LAB 1:
NỀN TẢNG NGÔN NGỮ LẬP TRÌNH C#

A. Mục tiêu

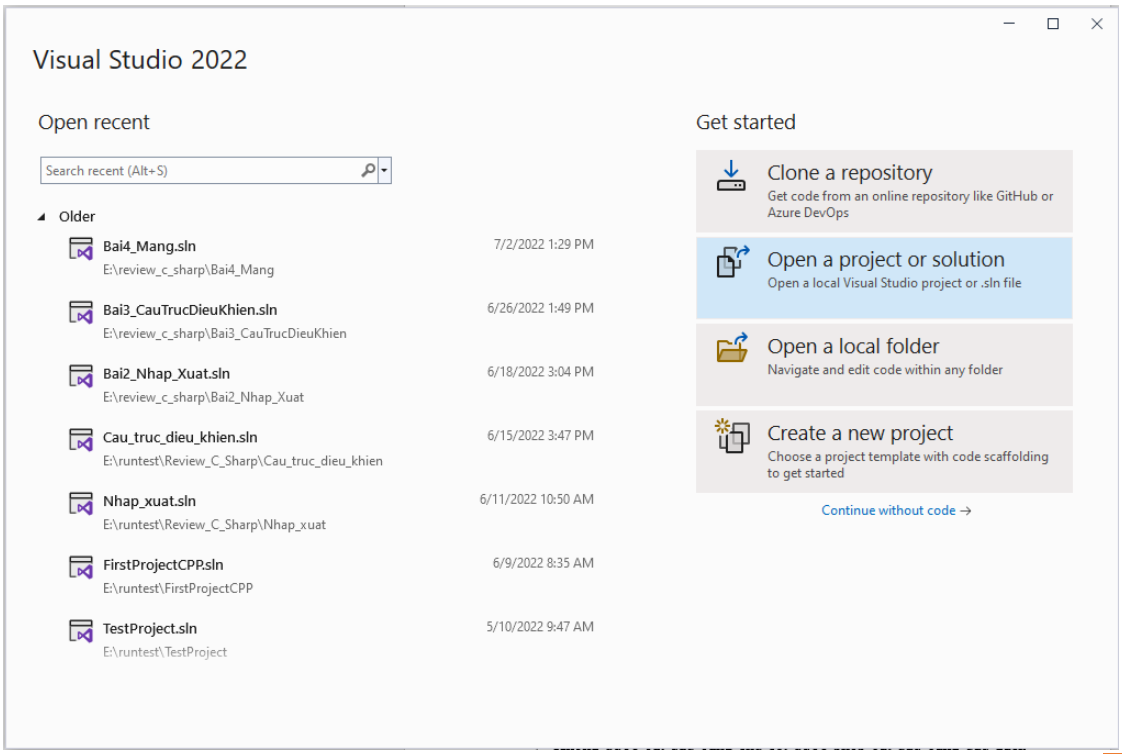
- Hướng dẫn sinh viên làm quen với ngôn ngữ lập trình C# qua thực hành các bài tập ứng dụng Console đơn giản.
- Hướng dẫn làm quen môi trường lập trình C# với phần mềm Microsoft Visual Studio 2022.
- Hướng dẫn sinh viên tập làm quen mới soạn thảo, chỉnh sửa mã nguồn, biên dịch, gỡ lỗi (debug), thực thi chương trình.
- Sinh viên ôn tập theo các đề mục giáo viên đã đề ra như: kiểu dữ liệu, biến số, các phép toán, biểu thức, câu lệnh điều kiện, các vòng lặp, từ khóa break, continue, mảng.

B. Nội dung

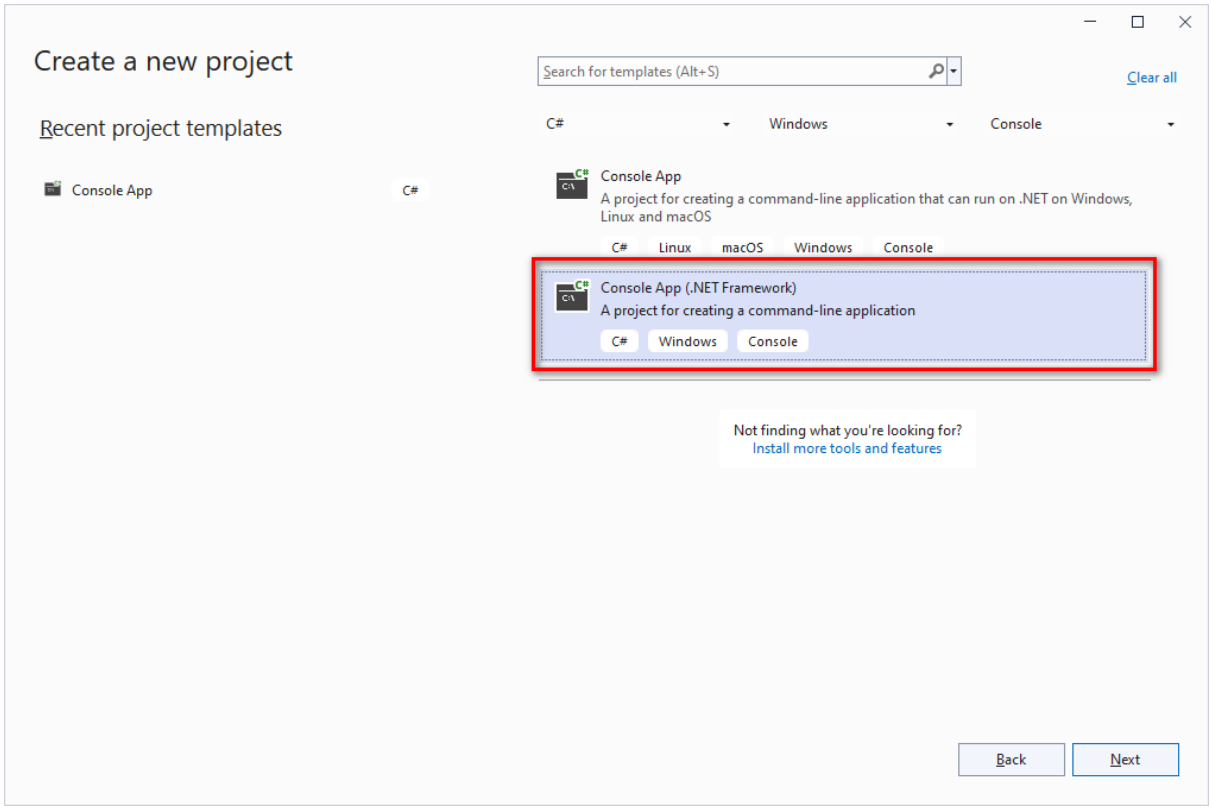
Thực hành 1: Sử dụng ngôn ngữ lập trình C# để tính và hiển thị chu vi, diện tích của hình chữ nhật có chiều dài a và chiều rộng b được nhập vào từ bàn phím.

1. Tạo Project Console Application trên Microsoft visual studio 2022

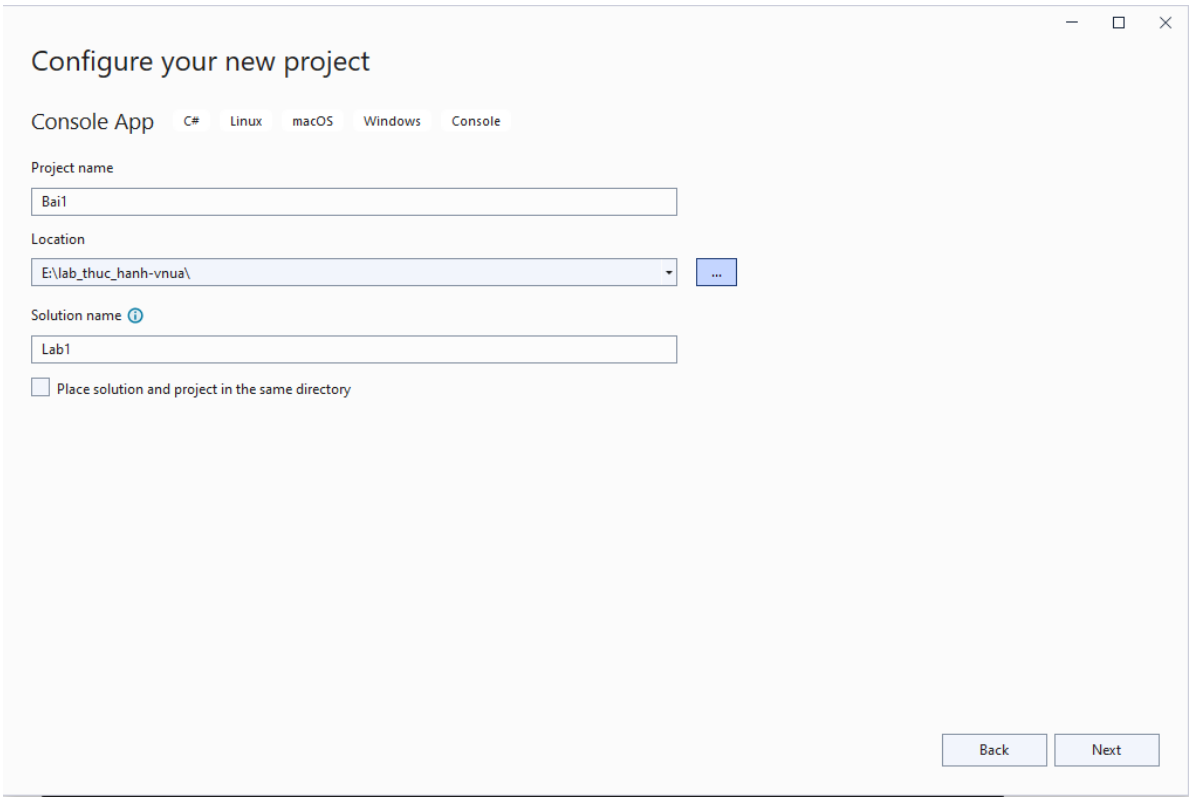
Bước 1: Khởi động chương trình visual studio 2022. Hình ảnh dưới đây là màn hình khởi động của ứng dụng:



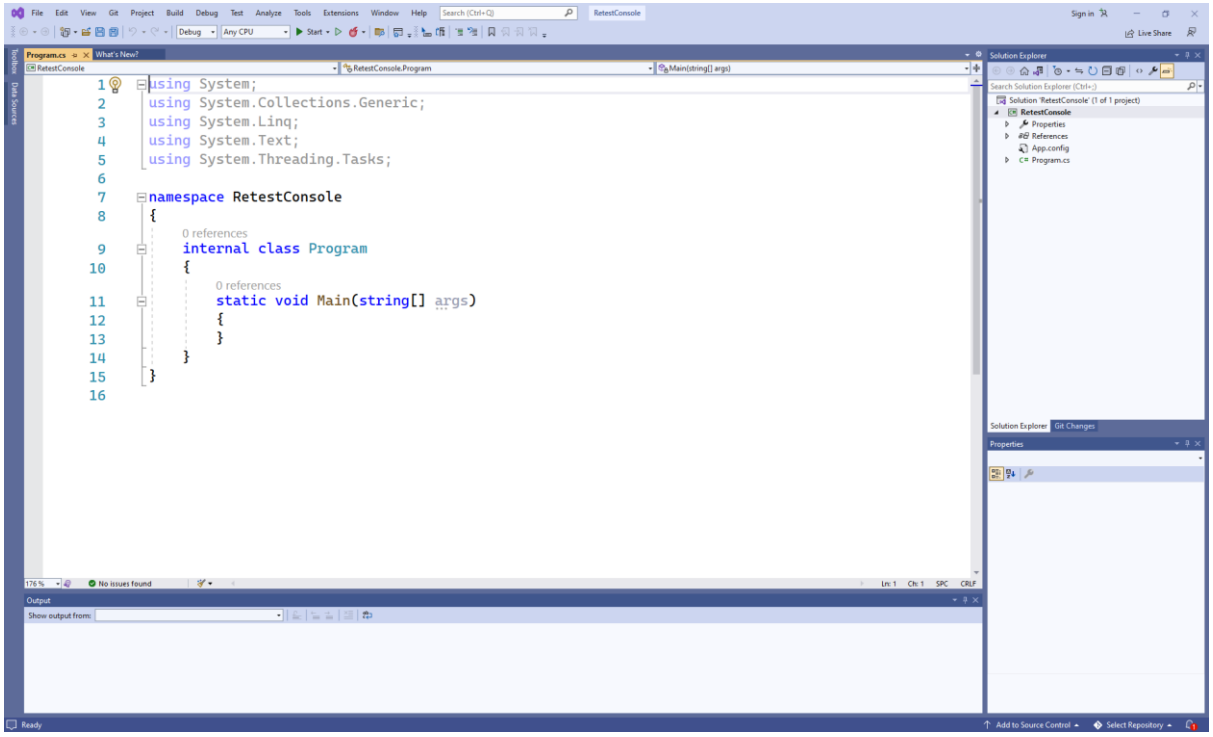
Bước 2: Chọn Create a new project trên màn hình khởi động. Trong màn hình tiếp theo, ở mục All languages → chọn C# trong danh sách ngôn ngữ lập trình. Mục All platforms → chọn Windows. Ở mục All project types → chọn Console.



Bước 3: Trong cửa sổ tiếp theo, bạn điền đầy đủ các thông tin về Project.



Đây là kết quả!



* Phân tích đề bài:

- Vào: chiều dài a, chiều rộng b.
- Ra: Chu vi P, diện tích S.

{Chương trình này thực hiện nhập chiều dài a và chiều rộng b từ bàn phím. Tính và hiển thị chu vi và diện tích của hình chữ nhật}

Program Tinh_CV_DT

1. Nhập dữ liệu

   Read(a, b);

2. Tính chu vi và diện tích

   P := (a + b) * 2;

   S:= a * b;

3. Hiển thị kết quả

   Write(P, S);

End.

Chương trình mẫu:

```
// Chương trình tính chu vi và diện tích hình chữ nhật
```

```
// Nhập dữ liệu
Console.Write("Nhap vao chieu dai: ");
double a = Convert.ToDouble(Console.ReadLine());
Console.Write("Nhap vao chieu rong: ");
double b = Convert.ToDouble(Console.ReadLine());

// Tính chu vi và diện tích
double P = (a + b) * 2;
double S = a * b;

// Hiển thị kết quả
Console.WriteLine("Chu vi la: " + P);
Console.WriteLine("Dien tich la: " + S);
```

2. Biến là gì?

- Biến là tên một ô nhớ trong bộ nhớ, dùng để chứa dữ liệu, giá trị của biến có thể thay đổi trong quá trình thực thi chương trình.
- Dữ liệu gồm 3 loại:
  + Dữ liệu đầu vào.
  + Dữ liệu đầu ra.
  + Dữ liệu trung gian trong quá trình xử lý dữ liệu.
- Quy tắc đặt tên biến:
  + Là một dãy các ký tự chữ cái (A-Z, a-z), chữ số (0-9), dấu gạch dưới.
  + Không được bắt đầu bằng chữ số, được phép bắt đầu bằng dấu gạch dưới.
  + Không được trùng với từ khóa trong C#.
  + C# PHÂN BIỆT KÝ TỰ HOA VÀ THƯỜNG.

3. Kiểu dữ liệu

C# là một ngôn ngữ lập trình kiểm soát chặt chẽ về kiểu dữ liệu. Nghĩa là bạn phải khai báo kiểu dữ liệu với mỗi một biến.

Bảng sau mô tả các kiểu dữ liệu được xây dựng sẵn:

Kiểu C#	Số byte	Kiểu .NET	Mô tả
byte	1	Byte	Số nguyên dương không dấu: 0 ÷ 255
char	2	Char	Ký tự Unicode
bool	1	Boolean	Giá trị logic true/ false
sbyte	1	Sbyte	Số nguyên có dấu: -128 ÷ 127
short	2	Int16	Số nguyên có dấu giá trị: -32768 ÷ 32767.
ushort	2	Uint16	Số nguyên không dấu: 0 ÷ 65.535
int	4	Int32	Số nguyên có dấu: -2.147.483.647 ÷ 2.147.483.647
uint	4	Uint32	Số nguyên không dấu: 0 ÷ 4.294.967.295
float	4	Single	Kiểu dấu chấm động giá trị xấp xỉ từ 3,4E-38 đến 3,4E+38, với 7 chữ số có nghĩa
double	8	Double	Kiểu dấu chấm động có độ chính xác gấp đôi, giá trị xấp xỉ từ 1,7E-308 đến 1,7E+308, với 1516 chữ số có nghĩa.
decimal	8	Decimal	Có độ chính xác đến 28 con số và giá trị thập phân, được dùng trong tính toán tài chính, kiểu này đòi hỏi phải có hậu tố "m" hay "M" theo sau giá trị.
long	8	Int64	Số nguyên có dấu: -9.223.370.036.854.775.808 ÷ 9.223.372.036.854.775.807
ulong	8	Uint64	Số nguyên không dấu: 0 ÷ 0xffffffffffffffff

Ví dụ về khai báo biến (declaration):

Kiểu_dữ_liệu tên_biến_1[, tên_biến_2, …] [=giá_trị_khởi_tạo];

```
// Ví dụ khai báo biến
int a = 1, b = 2;
int a1, a2 = 3;
string diaChi = "Hanoi";
float thanhTien = 456.9F;
bool kt = true;
char ch = 'a';
string s1 = "timoday";
string s2 = ".edu.vn";
string s3 = s1 + s2;
```

4. Các toán tử số học

+	Cộng
-	Trừ
*	Nhân

/	Chia
%	Chia lấy phần dư
++i, i++	Tăng giá trị i lên 1 đơn vị
--i, i--	Giảm giá trị i xuống 1 đơn vị

5. Chuyển đổi kiểu dữ liệu

Có 2 dạng chuyển đổi kiểu dữ liệu là ngầm định (implicit) và tường minh (explicit). Chuyển đổi theo dạng ngầm định do trình biên dịch thực hiện. Còn dạng tường minh thì bạn ép kiểu từ kiểu này sang kiểu khác.

5.1. Chuyển đổi ngầm (implicit conversion)

Chuyển đổi ngầm định sẽ được tự động thực hiện và bạn được đảm bảo rằng dữ liệu sẽ được không mất mát dữ liệu. Ví dụ như ta chuyển đổi một số nguyê kiểu short (2 byte) qua một số nguyên kiểu int (4 byte).

```
// Ví dụ chuyển đổi kiểu ngầm định
short x = 5;
int y = x;
```

Bảng sau chỉ rõ các kiểu được ngầm định chuyển đổi trong C#

Từ kiểu dữ liệu này..	Qua kiểu dữ liệu ...
sbyte	short, int, long, float, double, decimal
byte	short, ushort, int, uint, long, ulong, float, double, decimal
short	int, long, float, double, decimal
ushort	int, uint, long, ulong, float, double, decimal
int	long, float, double, decimal
uint	long, ulong, float, double, decimal
long, ulong	float, double, decimal
float	double
char	ushort, int, uint, long, ulong, float, double, decimal

5.2. Chuyển đổi tường minh (explicit conversion)

Nếu ta muốn chuyển đổi kiểu mà trình biên dịch không tự động thì bạn phải thực hiện ép kiểu. Khi ta ép một kiểu dữ liệu này chuyển đổi qua một kiểu dữ liệu, ta cố tình ép trình biên dịch phải tuân theo, thực hiện việc chuyển đổi.

```
// Ví dụ chuyển đổi kiểu tường minh
short x;
int y = 500;
x = (short)y;
```

*Kiểu dữ liệu bị ép về sẽ nằm trong cặp dấu ngoặc tròn, trước khi bị chuyển đổi.

Việc ép kiểu như này sẽ nguy hiểm đến dữ liệu. Vì nếu ép từ kiểu có miền lớn sang miền nhỏ hơn. Để đảm bảo, C# cung cấp cho bạn một toán tử **checked** dùng

kiểm tra việc ép kiểu có an toàn hay không. Nếu không an toàn thì nó sẽ tung ra một biệt lệ vào lúc chạy.

6. Nhập và xuất dữ liệu trong C#

6.1. Nhập dữ liệu từ bàn phím trong C#

Để nhập dữ liệu từ bàn phím, ta dùng phương thức **ReadLine** trong lớp **Console**. Phương thức **ReadLine** sẽ đọc theo từng dòng của dữ liệu nhập vào. Phương thức này trả về một giá trị nullable kiểu **string**. Khi đã hết dữ liệu nhập vào mà phương thức này vẫn được gọi phương thức sẽ trả về giá trị **null**.

Tuy nhiên, phương thức ReadLine sẽ trả về một xâu ký tự chứa nội dung được nhập từ bàn phím. Trong trường hợp bạn muốn có một số nguyên thì phải thực hiện việc ép kiểu. Chuyển đổi từ kiểu string sang kiểu int.

```
// Ví dụ về nhập dữ liệu từ bàn phím
int a = Convert.ToInt32(Console.ReadLine());
```

6.2. Hiển thị dữ liệu ra màn hình trong C#

C# hỗ trợ 2 phương thức giúp bạn hiển thị dữ liệu ra ngoài màn hình.

- Phương thức **WriteLine** cho phép chúng ta xuất dữ liệu ra màn hình và sẽ xuống dòng khi kết thúc câu lệnh.
- Ngược lại phương thức **Write** cũng sẽ cho phép chúng ta xuất dữ liệu ra màn hình nhưng khi kết thúc câu lệnh đó sẽ không in xuống dòng.

7. Các lệnh điều khiển

7.1. Lệnh điều kiện

7.1.1. Lệnh điều kiện if – thiếu

Cú pháp như sau:

if(<biểu thức điều kiện>)

{

    Tập các câu lệnh;

}

Cách thức hoạt động:

- Đầu tiên, ta xác định biểu thức điều kiện nhận giá trị True hoặc False.
- Nếu biểu thức điều kiện có giá trị True thì thực hiện Tập các câu lệnh.

– Còn ngược lại, ta không thực thi Tập các câu lệnh. Hay nói cách khác, ta chạy câu lệnh tiếp sau if.

Thực hành 2: Tìm số lớn hơn trong 2 số nguyên a và b được nhập vào từ bàn phím.

```
// Tìm max trong 2 số nguyên a, b

// Nhập dữ liệu
Console.Write("Nhap vao so nguyen a: ");
int a = Convert.ToInt32(Console.ReadLine());
Console.Write("Nhap vao so nguyen b: ");
int b = Convert.ToInt32(Console.ReadLine());

// Tìm số max
int max = 0;
if (a > b)
{
    max = a;
}
if (b > a)
{
    max = b;
}

// Hiển thị kết quả
Console.WriteLine("So lon hon trong 2 so la: " + max);
```

7.1.2. Lệnh điều kiện if – đủ

Cú pháp như sau:

if(<biểu thức điều kiện>)

{

Tập các câu lệnh của if;

}

else

{

Tập các câu lệnh của else;

}

Cách thức hoạt động:

– Đầu tiên, ta xác định biểu thức điều kiện nhận giá trị True hoặc False.
– Nếu biểu thức điều kiện có giá trị True thì thực hiện Tập các câu lệnh của if.

– Còn nếu biểu thức điều kiện có giá trị là False thì thực thi Tập các câu lệnh của else.

Tức là ở dạng if – đủ, ta sẽ kiểm tra cả 2 trường hợp: ĐÚNG thì làm những gì và SAI thì làm những gì.

Thực hành 3: Tìm số có giá trị lớn nhất trong 3 số nguyên a, b, c được nhập vào từ bàn phím.

```
// Tìm max trong 3 số nguyên a, b, c

// Nhập dữ liệu
Console.Write("Nhap vao so nguyen a: ");
int a = Convert.ToInt32(Console.ReadLine());
Console.Write("Nhap vao so nguyen b: ");
int b = Convert.ToInt32(Console.ReadLine());
Console.Write("Nhap vao so nguyen c: ");
int c = Convert.ToInt32(Console.ReadLine());

// Tìm số max
int max = 0;
if ((a > b) && (a > c))
{
    max = a;
}
else if ((b > a) && (b > c))
{
    max = b;
}
else
{
    max = c;
}

// Hiển thị kết quả
Console.WriteLine("So lon nhat trong 3 so la: {0}", max);
```

7.1.3. Lệnh điều kiện if – liệt kê

```
int diem = 8;

if (diem > 10 || diem < 0)
{
    Console.WriteLine("Diem khong hop le!");
}
else if (diem >= 9)
{
    Console.WriteLine("Xuat sac!");
}
else if (diem >= 8)
{
    Console.WriteLine("Gioi!");
}
else if (diem >= 6.5)
{
    Console.WriteLine("Kha!");
}
```

```
else if (diem >= 5)
{
    Console.WriteLine("Trung binh!");
}
else if (diem >= 3.5)
{
    Console.WriteLine("Yeu!");
}
else
{
    Console.WriteLine("Kem!");
}
```

7.2. Lệnh lựa chọn (switch … case)

Nếu để ý kỹ, bạn sẽ thấy if có những lúc hoạt động trên một tập các giá trị liên tục. Tuy nhiên, lệnh lựa chọn switch … case lại chỉ hoạt động với những giá trị rời rạc mà thôi.

Cú pháp:

case(biến_số_nguyên)

{

case 1:

lệnh của case 1;

break;

case 2:

lệnh của case 2;

break;

…

default:

lệnh của default;

break;

}

Thực hành 4: Cho biết một tháng có bao nhiêu ngày. Biết tháng, năm được nhập vào từ bàn phím.

```
// Tháng có bao nhiêu ngày

// Nhập dữ liệu
Console.Write("Nhap vao nam: ");
int nam = Convert.ToInt32(Console.ReadLine());
Console.Write("Nhap vao thang: ");
```

```
int thang = Convert.ToInt32(Console.ReadLine());

// Tìm số ngày trong tháng
switch(thang)
{
    case 1:
    case 3:
    case 5:
    case 7:
    case 8:
    case 10:
    case 12:
        Console.WriteLine("Thang co 31 ngay!");
        break;
    case 4:
    case 6:
    case 9:
    case 11:
        Console.WriteLine("Thang co 30 ngay!");
        break;
    case 2:
        if ((nam % 400 == 0) || ((nam % 4 == 0) && (nam % 100 != 0)))
        {
            Console.WriteLine("Thang co 29 ngay!");
        }
        else
        {
            Console.WriteLine("Thang co 28 ngay!");
        }
        break;
}
```

7.3. Lệnh lặp

7.3.1. Lệnh lặp for

Cú pháp:

for(biểu thức 1; biểu thức 2; biểu thức 3)

    Các câu lệnh;

Cách thức hoạt động:

- Đầu tiên, biểu thức 1 sẽ thực hiện khởi tạo các giá trị ban đầu.
- Tiếp đến kiểm tra điều kiện trong biểu thức 2.
- Nếu biểu thức điều kiện đúng thì thực hiện các câu lệnh. Ngược lại, dừng vòng lặp for.
- Cuối cùng, các giá trị cần thay đổi cho vòng lặp mới sẽ được thực hiện trong biểu thức 3. Và lại kiểm tra điều kiện ở biểu thức 2.

Ví dụ: tính tổng các số từ 1 đến 100.

```
int s = 0;
for(int i=1; i<=100; i++)
{
    s += 1;
}
```

7.3.2. Lệnh lặp while

Cú pháp:

while(biểu thức điều kiện)

Các câu lệnh;

Cách thức hoạt động:

Kiểm tra biểu thức điều kiện có nhận giá trị True hay không. Các câu lệnh được thực hiện lặp đi lặp lại cho đến khi nào biểu thức điều kiện mang giá trị False thì dừng lại.

Ví dụ:

```
int i = 1, s = 0;
while (i <= 100)
{
    s += i;
    i++;
}
```

7.3.3. Lệnh lặp do … while

Cú pháp:

do

{

Các câu lệnh;

}

while(biểu thức điều kiện);

Cách thức hoạt động:

Đầu tiên, thực hiện các câu lệnh.

Tiếp đến, kiểm tra điều kiện. Nếu điều kiện có giá trị True thì tiếp tục thực thi các lệnh lần tiếp theo. Ngược lại, dừng vòng lặp do … while.

Ví dụ:

```
int i = 1, s = 0;
do
{
    s += i;
    i++;
}
while (i <= 100);
```

Thực hành 5: Viết chương trình nhập vào một số nguyên n. Cho biết:

a) n là số chẵn hay số lẻ?
b) n là số âm hay số không âm?

Thực hành 6: Viết chương trình nhập vào 2 số thực dương chỉ chiều dài và chiều rộng của hình chữ nhật. In ra màn hình chu vi và diện tích của hình chữ nhật đó.

Thực hành 7: Viết chương trình nhập vào ba số thực chỉ độ dài của ba đoạn thẳng. Kiểm tra nếu ba đoạn thẳng này lập thành được một tam giá thì hiển thị chu vi và diện tích của tam giác đó.

Thực hành 8: Viết chương trình giải phương trình bậc 2: $ax^2 + bx + c = 0$.

8. Kiểu mảng (array) trong C#

8.1. Mảng trong C# là gì?

- Mảng (array) trong C# là một đối tượng cho phép chứa các phần tử có kiểu dữ liệu giống nhau, kích thước của mảng là cố định.
- Đặc trưng của mảng là cặp dấu ngoặc vuông [và]
- Về cách thức sử dụng. Các phần tử trong mảng được truy cập trực tiếp thông qua chỉ số. Mỗi một phần tử đều có một chỉ số riêng biệt. Chỉ số này có miền chạy từ 0 đến (n - 1), trong đó n là kích thước của mảng.

8.2. Khai báo và khởi tạo mảng trong C#

Ví dụ: để khai báo mảng arr có số lượng phần tử là 5, các phần tử có kiểu dữ liệu là int.

int[] arr = new int[5];

- *Câu lệnh trên đã tạo ra mảng arr có 5 phần tử là số 0.*
- Vừa khai báo và vừa khởi tạo mảng trong C#, ta lập trình:

```
int[] arr = { 3, 5, 1, 4, 9 };
int[] arr2 = new int[5] { 3, 5, 1, 4, 9 };
int[] arr3 = new int[] { 3, 5, 1, 4, 9 };
```

8.3. Một số tiện ích trong các thư viện C# hỗ trợ sử dụng mảng

- Để xác định kích thước hay số lượng các phần tử của mảng arr. Ta dùng hàm **Lengh**.
- Để sắp xếp mảng theo thứ tự tăng dần, ta dùng **Array.Sort(arr)**
- Sử dụng System.Linq, ta có nhiều phương thức xử lý mảng hữu ích khác như Min, Max, Sum

Thực hành 9: Tính tổng các phần tử trong mảng.

Thực hành 10: Dùng giải thuật sắp xếp chọn (Selection Sort) để sắp xếp tăng dần mảng các số nguyên. Mảng các số nguyên được lưu trong tệp văn bản có tên là “input_array.txt”.

Thực hành 11: Chèn thêm một số nguyên được nhạp vào từ bàn phím vào mảng đã sắp xếp tăng dần nhưng không làm mất tính tăng dần của mảng.

9. Collection trong C#

Link tham khảo:

https://viblo.asia/p/tong-quan-ve-collection-trong-c-WAyK8BmElxX