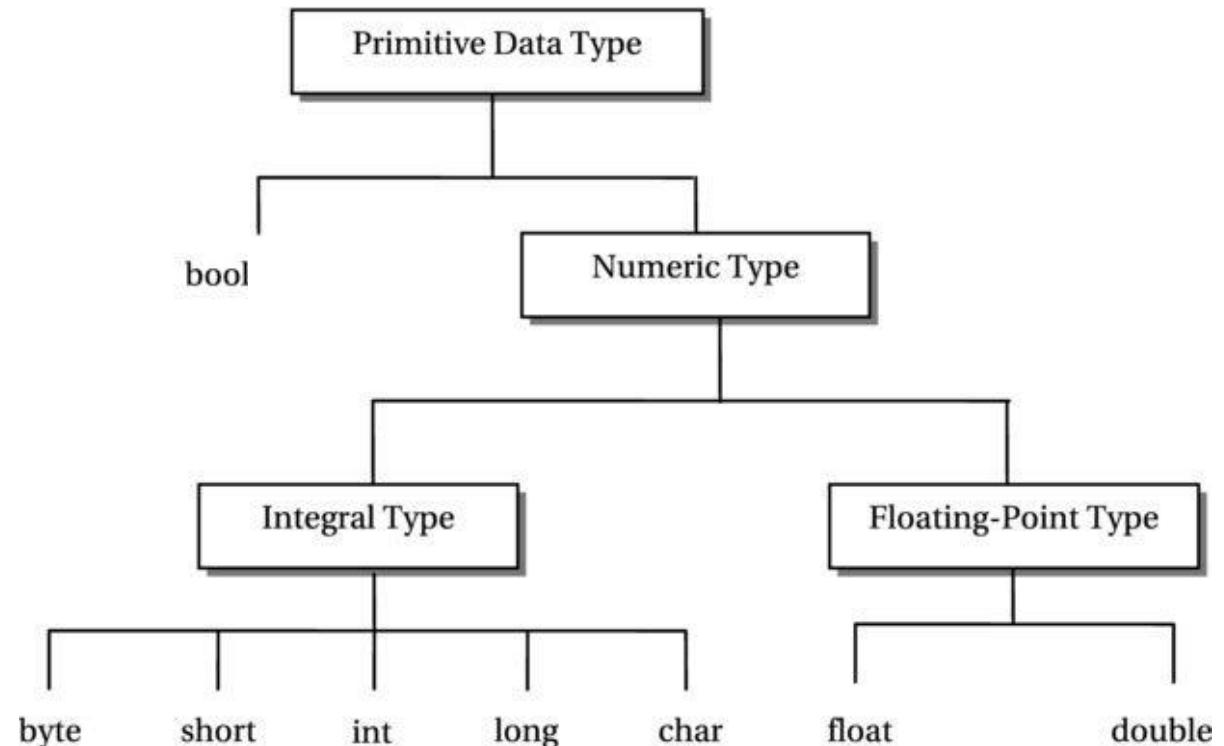
```
Console.WriteLine("{0} {1} {2}\n",
    "IT", "Design", "Korea");
```

Green Academy
IT - Design - Korea

Kiểu dữ liệu nguyên thuỷ

Primitive Data Types

- Kiểu dữ liệu nguyên thuỷ là kiểu được giữ lại từ ngôn ngữ C. Có 8 kiểu dữ liệu nguyên thuỷ.
- Ví dụ:
 - `int x = 6;`
 - `double kg;`



Biến và Hằng

Variables and Constant

- **Biến** là thành phần lưu trữ dữ liệu được chương trình sử dụng trong các biểu thức tính toán.
- Mỗi biến có kiểu dữ liệu riêng.
- **Hằng** là thành phần lưu trữ dữ liệu và không thể thay đổi giá trị sau khi khai báo.
- Thêm từ khóa **const** khi khai báo hằng.



Cách khai báo hằng số

const float PI = 3.14

Khai báo biến

Variable declaration

<Datatype> <Variable Name> [= starting value];

- **int a;**

Khai báo biến a không có giá trị khởi đầu.

- **int a, b = 6, c;**

Khai báo đồng thời nhiều biến cùng kiểu.

- **double b = 5.5;**

Khai báo biến b đồng thời set giá trị khởi đầu bằng 5.5.

- **c = 16;**

Gán giá trị cho biến c.

Khai báo biến không tường minh

Implicitly Typed Variables

```
var <Variable Name> = starting value ;
```

- `var a = 12;`

Khai báo biến a có giá trị là 12

→ Biến a có kiểu số nguyên.

- `var b = 7;`

Khai báo biến b có giá trị là 7

→ Biến b có kiểu số nguyên.

- `a = 8.8;`

→ generate a compiler error

- `c = 7.5;`

→ generate a compiler error



Casting

Ép kiểu

Input Data

Nhập dữ liệu từ bàn phím

- **Console.Read();**

Đọc 1 ký tự nhập từ bàn phím.

- **Console.ReadLine();**

Đọc 1 dòng chuỗi ký tự nhập từ bàn phím.

- **Console.ReadKey();**

Ứng dụng dừng màn hình chờ xem kết quả.

```
String name;
```

```
int year;
```

```
Console.Write("Input name: ");
```

```
name = Console.ReadLine();
```

```
Console.Write("Input birthyear: ");
```

```
year = Console.ReadLine();
```

Error: Cannot implicitly convert type 'string'
to 'int'.

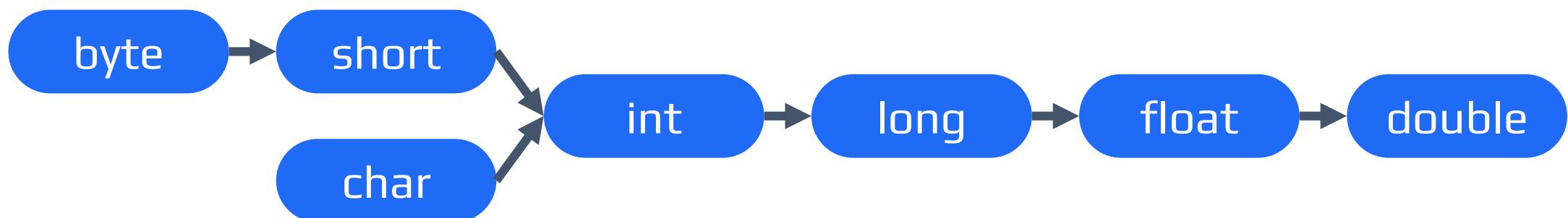
```
year = Convert.ToInt32(Console.ReadLine());
```

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

Ép kiểu

Casting

- Đối với kiểu dữ liệu nguyên thuỷ, ép kiểu tự động (implicit casting) xảy ra theo chiều mũi tên.



- Khai báo biến
- ```
int a = 5;

double b = 9.5;
```

- Ép kiểu tự động
- ```
b = a;
```

- Ép kiểu tường minh
- ```
a = (int) b;
```
- phần thập phân sẽ bị bỏ*

# Chuỗi → Kiểu dữ liệu khác

String → Other Data Type

Xét biểu thức 2

- **int a = Convert.ToInt16("3");**
- **int b = Convert.ToInt16(5.5);**
- **int c = a + b;**  
**→ c = 8**

## Chuỗi → Kiểu khác

Convert.ToBoolean(String)

Convert.ToByte(String)

Convert.ToChar(String);

Convert.ToInt16(String);

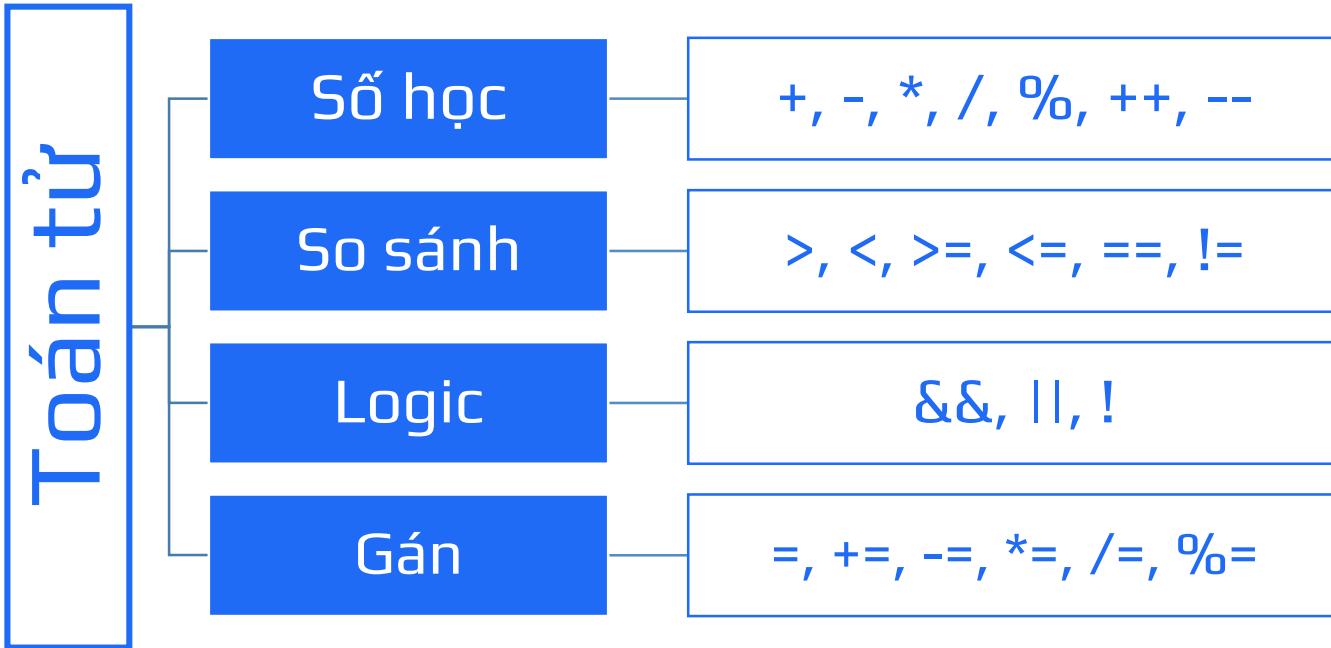
Convert.ToInt32(String);

Convert.ToDouble(String);

Convert.ToString(String);

# Toán tử và biểu thức

Operators and expressions



- **Biểu thức** là sự kết hợp giữa toán tử và toán hạng. Kết quả của biểu thức là một giá trị.

Hãy cho biết giá trị của biểu thức sau?

- a. int x = 11 % 4;
- b. boolean a = 9 < 2 && true || 4 > 3;

# Toán tử điều kiện

Conditional operator

**<điều kiện> ? <giá trị đúng> : <giá trị sai>;**

- Toán tử điều kiện là toán tử 3 ngôi.
- **Điễn giải:** Nếu biểu thức **<điều kiện>** có giá trị true thì kết quả của biểu thức là **<giá trị đúng>**, ngược lại là **<giá trị sai>**.
- **Ví dụ:** Tìm số lớn nhất trong 2 số a và b

```
int a = 1, b = 9;
```

```
int max = a > b ? a : b;
```

# Lệnh if

## If statement

```
if (<điều kiện>) {
 <công việc>
}
```

```
double diem = 4;
if (diem >= 5) {
 System.out.println("Đậu");
}
```

- **Điễn giải:** Nếu <điều kiện> có giá trị true thì <công việc> được thực hiện.
- **Kết quả:** Đoạn mã trên không xuất gì ra màn hình vì biểu thức **diem >= 5** là không đúng (false).

# Lệnh if ... else

If... else statement

```
if (<điều kiện>) {
 <công việc 1>
}

else {
 <công việc 2>
}
```

- **Điễn giải:** Nếu <điều kiện> có giá trị true thì <công việc 1> được thực hiện, ngược lại <công việc 2> được thực hiện.

```
double diem = 4;
if (diem < 5) {
 System.out.println("Rớt");
}
else {
 System.out.println("Đậu");
}
```

# Lệnh if ... else

If... else statement

- Kết quả: Đoạn mã xuất chữ "**Rớt**" vì điều kiện **diem < 5** là đúng (true).

# Nhiều lệnh if

Multiple if statement

```
if (<điều kiện 1>) {
 <công việc 1>
}

else if (<điều kiện 2>) {
 <công việc 2>
}

else {
 <công việc N+1>
}
```

- **Diễn giải:** Chương trình kiểm tra lần lượt từng **<điều kiện>** từ 1 đến N, nếu bắt gặp điều kiện i đầu tiên có giá trị true thì **<công việc i>** được thực hiện. Ngược lại, **<công việc N+1>** được thực hiện.

# Nhiều lệnh if

## Multiple if statement

### Biện luận và giải phương trình bậc 2

```
double delta = Math.pow(b,2) - 4 * a * c;
if (delta < 0) {
 System.out.println("Vô nghiệm");
}
else if (delta == 0) {
 System.out.println("Nghiệm kép");
}
else {
 System.out.println("2 nghiệm phân biệt");
}
```

# Lệnh switch

Switch statement

```
switch (<biểu thức>) {
 case <giá trị 1>:
 <công việc 1>
 break;
 case <giá trị 2>:
 <công việc 2>
 break;
 ...
 default:
 <công việc N+1>
 break;
}
```

## Diễn giải:

- So sánh giá trị của **<biểu thức>** với **<giá trị>** các case. Nếu bằng với giá trị của case nào thì sẽ thực hiện **<công việc>** của case đó, ngược lại sẽ thực hiện công việc của **default**.
- Nếu công việc của case không chứa lệnh **break** thì case tiếp sau sẽ được thực hiện.
- **default** là tùy chọn.

# Lệnh switch

## Switch statement

```
int option = 1;
switch (option) {
 case 1: System.out.println("Bạn chọn chức năng 1"); break;
 case 2: System.out.println("Bạn chọn chức năng 2"); break;
 case 3: ← không có break
 case 4: System.out.println("Đang xây dựng"); break;
 default: System.out.println("Vui lòng chỉ chọn từ 1-4");
 break;
}
```

# Lệnh lặp while

while statement

```
while (<điều kiện>) {
 <công việc>
}
```

```
int count = 1;
while (count < 11) {
 System.out.println(count);
 count++;
}
```

- **Diễn giải:** <công việc> được thực hiện trong khi <điều kiện> vẫn còn giá trị true.
- **Kết quả:** Đoạn mã trên xuất giá trị từ 1 đến 10 ra màn hình.

```
do {
 <công việc>
}
while (<điều kiện>);
```

```
int count = 1;
do {
 System.out.println(count);
 count++;
} while (count < 11);
```

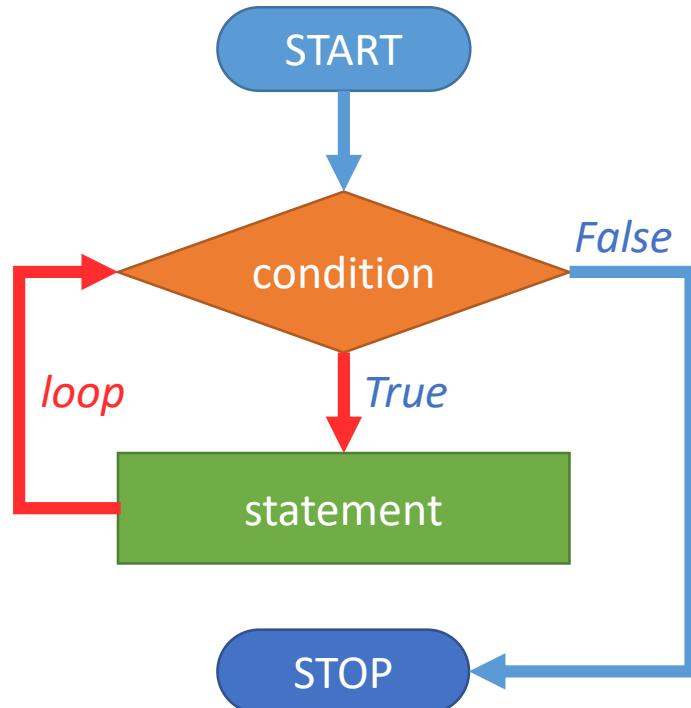
# Lệnh lặp do ... while

do... while statement

- **Diễn giải:** Điểm khác biệt giữa phát biểu do while và while là thời điểm kiểm tra điều kiện lặp tại phần cuối của vòng lặp.
- **Lưu ý:** Phát biểu do while thực hiện lệnh lặp trước, kiểm tra điều kiện sau. Vì vậy, luôn có ít nhất một lần lặp.

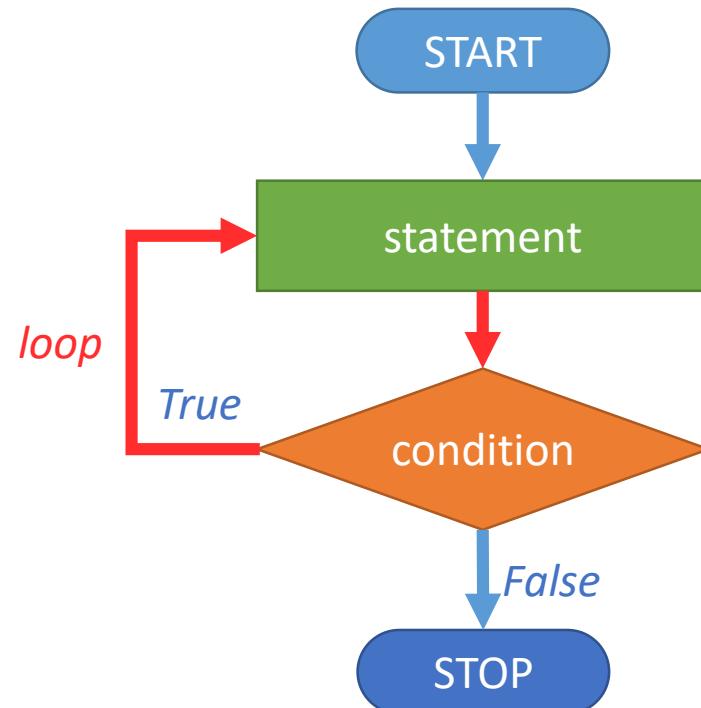
# While và Do ... While

Flowchart



**while**

Kiểm tra trước. Đúng mới lặp.  
Đôi khi vòng lặp không xảy ra.



**do ... while ...**

Thực hiện trước rồi mới kiểm  
tra điều kiện. Vòng lặp thực  
hiện ít nhất 01 lần.

# Lệnh lặp for

for statement

```
for (<khởi đầu> ; <điều kiện> ; <bước nhảy>) {
 <công việc>
}
```

- **Điễn giải:**

1. Thực hiện **<khởi đầu>**
2. Kiểm tra **<điều kiện>**, nếu còn đúng thì thực hiện bước 3. Ngược lại, thì kết thúc vòng lặp.
3. Thực hiện **<công việc>**
4. Thực hiện **<bước nhảy>**
5. Quay về bước 2.

# Lệnh lặp for

for statement

```
for (<khởi đầu> ; <điều kiện> ; <bước nhảy>) {
 <công việc>
}
```

- **Điễn giải:**

1. Thực hiện **<khởi đầu>**
2. Kiểm tra **<điều kiện>**, nếu còn đúng thì thực hiện bước 3. Ngược lại, thì kết thúc vòng lặp.
3. Thực hiện **<công việc>**
4. Thực hiện **<bước nhảy>**
5. Quay về bước 2.

# Lệnh lặp for

for statement

```
for (int count = 1; count <= 10 ; count++) {
 System.out.println(count);
}
```

- **Điễn giải:** Xét điểm **<khởi đầu>** có thoả mãn **<điều kiện>** thì **<công việc>** được thực hiện, đồng thời **<bước nhảy>** thay đổi cho đến khi điều kiện không còn đúng thì kết thúc vòng lặp.
- **Kết quả:** Đoạn mã trên xuất giá trị từ 1 đến 10 ra màn hình.

# Lệnh break và continue

## Điễn giải:

- **break** dùng để thoát khỏi vòng lặp.
- **continue** dùng để thực hiện lần lặp tiếp theo ngay lập tức.

<Lệnh lặp> {

...

**break;**

...

}

<Lệnh lặp> {

...

**continue;**

...

}

