

MỤC LỤC

PHẦN 1: VÌ SAO JAVACRIPT, PYTHON, C# LÀ 3 NGÔN NGỮ LẬP TRÌNH PHỔ BIẾN NHẤT HIỆN NAY?	8
I. Các yếu tố quan trọng để đánh giá một ngôn ngữ lập trình phổ biến.....	8
II. Hiện nay có rất nhiều ngôn ngữ lập trình được sử dụng, tuy nhiên có một số ngôn ngữ lập trình phổ biến nhất hiện nay.....	8
III. JavaScript, Python và C# là 3 trong số những ngôn ngữ lập trình phổ biến nhất hiện nay vì nhiều lý do khác nhau.....	9
IV. Tính ứng dụng thực tế của 3 ngôn ngữ cho thấy các ngôn ngữ là phổ biến với các công nghệ hiện tại nhất	12
V. Ví dụ sử dụng các ngôn ngữ lập trình thực tế	15
PHẦN 2: KIẾN THỨC, CÁCH ÁP DỤNG, BÀI TẬP THỰC HÀNH - NGÔN NGỮ C#	24
Chương 1: Tổng quan về ngôn ngữ C#	25
I. Tại sao nên học ngôn ngữ lập trình C#.....	25
II. C# bắt nguồn từ đâu.....	26
Chương 2: Kiến thức, cách áp dụng và bài tập thực hành	28
I. Khái niệm về ngôn ngữ lập trình C#	28
II. Các tính năng của C#.....	28
III. Các trình biên dịch sử dụng để lập trình C#	29
IV. Bài tập: Tải và cài đặt trình biên dịch soạn thảo code ngôn ngữ lập trình C# (visual studio, visual studio code).....	30

BÀI TẬP VÂN DỤNG	44
Bài tập về mảng, list	44
I. Bài tập về mảng một chiều trong C#.....	44
II. Mảng hai chiều trong C#	82
List trong lập trình C#	115
I. Collection trong C#	115
II. Lớp List<T>	116

PHẦN 3: KIẾN THỨC, CÁCH ÁP DỤNG, BÀI TẬP THỰC HÀNH	
- NGÔN NGỮ JAVASCRIPT	127
Chương 1: Tổng quan về ngôn ngữ JavaScript	127
I. Lý do phải học ngôn ngữ lập trình Java Script?	127
II. Lịch sử phát triển ngôn ngữ lập trình Java Script?	128
III. Các bước để cài đặt môi trường lập trình JavaScript (JS).....	129
Chương 2: Kiến thức, cách áp dụng và bài tập thực hành	130
I. JavaScript là gì?	130
II. Để chạy ứng dụng JavaScript đầu tiên	130
III. Biến và Khai báo biến.....	132
IV. Phép gán (Assignment) và hằng số	133
V. JavaScript có các kiểu dữ liệu cơ bản (nguyên thuỷ) sau	135
VI. Các toán tử có trong ngôn ngữ lập trình JavaScript	136
BÀI TẬP VÂN DỤNG	139

I. Bài tập	139
II. Ứng dụng ngôn ngữ Javascript (HTML/CSS) để lập trình ứng dụng đăng nhập hệ thống.....	157
III. Lời giải	159

PHẦN 4: KIẾN THỨC, CÁCH ÁP DỤNG, BÀI TẬP THỰC HÀNH - NGÔN NGỮ PYTHON	171
Chương 1: Tổng quan về ngôn ngữ Python	171
I. Tại sao nên học Python?.....	171
II. Python có cú pháp đơn giản, dễ đọc và dễ viết, đồng thời cũng có nhiều tính năng tiên tiến	172
III. Cách cài đặt môi trường lập trình Python	172
Chương 2: Kiến thức, cách áp dụng và bài tập thực hành	174
I. Khái niệm về ngôn ngữ lập trình Python.....	174
II. Các kiểu dữ liệu cơ bản trong Python	175
III. Cú pháp	179
IV. Các module trong Python.....	189
BÀI TẬP VẬN DỤNG	191
I. Bài tập	191
II. Thao tác trên tập tin	192
III. Xử lý hình ảnh, file JSON, XML	197
IV. Ví dụ thực tế	208
V. Lời giải.....	211

LỜI MỞ ĐẦU

Với nền tảng công nghệ hiện tại, sự ra đời của nhiều ngôn ngữ lập trình và liên tục thay đổi, liên tục cập nhật thì mỗi ngôn ngữ lập trình đều có những ưu và nhược điểm chung. Đối với những người mới học về công nghệ thông tin nói chung và định hướng phát triển theo lập trình viên nói riêng thì việc xác định hướng và mục tiêu học tập là hết sức cần và quan trọng.

Với mục tiêu đó tác giả đã cho ra đời cuốn sách “**Sổ tay kiến thức + thực hành 3 ngôn ngữ lập trình JavaScript, Python, C#**”. Nền tảng công nghệ và được cập nhật thường xuyên 3 ngôn ngữ có cú pháp và câu lệnh rất dễ học cho các bạn mới tiếp xúc. Nhóm tác giả đã nghiên cứu, đọc rất kỹ về các Tutorial công nghệ và muốn hướng bạn đọc tới phần cần học và chi tiết nhất. Các chương của cuốn sách tập trung giới thiệu về lịch sử, tính ứng dụng và trình biên dịch để bắt đầu soạn thảo từ những chương trình đầu tiên như “**Hello world**”. Phần 2 của cuốn sách viết về ngôn ngữ C#, phần 3 của cuốn sách về Javascript và phần 4 của cuốn sách nói về ngôn ngữ cơ bản và hiện đại đang rất phổ biến hiện nay Python.

Để khai thác hiệu quả và tính áp dụng của người đọc nội dung của cuốn sách đi kèm là các ví dụ và bài tập thực hành được hướng dẫn bởi tác giả trên trình biên dịch của từng ngôn ngữ. Bạn đọc hãy tự mình “**nghiên ngẫm**” lý thuyết và thực hành lại từng đoạn lệnh để hiểu bản chất và áp dụng được vào các dự án thực tế.

Lưu ý: Mã QR nhận video tặng kèm của từng ngôn ngữ được đặt ở trang đầu tiên trước khi đi vào chi tiết của từng ngôn ngữ.

PHẦN 1:

Vì sao JavaScript, Python, C# là
3 ngôn ngữ lập trình phổ biến
nhất hiện nay?



PHẦN I:

VÌ SAO JAVASCRIPT, PYTHON, C# LÀ 3 NGÔN NGỮ LẬP TRÌNH PHỔ BIẾN NHẤT HIỆN NAY?

I. Các yếu tố quan trọng để đánh giá một ngôn ngữ lập trình phổ biến

- › Sức mạnh và linh hoạt: Ngôn ngữ lập trình cần có đủ tính năng để giải quyết các vấn đề phức tạp trong lập trình, đồng thời linh hoạt để có thể sử dụng trong nhiều lĩnh vực khác nhau.
- › Cộng đồng hỗ trợ: Một ngôn ngữ lập trình phổ biến cần có một cộng đồng lớn hỗ trợ, cung cấp các tài liệu, thư viện, framework, tool, giúp các nhà phát triển tối ưu hóa quá trình phát triển sản phẩm.
- › Khả năng tương thích: Ngôn ngữ lập trình cần tương thích với nhiều hệ thống khác nhau, từ hệ điều hành đến các phần mềm và thiết bị phần cứng.
- › Hiệu suất và tốc độ: Các ứng dụng lớn cần phải xử lý lượng dữ liệu lớn, nên ngôn ngữ lập trình phổ biến cần có hiệu suất tốt và tốc độ xử lý nhanh.
- › Độ dễ học và sử dụng: Ngôn ngữ lập trình phổ biến cần dễ học, có cú pháp đơn giản và dễ hiểu, cho phép nhà phát triển viết mã một cách dễ dàng và hiệu quả.
- › Bảo mật: Ngôn ngữ lập trình phổ biến cần có các tính năng bảo mật để đảm bảo an toàn cho các ứng dụng được xây dựng bằng nó.
- › Đa nền tảng: Ngôn ngữ lập trình phổ biến cần được hỗ trợ trên nhiều nền tảng khác nhau, từ desktop đến mobile, web và cloud.

II. Hiện nay có rất nhiều ngôn ngữ lập trình được sử dụng, tuy nhiên một số ngôn ngữ lập trình phổ biến nhất hiện nay

- › JavaScript: Được sử dụng cho phía client-side và server-side, phát triển web, ứng dụng di động, game, chatbot,...
- › Python: Dễ học, đơn giản và có thể áp dụng vào nhiều lĩnh vực như khoa học dữ liệu, trí tuệ nhân tạo, web, ứng dụng di động,...

- › Java: Được sử dụng rộng rãi trong lập trình phần mềm, web, ứng dụng di động, game, hệ thống,...
- › C++: Được sử dụng trong lập trình phần mềm, game, hệ thống, ứng dụng di động,...
- › C#: Được sử dụng trong lập trình Windows, game, web, hệ thống,...

Tuy nhiên, việc xem xét ngôn ngữ lập trình phổ biến nhất phụ thuộc và mục đích và phạm vi ứng dụng của chương trình mà ta muốn phát triển. Cuốn sách này sẽ tập trung đi sâu vào phân tích 3 ngôn ngữ lập trình phổ biến và được áp dụng nhiều trên các nền tảng cũng như các app thực tế dễ dàng nhất đó là 3 ngôn ngữ: JavaScript, Python và C#.

III. JavaScript, Python và C# là 3 trong số những ngôn ngữ lập trình phổ biến nhất hiện nay vì nhiều lý do khác nhau

- › JavaScript: Là một trong những ngôn ngữ lập trình phổ biến nhất hiện nay vì nó được sử dụng rộng rãi trong các ứng dụng web, đặc biệt là trong các ứng dụng web động. JavaScript được sử dụng để tạo ra các tính năng động như chuyển động, hiệu ứng và phản hồi người dùng. Bên cạnh đó, JavaScript cũng được sử dụng để phát triển ứng dụng di động và máy tính để bàn.
- › Python: Là một trong những ngôn ngữ lập trình phổ biến nhất hiện nay vì nó dễ học, dễ sử dụng và có tính linh hoạt cao. Python được sử dụng trong nhiều lĩnh vực khác nhau, bao gồm khoa học dữ liệu, trí tuệ nhân tạo, phát triển web, và ứng dụng máy tính.
- › C#: Là một trong những ngôn ngữ lập trình phổ biến nhất hiện nay vì nó được sử dụng rộng rãi trong phát triển ứng dụng Windows và ứng dụng di động. C# được sử dụng để phát triển các ứng dụng trên nền tảng .NET Framework của Microsoft, bao gồm các ứng dụng máy tính, ứng dụng di động, ứng dụng web và trò chơi điện tử.

Ngoài ra, cả ba ngôn ngữ này đều có cộng đồng lập trình viên đông đảo và hỗ trợ tốt từ các công ty lớn như Microsoft, Google, Amazon, Facebook,... Điều này cũng đóng góp vào việc tăng tính phổ biến và ứng dụng rộng rãi của các ngôn ngữ này.

Để tìm hiểu sâu về 3 ngôn ngữ lập trình này phổ biến nhất chúng ta đi sâu vào tìm hiểu lý do tại sao từng ngôn ngữ lại đang phổ biến và được áp dụng rộng rãi nhất.

1. JavaScript được lựa chọn là ngôn ngữ lập trình phổ biến nhất bởi vì:

- › Nó được sử dụng rộng rãi trong các ứng dụng web động: JavaScript được sử dụng để tạo ra các tính năng động và phản hồi người dùng trong các ứng dụng web, như các trang web đa phương tiện, trang web thương mại điện tử và các ứng dụng web động khác.
- › JavaScript là một ngôn ngữ lập trình dễ học: JavaScript là một ngôn ngữ lập trình dễ học và dễ sử dụng. Nó có cú pháp đơn giản và giống với các ngôn ngữ lập trình khác như C++, Java, và Python.
- › JavaScript có tính linh hoạt cao: JavaScript là một ngôn ngữ lập trình đa nền tảng, có thể chạy trên nhiều trình duyệt và hệ điều hành khác nhau. Điều này làm cho nó được sử dụng rộng rãi trong các ứng dụng web đa nền tảng.
- › JavaScript có cộng đồng lập trình viên đông đảo: JavaScript có một cộng đồng lập trình viên đông đảo, cung cấp rất nhiều tài liệu, thư viện và framework cho người dùng.
- › Công nghệ mới phát triển nhanh: JavaScript liên tục được cập nhật và phát triển, có nhiều tính năng mới và các công nghệ mới được phát triển nhanh chóng. Các framework như React, Angular và Vue.js đang ngày càng trở nên phổ biến và được sử dụng rộng rãi.
- › JavaScript là ngôn ngữ lập trình mở: JavaScript là một ngôn ngữ lập trình mã nguồn mở, điều này có nghĩa là nó được phát triển bởi cộng đồng và có thể sử dụng miễn phí. Điều này làm cho nó trở nên phổ biến hơn trong cộng đồng lập trình viên.

2. Python là một trong những ngôn ngữ lập trình phổ biến nhất hiện nay vì nhiều lý do, bao gồm:

- › Dễ học và dễ sử dụng: Python có cú pháp đơn giản và dễ hiểu, giúp người dùng mới học có thể nhanh chóng bắt đầu viết code.

- › **Đa năng:** Python được sử dụng trong nhiều lĩnh vực khác nhau, bao gồm khoa học dữ liệu, trí tuệ nhân tạo, web, game và nhiều lĩnh vực khác.
- › **Cộng đồng lớn:** Python có một cộng đồng lập trình viên đông đảo và nhiều người dùng trên toàn thế giới, cung cấp nhiều tài liệu và hỗ trợ cho người dùng mới.
- › **Sử dụng rộng rãi** trong khoa học dữ liệu và trí tuệ nhân tạo: Python được sử dụng rộng rãi trong các ứng dụng khoa học dữ liệu và trí tuệ nhân tạo, như là một công cụ để xử lý và phân tích dữ liệu, và để xây dựng các mô hình máy học.
- › **Công cụ và thư viện đa dạng:** Python có nhiều công cụ và thư viện đa dạng để giúp các lập trình viên tăng tốc độ phát triển ứng dụng như: NumPy, Pandas, SciPy, TensorFlow, và PyTorch.
- › **Tính di động cao:** Python có thể chạy trên nhiều nền tảng khác nhau, bao gồm Windows, Linux, MacOS và các thiết bị di động.
- › **Ngôn ngữ mã nguồn mở:** Python là một ngôn ngữ lập trình mã nguồn mở, điều này có nghĩa là ai cũng có thể sử dụng nó miễn phí và phát triển các ứng dụng mà không cần trả phí cho giấy phép sử dụng.

3. Ngôn ngữ C# (C Sharp) là một trong những ngôn ngữ lập trình phổ biến nhất hiện nay vì nhiều lý do, bao gồm:

- › **Được phát triển bởi Microsoft:** Microsoft là một trong những công ty công nghệ lớn nhất thế giới, và C# là ngôn ngữ lập trình được phát triển bởi Microsoft, do đó nó có sự hỗ trợ mạnh mẽ từ Microsoft và được tích hợp tốt với các sản phẩm Microsoft như Visual Studio và .NET Framework.
- › **Đa năng:** C# được sử dụng trong nhiều lĩnh vực khác nhau, bao gồm phát triển ứng dụng máy tính, ứng dụng web, trò chơi điện tử và ứng dụng di động.
- › **Hỗ trợ đa nền tảng:** C# không chỉ chạy trên nền tảng Windows mà còn được hỗ trợ trên các nền tảng khác như Linux và macOS.
- › **Hỗ trợ cho .NET Framework:** C# được phát triển để tương tác với .NET Framework, một nền tảng mạnh mẽ cho phát triển ứng dụng trên Windows.

- › Hiệu suất cao: C# được thiết kế để có hiệu suất cao và có thể xử lý các tác vụ phức tạp.
- › An toàn và bảo mật: C# có tính năng an toàn và bảo mật, cho phép phát triển các ứng dụng an toàn và bảo mật hơn.
- › Cộng đồng lớn: C# có một cộng đồng lập trình viên đông đảo và nhiều người dùng trên toàn thế giới, cung cấp nhiều tài liệu và hỗ trợ cho người dùng mới.
- › Tính chất hướng đối tượng: C# là một ngôn ngữ lập trình hướng đối tượng, cho phép các lập trình viên thiết kế và triển khai các lớp và đối tượng, giúp quản lý mã nguồn dễ dàng hơn.

Tóm lại, C# là một ngôn ngữ lập trình mạnh mẽ và đa năng, với sự hỗ trợ mạnh mẽ từ Microsoft và cộng đồng lập trình viên đông đảo, do đó nó đã trở thành một trong những ngôn ngữ lập trình phổ biến nhất hiện nay.

IV. Tính ứng dụng thực tế của 3 ngôn ngữ cho thấy các ngôn ngữ là phổ biến với các công nghệ hiện tại nhất

1. Tính ứng dụng thực tế của ngôn ngữ Javascript

- › Javascript: Phát triển ứng dụng web: JavaScript được sử dụng để phát triển các ứng dụng web tương tác và động, nhưng cũng có thể sử dụng để phát triển các ứng dụng web phía server.
- › Phát triển game: JavaScript được sử dụng để phát triển các trò chơi trực tuyến, đặc biệt là các trò chơi casual và những trò chơi dành cho thiết bị di động.
- › Phát triển ứng dụng di động: JavaScript có thể sử dụng để phát triển ứng dụng di động cho cả iOS và Android, thông qua các công nghệ như React Native và PhoneGap.
- › Tích hợp với các công nghệ mới như AI và Machine Learning: JavaScript có thể được sử dụng để phát triển các ứng dụng liên quan đến trí tuệ nhân tạo và machine learning, nhờ vào các thư viện như TensorFlow.js.
- › Tạo các chức năng tương tác trực tiếp với người dùng: JavaScript có thể sử dụng để tạo các chức năng tương tác trực tiếp với người dùng

trên các trang web, chẳng hạn như hiển thị thông báo, xử lý đăng nhập và các hoạt động khác.

Tóm lại, JavaScript là một ngôn ngữ lập trình có nhiều ứng dụng thực tế và rất quan trọng đối với việc phát triển các ứng dụng web, game, ứng dụng di động và các công nghệ mới như AI và Machine Learning.

2. Tính ứng dụng thực tế của ngôn ngữ Python: Python là một ngôn ngữ lập trình đa năng và linh hoạt, có thể được sử dụng để phát triển các ứng dụng phong phú và đa dạng. Dưới đây là một số ứng dụng thực tế của Python:

- › Phát triển web: Python có thể được sử dụng để phát triển các ứng dụng web động với các framework như Django, Flask, Pyramid, Falcon, Tornado, Sanic, FastAPI, Bottle, Web2py, CherryPy.
- › Trí tuệ nhân tạo: Python là một trong những ngôn ngữ được sử dụng rộng rãi nhất trong lĩnh vực Trí tuệ nhân tạo (AI) với các thư viện và framework như TensorFlow, PyTorch, Keras, Scikit-learn, Pandas, NumPy.
- › Xử lý ngôn ngữ tự nhiên: Python được sử dụng để phát triển các ứng dụng xử lý ngôn ngữ tự nhiên (NLP) như chatbot, phân tích cảm xúc, dịch thuật, lọc tin rác với các thư viện như NLTK, Gensim, TextBlob, spaCy.
- › Xử lý dữ liệu: Python là một trong những ngôn ngữ phổ biến nhất được sử dụng để xử lý và phân tích dữ liệu với các thư viện như Pandas, NumPy, Matplotlib, Seaborn, Bokeh.
- › Game: Python cũng có thể được sử dụng để phát triển các trò chơi đơn giản với các thư viện như Pygame, PyOpenGL.
- › Hệ thống: Python được sử dụng để quản lý hệ thống, tự động hóa và đơn giản hóa các tác vụ hành chính với các thư viện như Fabric, Salt, Ansible.
- › Ứng dụng di động: Python cũng có thể được sử dụng để phát triển các ứng dụng di động với các framework như Kivy, BeeWare.

- › Kiểm thử và tự động hóa: Python có thể được sử dụng để phát triển các công cụ kiểm thử và tự động hóa với các thư viện như unittest, pytest, Robot Framework.

Trên đây là một số ví dụ về ứng dụng thực tế của Python, nhưng thực tế thì Python có thể được sử dụng để phát triển hầu hết mọi loại ứng dụng, từ các ứng dụng máy tính đơn giản đến các hệ thống phức tạp và quy mô lớn.

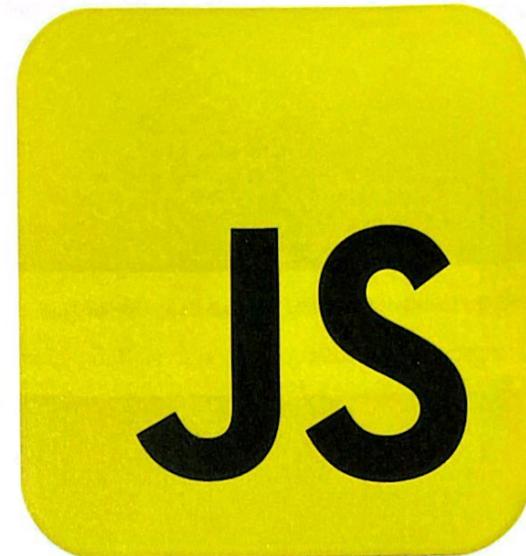
3. Tính ứng dụng thực tế của ngôn ngữ lập trình C#: Ngôn ngữ lập trình C# là một trong những ngôn ngữ phổ biến nhất hiện nay với rất nhiều ứng dụng thực tế. Dưới đây là một số ví dụ:

- › Phát triển ứng dụng Windows: C# là một trong những ngôn ngữ phổ biến nhất cho phát triển các ứng dụng Windows, đặc biệt là ứng dụng desktop. C# kết hợp với .NET Framework giúp các nhà phát triển tạo ra các ứng dụng có giao diện đồ họa, hiệu suất cao và an toàn.
- › Phát triển ứng dụng web: C# có thể được sử dụng để phát triển các ứng dụng web thông qua ASP.NET, một nền tảng web phổ biến. Với ASP.NET, các nhà phát triển có thể tạo ra các ứng dụng web có tính năng đa dạng, bảo mật cao và dễ bảo trì.
- › Phát triển game: C# là ngôn ngữ lập trình chính cho Unity, một trong những công cụ phát triển game phổ biến nhất. Unity cho phép các nhà phát triển tạo ra các trò chơi đa nền tảng với đồ họa 2D và 3D và các tính năng phong phú như vật lý, đa người chơi và xử lý AI.
- › Phát triển ứng dụng di động: Xamarin, một công cụ phát triển ứng dụng di động, cho phép các nhà phát triển sử dụng C# để phát triển các ứng dụng di động cho iOS và Android. Với Xamarin, các nhà phát triển có thể tận dụng các tính năng của C# để tạo ra các ứng dụng di động mạnh mẽ và đa nền tảng.
- › Phát triển các ứng dụng dữ liệu: C# có thể được sử dụng để phát triển các ứng dụng dữ liệu như các công cụ quản lý cơ sở dữ liệu hoặc các ứng dụng phân tích dữ liệu. Với các tính năng như LINQ và Entity Framework, các nhà phát triển có thể tương tác với cơ sở dữ liệu một cách dễ dàng và hiệu quả.

V. Ví dụ sử dụng các ngôn ngữ lập trình thực tế

1. Ngôn ngữ lập trình Javascript:

1.1 Logo của Javascript:



1.2 Ví dụ về cách sử dụng Javascript để thay đổi nội dung phần tử HTML:

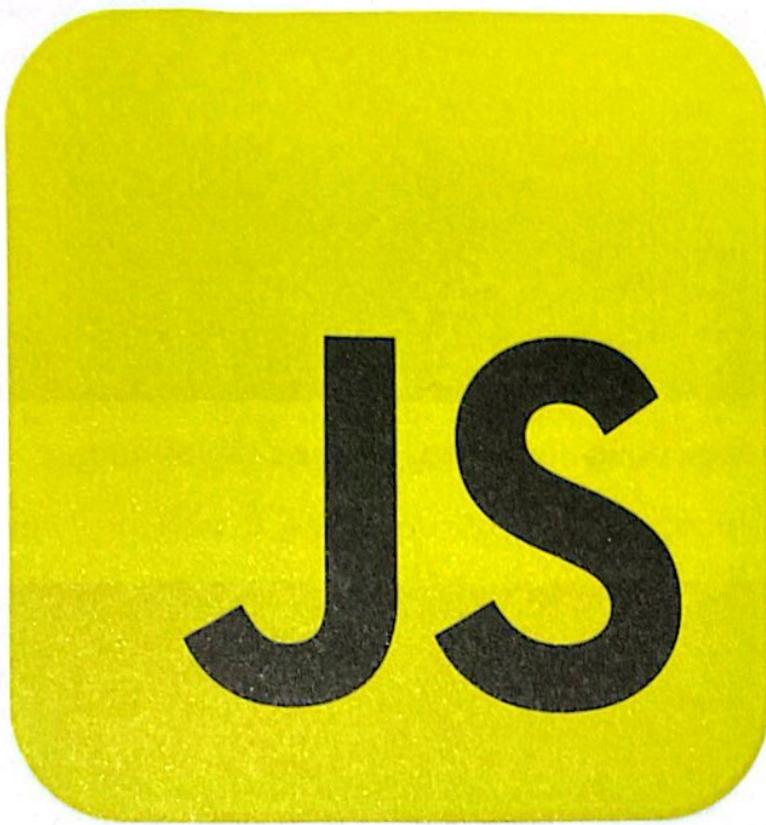
```
javascript
// Lấy phần tử có id là "demo"
var demo = document.getElementById("demo");

// Thay đổi nội dung của phần tử đó
demo.innerHTML = "Hello, world!";
```

V. Ví dụ sử dụng các ngôn ngữ lập trình thực tế

1. Ngôn ngữ lập trình Javascript:

1.1 Logo của Javascript:



1.2 Ví dụ về cách sử dụng Javascript để thay đổi nội dung của một phần tử HTML:

```
javascript
// Lấy phần tử có id là "demo"
var demo = document.getElementById('demo');

// Thay đổi nội dung của phần tử đó
demo.innerHTML = 'Hello, world!';
```

1.3 Cách sử dụng Javascript để kiểm tra xem một giá trị có phải là số hay không:

```
javascript
// Kiểm tra xem giá trị có phải là số hay không
function isNumber(value) {
    return typeof value === 'number' && isFinite(value);
}

// Sử dụng hàm kiểm tra số
console.log(isNumber(42)); // true
console.log(isNumber('42')) // false
```

1.4 Ví dụ về cách sử dụng Javascript để tạo một đối tượng:

Để tạo một đối tượng trong JavaScript, chúng ta sử dụng cú pháp sau:

```
var objectName = {
    propertyName1: PropertyValue1,
    propertyName2: PropertyValue2,
    methodName: function() {
        // method body
    }
};
```

Trong đó:

- ObjectName là tên của đối tượng,
- PropertyName1 và propertyName2 là các thuộc tính của đối tượng với giá trị tương ứng PropertyValue1 và PropertyValue2. methodName là tên của phương thức với nội dung trong cặp dấu ngoặc nhọn.

Ví dụ: Chúng ta có thể tạo một đối tượng person với các thuộc tính và age và phương thức sayHello như sau:

```
var person = {
    name: "Tom",
    age: 31,
    sayHello: function() {
        console.log("Hello, my name is " + this.name + " and I'm " + this.age + " years old.");
    }
};
```

Ở đây, this là một từ khóa đại diện cho đối tượng hiện tại. Phương thức sayHello sử dụng thuộc tính name và age của person để in ra một câu chào hỏi.

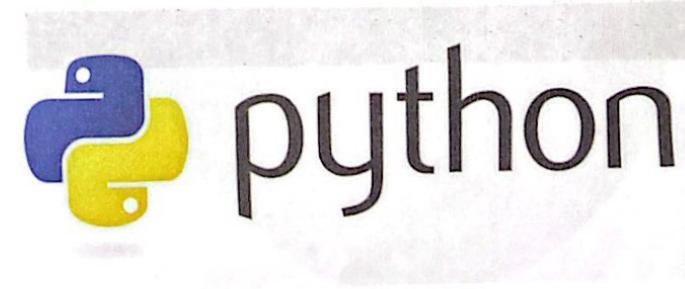
Chúng ta có thể gọi phương thức sayHello của đối tượng person sử dụng cú pháp sau:

```
person.sayHello();
```

Kết quả sẽ là "Hello, my name is Tom and I'm 31 years old." sẽ xuất hiện trên console.

2. Ngôn ngữ lập trình Python:

2.1 Logo ngôn ngữ lập trình Python:



Ví dụ: Chúng ta có thể tạo một đối tượng **person** với các thuộc tính **name** và **age** và phương thức **sayHello** như sau:

```
var person = {
    name: "Tom",
    age: 31,
    sayHello: function() {
        console.log("Hello, my name is " + this.name + " and I'm " + this.age + " years old.");
    }
};
```

Ở đây, **this** là một từ khóa đại diện cho đối tượng hiện tại (**person**) phương thức **sayHello** sử dụng thuộc tính **name** và **age** của đối tượng **person** để in ra một câu chào hỏi.

Chúng ta có thể gọi phương thức **sayHello** của đối tượng **person** bằng cách sử dụng cú pháp sau:

```
person.sayHello();
```

Kết quả sẽ là "**Hello, my name is Tom and I'm 31 years old.**" sẽ được in ra trên console.

2. Ngôn ngữ lập trình Python:

2.1 Logo ngôn ngữ lập trình Python:



2.2 Code Python cơ bản để in ra dòng chữ "Hello, World!":

```
python
```

 Copy code

```
print("Hello, world!")
```

Output:

```
Hello, world!
```

 Copy code

2.3 Một ví dụ về code Python để tính tổng của hai số:

```
python
```

 Copy code

```
a = 5  
b = 7  
c = a + b  
print(c)
```

Output:

```
12
```

 Copy code

2.4 Đọc dữ liệu từ tệp văn bản:

```
python
with open("example.txt", "r") as f:
    data = f.read()
print(data)
```

 Copy code

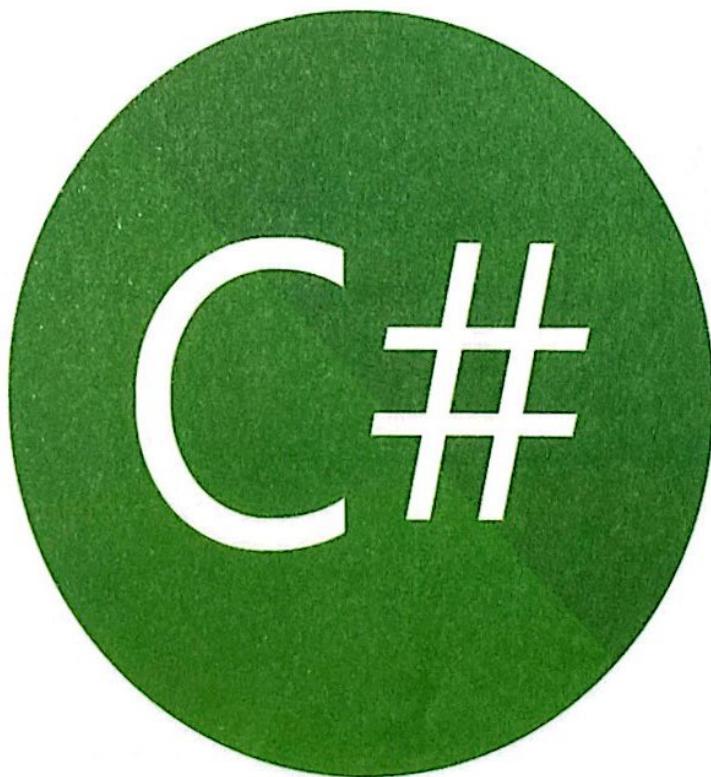
Output (nếu tệp `example.txt` chứa nội dung "This is an example."):

```
csharp
This is an example.
```

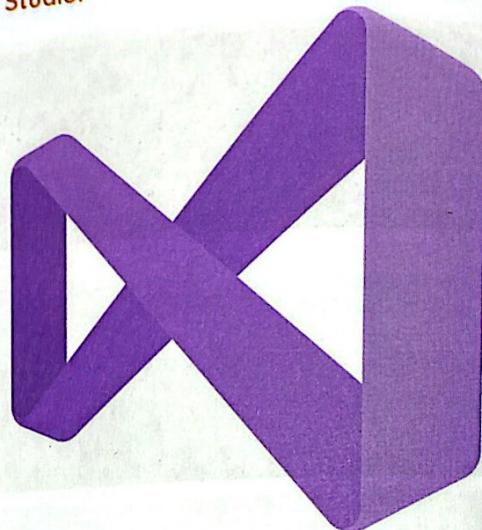
 Copy code

3. Ngôn ngữ lập trình C#

3.1 Logo C#:



3.2 IDE Visual Studio:



3.3 Thư viện .Net:



3.4 Khai báo biến

csharp

 Copy code

```
int age = 25;  
string name = "John";
```

3.5 Lập trình hướng đối tượng:

csharp

 Copy code

```
class Person {  
    private string name;  
    private int age;  
  
    public Person(string name, int age) {  
        this.name = name;  
        this.age = age;  
    }  
  
    public string GetName() {  
        return this.name;  
    }  
  
    public int GetAge() {  
        return this.age;  
    }  
}
```

3.6 Các câu lệnh điều kiện

csharp

 Copy code

```
if (age > 18) {  
    Console.WriteLine("Bạn đủ tuổi để trưởng thành.");  
}  
else {  
    Console.WriteLine("Bạn chưa đủ tuổi để trưởng thành.");  
}
```

3.7 Vòng lặp

```
csharp Copy code
for (int i = 0; i < 10; i++) {
    Console.WriteLine("Giá trị của i là: " + i);
}
```

Với việc học và sử dụng các ngôn ngữ này, bạn có thể phát triển các ứng dụng đa nền tảng, từ các ứng dụng web đến các ứng dụng di động và máy tính để bàn. Ngoài ra việc sử dụng các ngôn ngữ này cũng mang lại cho bạn nhiều cơ hội việc làm và tiếp cận với các dự án phát triển lớn.



PHẦN 2:

Kiến thức, cách áp dụng, bài tập
thực hành - ngôn ngữ C#



PHẦN 2:

KIẾN THỨC, CÁCH ÁP DỤNG, BÀI TẬP THỰC HÀNH - NGÔN NGỮ C#

Chương 1: Tổng quan về ngôn ngữ C#

I. Tại sao nên học ngôn ngữ lập trình C#

Dưới đây là một số lý do nên học ngôn ngữ lập trình C#:

- › C# là một ngôn ngữ lập trình hiện đại và phổ biến được sử dụng rộng rãi trong phát triển ứng dụng desktop, web, di động và game. C# được phát triển bởi Microsoft và được hỗ trợ rộng rãi bởi cộng đồng phát triển.
- › C# là một ngôn ngữ lập trình dễ học và dễ sử dụng cho người mới bắt đầu học lập trình. Cú pháp của C# tương đối giống với các ngôn ngữ lập trình khác như: Java, C++, do đó bạn có thể dễ dàng chuyển đổi sang C# từ những ngôn ngữ khác.
- › C# cung cấp rất nhiều tính năng hữu ích cho phát triển ứng dụng, bao gồm các tính năng như đa luồng, tính kế thừa, quản lý bộ nhớ tự động và phát triển ứng dụng đám mây.
- › C# là một ngôn ngữ lập trình được sử dụng rộng rãi trong lĩnh vực phát triển game. Nó cung cấp một số tính năng mạnh mẽ như: XNA Framework, Unity và Unreal Engine, cho phép phát triển game với chất lượng cao và đa nền tảng.
- › C# là một ngôn ngữ lập trình rất tốt cho phát triển ứng dụng đám mây. Với sự hỗ trợ của các dịch vụ như: Azure và Amazon Web Services, phát triển các ứng dụng đám mây trở nên dễ dàng và tiện lợi hơn.

Tóm lại, học C# là một lựa chọn tốt cho những người muốn học lập trình vì tính linh hoạt và sự phổ biến của nó. C# là một trong những ngôn ngữ lập trình quan trọng nhất hiện nay và có rất nhiều cơ hội việc làm dành cho những người có kỹ năng C#.

II. C# bắt nguồn từ đâu

C# (C sharp) là một ngôn ngữ lập trình hiện đại được phát triển bởi Microsoft vào những năm 2000. C# được thiết kế nhằm mục đích đơn giản hóa phát triển ứng dụng và tăng tính bảo mật so với các ngôn ngữ lập trình trước đó của Microsoft như C++ và Visual Basic.

C# được phát triển bởi một nhóm kỹ sư của Microsoft dưới sự chỉ đạo của Anders Hejlsberg. Trước đó, Anders Hejlsberg cũng đã tham gia phát triển các ngôn ngữ lập trình khác như Turbo Pascal và Delphi. C# được phát triển dựa trên các kinh nghiệm của Microsoft về phát triển các ứng dụng và hệ thống phần mềm trong môi trường Windows.

C# được phát hành lần đầu tiên vào năm 2002 và ngay lập tức được đón nhận rộng rãi trong cộng đồng lập trình viên. C# là một trong những ngôn ngữ lập trình quan trọng nhất hiện nay và được sử dụng rộng rãi cho phát triển ứng dụng desktop, web, mobile và game.

› Cài đặt môi trường lập trình C#: Để cài đặt môi trường lập trình C#, bạn có thể thực hiện các bước sau:

Bước 1: Tải và cài đặt **Visual Studio Community** (phiên bản miễn phí) từ trang web chính thức của Microsoft tại địa chỉ:

<https://visualstudio.microsoft.com/downloads/>.

Sau đó, khởi động **Visual Studio**.

Bước 2: Trong **Visual Studio**, chọn “**Create a new project**” để tạo một dự án mới.

Bước 3: Chọn “**Console App (.NET Core)**” để tạo một ứng dụng console C#.

Bước 4: Đặt tên cho dự án và chọn đường dẫn lưu trữ dự án.

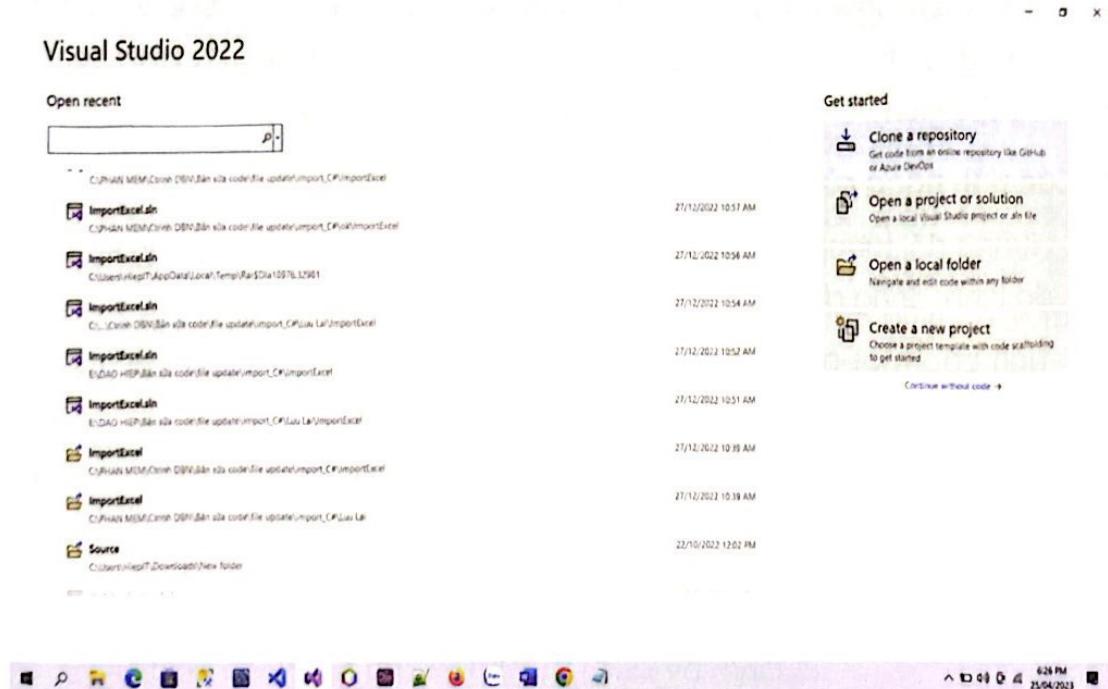
Bước 5: Tiếp theo, bạn sẽ được chuyển đến trình soạn thảo mã nguồn của **Visual Studio**, nơi bạn có thể viết mã C#.

Bước 6: Để chạy chương trình, bạn có thể nhấn tổ hợp phím **F5** hoặc chọn “**Debug**” > “**Start Debugging**”.

Nếu bạn gặp vấn đề khi cài đặt hoặc sử dụng Visual Studio, bạn có thể tìm kiếm hướng dẫn chi tiết hoặc tham khảo tài liệu hỗ trợ trên trang web chính thức của Microsoft.

Lưu ý rằng, Visual Studio là môi trường lập trình tích hợp đầy đủ và hỗ trợ nhiều ngôn ngữ lập trình khác nhau ngoài C#. Nếu bạn muốn chỉ cài đặt trình biên dịch C# và trình soạn thảo mã nguồn, bạn có thể cài đặt .NET Core SDK từ trang web chính thức của Microsoft tại địa chỉ: <https://dotnet.microsoft.com/download>. Sau khi cài đặt, bạn có thể sử dụng trình soạn thảo mã nguồn như Visual Studio Code hoặc Visual Studio Community để viết mã C# và biên dịch chương trình.

- Giao diện trình biên dịch sau khi cài đặt thành công sẽ có dạng:



Chương 2: Kiến thức, cách áp dụng và bài tập thực hành

I. Khái niệm về ngôn ngữ lập trình C#

C# (C sharp) là một ngôn ngữ lập trình hiện đại được phát triển bởi Microsoft vào những năm 2000. C# là một ngôn ngữ lập trình hướng đối tượng (Object-oriented programming - OOP) và được thiết kế nhằm mục đích đơn giản hóa phát triển ứng dụng và tăng tính bảo mật so với các ngôn ngữ lập trình trước đó của Microsoft như C++ và Visual Basic.

C# được phát triển dựa trên các kinh nghiệm của Microsoft về phát triển các ứng dụng và hệ thống phần mềm trong môi trường Windows. C# là một ngôn ngữ lập trình đa nền tảng, có thể chạy trên nhiều hệ điều hành khác nhau như: Windows, Linux, MacOS, iOS, Android,...

C# là một ngôn ngữ lập trình mạnh mẽ và linh hoạt, có thể được sử dụng để phát triển nhiều loại ứng dụng như desktop, web, mobile và game. C# cũng được tích hợp sẵn trong Framework .NET của Microsoft, cung cấp một số thư viện và công cụ hỗ trợ cho các lập trình viên.

II. Các tính năng của C#

- › Hướng đối tượng: C# hỗ trợ các tính năng hướng đối tượng như kế thừa, đa hình, đóng gói,...
- › Tính bảo mật cao: C# có tính năng bảo mật cao nhờ vào các cơ chế như kiểm soát quyền truy cập, quản lý bộ nhớ,...
- › Đơn giản và dễ học: C# có cú pháp dễ hiểu và dễ học, giúp cho việc phát triển ứng dụng dễ dàng và nhanh chóng.
- › Tính linh hoạt và đa nền tảng: C# có thể chạy trên nhiều nền tảng khác nhau và có thể tích hợp với các ngôn ngữ lập trình khác.
- › Các thư viện và công cụ hỗ trợ: C# tích hợp sẵn trong Framework .NET và cung cấp các thư viện và công cụ hỗ trợ phát triển ứng dụng.

III. Các trình biên dịch sử dụng để lập trình C#

C# là một ngôn ngữ lập trình được phát triển bởi Microsoft và được tích hợp trong .NET Framework, do đó Microsoft cung cấp các trình biên dịch (**compiler**) cho C# để phát triển ứng dụng trên nền tảng Windows. Các trình biên dịch C# phổ biến nhất hiện nay bao gồm:

- › Visual Studio: Đây là môi trường phát triển tích hợp (**Integrated Development Environment - IDE**) của Microsoft, cung cấp một bộ công cụ đầy đủ cho phát triển ứng dụng C# trên Windows. Visual Studio cung cấp trình biên dịch, trình gỡ lỗi, trình quản lý mã nguồn, trình kiểm tra phiên bản,...

Link tải: <https://visualstudio.microsoft.com/downloads/>

- › Visual Studio Code: Là một ứng dụng mã nguồn mở và miễn phí của Microsoft, hỗ trợ phát triển ứng dụng C# trên nhiều nền tảng khác nhau như Windows, macOS và Linux. Visual Studio Code cung cấp các tính năng như trình biên dịch, trình gỡ lỗi, hỗ trợ Git,...

Link tải: <https://code.visualstudio.com/download>

- › Mono: Là một nền tảng mã nguồn mở để phát triển ứng dụng đa nền tảng, bao gồm cả ứng dụng C#. Mono cung cấp một trình biên dịch cho C#, cho phép phát triển ứng dụng trên nhiều nền tảng như Windows, Linux, macOS và Android.

Link tải: <https://www.mono-project.com/>

- › .NET Core: Là một nền tảng mã nguồn mở của Microsoft cho phát triển các ứng dụng đa nền tảng, bao gồm cả ứng dụng C#. .NET Core cung cấp một trình biên dịch cho C#, cho phép phát triển ứng dụng trên nhiều nền tảng như Windows, Linux và macOS.

Link tải: <https://dotnet.microsoft.com/en-us/download>

- › Ngoài các trình biên dịch trên, còn có một số trình biên dịch C# miễn phí khác như SharpDevelop, Rider,... tùy vào nhu cầu và mục đích sử dụng, lập trình viên có thể lựa chọn trình biên dịch phù hợp để phát triển ứng dụng C#.

IV. Bài tập: Tải và cài đặt trình biên dịch soạn thảo code ngôn ngữ lập trình C# (visual studio, visual studio code)

1. Biến trong ngôn ngữ lập trình C#

1.1 Tìm hiểu về biến khi lập trình C#

Biến khi lập trình:

- › Do lập trình viên tạo ra nó,...
- › Việc lập trình sẽ rất khó khăn hoặc có thể dễ.
- › Được cấp phát ở nhớ Ram.
- › Tồn tại quá trình ứng dụng chạy.
- › Kiểu dữ liệu sẽ quyết định.

Trong lập trình C#, biến được sử dụng để lưu trữ giá trị dữ liệu trong quá trình thực thi chương trình. Mỗi biến trong C# phải được khai báo trước khi sử dụng và có kiểu dữ liệu cụ thể.

Cú pháp khai báo biến trong C# như sau:

```
<data_type> <variable_name>;
```

Trong đó:

- › <data_type> là kiểu dữ liệu của biến.

Ví dụ: Int, float, double, string, bool,...

- › <variable_name> là tên của biến.

Ví dụ:

```
Int age;
```

```
Float weight;
```

```
String name;
```

```
Bool isMale;
```

Sau khi khai báo biến, ta có thể gán giá trị cho biến đó bằng dấu =.

Ví dụ:

```
age = 20;  
weight = 60.5f;  
  
name = "John Doe";  
  
isMale = true;
```

Hoặc ta có thể khai báo và gán giá trị cho biến trong một câu lệnh duy nhất:

```
int age = 20;  
  
float weight = 60.5f;  
  
string name = "John Doe";  
  
bool isMale = true;
```

Ngoài ra, trong C# còn hỗ trợ khái niệm biến tạm thời (**temporary variable**), còn được gọi là biến cục bộ (**local variable**). Biến tạm thời được khai báo bên trong một khối lệnh (**block**) và chỉ có thể sử dụng trong khối lệnh đó. Khi khối lệnh kết thúc, biến tạm thời sẽ bị hủy và không còn sử dụng được nữa.

Ví dụ:

```
void Function() {  
  
    int x = 10; // biến tạm thời  
  
    Console.WriteLine(x);  
  
} // x bị hủy sau khi kết thúc hàm
```

Ngoài ra, trong C# còn có các loại biến đặc biệt như hằng số (**constant**) và biến tĩnh (**static variable**), được sử dụng để lưu trữ các giá trị không thay đổi và chia sẻ giữa các đối tượng trong chương trình.

1.2 Tìm hiểu về kiểu dữ liệu

Trong lập trình C#, kiểu dữ liệu là một khái niệm quan trọng, cho phép khai báo biến và hàm với kiểu dữ liệu cụ thể. Kiểu dữ liệu cho phép chương trình xử lý dữ liệu theo các quy tắc cụ thể như thực hiện các phép toán, so sánh giá trị,... C# hỗ trợ nhiều kiểu dữ liệu khác nhau, bao gồm:

- › Kiểu số nguyên (integer): Int, long, short, byte, sbyte, uint, ulong, ushort.
- › Kiểu số thực (floating-point): Float, double, decimal.
- › Kiểu ký tự (character): Char.
- › Kiểu chuỗi (string): String.
- › Kiểu logic (boolean): Bool.
- › Kiểu đối tượng (object): Object.
- › Kiểu giá trị null (null value): Null.

Mỗi kiểu dữ liệu có các đặc tính riêng, ví dụ như kiểu số nguyên có thể lưu trữ các giá trị nguyên không âm hoặc âm, trong khi kiểu số thực được sử dụng để lưu trữ các giá trị số thực, có chứa dấu phẩy động.

Ngoài các kiểu dữ liệu cơ bản trên, C# còn cho phép tạo kiểu dữ liệu mới thông qua cú pháp struct hoặc class, cho phép định nghĩa các kiểu dữ liệu phức tạp hơn, được tạo ra từ sự kết hợp của các kiểu dữ liệu cơ bản.

- › // khai báo biến kiểu số nguyên

```
int a = 10;
```

```
long b = 1000000;
```

```
byte c = 255;
```

- › // khai báo biến kiểu số thực

```
float x = 3.14f;
```

```
double y = 3.141592653589793;
```

```
decimal z = 100.5m;
```

- › // khai báo biến kiểu kí tự

```
char ch = 'A';
```
- › // khai báo biến kiểu chuỗi

```
string str = "Hello world";
```
- › // khai báo biến kiểu logic

```
bool isTrue = true;
```
- › // khai báo biến kiểu đối tượng

```
object obj = new object();
```
- › // khai báo biến kiểu giá trị null

```
int? nullableInt = null;
```
- › // khai báo kiểu dữ liệu mới thông qua struct

```
struct Person {
    public string name;
    public int age;
}
```

1.3 Tìm hiểu về các toán tử trong lập trình C#

Toán tử trong lập trình C# là các ký hiệu được sử dụng để thực hiện các phép tính hoặc phép so sánh giữa các giá trị hoặc biểu thức. C# hỗ trợ nhiều loại toán tử khác nhau, bao gồm:

- › Toán tử số học: + (cộng), - (trừ), * (nhân), / (chia), % (chia lấy dư).
- › Toán tử gán: = (gán), += (cộng và gán), -= (trừ và gán), *= (nhân và gán), /= (chia và gán), %= (chia lấy dư và gán).
- › Toán tử so sánh: == (bằng), != (khác), < (nhỏ hơn), > (lớn hơn), <= (nhỏ hơn hoặc bằng), >= (lớn hơn hoặc bằng).
- › Toán tử logic: && (và), || (hoặc), ! (phủ định).
- › Toán tử điều kiện: ?: (toán tử ba ngôi), if...else (rẽ nhánh điều kiện).

- › Toán tử thực hiện phép tính trên một biến trước hoặc sau đó: ++ (tăng lên một đơn vị), -- (giảm đi một đơn vị).

Ví dụ về việc sử dụng toán tử trong C#:

- › // toán tử số học

```
int a = 10 + 5; // a = 15
```

```
int b = 10 - 5; // b = 5
```

```
int c = 10 * 5; // c = 50
```

```
int d = 10 / 5; // d = 2
```

```
int e = 10 % 3; // e = 1
```

- › // toán tử gán

```
int f = 10;
```

```
f += 5; // f = 15
```

```
f -= 5; // f = 10
```

```
f *= 2; // f = 20
```

```
f /= 2; // f = 10
```

```
f %= 3; // f = 1
```

- › // toán tử so sánh

```
bool g = (10 == 5); // g = false
```

```
bool h = (10 != 5); // h = true
```

```
bool i = (10 < 5); // i = false
```

```
bool j = (10 > 5); // j = true
```

```
bool k = (10 <= 5); // k = false
```

```
bool l = (10 >= 5); // l = true
```

- › // toán tử logic
 - bool m = true;
 - bool n = false;
 - bool o = (m && n); // o = false
 - bool p = (m || n); // p = true
 - bool q = !m; // q = false
- › // toán tử điều kiện
 - int r = (10 > 5) ? 100 : 200; //

2. Cấu trúc điều khiển, vòng lặp

Trong lập trình, cấu trúc điều khiển vòng lặp được sử dụng để lặp lại một khối lệnh nhiều lần. C# hỗ trợ ba loại vòng lặp sau:

- › Vòng lặp while: Vòng lặp này được sử dụng để lặp lại một khối lệnh cho đến khi một điều kiện nhất định được thỏa mãn. Cú pháp của vòng lặp while như sau:

```
while (condition)
{
    // Khối lệnh được lặp lại
}
```

Ví dụ:

```
int i = 0;
while (i < 10)
{
    Console.WriteLine(i);
    i++;
}
```

- › Vòng lặp do-while: Vòng lặp này hoạt động tương tự như vòng lặp while, tuy nhiên khối lệnh trong vòng lặp do-while được thực hiện ít nhất một lần trước khi kiểm tra điều kiện để lặp lại khối lệnh. Cú pháp của vòng lặp do-while như sau:

```
do
{
    // Khối lệnh được lặp lại
} while (condition);
```

Ví dụ:

```
int i = 0;
do
{
    Console.WriteLine(i);
    i++;
} while (i < 10);
```

- › Vòng lặp for: Vòng lặp này được sử dụng để lặp lại một khối lệnh một số lần nhất định. Cú pháp của vòng lặp for như sau:

```
for (initialization; condition; increment/decrement)
{
    // Khối lệnh được lặp lại
}
```

Trong đó:

Initialization: Khởi tạo giá trị ban đầu cho biến lặp.

Condition: Kiểm tra điều kiện để lặp lại khối lệnh.

Increment/decrement: Tăng/giảm giá trị biến lặp sau mỗi lần lặp.

Ví dụ:

```
for (int i = 0; i < 10; i++)
{
    Console.WriteLine(i);
}
```

Ngoài ra, trong C# còn có thể sử dụng các câu lệnh điều khiển vòng lặp break và continue để thoát khỏi vòng lặp hoặc bỏ qua các lần lặp tiếp theo tùy thuộc vào điều kiện.

3. Mảng (Array) trong lập trình C#

Mảng là một cấu trúc dữ liệu quan trọng trong lập trình, được sử dụng để lưu trữ nhiều giá trị cùng kiểu dữ liệu trong một biến. Trong C#, mảng được định nghĩa bằng từ khóa “**Array**” và có thể được sử dụng với nhiều kiểu dữ liệu khác nhau.

Cú pháp để khai báo một mảng trong C# như sau:

```
type[] arrayName;
```

Trong đó:

- › **type**: Kiểu dữ liệu của mảng.
- › **arrayName**: Tên của mảng.

Ví dụ:

```
int[] numbers;
```

Để khởi tạo một mảng trong C#, ta sử dụng từ khóa “**new**” như sau:

```
type[] arrayName = new type[arraySize];
```

Trong đó:

- › **arraySize**: Kích thước của mảng.

Ví dụ:

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

Sau khi khởi tạo mảng, ta có thể truy cập đến các phần tử của mảng bằng cách sử dụng chỉ số của phần tử đó. Chỉ số bắt đầu từ 0 và kết thúc ở (kích thước mảng - 1).

```
int[] numbers = new int[5];  
numbers[0] = 10;  
numbers[1] = 20;  
numbers[2] = 30;  
numbers[3] = 40;  
numbers[4] = 50;  
Console.WriteLine(numbers[2]); // In ra 30
```

Ta cũng có thể khởi tạo mảng ngay từ khi khai báo mảng, như sau:

```
type[] arrayName = { value1, value2, ..., valueN };
```

Ví dụ:

```
int[] numbers = { 10, 20, 30, 40, 50 };
```

Có một số phương thức hữu ích khi làm việc với mảng trong C#, bao gồm:

- › Length: Trả về kích thước của mảng.
- › Clear: Xóa tất cả giá trị của mảng.
- › Copy: Sao chép các giá trị của mảng sang một mảng khác.
- › Sort: Sắp xếp các giá trị của mảng.

Ví dụ:

```
int[] numbers = { 5, 3, 7, 2, 8 };  
Console.WriteLine(numbers.Length); // In ra 5  
Array.Sort(numbers);  
Console.WriteLine(string.Join(", ", numbers)); // In ra "2, 3, 5,  
7, 8"
```

4. Lập trình hướng đối tượng với C#

C# là một ngôn ngữ lập trình hướng đối tượng (OOP), cho phép lập trình viên tạo ra các lớp và đối tượng để xử lý các tác vụ trong ứng dụng.

Trong lập trình hướng đối tượng, một lớp định nghĩa cấu trúc, thuộc tính và phương thức của đối tượng. Một đối tượng là một thể hiện của một lớp và có thể được sử dụng để thực hiện các tác vụ.

Để định nghĩa một lớp trong C#, ta sử dụng từ khóa "class" như sau:

```
class ClassName
{
    // Thuộc tính và phương thức
}
```

Một lớp có thể có các thuộc tính và phương thức. Thuộc tính định nghĩa các đặc tính của đối tượng, trong khi phương thức định nghĩa các hành động mà đối tượng có thể thực hiện.

Ví dụ:

```
class Person
{
    public string Name { get; set; }
    public int Age { get; set; }

    public void SayHello()
    {
        Console.WriteLine("Hello, my name is " + Name);
    }
}
```

Trong ví dụ trên, lớp "Person" có hai thuộc tính "Name" và "Age" và một phương thức "SayHello". Thuộc tính "Name" là kiểu chuỗi, còn "Age" là kiểu

số nguyên. Phương thức "SayHello" xuất ra một thông báo chào hỏi với tên của người.

Để tạo một đối tượng từ một lớp, ta sử dụng từ khóa "new" như sau:

```
ClassName objectName = new ClassName();
```

Ví dụ:

```
Person person = new Person();
person.Name = "John";
person.Age = 25;
person.SayHello(); // In ra "Hello, my name is John"
```

Ta cũng có thể sử dụng các khái niệm khác trong OOP như kế thừa, đa hình và trừu tượng để thiết kế các lớp phức tạp hơn.

5. Ví dụ thực tế

Dưới đây là một số bài tập ví dụ về lập trình C#:

5.1 Tính toán diện tích và chu vi hình tròn:

```
using System;
class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Enter the radius of the circle: ");
        double radius = double.Parse(Console.ReadLine());
        double area = Math.PI * radius * radius;
        double circumference = 2 * Math.PI * radius;
        Console.WriteLine("Area: " + area);
        Console.WriteLine("Circumference: " + circumference);
    }
}
```

5.2 Chương trình quản lý sinh viên:

```
using System;  
  
class Student  
{  
    public string name;  
    public int age;  
    public double gpa;  
    public Student(string name, int age, double gpa)  
    {  
        this.name = name;  
        this.age = age;  
        this.gpa = gpa;  
    }  
    public void PrintInfo()  
    {  
        Console.WriteLine("Name: " + name);  
        Console.WriteLine("Age: " + age);  
        Console.WriteLine("GPA: " + gpa);  
    }  
}  
  
class Program  
{  
    static void Main(string[] args)  
    {  
        Student[] students = new Student[3];  
    }  
}
```

```

students[0] = new Student("John", 20, 3.5);
students[1] = new Student("Mary", 22, 3.8);
students[2] = new Student("Tom", 19, 3.2);
foreach(Student s in students)
{
    s.PrintInfo();
}
}

```

5.3 Chương trình đăng nhập:

```

using System;
class Program
{
    static void Main(string[] args)
    {
        string username = "admin";
        string password = "password";

        Console.WriteLine("Enter username: ");
        string inputUsername = Console.ReadLine();
        Console.WriteLine("Enter password: ");
        string inputPassword = Console.ReadLine();

        if(inputUsername == username && inputPassword ==
password)
    }
}

```

```
{  
    Console.WriteLine("Login successful!");  
}  
else  
{  
    Console.WriteLine("Invalid username or password!");  
}  
}  
}
```

BÀI TẬP VẬN DỤNG

Để học tập tốt phần bài tập đề nghị các bạn xem và gõ lại code, tự tay gõ lại và thực hành nhiều lần từng đoạn code, mỗi lần thực hành thử debug lỗi theo kiểu nghiên sẽ giúp biến code thành của riêng bạn.

Bài tập về mảng, list

I. Bài tập mảng một chiều trong C#:

Bài 1: Đọc và in các phần tử mảng trong C#

```
using System;
using System.Text;
namespace Ba_Tap_ChuyenDe_Csharp
{
    class BaiTapCsharp
    {
        public static void Main()
        {
            Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để
hiển thị tiếng việt có dấu
            int[] arr = new int[10];
            int i;
            Console.WriteLine(" Đọc và in ra các phần tử của mảng trong
C#:"); // Sử dụng câu lệnh Console.WriteLine sẽ thay thế cho câu lệnh
Console.Write và ký tự \n
```

```
Console.WriteLine("-----\n");
Console.WriteLine("Nhập 10 phần tử của mảng:");
for (i = 0; i < 10; i++)
{
    Console.WriteLine("Phần tử thứ - {0} : ", i);
    arr[i] = Convert.ToInt32(Console.ReadLine());
}
Console.WriteLine("In ra các phần tử của mảng: ");
for (i = 0; i < 10; i++)
{
    Console.Write("{0} ", arr[i]);
}
Console.WriteLine("\n");
Console.ReadKey();
}
}
```

Kết quả chạy chương trình:

```
C:\Users\HD\Desktop\Console X + 
Đọc và in ra các phần tử của mảng trong C#:
Nhập 10 phần tử của mảng:
Phần tử thứ - 0 :
1
Phần tử thứ - 1 :
3
Phần tử thứ - 2 :
4
Phần tử thứ - 3 :
8
Phần tử thứ - 4 :
7
Phần tử thứ - 5 :
10
Phần tử thứ - 6 :
12
Phần tử thứ - 7 :
21
Phần tử thứ - 8 :
13
Phần tử thứ - 9 :
31
In ra các phần tử của mảng:
1 3 4 8 7 10 12 21 13 31
```

Bài 2: Cách in mảng theo chiều đảo ngược trong C#

```
using System;
using System.Text;
namespace Ba_Tap_ChuyenDe_Csharp
{
    class BaiTapCsharp
    {
        public static void Main()
        {
            int i, n;
            int[] a = new int[100];
```

```
Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để  
hiển thị tiếng việt có dấu  
  
Console.WriteLine("In mảng theo chiều đảo ngược trong C#:");
Console.WriteLine("-----");
Console.WriteLine("Nhập số phần tử cần lưu giữ trong mảng: ");
n = Convert.ToInt32(Console.ReadLine());
Console.WriteLine("Nhập {0} phần tử vào trong mảng:", n);
for (i = 0; i < n; i++)
{
    Console.Write("Phan tu - {0}: ", i);
    a[i] = Convert.ToInt32(Console.ReadLine());
}
Console.WriteLine("Các phần tử được lưu giữ trong mảng là:");
for (i = 0; i < n; i++)
{
    Console.WriteLine("{0} ", a[i]);
}
Console.WriteLine("In mảng theo chiều đảo ngược:");
for (i = n - 1; i >= 0; i--)
{
    Console.WriteLine("{0} ", a[i]);
}
Console.ReadKey();
}
```

```
 }  
 }
```

Kết quả chạy chương trình:

```
C:\Users\HD\Desktop\Console X +  
In mang theo chieu dao nguoc trong C#:  
-----  
Nhập số phần tử cần lưu giữ trong mang:  
9  
Nhập 9 phần tử vào trong mang:  
Phan tu - 0: 9  
Phan tu - 1: 8  
Phan tu - 2: 7  
Phan tu - 3: 6  
Phan tu - 4: 5  
Phan tu - 5: 4  
Phan tu - 6: 3  
Phan tu - 7: 2  
Phan tu - 8: 1  
Các phần tử được lưu giữ trong mang là:  
9  
8  
7  
6  
5  
4  
3  
2  
1  
In mang theo chieu dao nguoc:  
1  
2  
3  
4  
5  
6  
7  
8  
9
```

Bài 3: Chương trình C# để tìm tổng các phần tử mảng

```
using System;  
using System.Text;  
namespace Ba_Tap_ChuyenDe_Csharp  
{  
    class BaiTapCsharp
```

```

{
    public static void Main()
    {
        int[] a = new int[100];
        int i, n, sum = 0;
        Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để
        hiển thị tiếng việt có dấu
        Console.WriteLine("Tính tổng các phần tử trong mảng C#:");
        Console.WriteLine("-----");
        Console.WriteLine("Nhập số phần tử cần lưu giữ vào mảng: ");
        n = Convert.ToInt32(Console.ReadLine());
        Console.WriteLine("Nhập {0} phần tử vào mảng:", n);
        for (i = 0; i < n; i++)
        {
            Console.WriteLine("Phần tử - {0}: ", i);
            a[i] = Convert.ToInt32(Console.ReadLine());
        }
        for (i = 0; i < n; i++)
        {
            sum += a[i];
        }
        Console.WriteLine("Tổng các phần tử trong mảng là: {0}", sum);
        Console.ReadKey(); // lệnh thêm vào để dùng màn hình console
    }
}

```

Kết quả chạy chương trình:

```
C:\Users\HD\Desktop\Console X + 
Tinh tong cac phan tu trong mang C#:
-----
Nhập số phần tử cần lưu giữ vào mảng:
4
Nhập 4 phần tử vào mảng:
Phần tử - 0:
2
Phần tử - 1:
4
Phần tử - 2:
6
Phần tử - 3:
8
Tổng các phần tử trong mảng là: 20
```

Bài 4: Sao chép mảng trong C#

```
using System;
using System.Text;
namespace Ba_Tap_ChuyenDe_Csharp
{
    class BaiTapCsharp
    {
        public static void Main()
        {
            int[] arr1 = new int[100]; //day la mang ban dau
            int[] arr2 = new int[100]; //day la mang sao
            int i, n;
            Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để
hiển thị tiếng việt có dấu
            Console.WriteLine("Sao chép mảng trong C#:");
        }
    }
}
```

```

Console.WriteLine("-----");
Console.WriteLine("Nhập số phần tử cần lưu giữ trong mảng: ");
n = Convert.ToInt32(Console.ReadLine());
Console.WriteLine("Nhập {0} phần tử vào trong mảng:", n);
for (i = 0; i < n; i++)
{
    Console.WriteLine("Phần tử - {0}: ", i);
    arr1[i] = Convert.ToInt32(Console.ReadLine());
}
/* sao chép phần tử trong mảng arr1 và mảng arr2.*/
for (i = 0; i < n; i++)
{
    arr2[i] = arr1[i];
}
/* in các phần tử trong mảng arr1 */
Console.WriteLine("Các phần tử trong mảng ban đầu là:");
for (i = 0; i < n; i++)
{
    Console.WriteLine("{0} ", arr1[i]);
}
/* in các phần tử trong mảng arr2.*/
Console.WriteLine("Các phần tử trong mảng sau là:");
for (i = 0; i < n; i++)

```

```

    {
        Console.WriteLine("{0} ", arr2[i]);
    }

    Console.ReadKey();

}

}

```

Kết quả chạy chương trình:

```

C:\Users\HD\Desktop\Console1>
Sao chép mảng trong C#:
Nhập số phần tử cần lưu giữ trong mảng:
5
Nhập 5 phần tử vào trong mảng:
Phần tử - 0:
12
Phần tử - 1:
1
Phần tử - 2:
2
Phần tử - 3:
3
Phần tử - 4:
4
Các phần tử trong mảng ban đầu là:
12
1
2
3
4
Các phần tử trong mảng sau là :
12
1
2
3
4

```

Bài 5: Chương trình C# để tìm số phần tử giống nhau trong một mảng

```

using System;
using System.Text;
namespace Ba_Tap_ChuyenDe_Csharp
{

```

```
class BaiTapCsharp
```

```
{
```

```
    public static void Main()
```

```
{
```

```
    int[] arr1 = new int[100];
```

```
    int i, j, n, bien_dem = 0; // Khai báo tên biến và biến đếm
```

```
    Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để  
hiển thị tiếng việt có dấu
```

```
    Console.WriteLine("Tìm số phần tử giống nhau trong mảng C#");
```

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

```
    Console.Write("Nhập số phần tử cần lưu giữ trong mảng: ");
```

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

```
    Console.WriteLine("Nhập {0} phần tử vào mảng:", n);
```

```
    for (i = 0; i < n; i++)
```

```
{
```

```
        Console.WriteLine("Phần tử - {0}: ", i);
```

```
        arr1[i] = Convert.ToInt32(Console.ReadLine());
```

```
}
```

```
/*Tim kiếm các phần tử giống nhau*/
```

```
    for (i = 0; i < n; i++)
```

```
{
```

```
        for (j = i + 1; j < n; j++)
```

```
{
```

```
        /*Tăng biến đếm bien_dem khi thấy các phần tử giống nhau.*/
```

```
        if (arr1[i] == arr1[j])
```

```

        {
            bien_dem++;
            break;
        }
    }

    Console.WriteLine("Số phần tử giống nhau trong mảng là: {0}",
bien_dem);

    Console.ReadKey();
}
}
}

```

Kết quả chạy chương trình:

```

C:\Users\HD\Desktop\Console + 
Tìm số phần tử giống nhau trong mảng C#:

Nhập số phần tử cần lưu giữ trong mảng: 3
Nhập 3 phần tử vào mảng:
Phần tử - 0: 1
Phần tử - 1: 9
Phần tử - 2: 9
Số phần tử giống nhau trong mảng là: 1
|
```

Bài 6: In các phần tử duy nhất của mảng trong C#

```

using System;
using System.Text;
namespace Ba_Tap_ChuyenDe_Csharp

```

```

{
class BaiTapCsharp
{
    public static void Main()
    {
        int n, bien_dem = 0;
        int[] arr1 = new int[100];
        int i, j, k;

        Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để
hiển thị tiếng việt có dấu

        Console.WriteLine("In các phần tử duy nhất của mảng C#:");
        Console.WriteLine("-----");
        Console.WriteLine("Nhập số phần tử cần lưu giữ vào mảng: ");
        n = Convert.ToInt32(Console.ReadLine());
        Console.WriteLine("Nhập {0} phần tử vào mảng:", n);
        for (i = 0; i < n; i++)
        {
            Console.WriteLine("Phần tử - {0}: ", i);
            arr1[i] = Convert.ToInt32(Console.ReadLine());
        }
/*Kiểm tra các phần tử giống nhau*/
        Console.WriteLine("Các phần tử duy nhất được tìm thấy trong
mảng là:");
        for (i = 0; i < n; i++)
        {
    }
}

```

```

bien_dem = 0;

/*kiểm tra phần tử giống nhau trước vị trí hiện tại và tăng
bien_dem thêm 1 nếu tìm thấy.*/

for (j = 0; j < i - 1; j++)

{
    /*tăng biến đếm khi tìm thấy phần tử giống nhau.*/

    if (arr1[i] == arr1[j])

    {

        bien_dem++;

    }

}

/*Kiểm tra các phần tử giống nhau sau vị trí hiện tại và tăng biến
đếm thêm 1 nếu tìm thấy.*/

for (k = i + 1; k < n; k++)

{
    /*tăng biến đếm khi tìm thấy phần tử giống nhau.*/

    if (arr1[i] == arr1[k])

    {

        bien_dem++;

    }

}

/*In giá trị của vị trí hiện tại - là giá trị duy nhất khi con trỏ vẫn
chưa giá trị vị trí ban đầu của nó.*/

if (bien_dem == 0)

{

```

```

        Console.WriteLine("{0}", arr1[i]);

    }

}

Console.WriteLine("\n\n");

Console.ReadKey();

}

}

```

Kết quả chạy chương trình:

```

C:\Users\HD\Desktop\Console X + ✓

In các phần tử duy nhất của mảng C#:
Nhập số phần tử cần lưu giữ vào mảng:
9
Nhập 9 phần tử vào mảng:
Phần tử - 0:
1
Phần tử - 1:
1
Phần tử - 2:
1
Phần tử - 3:
2
Phần tử - 4:
3
Phần tử - 5:
4
Phần tử - 6:
5
Phần tử - 7:
6
Phần tử - 8:
7
Các phần tử duy nhất được tìm thấy trong mảng là:
2
3
4
5
6
7

```

Bài 7: Chương trình C# để trộn (ghép) hai mảng

```

using System;

using System.Text;

namespace Ba_Tap_ChuyenDe_Csharp

```

```

{
    class BaiTapCsharp
    {
        public static void Main()
        {
            int[] arr1 = new int[100];
            int[] arr2 = new int[100];
            int[] arr3 = new int[200];
            int s1, s2, s3;
            int i, j, k;

            Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để
            hiển thị tiếng việt có dấu

            Console.WriteLine("Trộn (ghép) 2 mảng trong C#.");
            Console.WriteLine("-----");
            Console.WriteLine("Nhập số phần tử cần lưu giữ của mảng arr1: ");
            s1 = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Nhập {0} phần tử vào mảng arr1:", s1);
            for (i = 0; i < s1; i++)
            {
                Console.WriteLine("Phần tử - {0}: ", i);
                arr1[i] = Convert.ToInt32(Console.ReadLine());
            }
            Console.WriteLine("Nhập số phần tử cần lưu giữ mảng arr2: ");
            s2 = Convert.ToInt32(Console.ReadLine());
        }
    }
}

```

```

Console.WriteLine("Nhập {0} phần tử vào mảng arr2:", s2);
for (i = 0; i < s2; i++)
{
    Console.WriteLine("Phần tử - {0}: ", i);
    arr2[i] = Convert.ToInt32(Console.ReadLine());
}

/* Kích thước của mảng trộn = tổng kích thước mảng arr1 và mảng
arr2 */

s3 = s1 + s2;

/* chèn các phần tử của 2 mảng arr1 và arr2 vào mảng thứ 3 */
for (i = 0; i < s1; i++)
{
    arr3[i] = arr1[i];
}

for (j = 0; j < s2; j++)
{
    arr3[i] = arr2[j];
    i++;
}

/* sắp xếp theo thứ tự tăng dần*/
for (i = 0; i < s3; i++)
{
    for (k = 0; k < s3 - 1; k++)
    {

```

```

if (arr3[k] >= arr3[k + 1])
{
    j = arr3[k + 1];
    arr3[k + 1] = arr3[k];
    arr3[k] = j;
}
}

/*in mảng arr3*/
Console.WriteLine("Mảng trộn đã được sắp xếp tăng dần:");
for (i = 0; i < s3; i++)
{
    Console.WriteLine("{0} ", arr3[i]);
}
Console.WriteLine("\n\n");
Console.ReadKey();
}
}

```

Kết quả chạy chương trình:

```
C:\Users\HD\Desktop\ConsoleX + Trộn (ghép) 2 mảng trong C#.
Nhập số phần tử cần lưu giữ của mảng arr1:
4
Nhập 4 phần tử vào mảng arr1:
Phần tử - 0:
1
Phần tử - 1:
2
Phần tử - 2:
3
Phần tử - 3:
4
Nhập số phần tử cần lưu giữ mảng arr2:
4
Nhập 4 phần tử vào mảng arr2:
Phần tử - 0:
5
Phần tử - 1:
6
Phần tử - 2:
7
Phần tử - 3:
8
Mảng trộn đã được sắp xếp tăng dần:
1
2
3
4
5
6
7
8
```

Bài 8: Chương trình C# để đếm số lần xuất hiện của từng phần tử trong mảng

```
using System;
using System.Text;
namespace Ba_Tap_ChuyenDe_Csharp
{
    class BaiTapCsharp
```

```
public static void Main()
{
    int[] arr1 = new int[100];
    int[] fr1 = new int[100];
    int n, i, j, bien_dem;

    Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để
hiển thị tiếng việt có dấu

    Console.WriteLine("Đếm số lần xuất hiện của phần tử trong mảng
C#:");
    Console.WriteLine("-----");
    Console.WriteLine("Nhập số phần tử cần lưu giữ trong mảng: ");
    n = Convert.ToInt32(Console.ReadLine());
    Console.WriteLine("Nhập {0} phần tử vào mảng:", n);
    for (i = 0; i < n; i++)
    {
        Console.WriteLine("Phần tử - {0}: ", i);
        arr1[i] = Convert.ToInt32(Console.ReadLine());
        fr1[i] = -1;
    }
    for (i = 0; i < n; i++)
    {
        bien_dem = 1;
        for (j = i + 1; j < n; j++)
        {

```

```

        if (arr1[i] == arr1[j])
    {
        bien_dem++;
        fr1[j] = 0;
    }
}

if (fr1[i] != 0)
{
    fr1[i] = bien_dem;
}

}

Console.WriteLine("Tần xuất suất hiện của phần tử trong mảng là:");
for (i = 0; i < n; i++)
{
    if (fr1[i] != 0)
    {
        Console.WriteLine("Phần tử {0} xuất hiện {1} lần", arr1[i], fr1[i]);
    }
}
Console.ReadKey();
}
}

```

Kết quả chạy chương trình:

```
C:\Users\HD\Desktop\Console X + 
Đếm số lần xuất hiện của phần tử trong mảng C#:
Nhập số phần tử cần lưu giữ trong mảng:
5
Nhập 5 phần tử vào mảng:
Phần tử - 0:
2
Phần tử - 1:
4
Phần tử - 2:
3
Phần tử - 3:
5
Phần tử - 4:
6
Tần xuất suất hiện của phần tử trong mảng là:
Phần tử 2 xuất hiện 1 lần
Phần tử 4 xuất hiện 1 lần
Phần tử 3 xuất hiện 1 lần
Phần tử 5 xuất hiện 1 lần
Phần tử 6 xuất hiện 1 lần
```

Bài 9: Chương trình C# để tìm phần tử lớn nhất, nhỏ nhất trong mảng

```
using System;
using System.Text;
namespace Bai_Tap_ChuyenDe_Csharp
{
    class BaiTapCsharp
    {
        public static void Main()
        {
            int[] arr1 = new int[100];
            int i, mx, mn, n;
            Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để
hiển thị tiếng việt có dấu
```

```
Console.WriteLine("Tìm phần tử lớn nhất, nhỏ nhất trong mảng  
C#:");
Console.WriteLine("-----");
Console.WriteLine("Nhập số phần tử cần lưu giữ trong mảng: ");
n = Convert.ToInt32(Console.ReadLine());
Console.WriteLine("Nhập {0} phần tử vào trong mảng:", n);
for (i = 0; i < n; i++)
{
    Console.WriteLine("Phần tử - {0}: ", i);
    arr1[i] = Convert.ToInt32(Console.ReadLine());
}
mx = arr1[0];
mn = arr1[0];
for (i = 1; i < n; i++)
{
    if (arr1[i] > mx)
    {
        mx = arr1[i];
    }
    if (arr1[i] < mn)
    {
        mn = arr1[i];
    }
}
```

```

        }
        Console.WriteLine("Phản tử lớn nhất trong mảng là: {0}", mx);
        Console.WriteLine("Phản tử nhỏ nhất trong mảng là: {0}", mn);
        Console.ReadKey();
    }
}

```

Kết quả chạy chương trình:

```

C:\Users\HD\Desktop\Console x + 
Tim phan tu lon nhat, nhieu nhat trong mang C#:
Nhập số phản tử cần lưu giữ trong mảng:
7
Nhập 7 phản tử vào trong mảng:
Phản tử - 0:
1
Phản tử - 1:
3
Phản tử - 2:
9
Phản tử - 3:
98
Phản tử - 4:
7
Phản tử - 5:
6
Phản tử - 6:
7
Phản tử lớn nhất trong mảng là: 98
Phản tử nhỏ nhất trong mảng là: 1
|
```

Bài 10: Chia mảng thành mảng chẵn, mảng lẻ trong C#

```

using System;
using System.Text;
namespace Ba_Tap_ChuyenDe_Csharp
{
    class BaiTapCsharp
    {

```

```

public static void Main()
{
    int[] arr1 = new int[10];

    int[] arr2 = new int[10]; // khai báo mảng chứa các phần tử chẵn
    int[] arr3 = new int[10]; // khai báo mảng chứa các phần tử lẻ

    int i, j = 0, k = 0, n;

    Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để
    hiển thị tiếng việt có dấu

    Console.WriteLine("Chia mảng thành mảng chẵn, mảng lẻ C#:");
    Console.WriteLine("-----");
    Console.WriteLine("Nhập số phần tử cần lưu giữ trong mảng: ");
    n = Convert.ToInt32(Console.ReadLine());

    Console.WriteLine("Nhập {0} phần tử vào trong mảng:", n);
    for (i = 0; i < n; i++)
    {
        Console.WriteLine("Phần tử - {0}: ", i);
        arr1[i] = Convert.ToInt32(Console.ReadLine());
    }
    for (i = 0; i < n; i++)
    {
        if (arr1[i] % 2 == 0)
        {
            arr2[j] = arr1[i];
            j++;
        }
    }
}

```

```

        j++;
    }

    else

    {
        arr3[k] = arr1[i];
        k++;
    }

}

Console.WriteLine("Các phần tử chẵn là: ");

for (i = 0; i < j; i++)
{
    Console.WriteLine("{0} ", arr2[i]);
}

Console.WriteLine("Các phần tử lẻ là:");

for (i = 0; i < k; i++)
{
    Console.WriteLine("{0} ", arr3[i]);
}

Console.WriteLine("\n\n");

Console.ReadKey();
}
}

```

Kết quả chạy chương trình:

```
C:\Users\HD\Desktop\Console X + 
Chia mảng thành mảng chẵn, mảng lẻ C#:
Nhập số phần tử cần lưu giữ trong mảng:
9
Nhập 9 phần tử vào trong mảng:
Phần tử - 0:
2
Phần tử - 1:
4
Phần tử - 2:
1
Phần tử - 3:
3
Phần tử - 4:
7
Phần tử - 5:
9
Phần tử - 6:
10
Phần tử - 7:
12
Phần tử - 8:
13
Các phần tử chẵn là:
2
4
10
12
Các phần tử lẻ là:
1
3
7
9
13
```

Bài 11: Sắp xếp mảng theo thứ tự tăng dần trong C#

```
using System;
using System.Text;
namespace Ba_Tap_ChuyenDe_Csharp
{
    class BaiTapCsharp
    {
        public static void Main()
        {
```

```

int[] arr1 = new int[10];

int n, i, j, tmp; // Khai vào biến và biến tạm tmp

Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để
hiển thị tiếng việt có dấu

Console.WriteLine("Sắp xếp mảng theo thứ tự tăng dần trong
C#:");

Console.WriteLine("-----\n");

Console.WriteLine("Nhập vào kích thước của mảng: ");

n = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Nhập {0} phần tử vào trong mảng:", n);

for (i = 0; i < n; i++)

{

    Console.WriteLine("Phần tử - {0}: ", i);

    arr1[i] = Convert.ToInt32(Console.ReadLine());

}

for (i = 0; i < n; i++)

{

    for (j = i + 1; j < n; j++)

    {

        if (arr1[j] < arr1[i])

        {

            // cách trao đổi giá trị

            tmp = arr1[i];

            arr1[i] = arr1[j];

            arr1[j] = tmp;

```

```
        arr1[j] = tmp;  
    }  
}  
  
Console.WriteLine("In các phần tử của mảng theo thứ tự tăng  
đần:");  
  
for (i = 0; i < n; i++)  
{  
    Console.WriteLine("{0} ", arr1[i]);  
}  
Console.WriteLine("\n\n");  
Console.ReadKey();  
}  
}  
}
```

Kết quả chạy chương trình:

```
C:\Users\HD\Desktop\ConsoleApp1> Sắp xếp mảng theo thứ tự tăng dần trong C#:  
-----  
Nhập vào kích thước của mảng:  
7  
Nhập 7 phần tử vào trong mảng:  
Phần tử - 0:  
0  
Phần tử - 1:  
10  
Phần tử - 2:  
4  
Phần tử - 3:  
6  
Phần tử - 4:  
3  
Phần tử - 5:  
5  
Phần tử - 6:  
17  
In các phần tử của mảng theo thứ tự tăng dần:  
0  
3  
4  
5  
6  
10  
17
```

Bài 12: Sắp xếp mảng theo thứ tự giảm dần trong C#

```
using System;  
using System.Text;  
namespace Bai_Tap_ChuyenDe_Csharp  
{  
    class BaiTapCsharp  
    {  
        public static void Main()  
        {  
            int[] arr1 = new int[10];  
            int n, i, j, tmp; //khai báo biến và biến tạm tmp
```

Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để hiển thị tiếng việt có dấu

```
Console.WriteLine("Sắp xếp mảng theo thứ tự giảm dần trong C#:");
Console.WriteLine("-----");
Console.WriteLine("Nhập kích thước của mảng: ");
n = Convert.ToInt32(Console.ReadLine());
Console.WriteLine("Nhập {0} phần tử vào trong mảng:", n);
for (i = 0; i < n; i++)
{
    Console.WriteLine("Phần tử - {0}: ", i);
    arr1[i] = Convert.ToInt32(Console.ReadLine());
}
for (i = 0; i < n; i++)
{
    for (j = i + 1; j < n; j++)
    {
        if (arr1[i] < arr1[j])
        {
            // cách trao đổi giá trị
            tmp = arr1[i];
            arr1[i] = arr1[j];
            arr1[j] = tmp;
        }
    }
}
```

```

        Console.WriteLine("In các phần tử của mảng theo thứ tự giảm dần:");
        for (i = 0; i < n; i++)
        {
            Console.WriteLine("{0} ", arr1[i]);
        }
        Console.WriteLine("\n\n");
        Console.ReadKey();
    }
}

```

Kết quả chạy chương trình:

```

C:\Users\HD\Desktop\Console>
Sắp xếp mảng theo thứ tự giảm dần trong C#:
Nhập kích thước của mảng:
9
Nhập 9 phần tử vào trong mảng:
Phần tử - 0:
10
Phần tử - 1:
1
Phần tử - 2:
2
Phần tử - 3:
3
Phần tử - 4:
6
Phần tử - 5:
5
Phần tử - 6:
4
Phần tử - 7:
7
Phần tử - 8:
8
In các phần tử của mảng theo thứ tự giảm dần:
10
8
7
6
5
4
3
2
1

```

Bài 13: Chương trình C# để xóa phần tử trong mảng

```
using System;
using System.Text;
namespace Ba_Tap_ChuyenDe_Csharp
{
    class BaiTapCsharp
    {
        public static void Main()
        {
            int i, pos, n; //pos là biến vị trí cần xoá
            int[] arr1 = new int[50];
            Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để
            hiển thị tiếng việt có dấu
            Console.WriteLine("Xoá phần tử trong mảng C#:");
            Console.WriteLine("-----");
            Console.WriteLine("Nhập kích thước của mảng: ");
            n = Convert.ToInt32(Console.ReadLine());
            /* Nhập các phần tử vào trong mảng */
            Console.WriteLine("Nhập {0} phần tử vào trong mảng theo thứ
            tự tăng dần:", n);
            for (i = 0; i < n; i++)
            {
                Console.WriteLine("Phần tử - {0}: ", i);
                arr1[i] = Convert.ToInt32(Console.ReadLine());
            }
        }
    }
}
```

```

Console.WriteLine("Nhập vị trí cần xoá:");
pos = Convert.ToInt32(Console.ReadLine());
/* Xác định vị trí của i trong mảng */
i = 0;
while (i != pos - 1)
{
    i++;
}
/* vị trí i trong mảng sẽ được thay thế bởi vị trí bên phải của nó */
while (i < n)
{
    arr1[i] = arr1[i + 1];
    i++;
}
n--;
Console.WriteLine("In mảng sau khi đã xoá vị trí phần tử:");
for (i = 0; i < n; i++)
{
    Console.WriteLine(" {0}", arr1[i]);
}
Console.WriteLine("\n\n");
Console.ReadKey();
}
}

```

Kết quả chạy chương trình:

```
C:\Users\HDI\Desktop\Console x + v
xoa phan tu trong mang C#:
-----
Nhập kích thước của mảng:
5
Nhập 5 phần tử vào trong mảng theo thứ tự tăng dần:
Phản từ - 0:
7
Phản từ - 1:
4
Phản từ - 2:
3
Phản từ - 3:
2
Phản từ - 4:
1
Nhập vị trí cần xóa:
4
In mảng sau khi đã xóa vị trí phản từ:
7
4
3
1
```

Bài 14: Tìm phần tử lớn thứ hai trong mảng C#

```
using System;
using System.Text;
namespace Ba_Tap_ChuyenDe_Csharp
{
    class BaiTapCsharp
    {
        public static void Main()
        {
            int n, i, j = 0, lon_nhat, lon_thu_hai;
            int[] arr1 = new int[50];
            Console.OutputEncoding = Encoding.UTF8; // Lệnh này thêm để
hiển thị tiếng việt có dấu
            Console.WriteLine("Tìm phần tử lớn thứ 2 trong mảng C#:");
            Console.WriteLine("-----\n");
            Console.WriteLine("Nhập kích thước mảng: ");
```

```

n = Convert.ToInt32(Console.ReadLine());

/* Nhập các phần tử vào trong mảng */

Console.WriteLine("Nhập {0} phần tử vào trong mảng:", n);

for (i = 0; i < n; i++)

{

    Console.WriteLine("Phần tử - {0}: ", i);

    arr1[i] = Convert.ToInt32(Console.ReadLine());

}

/* Tìm vị trí lớn nhất của phần tử trong mảng */

lon_nhat = 0;

for (i = 0; i < n; i++)

{

    if (lon_nhat < arr1[i])

    {

        lon_nhat = arr1[i];

        j = i;

    }

}

/* bỏ qua phần tử lớn nhất này và tìm phần tử lớn thứ 2 */

lon_thu_hai = 0;

for (i = 0; i < n; i++)

{

    if (i == j)

    {

        i++; /* bỏ qua phần tử lớn nhất */
    }
}

```

```

        i--;
    }
    else
    {
        if (lon_thu_hai < arr1[i])
        {
            lon_thu_hai = arr1[i];
        }
    }
    Console.WriteLine("Phần tử lớn thứ 2 trong mảng là: {0} \n\n",
lon_thu_hai);
    Console.ReadKey();
}
}

```

Kết quả chạy chương trình:

The screenshot shows a terminal window titled 'C:\Users\HD\Desktop\Console' with the following text output:

```

Tìm phần tử lớn thứ 2 trong mảng C#:

Nhập kích thước mảng: 5
Nhập 5 phần tử vào trong mảng:
Phần tử - 0: 9
Phần tử - 1: 8
Phần tử - 2: 7
Phần tử - 3: 21
Phần tử - 4: 22
Phần tử lớn thứ 2 trong mảng là: 21

```

Bài 15: Tìm phần tử nhỏ thứ hai trong mảng C#

```
using System;  
namespace Ba_Tap_ChuyenDe_Csharp  
{  
    class BaiTapCsharp  
    {  
        public static void Main()  
        {  
            int n, i, j = 0, nho_nhat, nho_thu_hai;  
            int[] arr1 = new int[50];  
  
            Console.Write("\nTim phan tu nho thu hai trong mang C#: \n");  
            Console.Write("-----\n");  
            Console.Write("Nhập kích cỡ mảng: ");  
            n = Convert.ToInt32(Console.ReadLine());  
            /* Nhập các phần tử vào trong mảng */  
            Console.Write("Nhập {0} phần tử vào trong mảng (nên nhập giá trị nhỏ hơn 99999):\n", n);  
            for (i = 0; i < n; i++)  
            {  
                Console.Write("Phần tử - {0}: ", i);  
                arr1[i] = Convert.ToInt32(Console.ReadLine());  
            }  
            /* Tìm vị trí của phần tử nhỏ nhất */  
            nho_nhat = 0;
```

```

for (i = 0; i < n; i++)
{
    if (nho_nhat > arr1[i])
    {
        nho_nhat = arr1[i];
        j = i;
    }
}

/* bo qua phan tu nho nhat va tim phan tu nho thu hai */
nho_thu_hai = 99999; //gan mot gia tri ban dau

for (i = 0; i < n; i++)
{
    if (i == j)
    {
        i++; /* bo qua phan tu nho nhat */
        i--;
    }
    else
    {
        if (nho_thu_hai > arr1[i])
        {
            nho_thu_hai = arr1[i];
        }
    }
}

```

```

        }

        Console.WriteLine("Phan tu nho thu hai trong mang la: {0} \n\n",
nho_thu_hai);

        Console.ReadKey();

    }

}

```

Kết quả chạy chương trình:

```

C:\Users\HOI\Desktop\Console1> Tim phan tu nho thu 2 trong mang C#:

Nhập kích thước mảng:
5
Nhập 5 phần tử vào trong mảng (nên nhập các giá trị nhỏ hơn hoặc bằng 99999):
Phản tử - 0:
3
Phản tử - 1:
4
Phản tử - 2:
5
Phản tử - 3:
6
Phản tử - 4:
9
Phản tử nho thu 2 trong mang la: 4

```

II. Mảng hai chiều trong C#

Mẫu đơn giản nhất của mảng đa chiều là mảng hai chiều. Một mảng hai chiều về bản chất là danh sách của các mảng một chiều.

Một mảng 2 chiều có thể được nghĩ như là một bảng, có x hàng và y cột. Dưới đây là một mảng hai chiều có 3 hàng và 4 cột.

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Như vậy, mỗi phần tử trong mảng a được định danh bởi một tên phần tử trong mẫu $a[i][j]$, với a là tên mảng và i, j là các subscript - chỉ số được xác định duy nhất mỗi phần tử trong a.

1. Khởi tạo mảng hai chiều trong C#

Các mảng đa chiều có thể được khởi tạo bởi xác định các giá trị trong dấu móc vuông cho mỗi hàng. Sau đây là một hàng với 3 hàng và mỗi hàng chứa 4 cột.

```
int[,] a = int [3,4] = {  
    {0, 1, 2, 3}, /* Khởi tạo cho hàng được đánh chỉ mục là 0 */  
    {4, 5, 6, 7}, /* Khởi tạo cho hàng được đánh chỉ mục là 1 */  
    {8, 9, 10, 11} /* Khởi tạo cho hàng được đánh chỉ mục là 2 */  
};
```

2. Bài tập mảng hai chiều trong C#

Bài 1: Đọc và in mảng hai chiều trong C#

```
using System;  
  
namespace Ba_Tap_ChuyenDe_Csharp  
{  
    class BaiTapCsharp  
    {  
        public static void Main()  
        {  
            int i, j;  
            int[,] arr1 = new int[3, 3];  
            Console.Write("\nDoc va in mang hai chieu trong C#: \n");  
            Console.Write("-----\n");
```

```

/* nhap cac phan tu vao trong mang*/
Console.WriteLine("Nhap cac phan tu vao trong mang hai chieu:\n");
for (i = 0; i < 3; i++)
{
    for (j = 0; j < 3; j++)
        Console.Write("Phan tu - [{0},{1}]: ", i, j);
    arr1[i, j] = Convert.ToInt32(Console.ReadLine());
}
Console.WriteLine("\nIn mang hai chieu: \n");
for (i = 0; i < 3; i++)
{
    Console.WriteLine("\n");
    for (j = 0; j < 3; j++)
        Console.Write("{0}\t", arr1[i, j]);
    Console.WriteLine("\n\n");
    Console.ReadKey();
}
}

```

Kết quả chạy chương trình:

```
Microsoft Visual Studio Debug Console

Doc va in mang hai chieu trong C#:
-----
Nhap cac phan tu vao trong mang hai chieu:
Phan tu - [0,0]: 2
Phan tu - [0,1]: 1
Phan tu - [0,2]: 3
Phan tu - [1,0]: 4
Phan tu - [1,1]: 2
Phan tu - [1,2]: 4
Phan tu - [2,0]: 1
Phan tu - [2,1]: 3
Phan tu - [2,2]: 7

In mang hai chieu:
2      1      3
4      2      4
1      3      7
```

Bài 2: Cộng hai ma trận trong C#

```
using System;
namespace Ba_Tap_ChuyenDe_Csharp
{
    class BaiTapCsharp
    {
        public static void Main()
        {
            int i, j, n;
            int[,] arr1 = new int[50, 50];
            int[,] arr2 = new int[50, 50];
            int[,] ma_tran_tong = new int[50, 50];
            Console.WriteLine("\nCong hai ma tran trong C#:\\n");
            Console.WriteLine("-----\\n");
```

```

Console.WriteLine("Nhập kích cỡ của hai mảng vuông (nho hơn 5):");
n = Convert.ToInt32(Console.ReadLine());
/* Nhập các phần tử vào trong mảng da chieu*/
Console.WriteLine("Nhập các phần tử vào trong mảng vuông:\n");
for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
    {
        Console.Write("Phần tử - [{0},{1}]: ", i, j);
        arr1[i, j] = Convert.ToInt32(Console.ReadLine());
    }
}
Console.WriteLine("Nhập các phần tử vào trong mảng vuông thứ hai:\n");
for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
    {
        Console.Write("Phần tử - [{0},{1}]: ", i, j);
        arr2[i, j] = Convert.ToInt32(Console.ReadLine());
    }
}
Console.WriteLine("\n\nIn mảng vuông thứ nhất:\n");
for (i = 0; i < n; i++)

```

```

{
    Console.WriteLine("\n");
    for (j = 0; j < n; j++)
        Console.Write("{0}\t", arr1[i, j]);
}

Console.WriteLine("\nIn ma tran thu hai:\n");
for (i = 0; i < n; i++)
{
    Console.WriteLine("\n");
    for (j = 0; j < n; j++)
        Console.Write("{0}\t", arr2[i, j]);
}

/* cong hai ma tran */
for (i = 0; i < n; i++)
    for (j = 0; j < n; j++)
        ma_tran_tong[i, j] = arr1[i, j] + arr2[i, j];
Console.WriteLine("\nMa tran tong cua hai ma tran tren la: \n");
for (i = 0; i < n; i++)
{
    Console.WriteLine("\n");
    for (j = 0; j < n; j++)
        Console.Write("{0}\t", ma_tran_tong[i, j]);
}

Console.WriteLine("\n\n");

```

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

Kết quả chạy chương trình:

The screenshot shows a terminal window titled 'C:\Users\HieplT\source\repos\ConsoleApp1\BaiTap2\bin\Debug\net6.0\BaiTap2.exe'. The program prompts the user to enter the dimensions of two 2x2 matrices. It then asks for matrix elements and prints them out. Finally, it calculates and prints the sum of the two matrices.

```
Cong hai ma tran trong C#  
Nhap kich co cua hai ma tran vuong (nho hon 5): 2  
Nhap cac phan tu vao trong ma tran dau tien...  
Phan tu [0,0]: 2  
Phan tu [0,1]: 1  
Phan tu [1,0]: 0  
Phan tu [1,1]: 1  
Nhap cac phan tu vao trong ma tran thu hai  
Phan tu [0,0]: 2  
Phan tu [0,1]: 2  
Phan tu [1,0]: 3  
Phan tu [1,1]: 3  
In ma tran thu nhat:  
2 1  
0 1  
In ma tran thu hai:  
2 2  
3 3  
Ma tran tong cua hai ma tran tren la:  
4 3  
3 4
```

Bài 3: Trừ ma trận trong C#

```
using System;  
  
namespace Ba_Tap_ChuyenDe_Csharp  
{  
    class BaiTapCsharp  
    {  
        public static void Main()  
        {  
            int i, j, n;
```

```

int[,] arr1 = new int[50, 50];
int[,] arr2 = new int[50, 50];
int[,] ma_tran_hieu = new int[50, 50];

Console.Write("\nTru ma tran trong C#: \n");
Console.Write("-----\n");
Console.Write("Nhap kich co cua hai ma tran (nho hon 5): ");
n = Convert.ToInt32(Console.ReadLine());
/* Nhap cac phan tu vao trong mang hai chieu */
Console.Write("Nhap cac phan tu vao trong ma tran thu
nhat:\n");
for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
    {
        Console.Write("Phan tu - [{0},{1}]: ", i, j);
        arr1[i, j] = Convert.ToInt32(Console.ReadLine());
    }
}
Console.Write("Nhap cac phan tu vao trong ma tran thu hai:\n");
for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
    {
}

```

```

        Console.WriteLine("Phan tu - [{0},{1}]: ", i, j);
        arr2[i, j] = Convert.ToInt32(Console.ReadLine());
    }

}

Console.WriteLine("\n\n ma tran thu nhat:\n");
for (i = 0; i < n; i++)
{
    Console.WriteLine("\n");
    for (j = 0; j < n; j++)
        Console.Write("{0}\t", arr1[i, j]);
}

Console.WriteLine("\n\n ma tran thu hai:\n");
for (i = 0; i < n; i++)
{
    Console.WriteLine("\n");
    for (j = 0; j < n; j++)
        Console.Write("{0}\t", arr2[i, j]);
}

/* tru ma tran */
for (i = 0; i < n; i++)
    for (j = 0; j < n; j++)
        ma_tran_hieu[i, j] = arr1[i, j] - arr2[i, j];

Console.WriteLine("\nMa tran hieu cua hai ma tran tren la: \n");
for (i = 0; i < n; i++)

```

```

    {
        Console.WriteLine("\n");
        for (j = 0; j < n; j++)
            Console.Write("{0}\t", ma_tran_hieu[i, j]);
    }
    Console.WriteLine("\n\n");
    Console.ReadKey();
}
}

```

Kết quả chạy chương trình:

```

C:\Users\HiepIT\source\repos\ConsoleApp1\BaiTap2.3\bin\Debug\

Truy mã tran trong C#:

Nhập kích cỡ của hai ma tran (nhỏ hơn 5): 2
Nhập các phần tử vào trong ma tran thu nhât:
Phần tử - [0,0]: 1
Phần tử - [0,1]: 2
Phần tử - [1,0]: 2
Phần tử - [1,1]: 1
Nhập các phần tử vào trong ma tran thu hai:
Phần tử - [0,0]: 2
Phần tử - [0,1]: 2
Phần tử - [1,0]: 3
Phần tử - [1,1]: 3

In ma tran thu nhât:
1      2
2      1
In ma tran thu hai:
2      2
3      3
Ma tran hieu cua hai ma tran tren la:
1      0
-1     -2

```

Bài 4: Chương trình C# để nhân hai ma trận

```
using System;  
namespace Ba_Tap_ChuyenDe_Csharp  
{  
    class BaiTapCsharp  
    {  
        public static void Main()  
        {  
            int i, j, k, r1, c1, r2, c2, sum = 0;  
            int[,] arr1 = new int[50, 50];  
            int[,] arr2 = new int[50, 50];  
            int[,] ma_tran_tich = new int[50, 50];  
            Console.Write("\nNhập hai ma trận trong C#: \n");  
            Console.WriteLine("-----\n");  
            Console.Write("\nNhập số hàng và số cột của ma trận thứ  
nhất: \n");  
            Console.Write("Nhập số hàng: ");  
            r1 = Convert.ToInt32(Console.ReadLine());  
            Console.Write("Nhập số cột: ");  
            c1 = Convert.ToInt32(Console.ReadLine());  
            Console.Write("\nNhập số hàng và số cột của ma trận thứ  
hai: \n");  
            Console.Write("Nhập số hàng: ");  
            r2 = Convert.ToInt32(Console.ReadLine());
```

```

Console.WriteLine("Nhập số cột: ");
c2 = Convert.ToInt32(Console.ReadLine());
if (c1 != r2)
{
    Console.WriteLine("Không thể nhân hai ma trận !!!");
    Console.WriteLine("\nSo cột của ma trận thu nhât phải bằng số
hang của ma trận thu hai.");
}
else
{
    Console.WriteLine("Nhập các phần tử của ma trận thu nhât:\n");
    for (i = 0; i < r1; i++)
    {
        for (j = 0; j < c1; j++)
        {
            Console.Write("Phần tử - [{0}],[{1}]: ", i, j);
            arr1[i, j] = Convert.ToInt32(Console.ReadLine());
        }
    }
    Console.WriteLine("Nhập các phần tử của ma trận thu hai:\n");
    for (i = 0; i < r2; i++)
    {
        for (j = 0; j < c2; j++)
        {

```

```

Console.WriteLine("Phan tu - [{0}],[{1}]: ", i, j);
arr2[i, j] = Convert.ToInt32(Console.ReadLine());

}

}

Console.WriteLine("\nIn ma tran dau tien:\n");
for (i = 0; i < r1; i++)
{
    Console.WriteLine("\n");
    for (j = 0; j < c1; j++)
        Console.Write("{0}\t", arr1[i, j]);
}
Console.WriteLine("\nIn ma tran thu hai:\n");
for (i = 0; i < r2; i++)
{
    Console.WriteLine("\n");
    for (j = 0; j < c2; j++)
        Console.Write("{0}\t", arr2[i, j]);
}
//nhan hai ma tran
for (i = 0; i < r1; i++)
    for (j = 0; j < c2; j++)
        ma_tran_tich[i, j] = 0;
for (i = 0; i < r1; i++) //hang cua ma tran thu nhat
{

```

```
for (j = 0; j < c2; j++) //cot cua ma tran thu hai  
{  
    sum = 0;  
    for (k = 0; k < c1; k++)  
        sum = sum + arr1[i, k] * arr2[k, j];  
    ma_tran_tich[i, j] = sum;  
}  
}  
Console.WriteLine("Ma tran tich cua hai ma tran tren la: \n");  
for (i = 0; i < r1; i++)  
{  
    Console.WriteLine("\n");  
    for (j = 0; j < c2; j++)  
    {  
        Console.Write("{0}\t", ma_tran_tich[i, j]);  
    }  
}  
Console.WriteLine("\n\n");  
Console.ReadKey();  
}  
}  
}
```

Kết quả chạy chương trình:

```
C:\Users\HiepIT\source\repos\ConsoleApp1\BaiTap2.4\bin\Debug\net6.0\{  
Nhập hai ma tran trong C#:  
-----  
Nhập số hàng và số cột của ma tran thu nhât:  
Nhập số hàng: 2  
Nhập số cột: 2  
  
Nhập số hàng và số cột của ma tran thu hai:  
Nhập số hàng: 2  
Nhập số cột: 2  
Nhập các phần tử của ma tran thu nhât:  
Phần tử - [0],[0]: 1  
Phần tử - [0],[1]: 2  
Phần tử - [1],[0]: 2  
Phần tử - [1],[1]: 1  
Nhập các phần tử của ma tran thu hai:  
Phần tử - [0],[0]: 2  
Phần tử - [0],[1]: 2  
Phần tử - [1],[0]: 3  
Phần tử - [1],[1]: 3  
  
In ma tran dau tien:  
1      2  
2      1  
In ma tran thu hai:  
2      2  
3      3  
Ma tran tích của hai ma tran trên là:  
8      8  
7      7
```

Bài 5: Tìm ma trận chuyển vị trong C#

```
using System;  
  
namespace Ba_Tap_ChuyenDe_Csharp  
{  
  
    class BaiTapCsharp  
    {  
  
        public static void Main()  
        {  
  
            int i, j, r, c;  
  
            int[,] arr1 = new int[50, 50];
```

```

int[,] ma_tran_chuyen_vi = new int[50, 50];

Console.WriteLine("\nTim ma tran chuyen vi trong C#: \n");

Console.WriteLine("-----\n");

Console.WriteLine("\nNhap so hang va so cot cua ma tran ban  
dau: \n");

Console.Write("Nhap so hang: ");

r = Convert.ToInt32(Console.ReadLine());

Console.Write("Nhap so cot: ");

c = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Nhap cac phan tu cua ma tran: \n");

for (i = 0; i < r; i++)
{
    for (j = 0; j < c; j++)
    {
        Console.Write("Phan tu - [{0}],[{1}]: ", i, j);
        arr1[i, j] = Convert.ToInt32(Console.ReadLine());
    }
}

Console.WriteLine("\nIn ma tran ban dau: \n");

for (i = 0; i < r; i++)
{
    Console.WriteLine("\n");
    for (j = 0; j < c; j++)
        Console.Write("{0}\t", arr1[i, j]);
}

```

```

}

//tim ma tran chuyen vi

for (i = 0; i < r; i++)
{
    for (j = 0; j < c; j++)
    {
        ma_tran_chuyen_vi[j, i] = arr1[i, j];
    }
}

Console.WriteLine("\n\nIn ma tran chuyen vi:");
for (i = 0; i < c; i++)
{
    Console.WriteLine("\n");
    for (j = 0; j < r; j++)
    {
        Console.Write("{0}\t", ma_tran_chuyen_vi[i, j]);
    }
    Console.WriteLine("\n\n");
    Console.ReadKey();
}
}
}

```

Kết quả chạy chương trình:

```
C:\Users\HiepIT\source\repos\ConsoleApp1\BaiTap2.5\bin\Debug\net6.0\Bai
Tim ma tran chuyen vi trong C#:

Nhap so hang va so cot cua ma tran ban dau:
Nhap so hang: 2
Nhap so cot: 2
Nhap cac phan tu cua ma tran:
Phan tu - [0],[0]: 2
Phan tu - [0],[1]: 1
Phan tu - [1],[0]: 1
Phan tu - [1],[1]: 2

In ma tran ban dau:
2      1
1      2

In ma tran chuyen vi:
2      1
1      2
```

Bài 6: Chương trình C# để tính tổng các phần tử trên đường chéo chính của ma trận

```
using System;

namespace Ba_Tap_ChuyenDe_Csharp

{
    class BaiTapCsharp

    {
        public static void Main()
        {
            int i, j, sum = 0, n;
            int[,] arr1 = new int[50, 50];
```

```

Console.WriteLine("\nTinh tong cac phan tu tren duong cheo chinh
cua ma tran trong C#: \n");

Console.WriteLine("-----\n-----\n");

Console.WriteLine("Nhap kich co ma tran vuong: ");
n = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Nhap cac phan tu cua ma tran: \n");

for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
    {
        Console.WriteLine("Phan tu - [{0}],[{1}]: ", i, j);
        arr1[i, j] = Convert.ToInt32(Console.ReadLine());
        //tinh tong cac phan tu tren duong cheo chinh
        if (i == j) sum = sum + arr1[i, j];
    }
}

Console.WriteLine("In ma tran: \n");
for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
        Console.Write("{0} ", arr1[i, j]);
    Console.WriteLine("\n");
}

```

```

        Console.WriteLine("Tong cac phan tu tren duong cheo chinh la: {0}\n",
sum);
Console.ReadKey();
}
}
}

```

Kết quả chạy chương trình:

```

C:\Users\HiepIT\source\repos\ConsoleApp1\BaiTap2.6\bin\Debug\net6.0\BaiTap2.6.exe

Tinh tong cac phan tu tren duong cheo chinh cua ma tran trong C#:
-----
Nhap kich co ma tran vuong: 2
Nhap cac phan tu cua ma tran:
Phan tu - [0],[0]: 2
Phan tu - [0],[1]: 1
Phan tu - [1],[0]: 1
Phan tu - [1],[1]: 2
In ma tran:
2 1
1 2
Tong cac phan tu tren duong cheo chinh la: 4

```

Bài 7: Tính tổng các phần tử trên đường chéo phụ của ma trận trong C#

```

using System;
namespace Ba_Tap_ChuyenDe_Csharp
{
    class BaiTapCsharp
    {
        public static void Main()
        {
            int i, j, sum = 0, n, m = 0;

```

```

int[,] arr1 = new int[50, 50];

Console.WriteLine("\nTinh tong cac phan tu tren duong cheo phu cua
ma tran trong C#: \n");

Console.WriteLine("-----\n");

Console.WriteLine("Nhap kich co cua ma tran vuong: ");
n = Convert.ToInt32(Console.ReadLine());

m = n;

Console.WriteLine("Nhap cac phan tu vao trong ma tran: \n");

for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
    {
        Console.Write("Phan tu - [{0}],[{1}]: ", i, j);
        arr1[i, j] = Convert.ToInt32(Console.ReadLine());
    }
}

Console.WriteLine("In ma tran: \n");

for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
        Console.Write("{0} ", arr1[i, j]);
    Console.WriteLine("\n");
}

```

```
// tinh tong cac phan tu tren duong cheo phu
for (i = 0; i < n; i++)
{
    m = m - 1;
    for (j = 0; j < n; j++)
    {
        if (j == m)
        {
            sum = sum + arr1[i, j];
        }
    }
}
Console.WriteLine("Tong cac phan tu tren duong cheo phu la: {0}\n",
sum);

Console.ReadKey();
}
```

Kết quả chạy chương trình:

```
C:\Users\HiepIT\source\repos\ConsoleApp1\BaiTap2.7\bin\Debug\net6.0\BaiTap2.7

Tinh tong cac phan tu tren duong cheo phu cua ma tran trong C#:
-----
Nhap kich co cua ma tran vuong: 2
Nhap cac phan tu vao trong ma tran:
Phan tu - [0],[0]: 3
Phan tu - [0],[1]: 1
Phan tu - [1],[0]: 1
Phan tu - [1],[1]: 3
In ma tran:
3 1
1 3
Tong cac phan tu tren duong cheo phu la: 2
```

Bài 8: Chương trình C# để tính tổng các hàng, các cột của ma trận

```
using System;

namespace Ba_Tap_ChuyenDe_Csharp

{
    class BaiTapCsharp

    {
        public static void Main()
        {
            int i, j, n;

            int[,] arr1 = new int[10, 10];

            int[] tong_hang = new int[10];

            int[] tong_cot = new int[10];

            Console.Write("\nTinh tong cac hang va cac cot cua ma tran
trong C#: \n");

            Console.Write("-----\n");
```

```

Console.WriteLine("Nhập kích thước ma trận vuông: ");
n = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Nhập các phần tử vào trong ma trận:\n");
for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
    {
        Console.Write("Phần tử - [{0}],[{1}]: ", i, j);
        arr1[i, j] = Convert.ToInt32(Console.ReadLine());
    }
}

Console.WriteLine("In ma trận:\n");
for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
        Console.Write("{0} ", arr1[i, j]);
    Console.WriteLine("\n");
}

/* tính tổng các hàng */
for (i = 0; i < n; i++)
{
    tong_hang[i] = 0;
    for (j = 0; j < n; j++)
        tong_hang[i] = tong_hang[i] + arr1[i, j];
}

```

```

    }

    /* tinh tong cac cot */

    for (i = 0; i < n; i++)

    {

        tong_cot[i] = 0;

        for (j = 0; j < n; j++)

            tong_cot[i] = tong_cot[i] + arr1[j, i];

    }

    Console.WriteLine("Tong cua cac hang va cac cot trong ma tran  

ia:\n");

    for (i = 0; i < n; i++)

    {

        for (j = 0; j < n; j++)

            Console.Write("{0} ", arr1[i, j]);

        Console.WriteLine("{0} ", tong_hang[i]);

        Console.WriteLine("\n");

    }

    Console.WriteLine("\n");

    for (j = 0; j < n; j++)

    {

        Console.Write("{0} ", tong_cot[j]);

    }

    Console.WriteLine("\n\n");

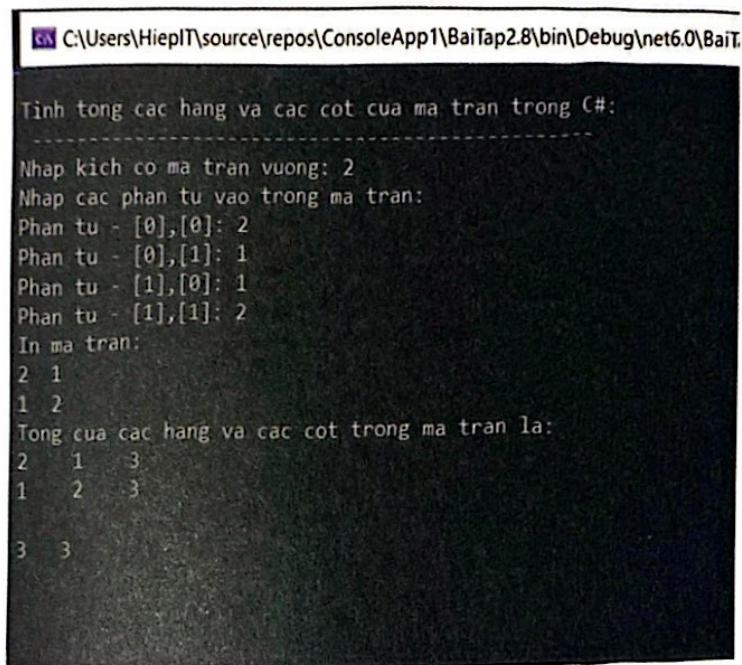
    Console.ReadKey();

}

```

```
    }  
}  
}
```

\Kết quả chạy chương trình:



```
Tinh tong cac hang va cac cot cua ma tran trong C#:  
-----  
Nhap kich co ma tran vuong: 2  
Nhap cac phan tu vao trong ma tran:  
Phan tu - [0],[0]: 2  
Phan tu - [0],[1]: 1  
Phan tu - [1],[0]: 1  
Phan tu - [1],[1]: 2  
In ma tran:  
2 1  
1 2  
Tong cua cac hang va cac cot trong ma tran la:  
2 1 3  
1 2 3  
3 3
```

Bài 9: In ma trận tam giác trên trong C#

```
using System;  
  
namespace Ba_Tap_ChuyenDe_Csharp;  
  
class BaiTapCsharp  
{  
  
    public static void Main()  
    {  
  
        int i, j, n;  
  
        int[,] arr1 = new int[10, 10];  
  
        Console.Write("\n\n ma tran tam giac tren trong C#: \n");  
  
        Console.Write("-----\n");
```

```

Console.WriteLine("Nhập kích thước của ma trận vuông: ");
n = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Nhập các phần tử vào trong ma trận:\n");
for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
    {
        Console.Write("Phần tử - [{0}],[{1}]: ", i, j);
        arr1[i, j] = Convert.ToInt32(Console.ReadLine());
    }
}

Console.WriteLine("In ma trận ban đầu:\n");
for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
        Console.Write("{0} ", arr1[i, j]);
    Console.WriteLine("\n");
}

Console.WriteLine("\nThiết lập các phần tử nằm dưới đường chéo chính bằng 0.\n");
for (i = 0; i < n; i++)
{
    Console.WriteLine("\n");
    for (j = 0; j < n; j++)

```

```

        if (i <= j)
            Console.Write("{0} ", arr1[i, j]);
        else
            Console.Write("{0} ", 0);
    }
    Console.WriteLine("\n\n");

    Console.ReadKey();
}

```

Kết quả chạy chương trình:

```

C:\Users\HiepIT\source\repos\ConsoleApp1\BaiTap2.9\bin\Debug\net6.0\BaiTap2.9.exe

In ma tran tam giac tren trong C#:
-----
Nhap kich co cua ma tran vuong: 2
Nhap cac phan tu vao trong ma tran:
Phan tu - [0],[0]: 2
Phan tu - [0],[1]: 1
Phan tu - [1],[0]: 6
Phan tu - [1],[1]: 1
In ma tran ban dau:
2 1
6 1

Thiet lap cac phan tu nam duoi duong cheo chinh bang 0.

2 1
0 1

```

Bài 10: Chương trình C# để in ma trận tam giác dưới

```
using System;  
namespace Ba_Tap_ChuyenDe_Csharp;  
class BaiTapCsharp  
{  
    public static void Main()  
    {  
        int i, j, n;  
        int[,] arr1 = new int[50, 50];  
        Console.WriteLine("\nIn ma tran tam giac duoi trong C#: \n");  
        Console.WriteLine("-----\n");  
        Console.Write("Nhap kich co cua ma tran vuong: ");  
        n = Convert.ToInt32(Console.ReadLine());  
        Console.Write("Nhap cac phan tu vao trong ma tran: \n");  
        for (i = 0; i < n; i++)  
        {  
            for (j = 0; j < n; j++)  
            {  
                Console.Write("Phan tu - [{0}],[{1}]: ", i, j);  
                arr1[i, j] = Convert.ToInt32(Console.ReadLine());  
            }  
        }  
        Console.WriteLine("In ma tran ban dau: \n");  
        for (i = 0; i < n; i++)
```

```

{
    for (j = 0; j < n; j++)
        Console.Write("{0} ", arr1[i, j]);
    Console.WriteLine("\n");
}

Console.WriteLine("Thiet lap cac phan tu nam ben tren duong cheo
chinh bang 0.\n");

for (i = 0; i < n; i++)
{
    Console.WriteLine("\n");
    for (j = 0; j < n; j++)
        if (i >= j)
            Console.Write("{0} ", arr1[i, j]);
        else
            Console.Write("{0} ", 0);
    }
    Console.WriteLine("\n\n");
}

Console.ReadKey();
}
}

```

Kết quả chạy chương trình:

```
C:\Users\HiepIT\source\repos\ConsoleApp1\BaiTap2.10\bin\Debug\net6.0\BaiTap2.10.exe

In ma tran tam giac duoi trong C#:
-----
Nhap kich co cua ma tran vuong: 2
Nhap cac phan tu vao trong ma tran:
Phan tu - [0],[0]: 3
Phan tu - [0],[1]: 1
Phan tu - [1],[0]: 1
Phan tu - [1],[1]: 3
In ma tran ban dau:
3 1
1 3

Thiet lap cac phan tu nam ben tren duong cheo chinh bang 0.

3 0
1 3
```

Bài 11: Tính định thức ma trận trong C#

```
using System;
namespace Ba_Tap_ChuyenDe_Csharp;
class BaiTapCsharp
{
    public static void Main()
    {
        int i, j;
        int[,] arr1 = new int[10, 10];
        int det = 0;
        Console.Write("\nTinh dinh thuc ma tran trong C#: \n");
        Console.Write("-----\n");
        Console.Write("Nhap ma tran vuong (3x3):\n");
        for (i = 0; i < 3; i++)
    }
```

```

{
    for (j = 0; j < 3; j++)
    {
        Console.Write("Phan tu - [{0}],[{1}]: ", i, j);
        arr1[i, j] = Convert.ToInt32(Console.ReadLine());
    }
}

Console.WriteLine("In ma tran:\n");
for (i = 0; i < 3; i++)
{
    for (j = 0; j < 3; j++)
        Console.Write("{0} ", arr1[i, j]);
    Console.WriteLine("\n");
}
//tinh dinh thuc ma tran vuong 3x3
for (i = 0; i < 3; i++)
{
    det = det + (arr1[0, i] * (arr1[1, (i + 1) % 3] * arr1[2, (i + 2) % 3] - arr1[1, (i + 2) % 3] * arr1[2, (i + 1) % 3]));
}
Console.WriteLine("\nDinh thuc cua ma tran vuong bang: {0}\n", det);
Console.ReadKey();
}
}

```

Kết quả chạy chương trình:

```
C:\Users\HiepT\source\repos\ConsoleApp1\BaiTap2.11\bin\Debug\net6.0\BaiTap2.11.e
Tinh dinh thuc ma tran trong C#:
Nhap ma tran vuong (3x3):
Phan tu - [0],[0]: 2
Phan tu - [0],[1]: 1
Phan tu - [0],[2]: 3
Phan tu - [1],[0]: 1
Phan tu - [1],[1]: 2
Phan tu - [1],[2]: 1
Phan tu - [2],[0]: 2
Phan tu - [2],[1]: 3
Phan tu - [2],[2]: 3
In ma tran:
2 1 3
1 2 1
2 3 3
Dinh thuc cua ma tran vuong bang: 2
```

List trong lập trình c#

I. Collection trong C#

Một collection (bộ, tập hợp) là một nhóm các đối tượng có sự liên quan đến nhau. Số đối tượng trong collect có thể thay đổi tăng giảm. Có nhiều loại collection, chúng được tập hợp vào namespace **System.Collections**. Thường thì một lớp collection có các phương thức để thêm, bớt, lấy tổng phần tử.

.NET cung cấp một số các lớp collection kiểu Generic như: **List<T>**, **Dictionary< TKey, TValue>**, **Stack<T>**,... những lớp generic này ở namespace **System.Collections.Generic**.

Ngoài ra, namespace **System.Collections** cũng có các lớp collection mà không sử dụng generic như: **ArrayList**, **Stack**, **Queue**,...

Các giao diện - interface về collect mà bạn có thể sử dụng.

Interface	Mô tả
IEnumerable<T>	Triển khai nó nếu muốn duyệt phần tử bằng foreach, nó định nghĩa phương thức GetEnumerator trả về một enumerator.
ICollection<T>	Giao diện này được triển khai bởi các generic collection. Với nó lấy tổng phần tử bằng thuộc tính Count, copy các phần tử vào mảng bằng CopyTo, thêm bớt phần tử với Add, Remove, Clear .
IList<T>	Giao diện này kế thừa ICollection<T> là một danh sách các phần tử truy cập được theo vị trí của nó. Nó có indexer, phương thức để chèn phần tử xóa phần tử Insert RemoveAt.

ISet<T>	Giao diện triển khai bởi các tập hợp.
IDictionary< TKey, TValue >	Giao diện để triển khai loại dữ liệu lưu trữ theo cặp key, value.
ILookup< TKey, TValue >	Giao diện để triển khai loại dữ liệu lưu trữ theo cặp key, value. Nhưng cho phép một key có nhiều giá trị.
IComparer< TKey, TValue >	Giao diện để triển khai cho phép so sánh để sắp xếp Collection.
IEqualityComparer< TKey, TValue >	Giao diện để triển khai cho phép so sánh bằng.

II. Lớp List<T>

Lớp collection **List** là lớp triển khai các giao diện **IList**, **ICollection**, **IEnumerable** nó quản lý danh sách các đối tượng cùng kiểu. Bạn có thể thêm, bớt, truy cập, sắp xếp các phần tử trong danh sách bằng các phương thức nó cung cấp như **Add**, **AddRange**, **Insert**, **RemoveAt**, **Remove**,... các phương thức này chi tiết theo ví dụ dưới.

Khởi tạo một danh sách List, mà các phần tử có kiểu element_type:

```
var list = new List<element_type>();
```

Ví dụ: Xây dựng danh sách các sản phẩm, sản phẩm có kiểu Product tự định nghĩa như sau - lớp sản phẩm hỗ trợ so sánh với sản phẩm khác nên triển khai **IComparable**, cho phép hiện lấy một chuỗi thông tin bằng **ToString** với định dạng nào đó nên triển khai giao diện **IFormattable**.

Mã nguồn xây dựng lớp **Product** trong file Product.cs như sau:

```
using System;
namespace Test_Generic
{
}
```

```
public class Product : IComparable<Product>, IFormattable
{
    public int ID {set; get;}
    public string Name {set; get;}      // tên
    public double Price {set; get;}     // giá
    public string Origin {set; get;}    // xuất xứ
    public Product(int id, string name, double price, string origin) {
        ID = id; Name = name; Price = price; Origin = origin;
    }
}
```

- //Triển khai Icomparable: Cho biết vị trí sắp xếp so với đối tượng khác.
- // trả về 0 - cùng vị trí; trả về > 0 đứng sau other; < 0 đứng trước trong danh sách.

```
public int CompareTo(Product other)
{
    // sắp xếp về giá
    double delta = this.Price - other.Price;
    if (delta > 0)    // giá lớn hơn xếp trước
        return -1;
    else if (delta < 0) // xếp sau, giá nhỏ hơn
        return 1;
    return 0;
}
```

- // Triển khai IFormattable: Lấy chuỗi thông tin của đối tượng theo định dạng.
- // format hỗ trợ "O" và "N".

```

public string ToString(string format, IFormatProvider formatProvider)
{
    if (format == null) format = "O";
    switch (format.ToUpper()) {
        case "O": // Xuất xứ trước
            return $"Xuất xứ: {Origin} - Tên: {Name} - Giá: {Price} - ID: {ID}";
        case "N": // Tên xứ trước
            return $"Tên: {Name} - Xuất xứ: {Origin} - Giá: {Price} - ID: {ID}";
        default: // Quăng lỗi nếu format sai
            throw new FormatException("Không hỗ trợ format này");
    }
}
// Nạp chồng ToString
override public string ToString() => $"Name: {Name} - Price: {Price}";
// Quá tải thêm ToString: Lấy chuỗi thông tin sản phẩm theo định dạng.
public string ToString(string format) => this.ToString(format, null);
}

```

1. Khởi tạo List<T>:

Để khởi tạo một danh sách rỗng, dùng toán tử new

```
var numbers = new List<int>(); // danh sách số nguyên
```

```
var products = new List<Product>(); // danh sách Product
```

Khởi tạo danh sách có sẵn một số phần tử, thì các phần tử liệt kê sau {}

```
var numbers = new List<int>() {1,2,3,4}; // khởi tạo 4 phần tử
```

```
var products = new List<Product>() // khởi tạo 1 phần tử
{
    new Product(1, "Iphone 6", 100, "Trung Quốc")
};
```

Thêm, xóa, chèn và đọc phần tử trong List<T>

Thêm phần tử vào cuối danh sách sử dụng phương thức Add

```
var p = new Product(2, "iPhone 14", 200, "Nhật bản");
products.Add(p); // Thêm p vào cuối List
products.Add(new Product(3, "iPhone 14 Promax", 400, "Nhật Bản")); // thêm đối tượng mới vào cuối List
```

Nếu muốn thêm nhiều phần tử một lúc (mảng các phần tử), dùng AddRange

```
var arrayProducts = new Product[] // Mảng 2 phần tử
{
    new Product(4, "Galaxy note 22", 500, "Việt Nam"),
    new Product(5, "Galaxy Note 8", 700, "Việt Nam"),
};
products.AddRange(arrayProducts); // Nối các phần tử của mảng vào danh sách
```

Chèn phần tử vào danh sách, phần tử sẽ ở vị trí chỉ ra dùng phương thức **Insert(index, object)** hoặc chèn cả một mảng **InsertRange(index, arrayObject)**. Trong đó index là vị trí chèn phần tử (0 là đầu tiên).

```
products.Insert(3, new Product(6, "Macbook Pro", 2000, "Mỹ")); // chèn phần tử vào vị trí index 3, (thứ 4)
```

Đọc phần tử trong List bạn dùng **indexer** với chỉ số (chỉ số bắt đầu từ 0).
Ví dụ lấy phần tử ở **Index = 1**:

```
var pro = products[2]; // đọc phần tử có index = 2
```

```
Console.WriteLine(pro.ToString());
```

Để duyệt qua các phần tử bạn có thể dùng lệnh for hoặc foreach

- // Duyệt qua tất cả các phần tử bằng for
- // products.Count = lấy tổng phần tử trong List

```
for (int i = 1; i < products.Count; i++)
```

```
{
```

```
    var pi = products[i - 1];
```

```
    Console.WriteLine(pi.ToString());
```

```
}
```

- // Duyệt qua các phần tử bằng foreach

```
foreach (var pi in products)
```

```
{
```

```
    Console.WriteLine(pi.ToString());
```

```
}
```

Xoá phần tử trong List - để xóa phần tử ở vị trí **index** dùng **RemoveAt(index)**, để xóa cả một đoạn count phần tử, từ vị trí **index** dùng **RemoveRange(index, count)**; để xóa toàn bộ (làm rỗng) gọi **Clear()**; hoặc **RemoveAll()**;

```
products.RemoveAt(0); // xóa phần tử đầu tiên
```

```
products.RemoveRange(products.Count - 2, 2); // xóa 2 phần tử cuối
```

Khi bạn có tham chiếu đến đối tượng đang có trong **List**, cũng có thể loại nó bằng **Remove(obj)**:

```
var pro_rm = products[1];
```

```
products.Remove(pro_rm); // xóa phần tử pro_rm
```

Tìm kiếm thông tin trong List

Một số phương thức cho phép tìm kiếm, tra cứu vị trí trí các phần tử trong List:

Phương thức	Mô tả
IndexOf(obj)	Tìm index của đối tượng trong List
LastIndexOf(obj)	Tìm index của phần tử cuối cùng có giá trị bằng obj trong List
FindIndex	Tìm kiếm trả về Index
FindLastIndex	Tìm kiếm trả về Index cuối
Find(Predicate)	Tìm kiếm trả về phần tử
FindAll(Predicate)	Tìm kiếm trả về danh sách phần tử
FindLast	Tìm kiếm trả về phần tử cuối tìm thấy

Trong các phương thức trên, có các phương thức ví dụ Find, chứa tham số là **delegate bool Predicate<in T>(T obj);**, nó là hàm callback, trả về true là phần tử phù hợp trả về (xem về sử dụng delegate trong C#). Ví dụ sau đây là một **Delegate** phù hợp gán cho tham số **Predicate**:

› // Delegate trả về true khi tên bằng "Galaxy 22"

```
(Product ob) => {  
    return (ob.Name == "Galaxy 22");  
}
```

Đoạn mã này có thể làm tham số cho Find, FindAll,...

```
Product foundpr1 = products.Find(  
    (Product ob) => { return (ob.Name == "Galaxy 22");}  
);
```

```
if (foundpr1 != null)
    Console.WriteLine("(found) " + foundpr1.ToString("O"));
    // (found) Xuất xứ: Việt Nam - Tên: Galaxy 22 - Giá: 700 - ID: 5
    Các delegate cũng có thể viết gọn lại
    // tìm index của đối tượng có xuất xứ là "Nhật bản"
    var ifound = products.FindIndex(x => x.Origin == "Nhật Bản");
    // tìm các sản phẩm có giá trên 100
    List<Product> p_100 = products.FindAll(product => product.Price > 100);
```

Nếu muốn tùy biến cao hơn **Delegate**, để tìm kiếm theo tham số tùy chọn, bạn có thể để **Delegate** trên vào lớp chức năng, ví dụ xây dựng lớp **SearchNameProduct**.

```
public class SearchNameProduct {
    string namesearch;
    public SearchNameProduct(string name) {
        namesearch = name;
    }
    // Hàm gán cho delegate
    public bool search(Product p) {
        return p.Name == namesearch;
    }
}
```

Thực hiện tìm kiếm, ví dụ

```
Product pr1 = products.Find((new SearchNameProduct("Galaxy
22")).search); // Tìm sản phẩm có tên Galaxy 22
```

```
Product pr2 = products.Find( new SearchNameProduct("iPhone 14")).search); // Tìm sản phẩm có tên iPhone 14
```

3. Sắp xếp các phần tử trong List

Để sắp xếp các phần tử trong danh sách, nếu phần tử đó có triển khai giao diện **IComparable** thì chỉ việc gọi **Sort()** để có danh sách theo thứ tự.

Ví dụ trên, lớp **Product** có triển khai **IComparable**, với phương thức **CompareTo**, thì sản phẩm nào có giá cao hơn xếp trước, có giá thấp hơn xếp sau.

```
products.Sort();
foreach (var pi in products)
{
    Console.WriteLine(pi.ToString("N"));
}
```

Bạn cũng có thể tùy biến cách thức sắp xếp bằng cách cung cấp hàm **callback** dạng **delegate** hai tham số kiểu cùng với kiểu phần tử cho **Search**, thay vì sắp xếp mặc định như trên.

Nhớ là trả về > 0 thì phần tử hiện tại xếp sau phần tử tham số.

Ví dụ: Hàm callback sau xếp ID nhỏ lên trước

```
products.Sort(
(p1, p2) => {
    if (p1.ID > p2.ID)
        return 1;
    else if (p1.ID == p2.ID)
        return 0;
    return -1;
})
```

```
};

foreach (var pi in products)
{
    Console.WriteLine(pi.ToString("N"));
}
```

Ngoài ra còn một số phương thức khác các bạn có thể tham khảo:

- › Contains(obj) kiểm tra có chứa phần tử obj.
- › Reverse() đảo thứ tự danh sách.
- › ToArray() copy các phần tử ra mảng.

