KHOA KỸ THUẬT VÀ CÔNG NGHỆ
BỘ MÔN CÔNG NGHỆ THÔNG TIN

THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH

HỌC KỲ I, NĂM HỌC 2024 - 2025

TÌM HIỂU JUPYTER BOOK VÀ BIÊN SOẠN TÀI LIỆU HỖ TRỢ LẬP TRÌNH PYTHON CƠ BẢN

Giáo viên hướng dẫn:
ThS. Nguyễn Bảo Ân

Sinh viên thực hiện:
Họ tên: Nguyễn Thiên Ân
MSSV: 110122030
Lớp: DA22TTA

Trà Vinh, tháng 12 năm 2024

KHOA KỸ THUẬT VÀ CÔNG NGHỆ
BỘ MÔN CÔNG NGHỆ THÔNG TIN

THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH

HỌC KỲ I, NĂM HỌC 2024 - 2025

TÌM HIỂU JUPYTER BOOK VÀ BIÊN SOẠN TÀI LIỆU HỖ TRỢ LẬP TRÌNH PYTHON CƠ BẢN

Giáo viên hướng dẫn:
ThS. Nguyễn Bảo Ân

Sinh viên thực hiện:
Họ tên: Nguyễn Thiên Ân
MSSV: 110122030
Lớp: DA22TTA

Trà Vinh, tháng 12 năm 2024

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

Trà Vinh, ngày ..... tháng …… năm ……

Giáo viên hướng dẫn

(Ký tên và ghi rõ họ tên)

NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

Trà Vinh, ngày ..... tháng …… năm ……

Thành viên hội đồng

(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Trước hết, tôi xin gửi lời cảm ơn sâu sắc đến các thầy cô trong khoa Kỹ thuật và Công nghệ trường Đại học Trà Vinh, những người đã tận tình giảng dạy và hướng dẫn tôi trong suốt quá trình học tập và nghiên cứu. Đặc biệt, tôi xin chân thành cảm ơn thầy Nguyễn Bảo Ân đã dành nhiều thời gian quý báu để hỗ trợ, chỉ bảo tôi trong suốt quá trình thực hiện báo cáo này.

Tôi cũng xin cảm ơn các bạn đồng môn, những người đã cùng tôi chia sẻ và trao đổi kiến thức, kinh nghiệm trong suốt quá trình học tập và nghiên cứu. Sự hỗ trợ, động viên của các bạn là nguồn động lực lớn đối với tôi.

Cuối cùng, tôi xin gửi lời cảm ơn đến gia đình và bạn bè đã luôn bên cạnh, động viên và tạo điều kiện tốt nhất cho tôi trong suốt thời gian qua. Những sự giúp đỡ và tình cảm chân thành của mọi người chính là nguồn động lực mạnh mẽ giúp tôi hoàn thành báo cáo này.

Tôi xin trân trọng cảm ơn!

MỤC LỤC

DANH MỤC HÌNH ẢNH – BẢNG BIỂU

DANH MỤC TỪ VIẾT TẮT

TỪ VIẾT TẮT	VIẾT THƯỜNG
HTML	HyperText Markup Language
PDF	Portable Document Format
CMS	Content Management System
URL	Uniform Resource Locator
YAML	YAML Ain't Markup Language
CSS	Cascading Style Sheets

TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH

Đề tài "Tìm hiểu Jupyter Book và biên soạn tài liệu hỗ trợ lập trình Python cơ bản" nghiên cứu việc sử dụng công cụ Jupyter Book để xây dựng tài liệu học tập cho lập trình Python. Mặc dù Python là một ngôn ngữ lập trình phổ biến và dễ học, nhưng việc tìm kiếm tài liệu chất lượng dành cho người mới bắt đầu vẫn còn hạn chế. Đặc biệt, các tài liệu chưa kết hợp đầy đủ giữa lý thuyết, ví dụ minh họa và bài tập thực hành trong một công cụ học tập thống nhất.

Để giải quyết vấn đề này, đồ án tập trung vào ba hướng chính:

- Nghiên cứu công cụ Jupyter Book, một phần mềm mã nguồn mở hỗ trợ xây dựng tài liệu điện tử, cho phép tích hợp văn bản, mã nguồn và kết quả tính toán trực tiếp.

- Tìm hiểu về mã Markdown, một ngôn ngữ đánh dấu văn bản nhẹ, được sử dụng phổ biến trong Jupyter Book để viết và định dạng văn bản trong tài liệu. Việc hiểu và sử dụng Markdown giúp tạo ra những tài liệu dễ đọc, dễ hiểu, đồng thời tích hợp các yếu tố như tiêu đề, danh sách, liên kết và hình ảnh một cách trực quan.

- Biên soạn tài liệu học lập trình Python cơ bản, với các chủ đề như cú pháp, kiểu dữ liệu, cấu trúc điều khiển và hàm trong Python. Tài liệu sẽ được thiết kế sao cho người học dễ dàng tiếp cận, nắm bắt lý thuyết và thực hành.

Đồ án có thể tiếp cận theo hai hướng:

- Tìm hiểu về Jupyter Book: Đồ án bắt đầu bằng việc nghiên cứu các tính năng của Jupyter Book, cách cài đặt và sử dụng công cụ này để xây dựng tài liệu học tập. Những tính năng nổi bật như khả năng tích hợp notebook Jupyter, hỗ trợ mã Markdown và xuất tài liệu dưới các định dạng HTML, PDF được tìm hiểu chi tiết.

- Biên soạn tài liệu Python cơ bản: Các chủ đề về lập trình Python cơ bản sẽ được lựa chọn và biên soạn thành các chương trong sách. Mỗi chương sẽ bao gồm lý thuyết, ví dụ minh họa và bài tập thực hành, giúp người học học lý thuyết và thực hành đồng thời.

Một số kết quả đạt được:

- Hoàn thành tài liệu học lập trình Python cơ bản với cấu trúc logic, dễ hiểu và dễ tiếp cận cho người mới bắt đầu.

- Tài liệu được xây dựng trên nền tảng Jupyter Book, giúp người học có thể tương tác với mã nguồn và xem kết quả trực tiếp trong tài liệu.

- Tài liệu có thể xuất ra các định dạng khác nhau (HTML, PDF) và dễ dàng chia sẻ qua các nền tảng trực tuyến.

MỞ ĐẦU

Lập trình Python hiện nay là một trong những ngôn ngữ phổ biến và dễ tiếp cận nhất đối với người mới bắt đầu học lập trình. Tuy nhiên, việc tìm kiếm tài liệu học tập phù hợp, dễ hiểu và có tính tương tác cao cho người học vẫn còn là một thách thức lớn. Đặc biệt đối với sinh viên mới bắt đầu, việc kết hợp lý thuyết và thực hành ngay trong tài liệu học có thể giúp quá trình học trở nên hiệu quả hơn. Với xu hướng học trực tuyến ngày càng phát triển, việc sử dụng công cụ như Jupyter Book để biên soạn tài liệu học tập mang lại nhiều lợi ích, giúp người học có thể thực hành ngay trong khi học lý thuyết. Chính vì vậy, đề tài "Tìm hiểu Jupyter Book và biên soạn tài liệu hỗ trợ lập trình Python cơ bản" được tôi lựa chọn để nghiên cứu và thực hiện.

Mục đích của đồ án là:

- Nghiên cứu, tìm hiểu các tính năng và ưu điểm của Jupyter Book, như khả năng tích hợp mã nguồn Python, kết quả tính toán và văn bản mô tả trong cùng một tài liệu, từ đó làm cơ sở để sử dụng công cụ này biên soạn tài liệu học lập trình Python.

- Xây dựng tài liệu học lập trình Python cho người mới bắt đầu, bao gồm các chủ đề cơ bản như cú pháp, kiểu dữ liệu, cấu trúc điều khiển và hàm, đồng thời kết hợp các bài tập thực hành để người học có thể rèn luyện kỹ năng lập trình ngay trong tài liệu.

Đối tượng nghiên cứu của đồ án bao gồm:

- Nghiên cứu và ứng dụng Jupyter Book trong việc biên soạn tài liệu học lập trình Python.

- Các khái niệm cơ bản về ngôn ngữ lập trình Python, từ cú pháp, kiểu dữ liệu cho đến các cấu trúc điều khiển và cách sử dụng các thư viện cơ bản trong Python.

- Những người mới bắt đầu học lập trình Python, đặc biệt là sinh viên năm nhất hoặc những người chưa có nền tảng lập trình vững chắc.

Phạm vi nghiên cứu:

- Đồ án sẽ nghiên cứu và ứng dụng Jupyter Book để tạo ra tài liệu học lập trình Python cơ bản. Phạm vi nghiên cứu chỉ giới hạn ở các tính năng cơ bản của Jupyter

Book, không bao gồm các tính năng nâng cao hoặc các vấn đề liên quan đến việc triển khai Jupyter Book trên quy mô lớn.

- Tài liệu học lập trình Python sẽ tập trung vào các khái niệm cơ bản, từ cú pháp, kiểu dữ liệu, cấu trúc điều khiển cho đến các bài tập thực hành. Các chủ đề nâng cao như lập trình hướng đối tượng hay các thư viện chuyên sâu sẽ được đề cập trong phạm vi nghiên cứu này, tuy nhiên sẽ không được chi tiết như các khái niệm cơ bản.

- Tài liệu sẽ được thiết kế chủ yếu cho sinh viên và người mới bắt đầu, giúp họ dễ dàng tiếp cận Python và có thể thực hành trực tiếp trên tài liệu.

CHƯƠNG 1: TỔNG QUAN

Python là một ngôn ngữ lập trình thông dụng, được sử dụng rộng rãi trong nhiều lĩnh vực như khoa học dữ liệu, trí tuệ nhân tạo, phát triển web, và tự động hóa. Với cú pháp đơn giản, dễ hiểu và khả năng áp dụng trong nhiều môi trường khác nhau, Python đã trở thành lựa chọn ưu tiên cho người mới bắt đầu học lập trình. Tuy nhiên, mặc dù có rất nhiều tài liệu học Python trên Internet, vấn đề quan trọng là làm thế nào để các tài liệu này không chỉ cung cấp lý thuyết mà còn giúp người học thực hành và áp dụng ngay kiến thức vào công việc thực tế.

Mặc dù Python có cú pháp dễ học, nhưng người mới bắt đầu vẫn gặp khó khăn trong việc tiếp cận và hiểu các khái niệm cơ bản như kiểu dữ liệu, cấu trúc điều khiển, hay các thao tác với thư viện. Việc thiếu các tài liệu học tập kết hợp giữa lý thuyết và thực hành khiến sinh viên khó có thể tự học và áp dụng ngay lập tức những gì đã học. Ngoài ra, các tài liệu học lập trình truyền thống thường chỉ bao gồm văn bản lý thuyết mà không tạo ra một môi trường học tập tương tác, điều này khiến người học cảm thấy thiếu động lực và không hiểu rõ cách áp dụng kiến thức vào thực tế.

Jupyter Book là một công cụ mã nguồn mở, cho phép xây dựng các tài liệu học tập dạng sách điện tử, trong đó người học có thể thực hành trực tiếp mã nguồn, xem kết quả tính toán và minh họa một cách dễ dàng. Jupyter Book hỗ trợ tích hợp mã nguồn Python vào các tài liệu dưới dạng các notebook Jupyter, giúp người học có thể tương tác với mã nguồn ngay trong khi đọc lý thuyết. Điều này mở ra cơ hội để tạo ra các tài liệu học lập trình vừa lý thuyết, vừa thực hành, tăng cường tính tương tác và hiệu quả trong việc học.

Tài liệu học lập trình Python cơ bản được biên soạn trên nền tảng Jupyter Book không chỉ cung cấp lý thuyết mà còn cho phép người học thực hành ngay trong tài liệu, giúp họ củng cố kiến thức qua các ví dụ thực tế. Các bài tập thực hành được tích hợp trong từng phần lý thuyết, giúp người học không chỉ hiểu lý thuyết mà còn có thể áp dụng ngay lập tức vào các bài toán cụ thể. Hơn nữa, tài liệu có thể xuất ra dưới các định dạng khác nhau (HTML, PDF) và dễ dàng chia sẻ trên nền tảng trực tuyến, giúp việc học trở nên thuận tiện hơn.

Mục tiêu của đồ án là:

- Nghiên cứu công cụ Jupyter Book để xây dựng tài liệu học lập trình Python cơ bản, với các chủ đề từ cú pháp, kiểu dữ liệu đến các cấu trúc điều khiển cơ bản.

- Biên soạn tài liệu học lập trình Python có tính chất tương tác cao, giúp người học không chỉ tiếp thu lý thuyết mà còn có thể thực hành ngay trong quá trình học.

- Đảm bảo tính ứng dụng thực tế của tài liệu, giúp người học dễ dàng áp dụng vào các bài tập thực hành và tình huống thực tế, đồng thời dễ dàng chia sẻ và tiếp cận qua các nền tảng trực tuyến.

CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

2.1. Cơ sở lý thuyết

2.1.1. Jupyter Book

Jupyter Book là một công cụ mạnh mẽ giúp tạo ra các tài liệu sách điện tử, bao gồm cả tài liệu học tập và tài liệu kỹ thuật, từ các mã nguồn Python và các tệp dữ liệu khác. Jupyter Book hỗ trợ tích hợp mã lệnh Python, văn bản, hình ảnh và biểu đồ một cách linh hoạt và dễ dàng chia sẻ với cộng đồng. Điều này làm cho nó trở thành một công cụ lý tưởng để biên soạn các tài liệu lập trình Python, đặc biệt là trong việc giảng dạy và nghiên cứu.

Jupyter Book được xây dựng trên nền tảng của Jupyter Notebooks, một môi trường phổ biến giúp lập trình viên viết mã, thực thi mã và ghi chú trực tiếp trong cùng một tài liệu. Jupyter Book mở rộng khả năng của Jupyter Notebooks bằng cách cho phép người dùng biên soạn sách chứa cả các đoạn mã, tài liệu văn bản và hình ảnh.

Các tính năng của Jupyter Book:

- Viết tài liệu bằng ngôn ngữ đánh dấu như Markdown, giúp quá trình viết và chỉnh sửa tài liệu trở nên đơn giản và dễ dàng.

- Jupyter Book cho phép nhúng các notebook Jupyter vào tài liệu, giúp người đọc có thể chạy mã trực tiếp trong tài liệu hoặc báo cáo.

- Jupyter Book có thể xuất ra các định dạng như HTML, PDF, LaTeX, và ePub, giúp chúng ta dễ dàng chia sẻ tài liệu của mình trên web hoặc dưới dạng tài liệu in ấn.

- Các mã nguồn có thể được thực thi trong tài liệu giúp tạo ra các biểu đồ, đồ thị hoặc kết quả tính toán trực tiếp từ tài liệu mà không cần phải chuyển qua lại giữa các phần mềm khác nhau.

- Jupyter Book cho phép liên kết đến các tệp dữ liệu, hình ảnh và các tài liệu khác, giúp mở rộng nội dung của sách.

- Jupyter Book hỗ trợ tùy biến giao diện của sách với các chủ đề khác nhau, giúp người dùng có thể thay đổi màu sắc, font chữ và bố cục trang sách một cách linh hoạt.

Cấu trúc của một Jupyter Book:

- Tài liệu chính: Các tệp Markdown chứa nội dung bài viết hoặc sách.

- Jupyter Notebooks: Các tệp notebook Jupyter (định dạng .ipynb) có thể chứa mã nguồn, đồ thị, biểu đồ, v.v.

- Cấu hình: Tệp config.yml giúp chúng ta cấu hình các tham số của sách như tiêu đề, tác giả, bố cục, v.v.

- Hình ảnh và tài liệu hỗ trợ: Các tệp hình ảnh, tài liệu hoặc dữ liệu liên quan đến nội dung của sách.

Quy trình tạo một Jupyter Book:

- Cài đặt Jupyter Book bằng cách sử dụng "pip" hoặc "conda" trong môi trường Python.

- Sử dụng câu lệnh "jupyter-book create <tên-sách>" để tạo một dự án Jupyter Book mới với cấu trúc thư mục chuẩn.

- Viết các trang sách bằng Markdown, thêm các đoạn mã Jupyter Notebook vào các trang đó.

- Sau khi hoàn thành, sử dụng câu lệnh "jupyter-book build" để tạo các tệp HTML, PDF để xuất bản sách của mình.

Ưu điểm của Jupyter Book:

- Jupyter Book cho phép kết hợp mã nguồn, văn bản mô tả, đồ họa và kết quả phân tích trong cùng một tài liệu, giúp tài liệu trở nên sinh động và dễ hiểu.

- Các tài liệu được tạo ra từ Jupyter Book có thể được chia sẻ trực tuyến, giúp cộng đồng dễ dàng tiếp cận và sử dụng tài liệu.

- Jupyter Book là một dự án mã nguồn mở, vì vậy chúng ta có thể dễ dàng tham gia đóng góp hoặc sửa đổi phần mềm để đáp ứng nhu cầu cụ thể của mình.

2.1.2. Markdown

Markdown là một ngôn ngữ đánh dấu (markup language) được thiết kế để dễ đọc, dễ viết, và dễ chuyển đổi sang các định dạng khác như HTML. Được phát triển bởi John Gruber vào năm 2004, Markdown đã trở thành một công cụ phổ biến để soạn thảo văn bản, đặc biệt là trong các nền tảng như GitHub, Reddit, hay trong các hệ thống quản lý nội dung (CMS) như Jekyll và Hugo.

Markdown sử dụng các ký hiệu đặc biệt để xác định các thành phần của tài liệu như tiêu đề, danh sách, chữ in đậm, chữ nghiêng, liên kết, và hình ảnh, mà không cần phải sử dụng các thẻ HTML phức tạp. Nhờ đó, người viết có thể tạo ra các tài liệu dễ đọc ngay cả khi chưa được chuyển đổi sang HTML.

Cấu trúc cơ bản của Markdown:

- Tiêu đề (Headings): Để tạo tiêu đề, sử dụng dấu thăng (#) theo sau là nội dung tiêu đề. Số lượng dấu # xác định cấp độ của tiêu đề (từ 1 đến 6).

```
# Tiêu đề cấp 1
## Tiêu đề cấp 2
### Tiêu đề cấp 3
#### Tiêu đề cấp 4
##### Tiêu đề cấp 5
###### Tiêu đề cấp 6
```

Hình 2.1: Ví dụ về tiêu đề trong Markdown

- Đoạn văn bản (Paragraphs): Đoạn văn bản được tạo đơn giản bằng cách viết văn bản bình thường. Để phân cách các đoạn văn, chỉ cần để trống một dòng giữa các đoạn.

```
Đây là một đoạn văn bản.

Đây là đoạn văn bản thứ hai.
```

Hình 2.2: Ví dụ về đoạn văn bản trong Markdown

- In đậm và in nghiêng:

+ In nghiêng: Sử dụng một dấu sao (*) hoặc dấu gạch dưới (_).

+ In đậm: Sử dụng hai dấu sao (**) hoặc hai dấu gạch dưới (__).

```
*In nghiêng*
**In đậm**
```

Hình 2.3: Ví dụ về in nghiêng và in đậm trong Markdown

- Danh sách:

+ Danh sách không có thứ tự (Unordered List): Dùng dấu -, +, hoặc * để tạo mục trong danh sách.

```
- Mục 1
- Mục 2
  - Mục con 1
  - Mục con 2
```

Hình 2.4: Ví dụ về danh sách có thứ tự trong Markdown

+ Danh sách có thứ tự (Ordered List): Dùng các số theo sau là dấu chấm (1., 2., ...).

```
1. Mục 1
2. Mục 2
   1. Mục con 1
   2. Mục con 2
```

Hình 2.5: Ví dụ về danh sách có thứ tự trong Markdown

- Đoạn mã (Code blocks):

+ Mã nguồn inline: Đặt mã trong dấu nháy đơn (`).

```
Đoạn mã inline: `print("Hello World")`
```

Hình 2.6: Ví dụ về mã nguồn trong Markdown

+ Mã nguồn dạng khối: Dùng ba dấu nháy ngược (```).

- Liên kết (Links): Để tạo liên kết, sử dụng cú pháp [Text hiển thị](URL)

```
[Google](https://www.google.com)
```

Hình 2.7: Ví dụ về chèn liên kết trong Markdown

- Hình ảnh (Images): Để thêm hình ảnh, cú pháp giống như liên kết, nhưng có thêm dấu chấm than (!) ở đầu. Cú pháp ![Alt text](URL của hình ảnh)

```
![Alt text](URL của hình ảnh)
```

Hình 2.8: Ví dụ về chèn hình ảnh trong Markdown

- Trích dẫn (Blockquotes): Để tạo trích dẫn, sử dụng dấu lớn hơn (>).

```
> Đây là một trích dẫn.
```

Hình 2.9: Ví dụ về cách trích dẫn trong Markdown

- Chèn đường ngang (Horizontal rules): Để tạo đường ngang, chúng ta có thể sử dụng ba dấu gạch ngang (---), ba dấu sao (***), hoặc ba dấu gạch dưới (___).

```
---
```

Hình 2.10: Ví dụ chèn đường ngang trong Markdown

2.1.3. Ngôn ngữ YAML

YAML (YAML Ain't Markup Language) là một ngôn ngữ đánh dấu (markup language) dùng để mô tả cấu trúc dữ liệu, được thiết kế để dễ đọc, dễ viết và dễ hiểu cho con người. YAML là viết tắt của "YAML Ain't Markup Language", nghĩa là YAML không phải là một ngôn ngữ đánh dấu như HTML. Nó được thiết kế đặc biệt để mô tả các cấu trúc dữ liệu như đối tượng, mảng, danh sách, và các cặp khóa-giá trị trong một cách thức dễ dàng và trực quan.

Các tính năng của YAML:

- YAML có cú pháp rất đơn giản và dễ hiểu. Điều này giúp người dùng có thể dễ dàng đọc và chỉnh sửa các tệp YAML mà không gặp khó khăn.

- Cấu trúc của YAML chủ yếu dựa vào việc thụt lề (indentation) mà không sử dụng dấu ngoặc hoặc dấu chấm phẩy, giúp giảm bớt sự phức tạp của cú pháp và giúp tệp YAML dễ đọc hơn.

- YAML hỗ trợ nhiều kiểu dữ liệu khác nhau như chuỗi, số, danh sách, và bản đồ (dictionary), làm cho nó phù hợp với nhiều mục đích sử dụng khác nhau.

Cấu trúc cơ bản của YAML:

- Cặp khóa-giá trị: Mỗi mục dữ liệu trong YAML được xác định bằng một cặp khóa-giá trị, với khóa và giá trị được phân cách bởi dấu hai chấm (:) theo sau một dấu cách.

```yaml
name: "John Doe"
age: 25
```

Hình 2.11: Ví dụ về cấu trúc cặp khóa-giá trị trong YAML

- Danh sách (List): Danh sách trong YAML được biểu diễn bằng cách liệt kê các mục trong một dòng và đặt dấu gạch ngang (-) ở đầu mỗi mục trong danh sách.

```yaml
fruits:
  - Apple
  - Banana
  - Orange
```

Hình 2.12: Ví dụ về cấu trúc danh sách trong YAML

- Bản đồ (Dictionary): Bản đồ trong YAML là các cặp khóa-giá trị lồng nhau, được phân cấp bằng cách thụt lề.

```yaml
address:
  street: 123 Main St
  city: New York
  zip: 10001
```

Hình 2.13: Ví dụ về cấu trúc bản đồ trong YAML

Các kiểu dữ liệu trong YAML:

- Chuỗi (String): Có thể được viết dưới dạng văn bản đơn giản hoặc có thể đặt trong dấu ngoặc kép nếu chuỗi chứa các ký tự đặc biệt.

```yaml
name: "Alice"
greeting: Hello, world!
```

Hình 2.14: Ví dụ về kiểu chuỗi trong YAML

- Số (Number): YAML hỗ trợ số nguyên và số thực.

```yaml
quantity: 100
price: 29.99
```

Hình 2.15: Ví dụ về kiểu số trong YAML

- Boolean: YAML hỗ trợ các giá trị true và false.

```yaml
active: true
```

Hình 2.16: Ví dụ về kiểu Boolean trong YAML

- Danh sách (List): Danh sách được biểu diễn bằng dấu gạch ngang.

```yaml
colors:
  - Red
  - Green
  - Blue
```

Hình 2.17: Ví dụ về kiểu danh sách trong YAML

- Bản đồ (Dictionary): Bản đồ là các cặp khóa-giá trị được thụt lề.

```yaml
person:
  name: "Bob"
  age: 30
```

Hình 2.18: Ví dụ về kiểu bản đồ trong YAML

Các tính năng khác của YAML:

- Chú thích: YAML cho phép người dùng thêm chú thích vào tệp dữ liệu. Chú thích được bắt đầu bằng dấu # và kéo dài đến hết dòng.

```yaml
# Đây là một chú thích
name: "John"
```

Hình 2.19: Ví dụ về tính năng chú thích trong YAML

- Dữ liệu nhiều dòng: YAML cho phép biểu diễn chuỗi văn bản dài và nhiều dòng bằng cách sử dụng ký tự đặc biệt | (giữ nguyên dòng mới) hoặc > (nối các dòng).

```yaml
description: |
  Đây là một đoạn văn bản
  có thể chứa nhiều dòng.
```

Hình 2.20: Ví dụ về tính năng dữ liệu nhiều dòng trong YAML

```yaml
description: >
  Đây là một đoạn văn bản dài
  và sẽ được nối lại thành một dòng.
```

Hình 2.21: Ví dụ về tính năng dữ liệu nhiều dòng trong YAML

- Dữ liệu kiểu null: YAML hỗ trợ kiểu null để biểu diễn các giá trị không xác định.

```yaml
middle_name: null
```

Hình 2.22: Ví dụ về dữ liệu kiểu null trong YAML

Ưu, nhược điểm của YAML:

- Ưu điểm:

+ Cú pháp dễ đọc và dễ viết cho con người.

+ Linh hoạt và dễ dàng mở rộng.

+ Hỗ trợ cấu trúc dữ liệu phức tạp, bao gồm danh sách lồng nhau và bản đồ.

- Nhược điểm:

+ Yêu cầu sự chú ý đặc biệt về thụt lề (indentation), nếu không sẽ dễ bị lỗi.

+ Không phải tất cả các ngôn ngữ lập trình đều có thư viện hỗ trợ đầy đủ cho YAML.

2.1.4. Python cơ bản

Python là một trong những ngôn ngữ lập trình phổ biến và dễ học nhất hiện nay. Python được sử dụng rộng rãi trong nhiều lĩnh vực, bao gồm phát triển web, phân tích dữ liệu, học máy và trí tuệ nhân tạo. Với cú pháp rõ ràng và dễ hiểu, Python đặc biệt thích hợp cho những người mới bắt đầu học lập trình.

Lập trình Python cơ bản bao gồm các khái niệm cơ bản như biến, kiểu dữ liệu, câu lệnh điều kiện, vòng lặp, hàm và lớp (classes). Những khái niệm này là nền tảng quan trọng để xây dựng các chương trình phức tạp hơn trong Python. Việc biên soạn tài liệu học lập trình Python cơ bản có thể giúp người học dễ dàng tiếp cận và nắm vững các kỹ năng lập trình từ những bước đầu tiên.

Các tính năng của Python:

- Python có cú pháp rất dễ học, đặc biệt với những người mới bắt đầu lập trình.

- Python không cần biên dịch trước khi chạy chương trình. Mã nguồn Python được thông dịch trực tiếp qua một chương trình gọi là "interpreter", điều này giúp việc phát triển phần mềm trở nên nhanh chóng và dễ dàng.

- Python hỗ trợ lập trình hướng đối tượng, giúp tổ chức và quản lý mã nguồn tốt hơn, dễ dàng tái sử dụng mã.

- Python đi kèm với một thư viện chuẩn rất lớn và có một cộng đồng phát triển mạnh mẽ, cung cấp hàng ngàn thư viện và module hỗ trợ mọi lĩnh vực từ web development, phân tích dữ liệu, đến học máy và trí tuệ nhân tạo.

- Python có thể chạy trên nhiều hệ điều hành khác nhau như Windows, macOS, Linux mà không cần thay đổi mã nguồn.

- Python không yêu cầu khai báo kiểu dữ liệu khi khai báo biến. Kiểu dữ liệu sẽ được gán tự động khi thực thi.

Cấu trúc của một chương trình Python:

- Biến: Dùng để lưu trữ dữ liệu.

- Câu lệnh điều kiện: Sử dụng "if, elif, else" để thực hiện các quyết định trong chương trình.

- Vòng lặp: "for" và "while" được dùng để lặp qua các dãy giá trị hoặc thực hiện một tác vụ nhiều lần.

- Hàm (Function): Hàm là các đoạn mã có thể tái sử dụng, giúp chương trình gọn gàng và dễ bảo trì.

- Cấu trúc dữ liệu: Python hỗ trợ nhiều loại cấu trúc dữ liệu như danh sách (list), bộ (tuple), từ điển (dictionary) và tập hợp (set).

```python
# Chương trình Python đơn giản để tính tổng của 2 số
def tinh_tong(a, b):
    return a + b

x = 5
y = 10
ket_qua = tinh_tong(x, y)
print("Tổng của", x, "và", y, "là:", ket_qua)
```

Hình 2.23: Ví dụ về cấu trúc cơ bản của Python

Ưu điểm của Python:

- Với cú pháp đơn giản, Python dễ dàng tiếp cận với người mới bắt đầu.

- Python có một cộng đồng người dùng và nhà phát triển lớn mạnh, cung cấp rất nhiều tài liệu học hỏi, thư viện và công cụ hỗ trợ.

- Python có thể sử dụng trong nhiều lĩnh vực, từ khoa học dữ liệu đến phát triển ứng dụng web và lập trình nhúng.

- Python có thể chạy trên nhiều hệ điều hành khác nhau mà không cần thay đổi mã nguồn.

2.1.5. Visual Studio Code

Visual Studio Code là một trình soạn thảo mã nguồn miễn phí và mã nguồn mở, được phát triển bởi Microsoft. Được ra mắt lần đầu tiên vào năm 2015, Visual Studio Code đã trở thành một trong những công cụ phổ biến nhất cho lập trình viên trên toàn thế giới nhờ vào tính năng mạnh mẽ, giao diện người dùng thân thiện, và khả năng mở rộng thông qua các plugin.

Visual Studio Code hỗ trợ một loạt các ngôn ngữ lập trình và công nghệ, bao gồm nhưng không giới hạn ở JavaScript, Python, C++, Java, HTML, CSS, và nhiều ngôn ngữ khác. Nó cung cấp các tính năng tiện ích như highlighting cú pháp, autocompletion, debugging, version control (Git), và rất nhiều extensions (tiện ích mở rộng) cho phép người dùng tùy chỉnh và mở rộng tính năng của nó theo nhu cầu cụ thể.

2.2. Phương pháp nghiên cứu

- Nghiên cứu các tài liệu và công cụ học lập trình Python, đặc biệt là Jupyter Book, để hiểu rõ về các tính năng và lợi ích của nó trong việc tạo tài liệu học lập trình.

- Áp dụng Jupyter Book để biên soạn tài liệu học lập trình Python cơ bản, với mục tiêu tạo ra một tài liệu dễ hiểu và dễ tiếp cận cho người học mới bắt đầu.

- Thực hiện khảo sát đối với người học và giảng viên để đánh giá mức độ hiệu quả của tài liệu học lập trình Python được biên soạn bằng Jupyter Book.

CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

3.1. Khởi tạo dự án

Công cụ mà tôi dùng để thực hiện nghiên cứu, tìm hiểu Jupyter Book là Visual Studio Code. Để thực hiện trước hết ta cần cài đặt Jupyter Book trong Terminal của Visual Studio Code. Để cài đặt ta dùng lệnh:

```
pip install -U jupyter-book
```

Hình 3.1: Cài đặt Jyputer Book bằng lệnh

Khởi tạo dự án bằng lệnh: jupyter-book create ten_sach

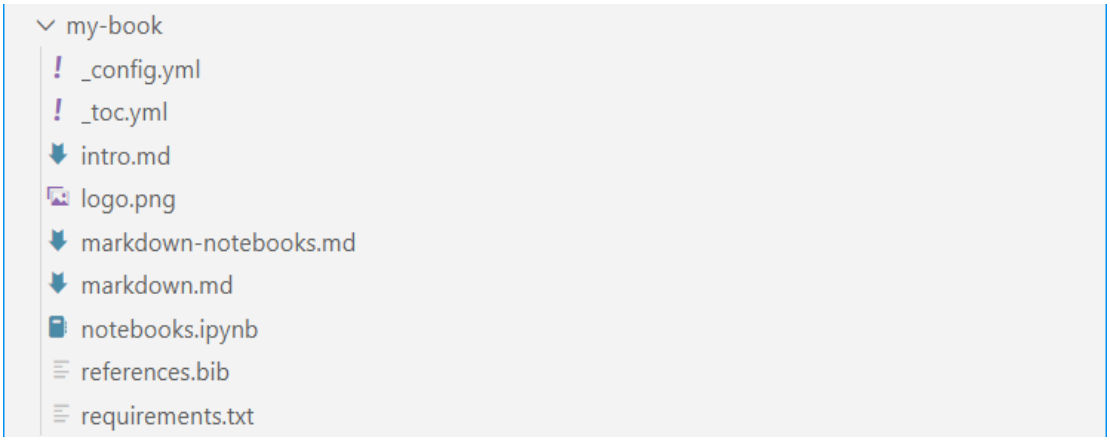
```
jupyter-book create my-book
```

Hình 3.2: Khởi tạo dự án bằng lệnh

Sau khi tạo được, ta sẽ có được thư mục như sau:

```
my-book/
├── _config.yml          # Cấu hình dự án
├── _toc.yml             # Mục lục
├── content/             # Nội dung của sách (chứa các file markdown và notebook)
│   ├── intro.md
│   └── chapter1.md
└── requirements.txt     # Các thư viện phụ thuộc
```

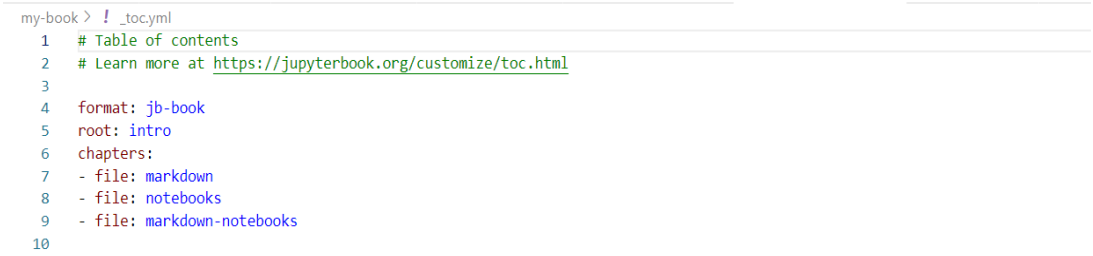
Hình 3.3: Cây thư mục sau khi tạo dự án Jupyter Book



Hình 3.4: Các file khi vừa tạo dự án Jupyter Book trong VS Code

3.2. Xác định cấu trúc tài liệu

Cấu trúc của tài liệu Jupyter Book thường được định nghĩa trong tệp _toc.yml (Table of Contents). Tệp này xác định cách sắp xếp các chương, phần và các tệp tài liệu trong sách.



Hình 3.5: Cấu trúc ban đầu của file _toc.yml

3.3. Biên tập mục lục

Tệp _toc.yml giúp tạo mục lục của sách, sắp xếp các chương và phần. Cấu trúc của mục lục có thể bao gồm các chương, phần con, và các tài liệu phụ.

```yaml
# _toc.yml
- file: intro
  title: Giới thiệu
- file: chapter1
  title: Chương 1: Lý thuyết cơ bản
  sections:
    - file: chapter1_section1
      title: Phần 1.1: Tổng quan
    - file: chapter1_section2
      title: Phần 1.2: Các khái niệm cơ bản
- file: chapter2
  title: Chương 2: Ứng dụng thực tế
```

Hình 3.6: Ví dụ về cách biên tập mục lục trong file _toc.yml

Với đề tài "Biên soạn tài liệu hỗ trợ lập trình Python cơ bản". Tôi đã tạo một mục lục đơn giản như sau:

mybook > ! _toc.yml

```yaml
# Table of contents
# Learn more at https://jupyterbook.org/customize/toc.html

format: jb-book
root: intro
chapters:
- file: bia
- file: markdown
- file: notebooks
- file: markdown-notebooks
- file: mucluc
- file: chuong1
- file: chuong2
- file: chuong3
- file: chuong4
- file: chuong5
- file: chuong6
- file: chuong7
- file: chuong8
- file: chuong9
- file: chuong10
- file: chuong11
- file: chuong12
- file: chuong13
- file: chuong14
- file: chuong15
- file: chuong16
```

Hình 3.7: Biên tập mục lục theo đề tài

3.4. Biên tập nội dung chương

- Tiêu đề và nội dung: Được tạo bằng cách sử dụng cú pháp Markdown.

mybook > chuong1.md > ## 1. Giới thiệu về Python > ### 1.5. Cài đặt Python trên các hệ điều hành

```markdown
## 1. Giới thiệu về Python

### 1.1. Python là gì?
Python là một ngôn ngữ lập trình bậc cao, thông dịch, và hướng đối tượng, được thiết kế với mục tiêu dễ học, dễ đọc và dễ sử dụng. Python được Guido van Rossum phát triển lần đầu tiên vào năm 1991 và đã trở thành một trong những ngôn ngữ lập trình phổ biến nhất trên thế giới. Python có cú pháp rõ ràng, dễ hiểu và hỗ trợ nhiều thư viện mạnh mẽ, giúp lập trình viên có thể nhanh chóng xây dựng các ứng dụng mà không phải lo lắng quá nhiều về chi tiết kỹ thuật. Python có thể được sử dụng cho nhiều mục đích khác nhau, từ phát triển phần mềm, khoa học dữ liệu, học máy (machine learning), trí tuệ nhân tạo (AI) đến các ứng dụng web và tự động hóa.

```

Hình 3.8: Tiêu đề và nội dung đã tạo

- Mã nguồn: Để nhúng mã nguồn trong Jupyter Book, tôi sử dụng cú pháp Markdown để nhúng mã Python hoặc các ngôn ngữ lập trình khác.

```python
class Dog:
    def __init__(self, name):
        self.name = name

    def bark(self):
        print(f"{self.name} barks!")
```

Hình 3.9: Nhúng mã nguồn vào dự án Jupyter Book

3.5. Biên dịch dự án

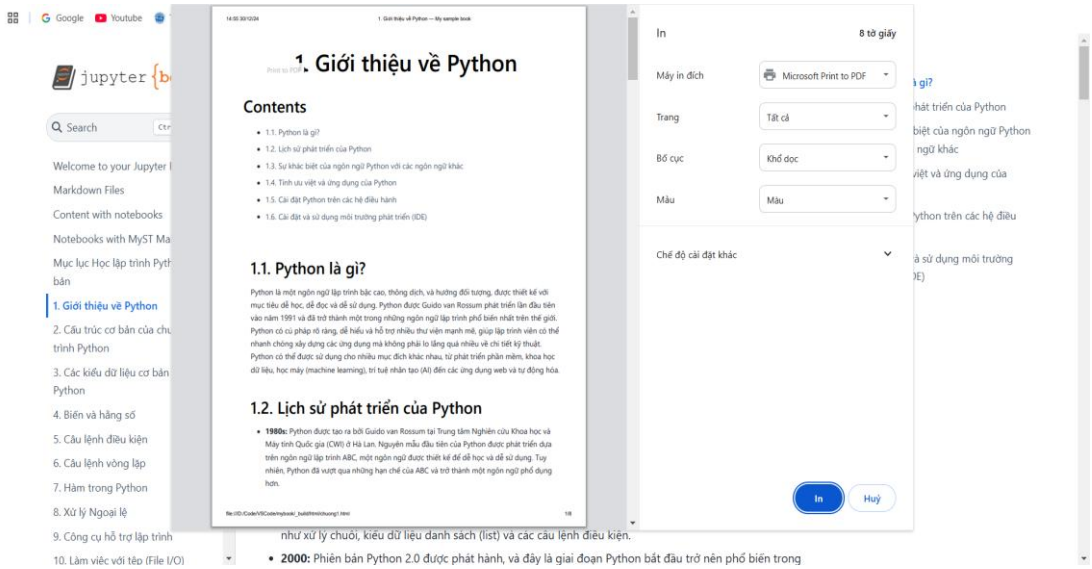
Sau khi biên soạn tài liệu và mục lục xong, ta có thể biên dịch dự án để kiểm tra trước khi xuất bản. Để biên dịch dự án, sử dụng lệnh: jupyter-book build ten_sach

```
jupyter-book build my-book/
```

Hình 3.10: Biên dịch dự án bằng lệnh

3.6. Xuất bản dự án thành file PDF

Sau khi biên dịch xong, ta có thể chạy tài liệu trực tiếp trên trình duyệt bằng file index theo đường dẫn "ten_sach/_build/html/index". Và ta có thể xuất file PDF trực tiếp trên đó.



Hình 3.11: Xuất file PDF trên trình duyệt

CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

4.1. Kết quả nghiên cứu

Qua quá trình nghiên cứu và thực nghiệm, tôi đã hoàn thành một tài liệu hướng dẫn lập trình Python cơ bản sử dụng Jupyter Book. Tài liệu này cung cấp cho người học những kiến thức nền tảng về lập trình Python, bao gồm các khái niệm cơ bản như biến, kiểu dữ liệu, cấu trúc điều kiện, vòng lặp và hàm. Ngoài ra, tài liệu còn bao gồm các ví dụ mã nguồn Python giúp người học có thể thực hành và tương tác trực tiếp với mã trong môi trường Jupyter Notebook.

4.2. Ưu điểm

Tài liệu được tổ chức một cách logic và dễ hiểu, với trình tự hợp lý từ các khái niệm cơ bản đến ứng dụng nâng cao. Điều này giúp người học dễ dàng tiếp cận và thực hành. Hơn nữa, tài liệu được thiết kế để dễ dàng xuất bản và chia sẻ dưới nhiều định dạng khác nhau như HTML và PDF, thuận tiện cho việc học tập và giảng dạy lập trình Python trong nhiều môi trường khác nhau. Việc tích hợp mã nguồn có thể chạy trực tiếp trong Jupyter Notebook giúp người học không chỉ hiểu lý thuyết mà còn củng cố kiến thức qua thực hành.

4.3. Nhược điểm

Một số nhược điểm của tài liệu là giao diện chưa thực sự bắt mắt, làm giảm trải nghiệm người dùng khi đọc và học. Mặc dù Jupyter Book cung cấp một nền tảng tiện lợi để chia sẻ tài liệu, nhưng giao diện của nó vẫn còn đơn giản và thiếu sự tinh chỉnh về mặt thẩm mỹ. Thêm vào đó, việc sử dụng Markdown để soạn thảo tài liệu cũng có hạn chế, như không hỗ trợ tốt việc căn giãn lề và bố cục của văn bản, khiến cho một số phần của tài liệu có thể trông không đồng đều hoặc thiếu sự chuyên nghiệp. Điều này có thể ảnh hưởng đến sự dễ đọc và tính thẩm mỹ của tài liệu, nhất là khi cần trình bày các thông tin quan trọng.

4.4. Hoàn thiện tài liệu

Dưới đây là một phần "Tài liệu hỗ trợ lập trình Python cơ bản" mà tôi đã biên soạn được qua quá trình tìm hiểu và thực nghiệm:

Lập trình Python Cơ bản

Contents

Trường: Đại học Trà Vinh

Khoa: Kỹ thuật và Công nghệ

Bộ môn: Công nghệ thông tin

Tên tài liệu: **Lập trình Python Cơ bản**

Tác giả: Nguyễn Thiên Ân + ChatGPT

Mục lục

Contents

Chương 1. Giới thiệu về Python

Chương 2. Cấu trúc cơ bản của chương trình Python

Chương 3. Các kiểu dữ liệu cơ bản trong Python

Chương 4. Biến và hằng số

Chương 1. Giới thiệu về Python

Contents

1.1. Python là gì?

Python là một ngôn ngữ lập trình bậc cao, thông dịch, và hướng đối tượng, được thiết kế với mục tiêu dễ học, dễ đọc và dễ sử dụng. Python được Guido van Rossum phát triển lần đầu tiên vào năm 1991 và đã trở thành một trong những ngôn ngữ lập trình phổ biến nhất trên thế giới. Python có cú pháp rõ ràng, dễ hiểu và hỗ trợ nhiều thư viện mạnh mẽ, giúp lập trình viên có thể nhanh chóng xây dựng các ứng dụng mà không phải lo lắng quá nhiều về chi tiết kỹ thuật. Python có thể được sử dụng cho nhiều mục đích khác nhau, từ phát triển phần mềm, khoa học dữ liệu, học máy (machine learning), trí tuệ nhân tạo (AI) đến các ứng dụng web và tự động hóa.

1.2. Lịch sử phát triển của Python

- **1980s:** Python được tạo ra bởi Guido van Rossum tại Trung tâm Nghiên cứu Khoa học và Máy tính Quốc gia (CWI) ở Hà Lan. Nguyên mẫu đầu tiên của Python được phát triển dựa trên ngôn ngữ lập trình ABC, một ngôn ngữ được thiết kế để dễ học và dễ sử dụng. Tuy nhiên, Python đã vượt qua những hạn chế của ABC và trở thành một ngôn ngữ phổ dụng hơn.

- **1991:** Python chính thức được phát hành lần đầu tiên với phiên bản 0.9.0, bao gồm các tính năng cơ bản như xử lý chuỗi, kiểu dữ liệu danh sách (list) và các câu lệnh điều kiện.
- **2000:** Phiên bản Python 2.0 được phát hành, và đây là giai đoạn Python bắt đầu trở nên phổ biến trong cộng đồng lập trình. Python 2.x tiếp tục được duy trì cho đến năm 2020.
- **2008:** Phiên bản Python 3.0 được phát hành, mang lại những cải tiến về cú pháp và hiệu suất, mặc dù không tương thích ngược với các phiên bản trước. Điều này gây ra sự phân chia trong cộng đồng, với nhiều người vẫn sử dụng Python 2 trong khi một số khác chuyển sang Python 3.
- **2020:** Python 2 chính thức hết hạn hỗ trợ, và Python 3 trở thành phiên bản chính thức và được duy trì. Python 3.x hiện tại đã trở thành phiên bản chính thức của ngôn ngữ này và có sự phát triển liên tục với nhiều cải tiến mạnh mẽ.

1.3. Sự khác biệt của ngôn ngữ Python với các ngôn ngữ khác

- **Cú pháp đơn giản và dễ đọc:** Cú pháp của Python rất rõ ràng và dễ đọc, gần gũi với ngôn ngữ tự nhiên, không yêu cầu dấu chấm phẩy (`;`) để kết thúc câu lệnh và không cần dấu ngoặc nhọn (`{}`) để phân biệt các khối mã (block of code). Thay vào đó, Python sử dụng thụt lề để xác định phạm vi (scope) của các khối mã.

```
if x > 10:
    print("x is greater than 10")
```

- **Kiểu dữ liệu động (Dynamic Typing):** Python là ngôn ngữ kiểu dữ liệu động (dynamically typed), tức là bạn không cần phải khai báo kiểu dữ liệu của biến trước khi sử dụng. Kiểu dữ liệu sẽ được gán tự động khi bạn gán giá trị cho biến.

```
x = 10  # x là kiểu int
x = "Hello"  # x thay đổi thành kiểu string
```

- **Quản lý bộ nhớ:** Python tự động quản lý bộ nhớ thông qua garbage collection (GC), nghĩa là bộ nhớ sẽ được giải phóng tự động khi các đối tượng không còn được tham chiếu.
- **Lập trình hướng đối tượng:** Python hỗ trợ lập trình hướng đối tượng (OOP) một cách linh hoạt và dễ dàng, nhưng không bắt buộc. Bạn có thể sử dụng các tính năng OOP (như kế thừa, đóng gói, và đa hình) mà không cần khai báo quá phức tạp.

```
class Dog:
  def __init__(self, name):
      self.name = name

  def bark(self):
      print(f"{self.name} barks!")
```

- **Quản lý thư viện và gói (Package Management):** Python có hệ thống quản lý gói rất mạnh mẽ và dễ sử dụng với `pip`, cho phép bạn dễ dàng cài đặt và quản lý thư viện bên ngoài. Python cũng có PyPI (Python Package Index), nơi lưu trữ hàng ngàn thư viện và công cụ sẵn có.
- **Tính tương thích đa nền tảng:** Python hỗ trợ đa nền tảng tuyệt vời, có thể chạy trên Windows, macOS, Linux mà không gặp phải nhiều vấn đề tương thích. Mã Python thường không cần thay đổi khi chuyển giữa các hệ điều hành.
- **Quản lý lỗi (Exception Handling):** Python sử dụng hệ thống xử lý ngoại lệ đơn giản và dễ hiểu. Các khối `try-except` giúp xử lý lỗi một cách mạch lạc.

```
try:
    x = 10 / 0
except ZeroDivisionError:
    print("Cannot divide by zero!")
```

- **Tốc độ thực thi:** Python là một ngôn ngữ diễn dịch (interpreted language), tức là mã nguồn sẽ được biên dịch và thực thi trực tiếp bởi trình thông dịch (interpreter) mà không cần biên dịch trước. Điều này khiến Python chậm hơn so với các ngôn ngữ biên dịch như C hay C++.
- **Sử dụng trong các lĩnh vực:** Python thường được sử dụng trong các lĩnh vực như khoa học dữ liệu, trí tuệ nhân tạo, tự động hóa, phát triển web (Django, Flask), và các ứng dụng học máy. Python nổi bật trong việc nhanh chóng phát triển các nguyên mẫu và các dự án nghiên cứu.

1.4. Tính ưu việt và ứng dụng của Python

Python có nhiều đặc điểm nổi bật giúp ngôn ngữ này được yêu thích trong nhiều lĩnh vực lập trình:

- **Cú pháp đơn giản, dễ học:** Python dễ tiếp cận đối với những người mới bắt đầu học lập trình, giúp họ nhanh chóng làm quen và phát triển các ứng dụng mà không cần phải học

cú pháp phức tạp.

- **Tính linh hoạt và đa năng:** Python có thể được sử dụng cho nhiều mục đích khác nhau, bao gồm phát triển phần mềm, tạo các ứng dụng web, phân tích dữ liệu, khoa học dữ liệu, học máy, trí tuệ nhân tạo, và tự động hóa.
- **Cộng đồng phát triển mạnh mẽ:** Python có một cộng đồng người dùng và lập trình viên đông đảo, giúp người dùng dễ dàng tìm kiếm sự hỗ trợ và học hỏi từ những người có kinh nghiệm.
- **Hỗ trợ thư viện phong phú:** Python có một kho thư viện đồ sộ, từ những thư viện hỗ trợ phát triển web như Django, Flask, đến các thư viện khoa học dữ liệu như NumPy, Pandas, Matplotlib.
- **Khả năng tương thích và tích hợp:** Python có khả năng tích hợp tốt với các ngôn ngữ lập trình khác và hỗ trợ nhiều nền tảng khác nhau. Bạn có thể sử dụng Python để giao tiếp với các hệ thống khác như C, C++, Java hoặc thậm chí tích hợp với các dịch vụ web.

Ứng dụng của Python rất rộng và bao gồm:

- **Phát triển ứng dụng web:** Với các framework như Django, Flask, Pyramid.
- **Khoa học dữ liệu và phân tích dữ liệu:** Python rất phổ biến trong việc xử lý, phân tích và trực quan hóa dữ liệu, với các thư viện như Pandas, Matplotlib, NumPy.
- **Học máy (Machine Learning) và Trí tuệ nhân tạo (AI):** Các thư viện như TensorFlow, Keras, và Scikit-learn giúp phát triển các mô hình học máy.
- **Tự động hóa:** Python có thể sử dụng để tự động hóa các tác vụ đơn giản hoặc phức tạp trên máy tính, chẳng hạn như xử lý tệp, gửi email tự động, v.v.
- **Phát triển phần mềm:** Python cũng có thể được dùng để phát triển các ứng dụng phần mềm với giao diện người dùng (GUI) bằng các thư viện như Tkinter hoặc PyQt.

1.5. Cài đặt Python trên các hệ điều hành

Để bắt đầu lập trình Python, bạn cần cài đặt Python trên máy tính của mình. Quá trình cài đặt khác nhau tùy vào hệ điều hành mà bạn đang sử dụng.

- **Cài đặt trên Windows:**
    1. Truy cập trang web chính thức của Python tại python.org.
    2. Tải phiên bản Python mới nhất dành cho Windows.

3. Chạy tệp cài đặt và chọn tùy chọn “Add Python to PATH” trước khi nhấn nút “Install Now”.
4. Sau khi cài đặt, mở Command Prompt và gõ `python` để kiểm tra xem Python đã được cài đặt thành công hay chưa.

- **Cài đặt trên macOS:**
  1. Python thường đã được cài sẵn trên macOS, nhưng bạn có thể tải bản mới nhất từ python.org.
  2. Chạy tệp .pkg để cài đặt Python.
  3. Kiểm tra cài đặt bằng cách mở Terminal và gõ `python3 --version`.
- **Cài đặt trên Linux:**
  1. Mở terminal và sử dụng trình quản lý gói của hệ điều hành để cài đặt Python. Ví dụ, trên Ubuntu, bạn có thể dùng lệnh:

     ```
     sudo apt update
     sudo apt install python3
     ```

  2. Kiểm tra cài đặt bằng lệnh:

     ```
     python3 --version
     ```

Hoặc có thể cài đặt đặt Python qua Anaconda như sau:

Để cài đặt Python thông qua **Anaconda**, bạn có thể làm theo các bước sau. Anaconda là một môi trường quản lý các thư viện và môi trường Python, giúp bạn quản lý các gói thư viện và môi trường ảo rất tiện lợi, đặc biệt trong các dự án khoa học dữ liệu, học máy, hoặc phân tích dữ liệu.

- **Cài đặt Anaconda trên Windows:**

1. Truy cập trang web Anaconda: Truy cập trang web chính thức của Anaconda tại https://www.anaconda.com.
2. Tải Anaconda: Chọn phiên bản Anaconda phù hợp với hệ điều hành của bạn (Windows), sau đó tải tệp cài đặt **Anaconda Distribution**.
3. Chạy tệp cài đặt:
   - Mở tệp tải về và chạy chương trình cài đặt.
   - Trong quá trình cài đặt, bạn có thể chọn các tùy chọn sau:

- **Add Anaconda to my PATH environment variable** (để dễ dàng sử dụng Anaconda từ Command Prompt).
- **Install for me only** (cài đặt cho người dùng hiện tại).
- Bạn cũng có thể chọn **Install Microsoft VSCode** nếu cần.

4. Hoàn tất cài đặt: Sau khi cài đặt hoàn tất, mở **Anaconda Navigator** (giao diện đồ họa) hoặc sử dụng **Anaconda Prompt** (dòng lệnh) để bắt đầu làm việc với Python.
5. Kiểm tra cài đặt: Mở **Command Prompt** và gõ lệnh sau để kiểm tra xem Anaconda và Python đã được cài đặt thành công:

```
conda --version
python --version
```

- **Cài đặt Anaconda trên macOS:**

1. Truy cập trang web Anaconda: Truy cập trang web chính thức của Anaconda tại https://www.anaconda.com.
2. Tải Anaconda: Chọn phiên bản Anaconda phù hợp với macOS, sau đó tải tệp cài đặt **Anaconda Distribution**.
3. Chạy tệp cài đặt: Mở tệp tải về (.pkg) và làm theo các bước hướng dẫn cài đặt để cài đặt Anaconda trên hệ điều hành macOS.
4. Hoàn tất cài đặt: Sau khi cài đặt xong, bạn có thể mở **Anaconda Navigator** hoặc sử dụng **Terminal** với các lệnh conda.
5. Kiểm tra cài đặt: Mở **Terminal** và gõ lệnh sau để kiểm tra:

```
conda --version
python --version
```

- **Cài đặt Anaconda trên Linux** (ví dụ với Ubuntu):

1. Truy cập trang web Anaconda: Truy cập trang web chính thức của Anaconda tại https://www.anaconda.com.
2. Tải Anaconda: Chọn phiên bản phù hợp cho Linux (tệp .sh), sau đó tải về.
3. Cài đặt Anaconda:
   - Mở **Terminal** và di chuyển đến thư mục chứa tệp cài đặt `.sh`.
   - Chạy lệnh sau để cài đặt:

```
bash Anaconda3-<version>-Linux-x86_64.sh
```

   - Làm theo các hướng dẫn cài đặt trong terminal.
4. Hoàn tất cài đặt: Sau khi cài đặt hoàn tất, bạn có thể sử dụng **Anaconda Navigator** hoặc sử dụng **Terminal** để làm việc với môi trường Python.
5. Kiểm tra cài đặt: Mở **Terminal** và kiểm tra cài đặt bằng lệnh:

```
conda --version
python --version
```

Lưu ý thêm:

- **Anaconda Navigator** là một ứng dụng GUI (giao diện đồ họa người dùng) giúp bạn dễ dàng quản lý môi trường, cài đặt các gói và thư viện.
- Bạn cũng có thể sử dụng **Anaconda Prompt** (hoặc Terminal trên macOS/Linux) để chạy các lệnh conda và quản lý các môi trường Python của mình.
- Khi sử dụng **Anaconda**, bạn có thể tạo các **môi trường ảo** riêng biệt cho từng dự án để tránh xung đột giữa các gói thư viện. Ví dụ:

```
conda create --name myenv python=3.9
conda activate myenv
```

1.6. Cài đặt và sử dụng môi trường phát triển (IDE)

Một môi trường phát triển tích hợp (IDE) là công cụ rất hữu ích giúp lập trình viên viết mã Python một cách dễ dàng và hiệu quả. Dưới đây là một số IDE phổ biến:

- **PyCharm:** PyCharm là một IDE mạnh mẽ được phát triển bởi JetBrains, rất phổ biến trong cộng đồng lập trình Python. Nó hỗ trợ các tính năng như gỡ lỗi, tự động hoàn thành mã, kiểm tra mã, và tích hợp với Git.
   - Cài đặt: Truy cập PyCharm để tải và cài đặt.
- **Visual Studio Code (VS Code):** VS Code là một trình soạn thảo mã nguồn miễn phí và phổ biến, hỗ trợ Python thông qua các tiện ích mở rộng. Đây là một lựa chọn nhẹ nhàng nhưng mạnh mẽ, phù hợp với lập trình viên Python.

    - Cài đặt: Truy cập Visual Studio Code và cài đặt Python extension.
- **Jupyter Notebook:** Jupyter Notebook là một công cụ lý tưởng cho việc phân tích dữ liệu và làm việc với mã Python trong các bài toán khoa học dữ liệu. Nó cho phép bạn viết mã và chạy thử ngay trong trình duyệt, dễ dàng kết hợp giữa mã nguồn và văn bản.
    - Cài đặt: Cài đặt thông qua Anaconda hoặc sử dụng pip:

```
pip install notebook
```

- **IDLE (Integrated Development and Learning Environment):** Đây là IDE cơ bản đi kèm với Python. IDLE nhẹ và dễ sử dụng, phù hợp cho những người mới bắt đầu học Python.
    - Cài đặt: IDLE đã được cài đặt sẵn cùng với Python.

Sau khi cài đặt một IDE, bạn có thể bắt đầu viết và chạy các chương trình Python đầu tiên của mình trong môi trường này. Mỗi IDE sẽ có những tính năng hỗ trợ khác nhau, do đó bạn có thể lựa chọn IDE phù hợp với nhu cầu và sở thích cá nhân.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Qua quá trình tìm hiểu Jupyter Book và biên soạn tài liệu hỗ trợ lập trình Python cơ bản, tôi đã đạt được một số kết quả quan trọng. Cụ thể, việc ứng dụng Jupyter Book giúp tạo ra các tài liệu học tập dễ chỉnh sửa, chia sẻ và tích hợp mã nguồn trực tiếp vào nội dung, mang đến một phương pháp học trực quan và hiệu quả. Tài liệu hỗ trợ lập trình Python cơ bản đã hệ thống hóa các kiến thức một cách rõ ràng, dễ hiểu, phù hợp cho người mới bắt đầu học lập trình.

Đóng góp của nghiên cứu này là cung cấp một công cụ tiện lợi cho việc biên soạn tài liệu học lập trình Python, giúp nâng cao hiệu quả học tập và tiếp cận kiến thức lập trình. Tài liệu này cũng dễ dàng mở rộng và có thể bổ sung thêm các ví dụ, bài tập thực hành cụ thể để đáp ứng nhu cầu học tập đa dạng của người học.

Về hướng phát triển tiếp theo, có thể nghiên cứu và triển khai thêm các tính năng nâng cao trong Jupyter Book, như tích hợp các thư viện Python phức tạp hơn hoặc xây dựng các tài liệu hướng dẫn cho các chủ đề lập trình Python nâng cao. Bên cạnh đó, kết hợp Jupyter Book với các nền tảng học trực tuyến và xây dựng cộng đồng học viên có thể là một hướng nghiên cứu đầy tiềm năng trong tương lai.

DANH MỤC TÀI LIỆU THAM KHẢO

[1] W. McKinney, Python for Data Analysis, O'Reilly Media, 2017.

[2] V. S. Code, "Visual Studio Code," n.d. [Online]. Available: https://code.visualstudio.com. [Accessed 06 01 2025].

[3] P. S. Foundation, "Python Documentation," n.d. [Online]. Available: https://docs.python.org. [Accessed 06 01 2025].

[4] GitHub, "Mastering Markdown," n.d. [Online]. Available: https://guides.github.com/features/mastering-markdown/. [Accessed 06 01 2025].

[5] M. Guide, "Markdown Guide," n.d. [Online]. Available: https://www.markdownguide.org/. [Accessed 06 01 2025].

[6] OpenAI, "ChatGPT," 2022. [Online]. Available: https://chatgpt.com/. [Accessed 06 01 2025].

[7] E. B. Project, "Jupyter Book: Documentation," n.d. [Online]. Available: https://jupyterbook.org/. [Accessed 06 01 2025].

[8] J. Project, "Jupyter: The Project," n.d. [Online]. Available: https://jupyter.org/. [Accessed 06 01 2025].

[9] TutorialsPoint, "Python Tutorial," n.d. [Online]. Available: https://www.tutorialspoint.com/python/index.htm. [Accessed 06 01 2025].

[10] TutorialsPoint, "YAML Tutorial," n.d. [Online]. Available: https://www.tutorialspoint.com/yaml/index.htm. [Accessed 06 01 2025].

[11] YAML, "YAML Ain't Markup Language," n.d. [Online]. Available: https://yaml.org/. [Accessed 08 01 2025].