



**C# and .NET
Development
with VS Code**

FOR BEGINNERS



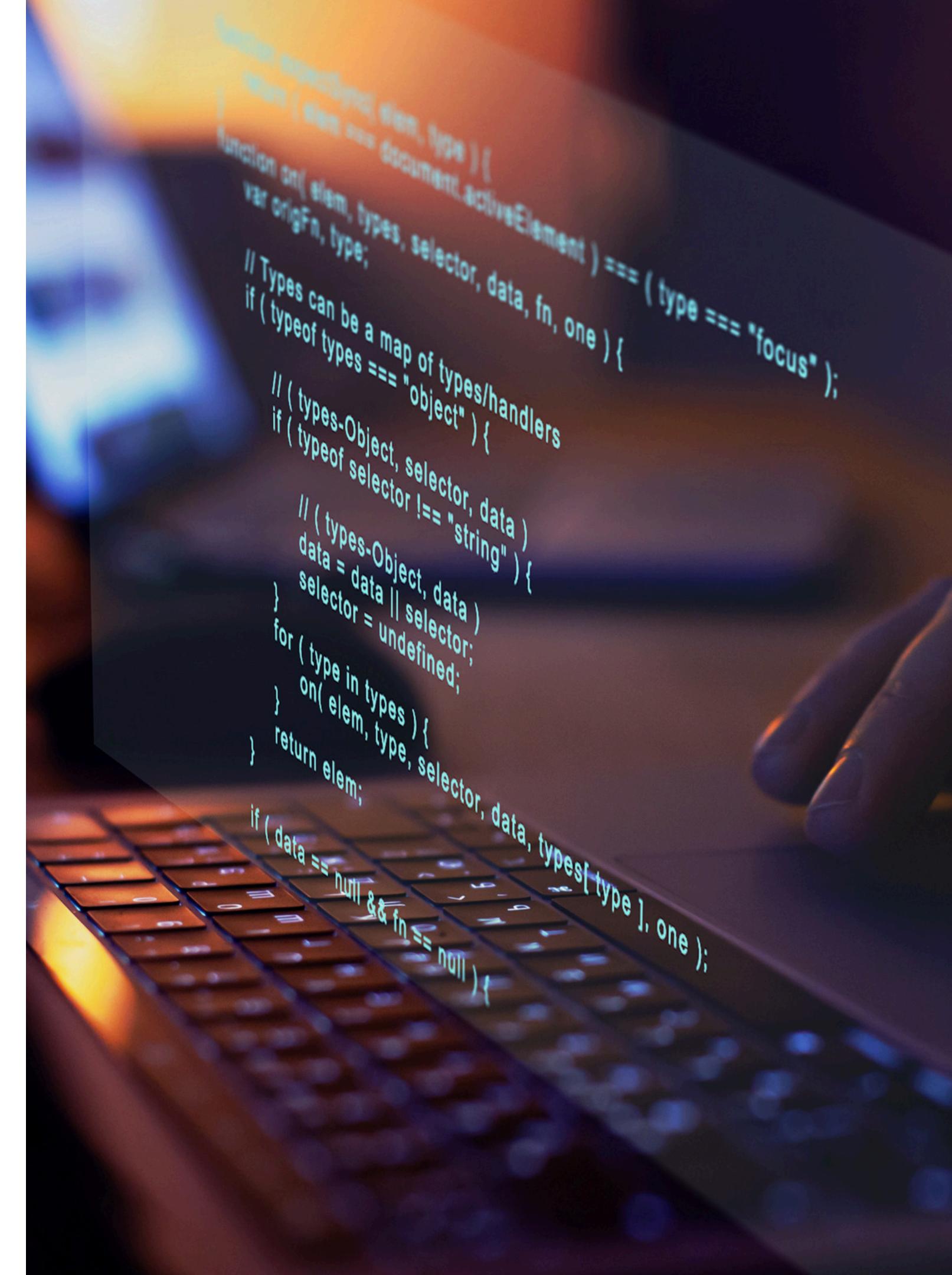
PHẦN 1 C# FOUNDATION

XÂY DỰNG TƯ DUY NỀN TẢNG LẬP
TRÌNH VỚI C#

NỘI DUNG

- Cài đặt môi trường và công cụ lập trình
- Tìm hiểu các khái niệm cơ bản trong lập trình
- Cách tư duy giải quyết vấn đề trong lập trình
- Hướng dẫn thực hành qua ví dụ về console app

```
1 //Note command
2 //Input và Output
3 Console.WriteLine("Nhập vào giá trị: "); //output
4 string? input = Console.ReadLine(); //input
5 Console.WriteLine($"Giá trị bạn vừa nhập là: {input}"); //output
```





CÀI ĐẶT CÔNG CỤ VÀ MÔI TRƯỜNG

1. CÀI ĐẶT CÔNG CỤ RIDER (JET JETBRAINS)



- Link cài đặt: <https://www.jetbrains.com/rider/>

1

2

3

4

5

6

7

8

9

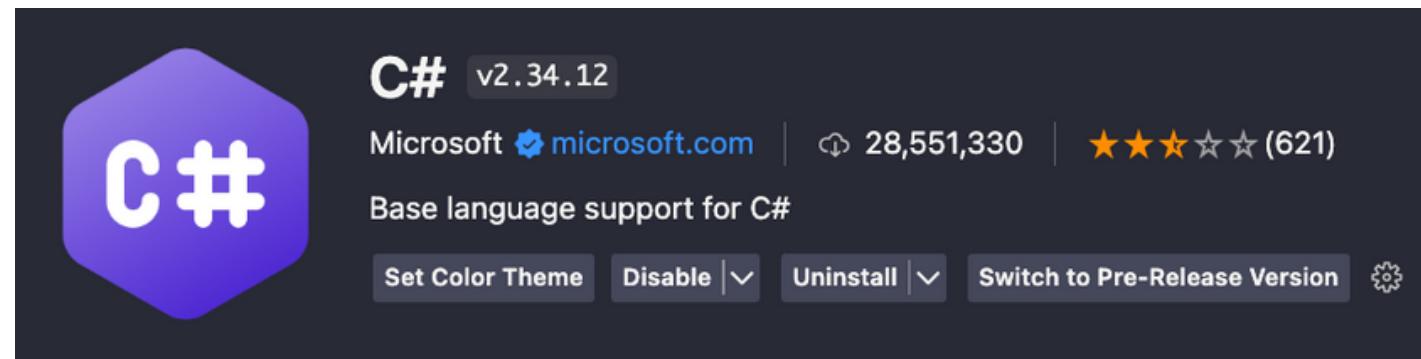
10

2. CÀI ĐẶT .NET CORE (SDK) TÙY HỆ ĐIỀU HÀNH MÀ CÀI BẢN TƯƠNG ỨNG

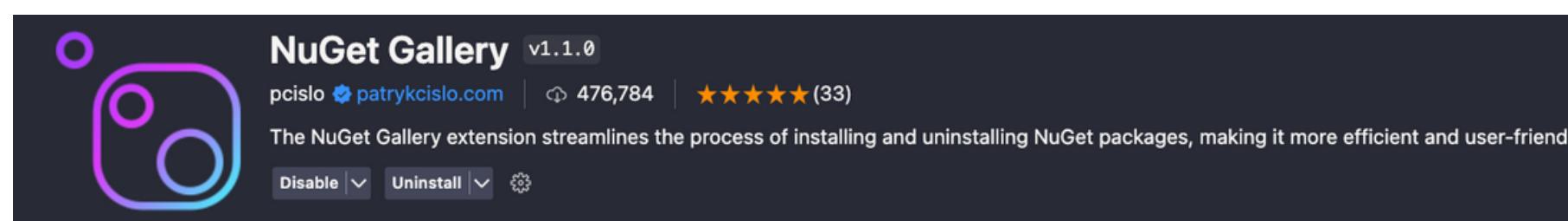


- Link cài đặt: <https://dotnet.microsoft.com/en-us/download/dotnet/8.0>

3. CÀI ĐẶT EXTENSION NẾU NHƯ SỬ DỤNG VISUAL CODE (TA SẼ HỌC PHẦN ĐẦU VỚI VISUAL CODE ĐỂ HIỂU DỰ ÁN)



4. CÀI ĐẶT EXTENSION SETUP CHO CÁC THƯ VIỆN TRONG .NET



5. KHỞI TẠO DỰ ÁN VỚI .NET CONSOLE APP BẰNG VISUAL CODE VÀ CHẠY DỰ ÁN

```
dotnet new console -n [ten_du_an]
```

```
dotnet run
```

6. LỆNH ĐÓNG GÓI VÀ BUILD DỰ ÁN

```
dotnet publish -c Release -r osx-x64 --self-contained
```

```
dotnet publish -c Release -r win-x64 --self-contained
```

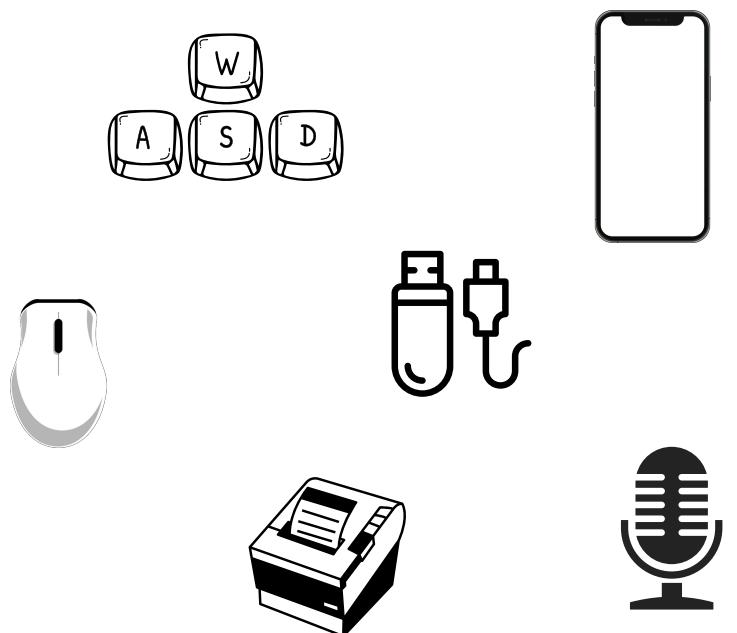
MAC OS

WINDOW OS



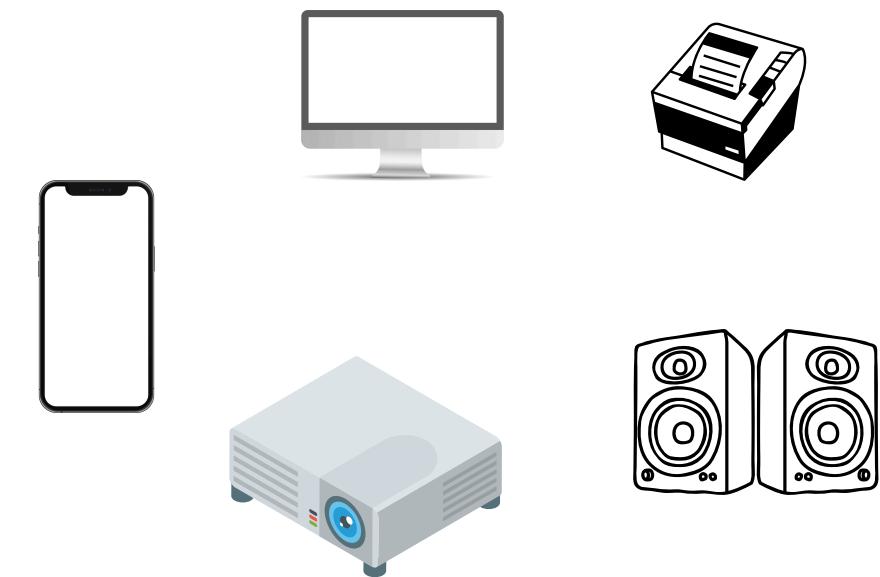
CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

Input (Thiết bị nhập)



Khi nhắc đến input hay input device là những công cụ hỗ trợ ta cung cấp dữ liệu cho máy tính

Output (Thiết bị xuất)



Khi nhắc đến output hay output device là nói đến công cụ hỗ trợ xuất thông tin hoặc kết quả mà ta nhận được từ một thiết bị hoặc hệ thống ta đang sử dụng

1

2

3

4

5

6

7

8

9

10



CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

1

2

3

4

5

6

7

8

9

10

CÚ PHÁP: NHẬP XUẤT DỮ LIỆU

1. **Input (Lệnh nhập)**

- Dữ liệu nhập từ người dùng luôn
lương là kiểu **string**

```
string input = Console.ReadLine();
```

Lưu ý: Mỗi câu lệnh sẽ bắt buộc kết thúc bằng dấu ;

```
C# Program.cs ×  
app > C# Program.cs  
1 //input  
2 string? input = Console.ReadLine();  
3  
4  
5  
6
```



CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

1

2

3

4

5

6

7

8

9

10

CÚ PHÁP: NHẬP XUẤT DỮ LIỆU

2. output (Lệnh xuất)

- Dữ liệu xuất ra màn hình có thể là number, string, boolean ...

```
Console.WriteLine($"Output: {value}");
```

Lưu ý: Mỗi câu lệnh sẽ bắt buộc kết thúc bằng dấu ;

```
app > C# Program.cs
1 //Lệnh xuất (output): Màn hình sẽ hiển thị dòng này khi run ứng dụng
2 Console.WriteLine("Nhập vào tên của bạn: ");
3 //Lệnh nhập (input): Lưu giữ giá trị người dùng nhập
4 string? name = Console.ReadLine();
5 //Kết quả (output): In ra lời chào và tên người dùng nhập
6 Console.WriteLine($"Nhập vào tên của bạn: {name}");
7
8
9
10
11
12
13
14
```



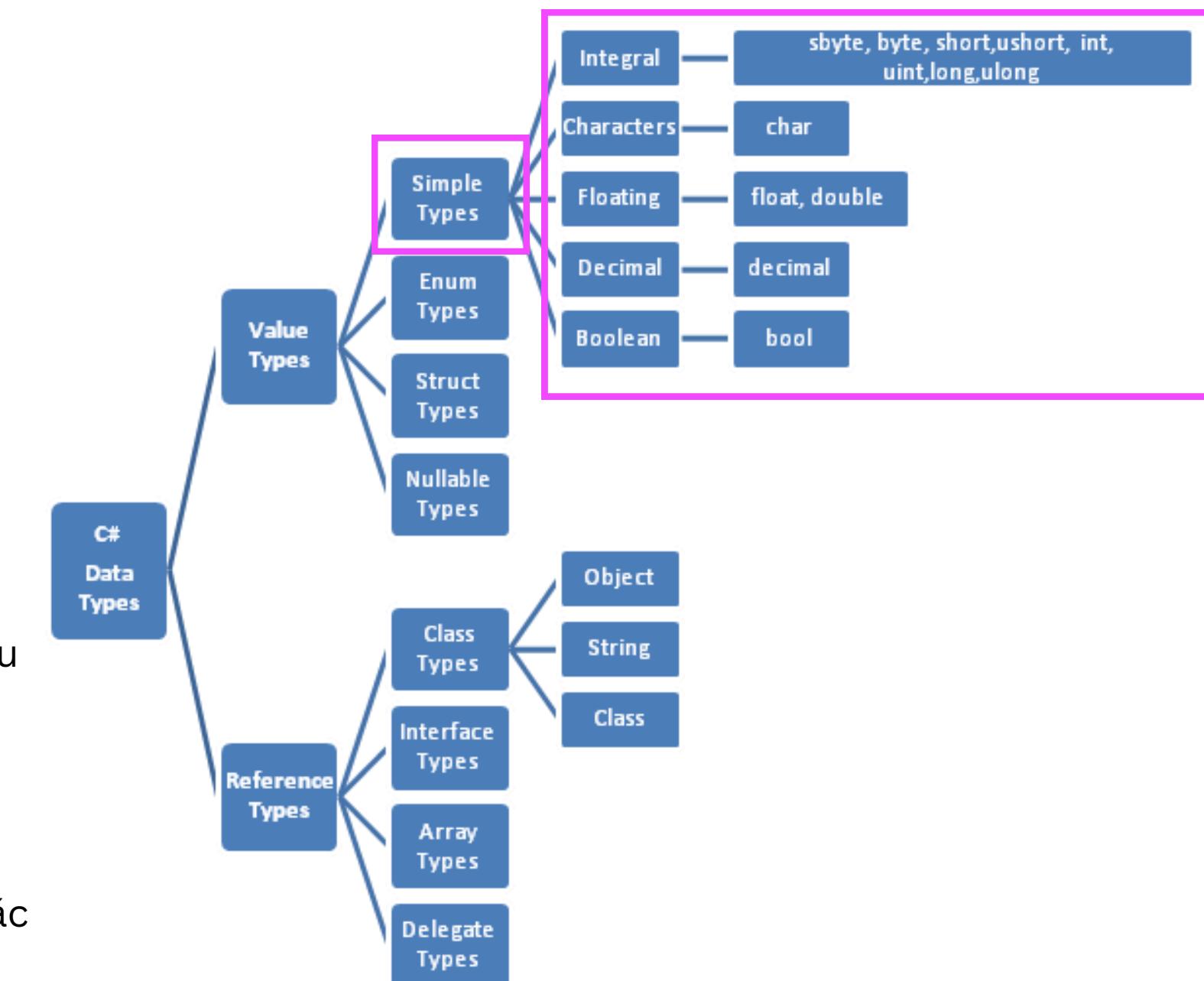
CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

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

BIÊN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

3. Data type

- Để máy tính có thể phân biệt được các xử lý tính toán hiển thị lưu trữ v...v... trong lập trình thì các giá trị sẽ được phân loại thành các kiểu dữ liệu gọi là **datatype**
- Ví dụ:
 - Để tính toán ta dùng **integral** (số nguyên, thập phân,...), **floating**(số thực), **Decimal**(tiền tệ)
 - Để xử lý đúng sai (hợp lệ, không hợp lệ) ta dùng **Boolean**)
 - Để lưu trữ thông tin ta (họ tên, email, ...) ta sử dụng Characters (ký tự, hoặc chuỗi ký tự **string**)
 - Các kiểu dữ liệu ở nhóm Enum, Struct, sẽ là các kiểu dữ liệu mà C# cho phép ta tự định nghĩa.
 - Reference type cũng tương tự C# hỗ trợ ta định nghĩa các kiểu liệu mới do ta qui định. Ta sẽ tìm hiểu các kiểu dữ liệu tự định nghĩa ở các phần sau.
 - Trong C#, Nullable<T> là một cấu trúc (struct) cho phép các kiểu giá trị (value types) như int, bool, float, v.v., có thể chứa giá trị null. Điều này rất hữu ích khi bạn cần biểu diễn tình trạng "không có giá trị" cho các kiểu dữ liệu không phải là kiểu tham chiếu (reference types). (Tóm lại null không chứa giá trị hoặc không tham chiếu giá trị).





CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

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

BIÊN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

4. Biên - Khai báo biến

Để máy tính có thể phân biệt được các xử lý tính toán hiển thị lưu trữ v...v... trong lập trình thì các giá trị sẽ được phân loại thành các kiểu dữ liệu gọi là datatype.

[Kiểu dữ liệu] [Tên biến] = [Giá trị];

```
Console.WriteLine("Hello cybersoft");
Console.WriteLine("Hello cybersoft");
Console.WriteLine("Hello cybersoft");
```

Khai báo biến
Tên biến không bắt đầu từ số hay ký tự đặc biệt
`string mess = "hello cybersoft";`



CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

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

BIÊN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

4. Biên - Khai báo biến

Qui tắc khai báo biến

- Lưu trữ dữ liệu tạm thời để xử lý
- Đại diện vùng nhớ được cấp phát
- Phải khai báo trước khi sử dụng
- Phân biệt hoa và thường
- Phải bắt đầu bằng kí tự là chữ, hoặc gạch dưới (_).
- Không được có khoảng trắng giữa các từ (giá trị - giaTri)
- Không được sử dụng tiếng Việt có dấu

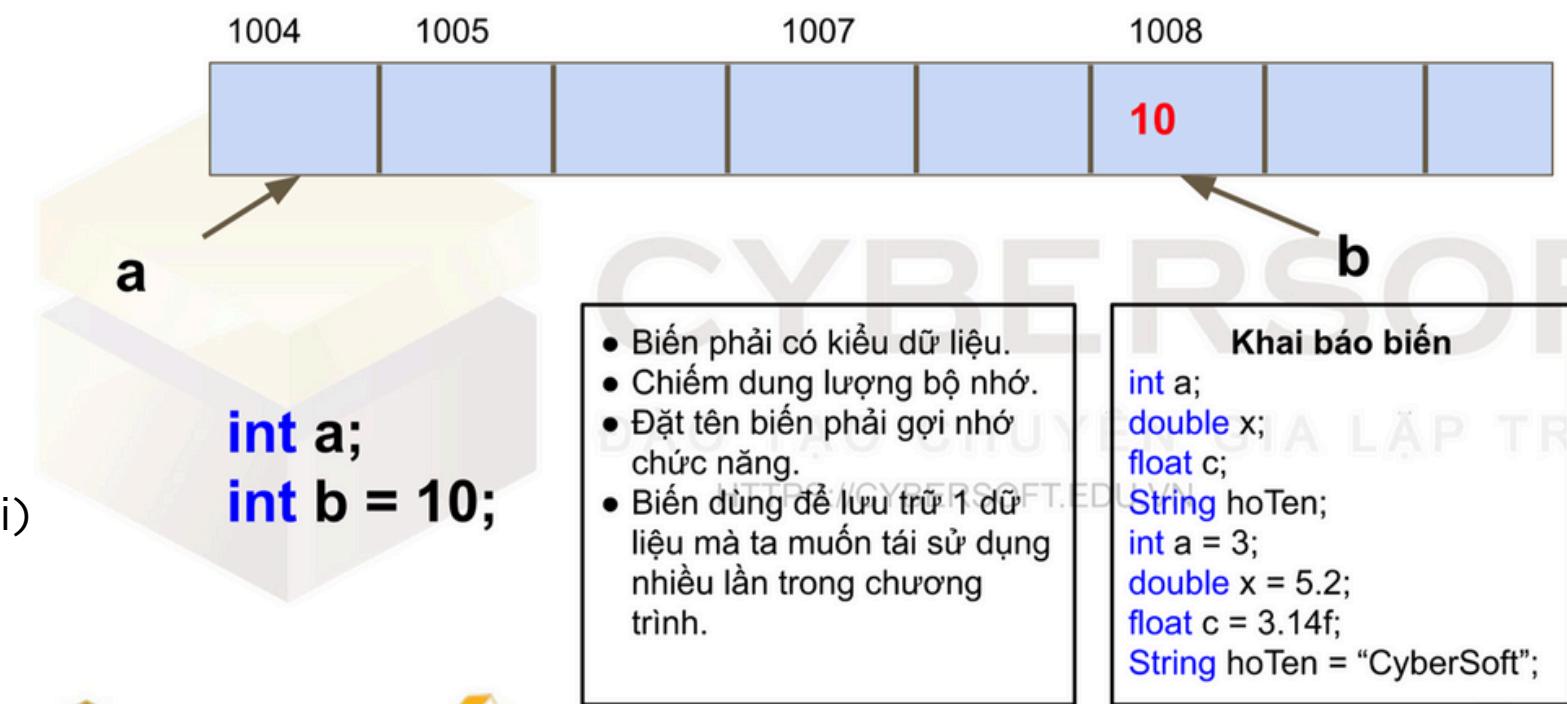
Qui tắc đặt tên

1. Đặt tên biến, tên hàm, tên lớp có ý nghĩa
2. Tên biến viết thường chữ đùa (kiểu Lạc đà - camelCase)

Ví dụ:

- int soLuong;
- float donGia;

3. Xài tiết kiệm -> tăng performance





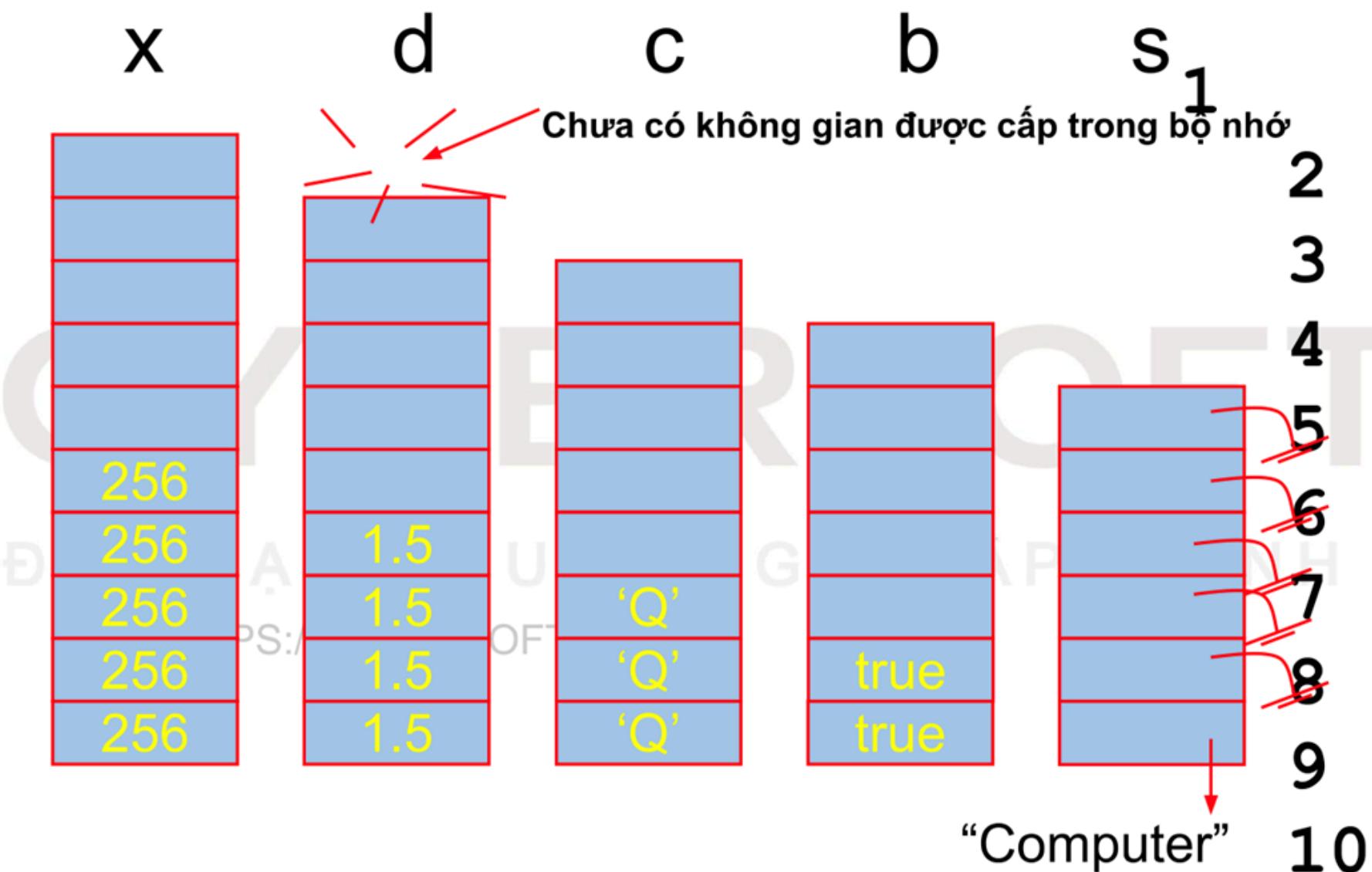
CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

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

BIẾN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

4. Biến - Khai báo biến

```
int x;  
double d;  
char c;  
boolean b;  
String s;  
x = 256;  
d = 1.5;  
c = 'Q' ;  
b = true;  
s = "Computer"
```





CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

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

BIÊN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

5. Numeric toán tử

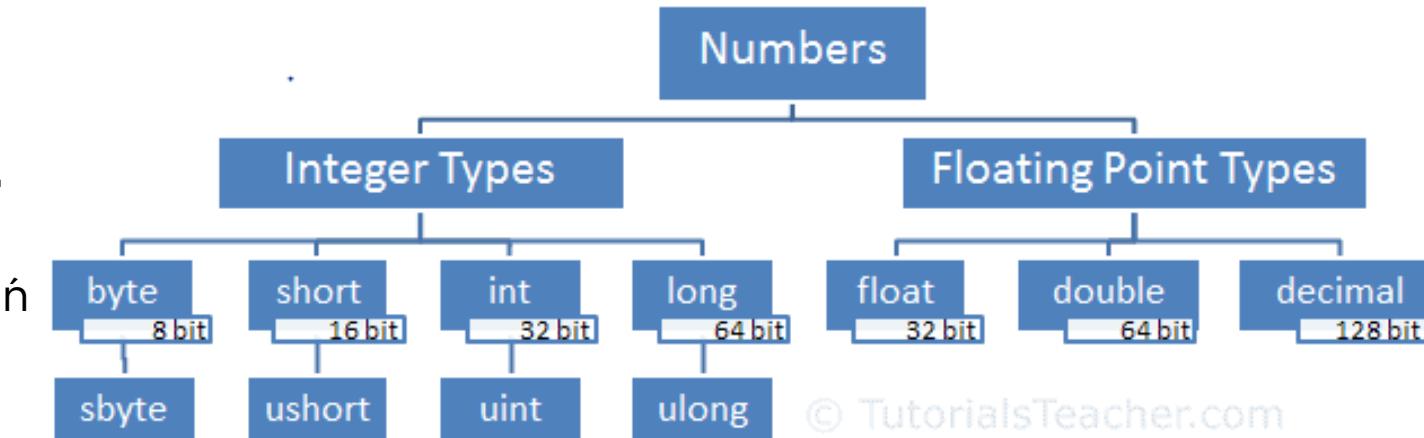
Trong C#, các kiểu dữ liệu số học được chia thành hai loại chính: số nguyên (Integral Types) và số dấu phẩy động (Floating-point Types).

- Số nguyên:

- **int**: Kiểu dữ liệu số nguyên 32-bit, có giá trị từ -2,147,483,648 đến 2,147,483,647.
- **long**: Kiểu dữ liệu số nguyên 64-bit, có giá trị từ -9,223,372,036,854,775,808 đến 9,223,372,036,854,775,807.
- **short**: Kiểu dữ liệu số nguyên 16-bit, có giá trị từ -32,768 đến 32,767.
- **byte**: Kiểu dữ liệu số nguyên không dấu 8-bit, có giá trị từ 0 đến 255.
- **sbyte**: Kiểu dữ liệu số nguyên 8-bit có dấu, có giá trị từ -128 đến 127.
- **uint, ulong, ushort**: Tương tự như int, long, short nhưng không dấu.

- Số thực:

- **float**: 32-bit, độ chính xác khoảng 7 chữ số thập phân.
- **double**: 64-bit, độ chính xác khoảng 15-16 chữ số thập phân.
- **decimal**: 128-bit, độ chính xác cao nhất, khoảng 28-29 chữ số thập phân, thường dùng cho tính toán tài chính.



```

c# Program.cs x
app > C# Program.cs
1 // Kiểu dữ liệu số nguyên (byte, short, int, long)
2 byte byteNumber = 255; // byte: từ 0 đến 255
3 short shortNumber = -32768; // short: từ -32768 đến 32767
4 int intNumber = 2147483647; // int: từ -2147483648 đến 2147483647
5 long longNumber = 9223372036854775808; // long: từ -9223372036854775808 đến 9223372036854775808
6
7 // Kiểu dữ liệu số nguyên không dấu (ushort, uint, ulong)
8 ushort ushortNumber = 65535; // ushort: từ 0 đến 65535
9 uint uintNumber = 4294967295; // uint: từ 0 đến 4294967295
10 ulong ulongNumber = 18446744073709551615; // ulong: từ 0 đến 18446744073709551615
11
12 // Kiểu dữ liệu số thực dấu phẩy động (float, double, decimal)
13 float floatNumber = 3.14F; // float: độ chính xác 7 chữ số, cần hậu tố 'F'
14 double doubleNumber = 3.14159265359; // double: độ chính xác 15-16 chữ số
15 decimal decimalNumber = 79228162514264337593543950335M; // decimal: độ chính xác 28-29 chữ số, cần hậu tố 'M'

```



CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

1

2

3

4

5

6

7

8

9

10

BIÊN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

5. Numeric toán tử

Toán tử số học (Arithmetic Operators):

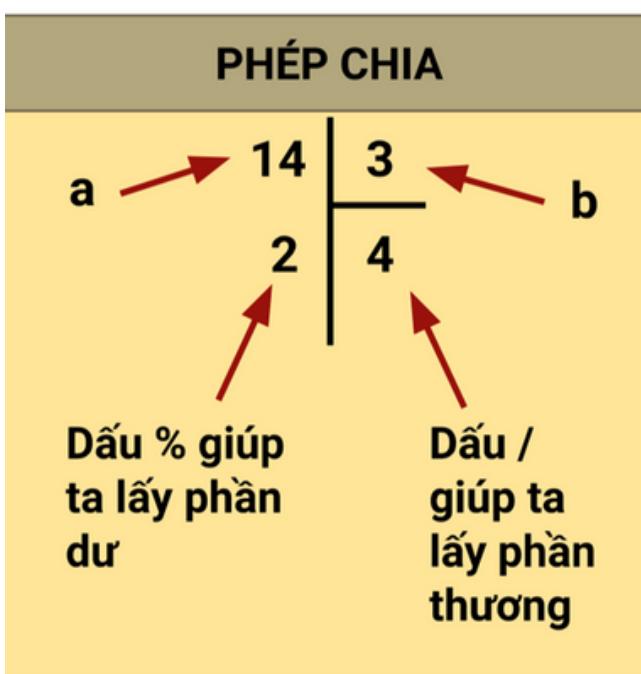
+: Cộng hai giá trị.

-: Trừ 2 giá trị.

*: Nhân hai giá trị.

/: Chia giá trị lấy kết quả là số thập phân.

%: Chia lấy phần dư của kết quả.



C# Program.cs ×

```
app > C# Program.cs
1 int a = 10;
2 int b = 3;
3 int sum = a + b; // 13
4 int diff = a - b; // 7
5 int product = a * b; // 30
6 int quotient = a / b; // 3
7 int remainder = a % b; // 1
8 a++; // a = 11
9 b--; // b = 2
```



CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

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

BIẾN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

5. Numberic toán tử

Toán tử gán (Assignment Operators):

- =: Gán giá trị bên phải cho biến bên trái.
- +=: Cộng giá trị bên phải với giá trị hiện tại của biến và gán lại.
- =: Trừ giá trị bên phải từ giá trị hiện tại của biến và gán lại.
- *=: Nhân giá trị bên phải với giá trị hiện tại của biến và gán lại.
- /=: Chia giá trị hiện tại của biến cho giá trị bên phải và gán lại.
- %=: Lấy phần dư của phép chia giá trị hiện tại của biến cho giá trị bên phải và gán lại.

```
C# Program.cs ×  
app > C# Program.cs  
1 int a = 10;  
2 int b = 3;  
3 int sum = a + b; // 13  
4 int diff = a - b; // 7  
5 int product = a * b; // 30  
6 int quotient = a / b; // 3  
7 int remainder = a % b; // 1  
8 a++; // a = 11  
9 b--; // b = 2
```



CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

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

BIẾN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

5. Numeric toán tử

Toán tử gán (Assignment Operators):

- =: Gán giá trị bên phải cho biến bên trái.
- +=: Cộng giá trị bên phải với giá trị hiện tại của biến và gán lại.
- =: Trừ giá trị bên phải từ giá trị hiện tại của biến và gán lại.
- *=: Nhân giá trị bên phải với giá trị hiện tại của biến và gán lại.
- /=: Chia giá trị hiện tại của biến cho giá trị bên phải và gán lại.
- %=: Lấy phần dư của phép chia giá trị hiện tại của biến cho giá trị bên phải và gán lại.

```
C# Program.cs ×  
app > C# Program.cs  
1 int a = 10;  
2 int b = 3;  
3 int sum = a + b; // 13  
4 int diff = a - b; // 7  
5 int product = a * b; // 30  
6 int quotient = a / b; // 3  
7 int remainder = a % b; // 1  
8 a++; // a = 11  
9 b--; // b = 2
```



CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

1

2

3

4

5

6

7

8

9

10

BIẾN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

5. Numeric toán tử

Toán tử null-coalescing:

- ?? : Trả về vế trái nếu không phải null, nếu vế trái là null thì trả về vế phải.
- ??= : Gán giá trị cho biến chỉ khi biến đó đang là null.

```
C# Program.cs X
demo > C# Program.cs
1 //Trả về giá trị của vế trái nếu nó không phải là null.
2 // Nếu vế trái là null, thì trả về giá trị của vế phải.
3 int? x = null;
4 int y = x ?? 10; // y sẽ là 10 vì x là null
5
6 int? z = null;
7 z ??= 5; // z sẽ được gán giá trị 5 vì ban đầu nó là null
8
9 int? a = 3;
10 a ??= 7; // a vẫn sẽ là 3 vì ban đầu nó không phải là null
11
12
13
14
15
16
17
```



CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

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

BIẾN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

6. Chuyển đổi kiểu dữ liệu

Ép kiểu trong C# là quá trình chuyển đổi dữ liệu từ kiểu này sang kiểu khác. Có hai loại chính:

- **Ép kiểu ngầm định (Implicit Casting):**
 - Tự động thực hiện khi chuyển từ kiểu nhỏ hơn sang kiểu lớn hơn (an toàn).
 - Ví dụ: int sang double
- **Ép kiểu tường minh (Explicit Casting):**
 - Cần thực hiện thủ công khi chuyển từ kiểu lớn hơn sang kiểu nhỏ hơn hoặc không tương thích (có thể mất dữ liệu).
 - Ví dụ: double sang int.

```
C# Program.cs X
demo > C# Program.cs
1 // $"..." gọi là Interpolated Strings, cho phép chèn giá trị biến vào chuỗi.
2 // Ép kiểu trong C# là quá trình chuyển đổi dữ liệu giữa các kiểu.
3
4 // Ép kiểu ngầm định (Implicit Casting)
5 // Tự động khi chuyển từ kiểu nhỏ hơn sang kiểu lớn hơn (an toàn).
6 int a = 10;
7 double b = a;
8 Console.WriteLine($"Giá trị của b: {b}"); // Output: Giá trị của b: 10
9 // Ép kiểu tường minh (Explicit Casting)
10 // Cần khi chuyển từ kiểu lớn hơn sang kiểu nhỏ hơn (có thể mất dữ liệu).
11 double x = 9.78;
12 int y = (int)x;
13 Console.WriteLine($"Giá trị của y: {y}"); // Output: Giá trị của y: 9
14
15 // Chuyển từ string sang int để tính toán
16 string str = "123";
17 int num = int.Parse(str); // Chuyển đổi chuỗi thành số nguyên
18 int result = num + 10;
19 Console.WriteLine($"Kết quả sau khi cộng: {result}"); // Output: Kết quả sau khi cộng: 133
20
21 // Chuyển từ int sang string
22 int number = 456;
23 string numberStr = number.ToString(); // Chuyển đổi số nguyên thành chuỗi
24 Console.WriteLine($"Chuỗi sau khi chuyển đổi: {numberStr}"); // Output: Chuỗi sau khi chuyển đổi: 456
25
26 // => Ép kiểu giúp đảm bảo dữ liệu phù hợp với yêu cầu xử lý.
```



CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

1

2

3

4

5

6

7

8

9

10

BIẾN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

6. Chuyển đổi kiểu dữ liệu

Ép kiểu trong C# là quá trình chuyển đổi dữ liệu từ kiểu này sang kiểu khác. Có hai loại chính:

- **Ép kiểu ngầm định (Implicit Casting):**

- Tự động thực hiện khi chuyển từ kiểu nhỏ hơn sang kiểu lớn hơn (an toàn).
- Ví dụ: int sang double

- **Ép kiểu tương minh (Explicit Casting):**

- Cần thực hiện thủ công khi chuyển từ kiểu lớn hơn sang kiểu nhỏ hơn hoặc không tương thích (có thể mất dữ liệu).
- Ví dụ: double sang int.

- **Chuyển đổi bằng phương thức Convert:**

- Phương thức **Convert** cung cấp các cách thức chuyển đổi giữa các kiểu dữ liệu một cách rõ ràng và linh hoạt.
- **Convert** cũng cung cấp khả năng xử lý ngoại lệ nếu việc chuyển đổi không hợp lệ (ví dụ khi chuyển từ string sang int nhưng string không chứa giá trị số hợp lệ).

Chuyển đổi bằng phương pháp ép kiểu

```
C# Program.cs X
demo > C# Program.cs

1 // $"..." gọi là Interpolated Strings, cho phép chèn giá trị biến vào chuỗi.
2 // Ép kiểu trong C# là quá trình chuyển đổi dữ liệu giữa các kiểu.
3
4 // Ép kiểu ngầm định (Implicit Casting)
5 // Tự động khi chuyển từ kiểu nhỏ hơn sang kiểu lớn hơn (an toàn).
6 int a = 10;
7 double b = a;
8 Console.WriteLine($"Giá trị của b: {b}"); // Output: Giá trị của b: 10
9 // Ép kiểu tương minh (Explicit Casting)
10 // Cần khi chuyển từ kiểu lớn hơn sang kiểu nhỏ hơn (có thể mất dữ liệu).
11 double x = 9.78;
12 int y = (int)x;
13 Console.WriteLine($"Giá trị của y: {y}"); // Output: Giá trị của y: 9
14
15 // Chuyển từ string sang int để tính toán
16 string str = "123";
17 int num = int.Parse(str); // Chuyển đổi chuỗi thành số nguyên
18 int result = num + 10;
19 Console.WriteLine($"Kết quả sau khi cộng: {result}"); // Output: Kết quả sau khi cộng: 133
20
21 // Chuyển từ int sang string
22 int number = 456;
23 string numberStr = number.ToString(); // Chuyển đổi số nguyên thành chuỗi
24 Console.WriteLine($"Chuỗi sau khi chuyển đổi: {numberStr}"); // Output: Chuỗi sau khi chuyển đổi: 456
25
26 // => Ép kiểu giúp đảm bảo dữ liệu phù hợp với yêu cầu xử lý.
```

Chuyển đổi bằng phương thức Convert:

```
22 string str = "123";
23 int number = Convert.ToInt32(str); // Chuyển đổi từ string sang int
24
25 double d = 123.45;
26 int i = Convert.ToInt32(d); // Chuyển đổi từ double sang int, i sẽ là 123
```



CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

1

2

3

4

5

6

7

8

9

10

BIẾN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

6. Chuyển đổi kiểu dữ liệu

- Toán tử trong ép kiểu:
 - Toán tử is và as trong C# được sử dụng để kiểm tra và ép kiểu dữ liệu một cách an toàn. Dưới đây là giải thích ngắn gọn về hai toán tử này.
- Toán tử is
 - Mục đích: Kiểm tra xem một đối tượng có thuộc một kiểu dữ liệu cụ thể hay không.
- Toán tử as
 - Mục đích: Thực hiện ép kiểu một cách an toàn.

```
C# Program.cs X
demo > C# Program.cs
1 // Toán tử is và as trong C# được sử dụng để kiểm tra và ép kiểu dữ liệu một cách an toàn.
2 // 1. Toán tử is
3 // Mục đích: Kiểm tra xem một đối tượng có thuộc một kiểu dữ liệu cụ thể hay không.
4 // Cách sử dụng:
5 // - Nếu đối tượng thuộc kiểu dữ liệu được kiểm tra, is trả về true.
6 // - Nếu không, is trả về false.
7 object value = "123";
8 bool isString = value is string;
9 Console.WriteLine($"Kiểm tra is: value có phải là string? {isString}");
10
11 // Trong ví dụ này, `value` là một chuỗi, nên `is string` trả về `true`.
12
13 // 2. Toán tử as
14 // Mục đích: Thực hiện ép kiểu một cách an toàn.
15 // Cách sử dụng:
16 // - Nếu ép kiểu thành công, as trả về đối tượng đã được ép kiểu.
17 // - Nếu không thành công (do không tương thích kiểu), as trả về null thay vì ném ra ngoại lệ.
18
19 string? strValue = value as string;
20 Console.WriteLine($"Kết quả ép kiểu bằng as: {strValue}");
21
22 // Trong ví dụ này, `value` được ép kiểu sang `string` bằng `as`.
23 // Nếu `value` không phải là `string`, `strValue` sẽ là `null`.
24
25 // Giải thích kết quả:
26 // - `is` kiểm tra và trả về `true` vì `value` là kiểu `string`.
27 // - `as` thực hiện ép kiểu và trả về giá trị "123" vì `value` có thể được ép kiểu thành `string`.
```



CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

1

2

3

4

5

6

7

8

9

10

BIÊN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

6. Toán tử khác

- Các từ khóa như `typeof`, `sizeof`, `default`, ... được gọi chung là biểu thức đặc biệt hoặc toán tử đặc biệt trong C#.
- Chúng được sử dụng để cung cấp các chức năng đặc biệt liên quan đến việc xử lý và kiểm soát kiểu dữ liệu, bộ nhớ, và các thao tác số học trong quá trình chạy chương trình. Mục đích chính là giúp lập trình viên viết mã an toàn, dễ bảo trì, và tối ưu hóa hiệu suất.

```
C# Program.cs X
demo > C# Program.cs
1 // typeof: Trả về đối tượng Type của một kiểu dữ liệu.
2 Type typeInfo = typeof(int);
3 Console.WriteLine($"typeof: {typeInfo}"); // Output: "System.Int32"
4
5 // sizeof: Trả về kích thước (số byte) của một kiểu dữ liệu nguyên thủy.
6 int size = sizeof(int);
7 Console.WriteLine($"sizeof: {size} bytes"); // Output: "4 bytes"
8
9 // default: Trả về giá trị mặc định của một kiểu dữ liệu.
10 int defaultValue = default(int);
11 Console.WriteLine($"default: {defaultValue}"); // Output: "0"
12
```



CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

1

2

3

4

5

6

7

8

9

10

BIẾN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

Thảo luận (Chia nhóm thảo luận)

1. Phép cộng (+) giữa String và số, String và String, String và char ?
2. Phép chia (/) trên số thực?
3. Kết luận :
 - String và số nối + nhau sẽ thành phép nối chuỗi
 - Trong biểu thức chỉ cần 1 biến là kiểu dữ liệu “lớn nhất”, theo thứ tự như sau: int < long < float < double
 - Đồng nhất kiểu dữ liệu?
 - Tránh mất dữ liệu



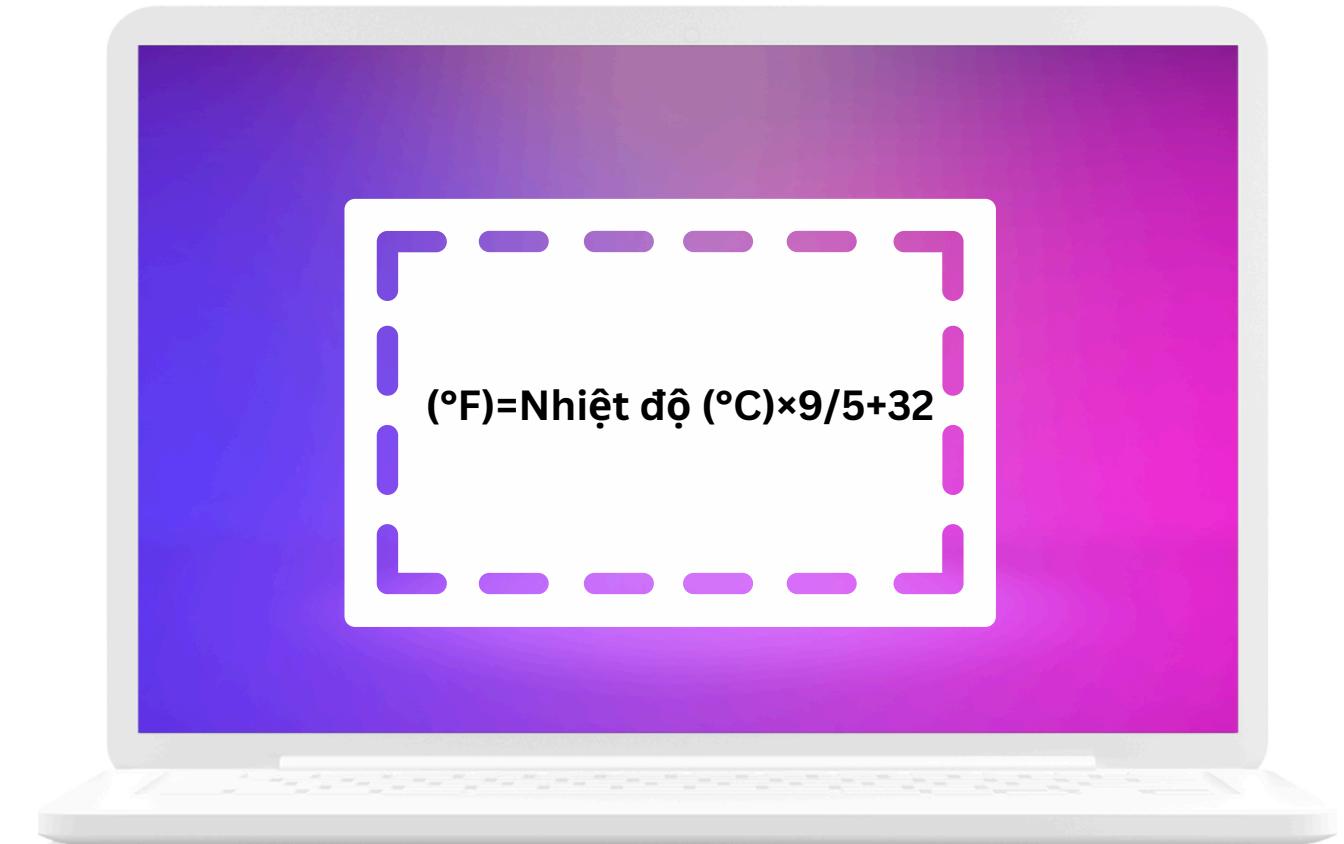
CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

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

BIÊN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

Bài tập:

Viết chương trình cho phép người dùng nhập vào độ C chuyển đổi sang độ F.





CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

1

2

3

4

5

6

7

8

9

10

BIÊN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

Bài tập:

Tính chu vi và diện tích hình tròn
Viết chương trình nhận vào bán kính
của hình tròn. Tính và in ra chu vi và diện tích của hình tròn đó.

Tính chu vi và diện tích hình tròn
Viết chương trình Python nhận vào
bán kính của hình tròn. Tính và in ra chu vi và diện tích của hình tròn đó.

$$P = 2 \pi r$$

$$A = \pi r^2$$





CÁC KHÁI NIỆM CƠ BẢN TRONG LẬP TRÌNH

1

2

3

4

5

6

7

8

9

10

BIÊN, KIỂU DỮ LIỆU VÀ GIÁ TRỊ

Bài tập:

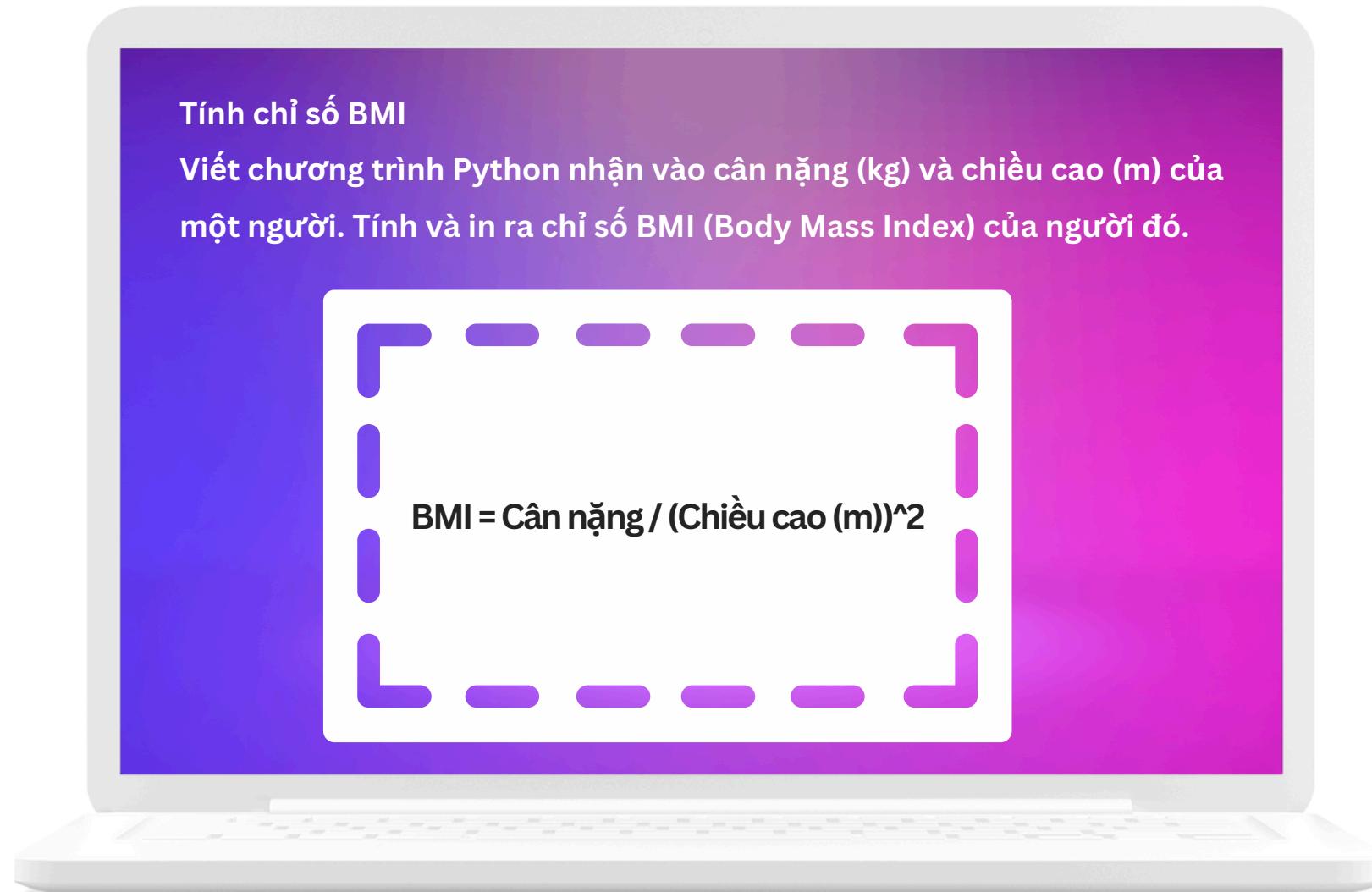
Viết chương trình nhập vào cân nặng (kg) và chiều cao (m) của một người. Tính và in ra chỉ số BMI (Body Mass Index) của người đó.



Tính chỉ số BMI

Viết chương trình Python nhận vào cân nặng (kg) và chiều cao (m) của một người. Tính và in ra chỉ số BMI (Body Mass Index) của người đó.

$$\text{BMI} = \text{Cân nặng} / (\text{Chiều cao (m)})^2$$





CÁCH TƯ DUY GIẢI QUYẾT VẤN ĐỀ TRONG LẬP TRÌNH

QUI TRÌNH ĐỂ GIẢI QUYẾT 1 VẤN ĐỀ

1

3

3

4

5

6

7

8

9

10

input**process****output**

1/Giá trị được cấu hình sẵn
(như hằng số hay hệ thống
mặc định)
2/Giá trị đề bài cho
3/Giá trị người dùng nhập
vào

- Các bước xử lý
- Dùng công thức hoặc if
else hoặc vòng lặp hoặc
kết hợp nhiều thứ

Giá trị chức năng cần hoặc
giá trị hiển thị ra màn hình

Biên s



ÔN TẬP XÂY DỰNG CONSOLE APP

Bài tập luyện thêm:

1

2

3

4

5

6

7

8

9

10

Bài tập 1: Tính số ngày trong tuần và số ngày lẻ

Yêu cầu người dùng nhập số ngày và tính toán bao nhiêu tuần và bao nhiêu ngày lẻ còn lại. Ví dụ, nếu người dùng nhập vào 10 ngày, kết quả sẽ là 1 tuần và 3 ngày.

Bài tập 2 : Tính tổng giá trị đơn hàng sau khi áp dụng giảm giá

Yêu cầu người dùng nhập vào giá trị của một đơn hàng và phần trăm giảm giá. Tính toán số tiền giảm giá và tổng số tiền phải thanh toán sau khi áp dụng giảm giá.

Bài tập 3: Chuyển đổi thời gian từ phút sang giờ và phút

Yêu cầu người dùng nhập vào một số phút và chuyển đổi số phút này thành giờ và phút. Ví dụ, nếu người dùng nhập vào 130 phút, kết quả sẽ là 2 giờ và 10 phút.

Bài tập 4: Tính tổng số tiền sau khi cộng thêm thuế VAT

Yêu cầu người dùng nhập vào số tiền gốc và tỷ lệ thuế VAT (ví dụ: 10%). Tính và in ra tổng số tiền sau khi đã cộng thêm thuế.

Bài tập 5: Chuyển đổi đơn vị tiền tệ

Yêu cầu người dùng nhập vào một số tiền bằng USD và tỷ giá chuyển đổi từ USD sang VND. Tính và in ra số tiền tương ứng bằng VND.

Bài tập 6: Tính số dư sau khi rút tiền từ tài khoản

Yêu cầu người dùng nhập vào số dư tài khoản hiện tại và số tiền muốn rút. Tính và in ra số dư còn lại sau khi rút tiền (lưu ý không kiểm tra số dư âm ở bài này).

Bài tập 7: Tính tốc độ trung bình

Yêu cầu người dùng nhập vào quãng đường đã đi (km) và thời gian đã đi (giờ). Tính và in ra tốc độ trung bình (km/h).

Bài tập 8: Tính tỷ lệ phần trăm

Yêu cầu người dùng nhập vào một số và một tổng số, sau đó tính và in ra tỷ lệ phần trăm của số đó trong tổng số.

Bài tập 9: Chuyển đổi từ km/h sang m/s

Yêu cầu người dùng nhập vào vận tốc bằng km/h và chuyển đổi nó sang m/s theo công thức: $m/s = km/h \div 3.6$. In ra kết quả sau khi chuyển đổi.

Bài tập 10: Tính lượng calo tiêu thụ

Yêu cầu người dùng nhập vào số phút đã tập thể dục và loại hình tập thể dục (chọn từ các giá trị đã định trước như chạy, đạp xe, bơi lội). Tính và in ra lượng calo tiêu thụ dựa trên số phút và loại hình tập thể dục (sử dụng hệ số calo tiêu thụ giả định cho mỗi loại hình).