



MINISTRY OF EDUCATION AND TRAINING



VNU HCM CITY UNIVERSITY OF TECHNOLOGY

Capstone Project: **Develop a Web application to detect pests on coffee leaves**

College: VNU - Ho Chi Minh City University
 of Technology
Discipline: Computer Science

Student ID: 1952998

Student Name: Trần Đức Thiện

Advisor 1: Ph.D Nguyễn Tiến Thịnh

Ho Chi Minh City, August 2023

ACKNOWLEDGEMENT

Firstly, I would like to thank my teaching advisor, Mr Thinh, for his invaluable advice and guidance on this article. Without his support and instructions, i couldn't have completed this paper on time and sufficient enough, he has been been very generous and patiently guide me through challenges despite my stubborn and lack of communication skill.

Secondly, I also want to extend my thanks to my family and friends, especially Phong, who have always supported and encouraged me throughout this journey. They have been a constant source of inspiration, motivation and strength, and I am grateful for their presence in my life.

Additionally, I want to thank my reviewers, who have listened, and given me some hard, cold facts, and point out my logic flaws and mistakes in this paper through their entire gasp of knowledge. Thanks to your support, I have been able to fill some hole in my perspective, and improve the quality of my work.

Lastly, I express my gratitude to all the authors, readers, and reviewers, as your input and evaluation have been crucial to me. They have helped me identify the shortcomings in my writing and provided insights into the direction of my research. Thank you all!

Once again, I want to send my thanks and appreciation to all those who have supported and encouraged me.

ABSTRACT

Something something abstract

KEYWORDS: Capstone Project; HCMUT; Coffee leaf disease; Object Detection; Classification; Deep learning; Computer Vision; Web application

ABBREVIATIONS

Chữ viết tắt	Tiếng Anh
DNA	Deoxy Nucleotide Acid
PCR	Polymerase Chain Reaction
AI	Artificial Intelligence
SNP	Single nucleotide polymorphism
ML	Machine Learning

List of Figures

Figure 1: Typst logo	1
Figure 2: “Typst Logo”	3
Figure 3: “Crate Logo”	3
Figure 4: “Typst Long Logo”	4
Figure 5: “Typst Logo”	8
Figure 1: “XYZZZZ”	14

List of Tables

Table 1: Bảng mẫu phô mai	1
Table 2: Bảng mẫu phô mai	1
Table 3: ZJGSU ACMer	1
Table 4: Collection data used in this study	2
Table 5: Caption	2
Table 1: ZJGSU ACMer	14
Table 2: Bảng mẫu phô mai	14

List of Equations

Equation (0.1)	1
Equation (0.1)	1
Equation (0.1)	1
Equation (0.1)	1
Equation (0.2)	1
Equation (0.2)	1
Equation (0.3)	1
Equation (0.3)	1
Equation (0.3)	1
Equation (0.3)	1
Equation (0.4)	1
Equation (0.4)	1
Equation (0.1)	15
Equation (0.1)	15
Equation (0.1)	15
Equation (0.1)	15
Equation (0.2)	15
Equation (0.2)	15
Equation (0)	16

Table of Contents

ABSTRACT	I
ABBREVIATIONS	II
LIST OF FIGURES	III
LIST OF TABLES	IV
LIST OF FORMULAS	V
LIST OF EQUATIONS	VI
TABLE OF CONTENTS	VII
ACKNOWLEDGEMENT	1
List of Figures	4
List of Tables	5
Chapter I. Introduction	1
1.1 Introduction on Leaf disease detection	2
1.2 Ngôn ngữ Rust	4
1.2.1 Sinh viên	5
1.2.2 Các công ty	5
1.2.3 Các nhà phát triển mã nguồn mở	5
1.2.4 Những người quan tâm đến tốc độ và ổn định	5
1.2.5 Cuốn sách này dành cho ai	5
1.2.6 Sử dụng quyền sách này như thế nào	6
1.2.7 Tiêu đề	12
Chapter II. giới thiệu mẫu	13
2.1 Tổng quan về mẫu	13
2.2 trích dẫn	13
Chapter III. Mẫu biểu đồ	14
3.1 mẫu biểu đồ	14
3.2 Mẫu ví dụ	14
Chapter IV. mẫu công thức	16
4.1 công thức nội tuyến	16
4.2 công thức độc lập	16
Chapter V. danh sách ví dụ	17
5.1 danh sách không có thứ tự	17
5.2 danh sách sắp xếp	17
Chapter VI. Đây là phần giữ chỗ của chương	18

6.1 trình giữ chỗ cho tiêu đề cấp hai 1	18
6.2 Giữ chỗ tiêu đề phụ 2	18
6.3 tiêu đề phụ giữ chỗ 3	18
6.4 tiêu đề phụ giữ chỗ 4	18
6.4.1 giữ chỗ tiêu đề cấp 3 1	18
6.4.2 giữ chỗ tiêu đề cấp 3 2	18
6.4.2.1 trình giữ chỗ cấp 4 tiêu đề 1	18
6.4.2.2 trình giữ chỗ cấp 4 tiêu đề 2	18
6.5 tiêu đề phụ giữ chỗ 5	18
6.6 tiêu đề phụ giữ chỗ 6	18
References	19
Appendix 7. How to train large language models	20
Appendix 8. How to train very huge image-language datasets	20

Chapter I. Introduction

Table 1.1 Bảng mẫu phô mai


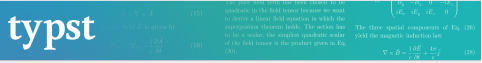
	Area	Parameters
	$\pi h \frac{D^2 - d^2}{4} (1.1)$	<p>h: height D: outer radius d: inner radius</p>
	$\frac{\sqrt{2}}{12} a^3 (1.2)$	<p>a: kích thước chiều dài</p>



Figure 1.1 Typst logo

Figure 1.1 is the official logo for Typst.

Table 1.2 Bảng mẫu phô mai



	Area	Parameters
	$\pi h \frac{D^2 - d^2}{4} (1.3)$	<p>h: height D: outer radius d: inner radius</p>
	$\frac{\sqrt{2}}{12} a^3 (1.4)$	<p>a: kích thước chiều dài</p>

Table 1.3 ZJGSU ACMer

gọi như thế nào	bộ phận bạn đang ở	một câu để giới thiệu	một liên kết
Mauve	Jike 2018	Người Ali	https://hukeqing.github.io
jujimeizuo	Jike 2019	Gà rau	http://www.jujimeizuo.cn
kaka	Jike 2019	Nghiên cứu Hangdian	https://ricar0.github.io
lx_tyin	2020 Ji Ke	Cao thủ huy chương vàng	lxtyin.ac.cn

Table 1.4 Collection data used in this study

Type	Original Task	Categorys			Total
		Benign	Malignant	Normal	
Lin	Object detection	9,932	15,475	0	25,407
Al-Dhabyani	Segmentation	437	210	133	780
Rodrigues	Not applicable	100	150	0	250
Total	—	10,469	15835	133	26,437

```

1 printf("Hello world!\n");
2
3 // Comment
4 for (int i = 0; i < m; i++) {
5     for (int j = 0; j < n; j++) {
6         sum += 'a';
7     }
8 }

```

```

1 def sqrt(x):
2     return x ** 0.5

```

Table 1.5 Caption

Header 1	Header 2	Header 3
Row 1	Row 1	Row 1
Row 2	Row 2	Row 2

1.1 Introduction on Leaf disease detection

1. Typst is the latest and hottest markup text language. Its positioning is similar to LaTeX. It has strong typesetting capabilities, writes documents through certain grammar, and then generates pdf files. Compared with LaTeX, it has the following advantages:
2. Compilation is extremely fast: because it provides the function of incremental compilation, it can basically compile a pdf file within one second after modification. typst provides the

function of monitoring modification and automatic compilation, and can read the effect while writing like Markdown.

3. The environment is simple to build: natively supports non-Latin languages such as China, Japan, and Korea, so there is no need to worry about character compatibility issues and download several G environments. Just download the command line program to start compiling and generating pdf.
4. Grammatically friendly: For ordinary typesetting needs, the difficulty of getting started is comparable to Markdown, and the text source code is highly readable: no more backslashes and curly braces

Personal opinion: as easy to use as Markdown, as powerful as LaTeX



Figure 1.2 “Typst Logo”




Ferris	Meaning
	Code này không dịch được!
	Code này trông phát ớn!
	Code này sẽ không tạo ra kết quả mong muốn.

Figure 1.3 “Crate Logo”

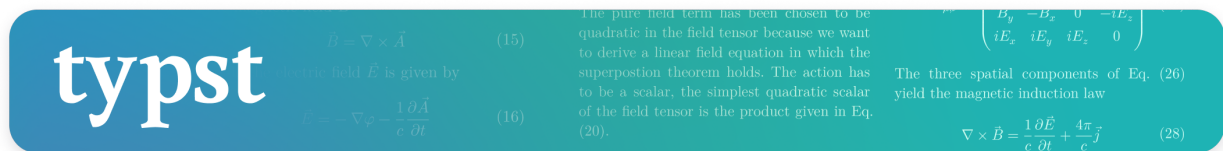


Figure 1.4 “Typst Long Logo”

1.2 Ngôn ngữ Rust

Giới thiệu *Ghi chú: Phiên bản này cũng chính là phiên bản in The Rust Programming Language và ebook No Starch Press của sách.*

- Chào mừng bạn đến với Ngôn ngữ lập trình Rust, một cuốn sách giới thiệu về Rust. “Ngôn ngữ lập trình Rust” sẽ giúp bạn viết các phần mềm nhanh và tin cậy hơn. Việc thiết kế ngôn ngữ lập trình luôn phải giải quyết bài toán xung đột giữa kiểm soát ở cấp thấp và việc hỗ trợ con người ở bậc cao; Rust thách thức sự xung đột này. Thông qua khả năng cân bằng giữa khả năng tiếp thu công nghệ mạnh mẽ và kinh nghiệm phát triển tuyệt vời, Rust cung cấp cho bạn khả năng kiểm soát các chi tiết ở cấp độ thấp (chẳng hạn việc sử dụng bộ nhớ) mà vẫn tránh được những phiền toái vốn hay gặp phải khi phải làm việc ở cấp độ này.

Rust được dành cho ai Rust khá lý tưởng cho nhiều người vì nhiều lý do khác nhau. Hãy cũng xem qua một vài trong số những nhóm lý do quan trọng nhất.

Các nhóm phát triển

- Rust đang dần chứng minh như một công cụ hữu hiệu cho việc cộng tác giữa những nhóm lớn nhà phát triển với các mức độ kiến thức về lập trình hệ thống khác nhau. Mã lệnh cấp thấp thường dính phải các loại lỗi ẩn, vốn chỉ có thể tìm thấy thông qua việc kiểm thử hoặc review cẩn thận bởi các nhà phát triển nhiều kinh nghiệm. Trong Rust, trình biên dịch đóng vai trò như người gác cổng bằng cách từ chối các loại lỗi như vậy, bao gồm cả các lỗi liên quan đến việc xử lý đồng thời. Bằng cách kết hợp với trình dịch, nhóm phát triển có thể dành thời gian tập trung cho logic chương trình hơn là tìm kiếm các lỗi.

1. Rust cũng đồng thời cung cấp các công cụ hỗ trợ phát triển đến cho thế giới lập trình hệ thống:

Cargo, công cụ quản lý thư viện và build, cho phép thêm, dịch, quản lý các thư viện phụ thuộc dễ dàng và đồng bộ xuyên suốt hệ sinh thái Rust. Công cụ định dạng code Rustfmt đảm bảo sự đồng nhất về phong cách viết code giữa các nhà phát triển. The Rust Language Server cung cấp sức mạnh cho các Integrated Development Environment (IDE) để hỗ trợ các tính năng như code completion và các thông báo lỗi tại chỗ. Bằng cách sử dụng các công cụ ở trên cũng như một số công cụ khác trong hệ sinh thái Rust, các nhà phát triển có thể viết một cách hiệu quả các mã lệnh cấp hệ thống.

1.2.1 Sinh viên

Rust được dành cho sinh viên và bất kỳ ai yêu thích việc học các khái niệm hệ thống. Nhiều người thông qua sử dụng Rust đã học về những chủ đề như phát triển hệ điều hành. Cộng đồng Rust cũng rất sẵn sàng chào đón và hỗ trợ trả các câu hỏi của sinh viên. Thông qua những nỗ lực tương tự như cuốn sách này, các nhóm Rust muốn làm cho việc hiểu các khái niệm về hệ thống dễ dàng hơn với nhiều người, đặc biệt những người mới làm quen với lập trình.

1.2.2 Các công ty

Hàng trăm công ty, cả lớn và nhỏ, sử dụng Rust cho nhiều nhiệm vụ khác nhau trong hoạt động của họ. Những nhiệm vụ đó bao gồm các công cụ dòng lệnh, dịch vụ web, các công cụ DevOps, các thiết bị nhúng, các trình phân tích và mã hóa âm thanh hình ảnh, tiền mã hóa, tin sinh học, các máy tìm kiếm, các ứng dụng IoT, học máy, và thậm chí các phần chính của trình duyệt web FireFox.

1.2.3 Các nhà phát triển mã nguồn mở

Rust dành cho những người tạo nên ngôn ngữ, cộng đồng, công cụ phát triển và các thư viện. Chúng tôi sẽ rất vui khi được bạn góp phần xây dựng ngôn ngữ Rust.

1.2.4 Những người quan tâm đến tốc độ và ổn định

Rust được dành cho những người đam mê tốc độ và sự ổn định trong một ngôn ngữ. Với tốc độ, chúng tôi nói về cả tốc độ thực thi chương trình và cả tốc độ mà Rust cho phép bạn viết ra chúng. Các phép kiểm tra của trình dịch Rust đảm bảo sự ổn định thông qua các đặc tính thêm vào và việc tái cấu trúc (refactoring). Điều này ngược lại với các mã lệnh dễ-mắc-lỗi khi viết bằng các ngôn ngữ không có các phép tra đó, vốn làm cho các nhà phát triển ngại việc chỉnh sửa. Bằng việc đánh vào sự trừu tượng, hoặc các đặc tính ở cấp độ cao không gây phát sinh ra thêm chi phí khi dịch ra mã lệnh cấp thấp, Rust cho phép các mã lệnh này thực thi nhanh và an toàn như khi viết bằng tay.

Ngôn ngữ Rust cũng hy vọng hỗ trợ thêm nhiều người dùng khác; những người được nhắc đến ở đây chỉ đơn thuần là một trong số những người tham gia nhiều nhất. Tổng thể lại, tham vọng lớn nhất của Rust là loại bỏ những sự đánh đổi mà lập trình viên phải chấp nhận trong hàng thập kỷ qua, để cung cấp sự an toàn và năng suất, tốc độ và sự hỗ trợ bậc cao của ngôn ngữ. Hãy thử và xem Rust liệu có thể trở thành lựa chọn cho công việc của bạn.

1.2.5 Cuốn sách này dành cho ai

Cuốn sách này cho rằng bạn đã viết code bằng một ngôn ngữ khác nhưng không chỉ rõ là ngôn ngữ nào. Chúng tôi đã cố gắng tạo ra các tài liệu có thể dùng được bởi nhiều người với

nhiều nền tảng lập trình khác nhau. Chúng tôi không dành nhiều thời gian để nói về những thứ như lập trình là gì và bạn nghĩ về nó như thế nào. Nếu bạn là người hoàn toàn mới với việc lập trình, có lẽ tốt hơn là bạn nên đọc trước một quyển sách chuyên về nhập môn lập trình.

1.2.6 Sử dụng quyển sách này như thế nào

Về tổng thể, quyển sách này cho là bạn đang đọc tuần tự từ đầu đến cuối. Các chương sau được xây dựng dựa trên các khái niệm được giới thiệu trong các chương trước, và các chương đầu có lẽ sẽ không đi vào chi tiết về một chủ đề cụ thể, thay vì vậy chúng ta sẽ quay lại chủ đề đó trong các chương sau.

- Bạn sẽ tìm thấy hai loại chương trong cuốn sách này: các chương khái niệm và các chương dự án. Trong các chương khái niệm, bạn sẽ học về một khía cạnh nào đó của Rust. Trong các chương dự án, chúng ta sẽ cùng với nhau xây dựng các chương trình nhỏ, áp dụng những gì các bạn đã học. Các chương 2, 12 và 20 là các chương dự án; còn lại là các chương khái niệm.
1. Chương 1 hướng dẫn cách cài đặt Rust, làm sao để viết một chương trình “Hello, world!”, và làm sao sử dụng Cargo, công cụ quản lý các gói thư viện và build chương trình. Chương 2 là một phần giới thiệu kiểu “trên tay” về cách viết chương trình trong ngôn ngữ Rust, bạn cũng sẽ xây dựng một trò chơi đoán số. Chúng ta sẽ xem sơ các khái niệm ở cấp cao, rồi các chương sau đó sẽ cung cấp thêm các chi tiết. Nếu bạn muốn vọc vạch ngay, chương này là dành cho bạn. Đầu tiên, có lẽ bạn sẽ muốn bỏ qua chương 3, vốn giới thiệu các đặc tính của Rust tương tự trong các ngôn ngữ khác, và nhảy ngay đến chương 4 để học về hệ thống ownership của Rust. Tuy nhiên, nếu bạn là một người học cẩn trọng muốn học tất cả các chi tiết trước khi chuyển đến phần kế tiếp, có lẽ bạn sẽ bỏ qua chương 2 và đi thẳng đến chương 3, trở về lại chương 2 khi bạn muốn áp dụng những chi tiết mà bạn đã học.
 2. Chương 5 thảo luận về struct và method, chương 6 giới thiệu về enum, các biểu thức match, và cấu trúc điều khiển if let. Bạn sẽ dùng struct và enum để tạo ra các kiểu tùy biến trong Rust.
 3. Trong chương 7, bạn sẽ học về hệ thống module của Rust và về các quy tắc riêng tư khi tổ chức mã nguồn và hệ thống Application Programming Interface (API) của nó. Chương 8 thảo luận về một số kiểu tập hợp (collection) mà các thư viện chuẩn cung cấp, như vector, string và hash map. Chương 9 khám phá các triết lý cũng như kỹ thuật của Rust trong việc xử lý lỗi.
 4. Chương 10 đào sâu và generic, trait và vòng đời, thứ mang lại sức mạnh để tạo ra các mã lệnh có thể dùng được với nhiều kiểu dữ liệu khác nhau. Chương 11 nói hoàn toàn về kiểm thử, thứ cần có để đảm bảo logic chương trình của bạn là chính xác, ngay cả khi đã có hệ thống an toàn của Rust. Trong chương 12, chúng ta sẽ xây dựng một tập con các tính năng từ câu

lệnh grep, cho phép tìm kiếm các đoạn văn bản bên trong các file. Để làm điều này, chúng ta sẽ dùng nhiều các khái niệm đã thảo luận trong các chương trước.

5. Chương 13 khám phá closure và iterator: các tính năng của Rust vốn đến từ các ngôn ngữ lập trình hàm (functional programming). Trong chương 14, chúng ta sẽ khảo sát lại Cargo sâu hơn và nói về các phương pháp hay nhất để chia sẻ thư viện của bạn với những người khác. Chương 15 thảo luận về các con trỏ thông minh mà các thư viện chuẩn cung cấp, đồng thời nói về các trait cho phép chúng hoạt động.
6. Trong chương 16, chúng ta sẽ dạo qua các mô hình khác nhau của lập trình song song và nói về cách Rust hỗ trợ bạn viết đa luồng một cách dễ dàng. Chương 17 so sánh một số thành phần trong Rust với các khái niệm hướng đối tượng mà có lẽ bạn đã quen thuộc.
7. Chương 18 là phần tham khảo về mẫu và khớp mẫu, vốn là những cách thức mạnh mẽ để biểu đạt các ý tưởng trong suốt các chương trình Rust. Chương 19 là một bữa đại tiệc với nhiều chủ đề nâng cao khác nhau, bao gồm unsafe Rust, macro, và nói thêm về vòng đời, trait, type, function và closure.
8. Trong chương 20, chúng ta sẽ hoàn thành một máy chủ web đa luồng cấp thấp.
 - Cuối cùng, các phụ lục sẽ chứa nhiều thông tin hữu ích về ngôn ngữ theo dạng liệt kê dễ dàng để tham khảo. Phụ lục A chứa các từ khóa của Rust, phụ lục B chứa các toán tử và ký hiệu, phụ lục C chứa các trait có thể dẫn xuất lại được cung cấp bởi thư viện chuẩn, phụ lục D nói về các công cụ phát triển hữu ích, và phụ lục E nói về các phiên bản của Rust.
 - Sẽ không có một cách sai để đọc quyển sách này: nếu bạn muốn nhảy qua một vài phần, cứ việc! Bạn cũng có thể nhảy ngược lại các chương trước khi gặp phải điều gì khó hiểu. Cứ đơn giản làm cái gì bạn cảm thấy hiệu quả.
 - Một phần quan trọng khi học Rust là học cách đọc các thông báo lỗi mà trình biên dịch hiển thị: Chúng sẽ giúp bạn tạo ra các code làm việc được. Do đó, chúng tôi sẽ cung cấp nhiều ví dụ không thể dịch được cùng với thông báo lỗi trình dịch sẽ hiển thị trong trường hợp tương ứng. Hãy nhớ nếu bạn nhập và chạy một ví dụ ngẫu nhiên, nó có thể bị lỗi! Hãy đảm bảo bạn đã đọc những nội dung xung quanh để xem liệu code bạn đang thử chạy có tạo ra lỗi hay không. Ferris cũng sẽ giúp bạn phân biệt những code nào sẽ không chạy:



Figure 1.5 “Typst Logo”

Nguồn tham khảo tại đây <https://daohainam.github.io/rust-book/ch00-00-introduction.html> và tại đây.

Hiểu biết cá nhân: typst có hai môi trường, mã và nội dung, trong môi trường mã, nó sẽ được thực thi theo mã và trong môi trường nội dung, nó sẽ được phân tích thành văn bản thông thường, môi trường mã được biểu thị bằng `{}` và môi trường nội dung được đại diện bởi `[]`. Nội dung bắt đầu bằng `#` để kết nối với một đoạn mã, chẳng hạn như `#set` quy tắc, và `#` không cần thiết để gọi mã trong khối được đặt trong dấu ngoặc nhọn.

```
1 void setFib(void)
2 {
3     fib(1 | 2 | 3 | 5 | 8 | 13 | 21 | 34 | 55 | 89, 10);
4 }
```

```
1 def sqrt(x):
2     return x**0.5
```

```
1 #include <math.h>
2 #include <stdio.h>
3 #include <stdlib.h>
4
5 // Train Data
6 float td[][2] = {
7     // C    F
```

```
8      {-40, -40},
9      {-20, -4 },
10     {0, 32 },
11     {20, 68 },
12     {40, 104},
13     {60, 140},
14     {80, 176},
15     {100, 212},
16 };
17
18 #define N 8 // Number of item in Train Data
19
20
21 // Define the learning alpha
22 #define ALPHA 0.00001
23
24 // Define the number of iterations
25 #define EPOCHS 100 * 1000
```

```

1 // Define a function to compute the mean squared error
2 double cost(double w, double b) {
3     double error = 0.0;
4     for (int i = 0; i < N; ++i) {
5         double x = td[i][0];
6         double y = td[i][1];
7         double d = y - (w * x + b);
8         error += d * d;
9     }
10    return error / (int) N;
11 }
12
13 // Define a function to perform gradient descent
14 void gradient_descent(double *w, double *b) {
15     // Derivative of cost function with respect to w or b
16     double dw = 0.0;
17     double db = 0.0;
18     for (int i = 0; i < N; i++) {
19         double x = td[i][0];
20         double y0 = td[i][1];
21         double y = *w * x + *b;
22         dw += x * (y - y0);
23         db += (y - y0);
24     }
25     // Update w and b using the learning rate and the derivatives
26     *w = *w - ALPHA * dw;
27     *b = *b - ALPHA * db;
28 }
29
30 // Define a function to train the neuron using gradient descent
31 void train(double *w, double *b) {
32     for (int i = 0; i < EPOCHS; i++) {
33         gradient_descent(w, b);
34         if (i % 101000 == 0)
35             printf("Iteration: %d, Cost:%3.3f w=%.3lf b=%.3lf\n", i, cost(*w,
36 *b), *w, *b);
37     }
38 }
39
40 // Define a function to predict the output using the neuron

```

```

41 double predict(double x, double w, double b) {
42     return w * x + b;
43 }

1  int main() {
2      // Initialize w and b randomly
3      double w = (double) rand() / RAND_MAX;
4      double b = (double) rand() / RAND_MAX;
5
6      // Train the neuron using gradient descent
7      train(&w, &b);
8
9      // Print the final values of w and b
10     printf("\nFinal values are: w = %f and b = %f\n\n", w, b);
11
12     //Test the neuron with some new inputs
13     double x_new = 50;                // Celsius
14     double y_new = predict(x_new, w, b); // Fahrenheit
15     printf("Fahrenheit of 50C: 122F\n");
16     printf("Prediction of 50C: %.1fF\n", y_new);
17
18     return 0;
19 }

```

The output of running the script shows that the weight parameter converges to 2, which is the true slope of the data set. The bias parameter converges to 0, which is the true intercept of the data set. The cost function reaches its minimum value when the parameters are optimal. This means that our script successfully learns the linear relationship between x and y from the data set.

```

1  Iteration: 0, Cost:3519.467 w=1.146 b=0.399
2  Iteration: 10000, Cost:227.957 w=1.981 b=13.954
3  Iteration: 20000, Cost:74.384 w=1.903 b=21.692
4  Iteration: 30000, Cost:24.272 w=1.859 b=26.112
5  Iteration: 40000, Cost:7.920 w=1.834 b=28.636
6  Iteration: 50000, Cost:2.584 w=1.819 b=30.079
7  Iteration: 60000, Cost:0.843 w=1.811 b=30.902
8  Iteration: 70000, Cost:0.275 w=1.806 b=31.373
9  Iteration: 80000, Cost:0.090 w=1.804 b=31.642
10 Iteration: 90000, Cost:0.029 w=1.802 b=31.795

```

```
11
12 Final values are: w = 1.801169 and b = 31.883127
13
14 Fahrenheit of 50C: 122F
15 Prediction of 50C: 121.9F
```

1.2.7 Tiêu đề

Tương tự như markdown, # được dùng để chỉ tiêu đề, trong typst, = được dùng để chỉ tiêu đề, một = được dùng cho tiêu đề cấp một, hai = được dùng cho tiêu đề cấp hai, v.v.

Khoảng cách, phong chữ, v.v. đều do tôi sắp chữ. Nhưng chú ý cần thêm 12pt sau mỗi đoạn nhé! ! !

Chapter II. giới thiệu mẫu

2.1 Tổng quan về mẫu

Dự án này được viết lại bằng ngôn ngữ Typst để giúp sinh viên chưa tốt nghiệp của Đại học Công Thương Chiết Giang viết luận văn tốt nghiệp thuận tiện hơn. Mẫu này được tạo dựa trên hệ thống Typst, so với Latex [1], nó là một phần mềm sắp chữ có cú pháp đơn giản hơn và có thể được sử dụng để sản xuất các bài báo và ấn phẩm khoa học chất lượng cao. Dự án hiện bao gồm trang bìa, tóm tắt, văn bản, tài liệu tham khảo, v.v. của bài báo và người dùng có thể sửa đổi và tùy chỉnh nó theo nhu cầu của họ.

2.2 trích dẫn

Đây là một tham chiếu đến các tài liệu tham khảo trong mẫu Latex cộng đồng nguồn mở [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17], Bạn có thể bấm vào số thứ tự để chuyển đến cuối văn bản để xem định dạng trích dẫn.

Chapter III. Mẫu biểu đồ

3.1 mẫu biểu đồ

Như được hiển thị bởi **Figure 3.1** là một hình ảnh mẫu.

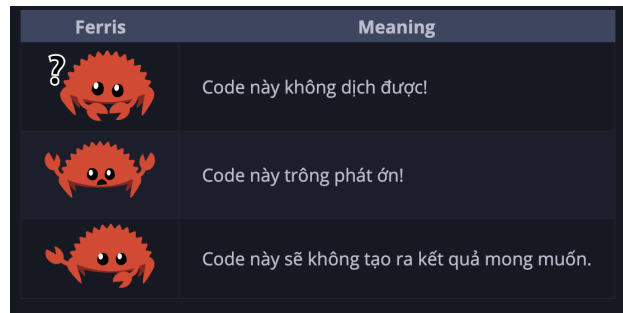


Figure 3.1 “XYZZZZ”

3.2 Mẫu ví dụ

Table 1.3 Hiện một số blogger.

Table 3.1 ZJGSU ACMer


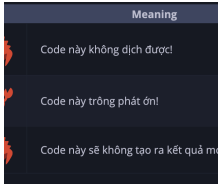
gọi như thế nào	bộ phận bạn đang ở	một câu để giới thiệu	một liên kết
Mauve	Jike 2018	Người Ali	https://hukeqing.github.io
jujimeizuo	Jike 2019	Gà rau	http://www.jujimeizuo.cn
kaka	Jike 2019	Nghiên cứu Hangdian	https://ricar0.github.io
lx_tyin	2020 Ji Ke	Cao thủ huy chương vàng	lxtyin.ac.cn

Biểu mẫu tương tự như hình, nhưng cách nhập biểu mẫu phức tạp hơn một chút, nên vào trang web chính thức của typst để tìm hiểu, mức độ tự do rất cao và khả năng tùy biến rất mạnh.

Hãy xem **Table 1.1** , những ô không có trường là tất cả nội dung của ô (mỗi ô được bao bọc bởi []), được sắp xếp theo chiều ngang khi căn chỉnh nằm ngang và bao bọc sau khi kết thúc.

Table 3.2 Bảng mẫu phô mai

	Area	Parameters
--	------	------------

	$\pi h \frac{D^2 - d^2}{4} \quad (3.1)$	<p>h: height D: outer radius d: inner radius</p>
	$\frac{\sqrt{2}}{12} a^3 \quad (3.2)$	<p>a: kích thước chiều dài</p>

Chapter IV. mẫu công thức

Công thức được bao bọc bởi hai \$, nhưng ngữ pháp không giống với LaTeX, nếu bạn có nhiều nhu cầu về công thức, trước tiên nên đọc hướng dẫn trên trang web chính thức, nhưng typst vẫn còn tương đối sớm và khả năng không thể loại trừ việc thêm ngữ pháp tương thích trong tương lai.

4.1 công thức nội tuyến

Công thức nội tuyến $a^2 + b^2 = c^2$ Công thức nội tuyến.

4.2 công thức độc lập

Chapter V. danh sách ví dụ

5.1 danh sách không có thứ tự

- danh sách không có thứ tự 1: 1
- Danh sách không có thứ tự 2: 2

5.2 danh sách sắp xếp

1. Danh sách thứ tự 1
2. Danh sách thứ tự 2

Nếu muốn tự xác định, bạn có thể tự đặt đánh số, nên bọc nó bằng #[] để đảm bảo rằng nó chỉ có hiệu lực trong phạm vi này:

- 1) danh sách tùy chỉnh 1
 - a) Danh sách tùy chỉnh 1.1
- 2) danh sách tùy chỉnh 2
 - a) Danh sách tùy chỉnh 2.1

Chapter VI. Đây là phần giữ chỗ của chương

6.1 trình giữ chỗ cho tiêu đề cấp hai 1

6.2 Giữ chỗ tiêu đề phụ 2

6.3 tiêu đề phụ giữ chỗ 3

6.4 tiêu đề phụ giữ chỗ 4

6.4.1 giữ chỗ tiêu đề cấp 3 1

6.4.2 giữ chỗ tiêu đề cấp 3 2

6.4.2.1 trình giữ chỗ cấp 4 tiêu đề 1

6.4.2.2 trình giữ chỗ cấp 4 tiêu đề 2

6.5 tiêu đề phụ giữ chỗ 5

6.6 tiêu đề phụ giữ chỗ 6

References

- [1] D. E. Knuth, *The Tex Book*, 15th, Reading, MA: Addison-Wesley Publishing Company, 1989.
- [2] V. Nikiforov, "Analytic methods for uniform hypergraphs," *Linear Algebra Its Appl.*, vol. 457, pp. 455–535, 2014.
- [3] 聂灵沼, and 丁石孙, 代数学引论, 北京: 高等教育出版社, 2000.
- [4] L. Lu, and S. Man, "Connected hypergraphs with small spectral radius," *Linear Algebra Its Appl.*, vol. 509, pp. 206–227, 2016, doi: 10.1016/j.laa.2016.07.013.
- [5] S. Hu, L. Qi, and J. Shao, "Cored hypergraphs, power hypergraphs and their Laplacian H-eigenvalues," *Linear Algebra Its Appl.*, vol. 439, pp. 2980–2998, 2013, doi: 10.1016/j.laa.2013.08.028.
- [6] H. Lin, and B. Zhou, "Distance spectral radius of uniform hypergraphs," *Linear Algebra Its Appl.*, vol. 506, pp. 564–578, 2016, doi: 10.1016/j.laa.2016.06.011.
- [7] L. Kang, and V. Nikiforov, "Extremal problems for the \mathbb{P} -spectral radius of graphs," *Electronic J. Combinatorics*, vol. 21, no. 3, 2014.
- [8] L. Qi, " H^+ -eigenvalues of Laplacian and signless Laplacian tensors," *Commun. Math. Sciences*, vol. 12, no. 6, pp. 1045–1064, 2014.
- [9] V. Nikiforov, "Hypergraphs and hypermatrices with symmetric spectrum," *Linear Algebra Its Appl.*, vol. 519, pp. 1–18, 2017, doi: 10.1016/j.laa.2016.12.038.
- [10] C. Bu, Y. Fan, and J. Zhou, "Laplacian and signless Laplacian Z-eigenvalues of uniform hypergraphs," *Frontiers Math. China*, vol. 11, no. 3, pp. 511–520, 2016.
- [11] R. Impagliazzo, and R. Paturi, "On the complexity of k-sat," *J. Comput. System Sciences*, vol. 62, no. 2, pp. 367–375, 2001.
- [12] R. Impagliazzo, R. Paturi, and F. Zane, "Which problems have strongly exponential complexity?," *J. Comput. System Sciences*, vol. 63, no. 4, pp. 512–530, 2001.
- [13] J. Elffers, and M. de Weerd, "Scheduling with two non-unit task lengths is np-complete," *Arxiv Preprint Arxiv:1412.3095*, 2014.
- [14] M. Chrobak, U. Feige, et al., "A greedy approximation algorithm for minimum-gap scheduling," *J. Scheduling*, vol. 20, no. 3, pp. 279–292, 2017.
- [15] R. Paturi, P. Pudlák, and F. Zane, "Satisfiability coding lemma," in *Proc. 38th Annu. Symp. Foundations Comput. Sci.*, 1997, pp. 566–574.
- [16] R. V. Book, "Michael r. garey and david s. johnson, computers and intractability: a guide to the theory of \mathcal{NP} -completeness," *Bull. (New Series) Amer. Math. Soc.*, vol. 3, no. 2, pp. 898–904, 1980.
- [17] C. H. Papadimitriou, and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Courier Corporation, 1998.

Appendix 7. How to train large language models

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aequale doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et.

Appendix 8. How to train very huge image-language datasets

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aequale doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et.