ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM

BÁO CÁO ĐỒ ÁN

ĐỀ TÀI

Phần mềm ứng dụng vẽ đồ thị

Môn học	**: Lập trình trực quan**
Giảng viên lý thuyết	**: Huỳnh Tuấn Anh**
Giảng viên lý thực hành	**: Quan Chí Khánh An**
Nhóm thực hiện	**: Trần Ngọc Phú -**
	Võ Đức Kha - 22520601

TP. Hồ Chí Minh, tháng 12 năm 2023

LỜI MỞ ĐẦU

Trong xã hội ngày nay, các ứng dụng tin học đang ngày càng phát triển và được ứng dụng trong rất nhiều lĩnh vực đời sống từ kinh tế đến khoa học xã hội...Và giáo dục cũng không là ngoại lệ, việc sử dụng các ứng dụng Công nghệ Thông tin là rất hợp lý và cần thiết, nhằm nâng cao chất lượng học tập của học sinh và sinh viên. Trên cơ sở đó, nhóm em tạo ra phần mềm này với hi vọng nó có thể giúp ích cho quá trình học tập của các bạn, đồng thời bọn em cũng có cơ hội ứng dụng các kiến thức học được ở môn Lập trình trực quan.

Đồ án Phần mềm Vẽ Đồ Thị là một ứng dụng được phát triển trên nền tảng Windows, sử dụng ngôn ngữ lập trình C# và framework WPF (Windows Presentation Foundation) để tạo giao diện người dùng đồ họa. Ứng dụng này cho phép người dùng vẽ và hiển thị đồ thị các hàm số toán học theo biểu đồ.

Trong thời gian nghiên cứu cũng như trong quá trình làm báo cáo đề tài, không tránh khỏi thiếu sót. Chúng em rất mong nhận được sự góp ý quý báu của Thầy.

Xin chân thành cảm ơn!!!

Nội dung

I. Tổng quan :

1. Lý do chọn đề tài :

- Giáo dục và Học tập: Đồ án này nhằm hỗ trợ việc học và thực hành Toán học một cách trực quan và thú vị.

- Ứng dụng Thực tế: Ứng dụng có thể hữu ích cho sinh viên, giáo viên hoặc những người đang cần biểu diễn đồ thị hàm số trong công việc hoặc học tập.

2. Mục đích nghiên cứu đề tài :

- Cho phép người dùng nhập và biểu diễn đồ thị các hàm số toán học đơn giản.

- Cung cấp giao diện thân thiện, dễ sử dụng để thêm, xóa và chỉnh sửa các hàm số.

- Hỗ trợ việc lưu trữ và nạp lại các biểu đồ đã tạo thông qua tệp JSON.

- Hiển thị các điểm giao nhau của các hàm số.

3. Công nghệ sử dụng :

- Ngôn ngữ lập trình: C#

- Framework: Windows Presentation Foundation (WPF).

- Thư viện sử dụng: OxyPlot, MaterialDesignThemes.Wpf, Newtonsoft.Json.

4. Cấu trúc mã nguồn :

- Mã nguồn chính của ứng dụng này được chia thành các phần chức năng chính sau:

+ MainWindow.xaml.cs: Đây là file chứa mã nguồn chính của giao diện và logic xử lý sự kiện của ứng dụng.

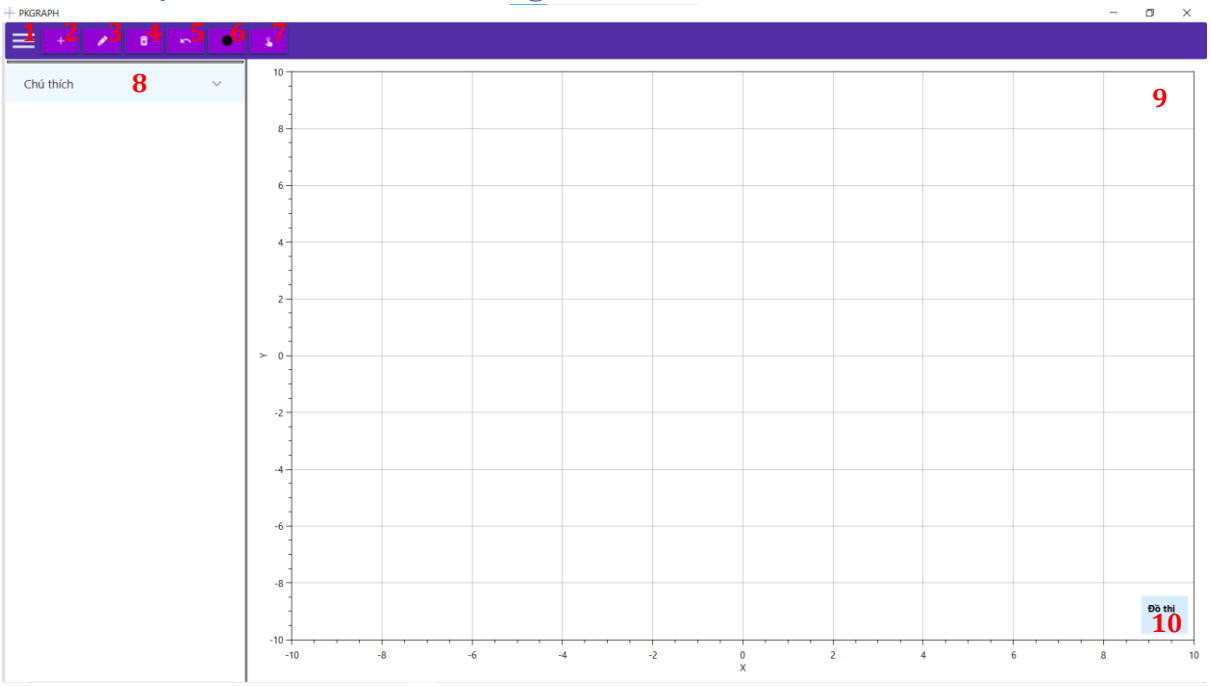
+ Models: Thư mục chứa các lớp mô hình dữ liệu cho đồ thị và hàm số.

+ ColorDialog.cs: Chứa mã nguồn xử lý hộp thoại chọn màu sắc.

+ PointWindow.xaml.cs: Xử lý cửa sổ hiển thị điểm giao nhau.

+ SavedGraphModel.cs: Lớp mô hình cho việc lưu trữ dữ liệu đồ thị đã lưu.

II. Giao diện màn hình và chức năng chính



STT	Tên gọi	Chức năng
1	Menu	Cung cấp các chức năng : File mới (New), mở file (Open), lưu file (Save), lưu file mới (Save as).
2	Thêm mới	Cung cấp thao tác nhập hàm số mới.
3	Vẽ	Thực hiện chức năng vẽ đồ thị hàm số đã được nhập trước đó.
4	Xóa	Xóa hàm số muốn xóa.
5	Undo	Hoàn tác lại hành động vừa xảy ra.
6	Color	Thay đổi màu sắc của đồ thị hàm số trước khi vẽ.
7	Tạo giao điểm	Tạo giao điểm của các đồ thị hàm số được chọn.
8	Chú thích	Hiển thị tên hàm, biểu thị toán học, chức năng.
9	Bảng vẽ	Hiển thị các đồ thị hàm số mà người dùng nhập vào.
10	Ô chú thích	Hiển thị chi tiết các đồ thị hàm số có trong bảng vẽ, có thể ẩn hiện các đồ thị mình vẽ.

III. Kỹ thuật xử lý và mô hình hóa :

1. Kỹ thuật xử lý :

1.1 Sử dụng Framework WPF và C# :

- Windows Presentation Foundation (WPF): Sử dụng WPF để tạo giao diện người dùng (GUI) thân thiện và linh hoạt.

- Ngôn ngữ lập trình C#: Sử dụng C# để viết logic xử lý sự kiện, xây dựng các lớp và thực hiện các chức năng của ứng dụng.

1.2 Sử dụng Thư viện OxyPlot :

- Vẽ Đồ thị Hàm Số: Sử dụng OxyPlot để tạo và hiển thị đồ thị hàm số trong giao diện người dùng.

- Cài Đặt Các Thuộc Tính Đồ Thị: Sử dụng OxyPlot để tùy chỉnh các thuộc tính như màu sắc, định dạng trục, và hiển thị đường cong.

1.3 Sử dụng thư viện Material Design:

- Tích hợp thư viện Material Design vào giao diện ứng dụng là một kỹ thuật hữu ích giúp tạo ra giao diện người dùng thẩm mỹ và tương tác tốt.

1.4 Xử lý Thêm và Xóa Hàm Số :

- Kiểm Tra Hợp Lệ của Hàm Số: Trước khi vẽ đồ thị, xác minh tính hợp lệ của hàm số được nhập bởi người dùng.

- Thêm và Xóa Hàm Số: Thêm hàm số vào danh sách để vẽ đồ thị. Xóa hàm số không cần thiết khỏi danh sách. Tạo tập điểm từ hàm số với model tự tạo sau đó vẽ tập điểm lên đồ thị

1.5 Lưu và Nạp Tệp JSON :

- Tạo và Đọc Tệp JSON: Sử dụng thư viện Newtonsoft.Json để lưu trữ danh sách các hàm số đã nhập thành tệp JSON. Đọc và tải danh sách từ tệp JSON để tái sử dụng đồ thị đã lưu.

1.6 Hiển thị Điểm Giao Nhau :

- Tính Toán và Vẽ Điểm Giao Nhau: Dựa trên danh sách các hàm số, thực hiện tính toán để tìm điểm giao nhau và hiển thị chúng trên đồ thị.

1.7 Thiết kế Giao diện Thân thiện :

- Sử dụng Controls WPF: Tạo giao diện người dùng dựa trên các controls có sẵn trong WPF như TextBox, Button, ListBox để thực hiện các chức năng.

- Hộp Thoại và Nút Điều Khiển: Sử dụng hộp thoại và nút điều khiển để cho phép người dùng tương tác dễ dàng với ứng dụng.

1.8 Kỹ thuật Sử dụng :

- Xử lý Sự kiện: Sử dụng cơ chế xử lý sự kiện để phản hồi khi người dùng thực hiện các hành động như thêm, xóa, lưu, và nạp tệp.

- Kiểm Tra Điều Kiện: Kiểm tra tính hợp lệ của dữ liệu nhập vào trước khi xử lý.

2. Mô hình hóa :

2.1 Use Case Diagram:

- Mô hình Use Case Diagram cho thấy các chức năng mà người dùng có thể thực hiện trong ứng dụng.

2.2 Diagram Sequence:

- Biểu đồ Sequence Diagram minh họa quy trình tương tác giữa người dùng và hệ thống khi thực hiện một chức năng nhất định.

2.3 Class Diagram:

- Biểu đồ Class Diagram biểu thị cấu trúc các lớp và mối quan hệ giữa chúng trong ứng dụng.

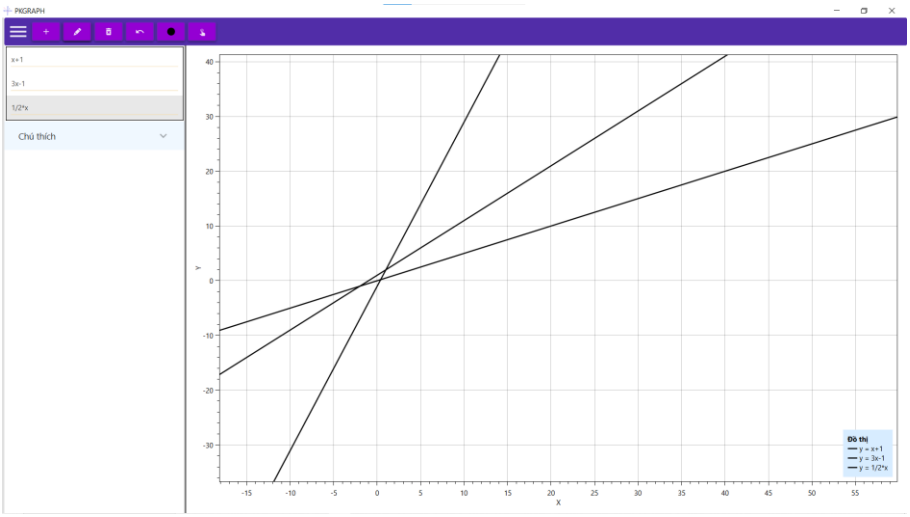
2.4 Activity Diagram:

- Biểu đồ Activity Diagram mô tả các hoạt động cụ thể trong quá trình thực hiện một chức năng hoặc quy trình.

IV. Hiện thực hóa một số chức năng nổi bật

1. Thêm và Xóa Hàm Số :

- Thêm Hàm Số: Người dùng có thể nhập hàm số vào ô văn bản và thêm chúng vào danh sách để vẽ đồ thị.Chương trình kiểm tra tính hợp lệ của hàm số trước khi thêm vào danh sách.



- Xóa Hàm Số: Cho phép người dùng chọn hàm số cần xóa từ danh sách và loại bỏ nó khỏi đồ thị.

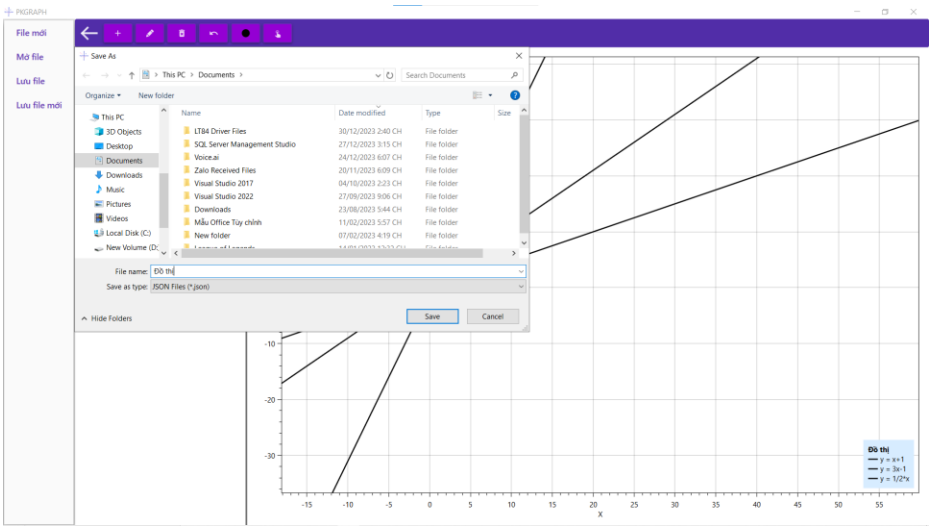
2. Lưu và Nạp Tệp JSON :

- Lưu Đồ Thị: Người dùng có thể lưu danh sách các hàm số đã thêm thành một tệp JSON để sử dụng sau này.

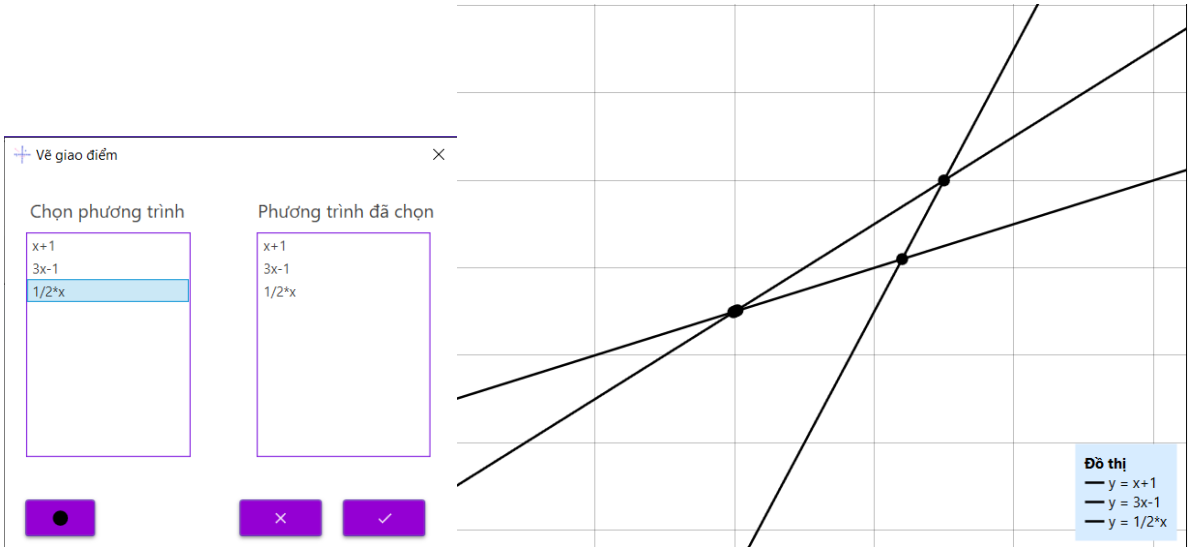
- Nạp Tệp Đã Lưu: Ứng dụng cho phép người dùng nạp tệp JSON để tái tạo lại danh sách hàm số và vẽ đồ thị tương ứng.

3. Hiển Thị Điểm Giao Nhau :

- Tìm Điểm Giao Nhau: Ứng dụng tính toán và tìm các điểm giao nhau của các hàm số có trong danh sách.



- Vẽ Điểm Giao Nhau: Đồ thị hiển thị các điểm giao nhau với màu sắc hoặc biểu tượng để phân biệt.



4. Tính Năng Thân Thiện Với Người Dùng :

- Giao Diện Thân Thiện: Sử dụng các controls và layout phù hợp để tạo giao diện dễ sử dụng và trực quan cho người dùng.

- Thông Báo và Phản Hồi: Cung cấp thông báo khi người dùng thực hiện các hành động như thêm, xóa, lưu, và nạp tệp.

5. Tính Linh Hoạt và Thực Tế :

- Đồng Nhất và Linh Hoạt: Ứng dụng có khả năng xử lý nhiều hàm số cùng một lúc và cho phép người dùng thao tác linh hoạt trên đồ thị.

- Phản Hồi Nhanh Chóng: Đáp ứng ngay lập tức khi người dùng thực hiện các thao tác và cung cấp kết quả vẽ đồ thị một cách nhanh chóng.

V. Tổng kết :

1. Kết quả đạt được :

1.1 Tiện Ích và Linh Hoạt :

- Tính Năng Đầy Đủ: Ứng dụng cung cấp các chức năng thêm, xóa hàm số, lưu và nạp tệp JSON, cũng như hiển thị điểm giao nhau, tạo điều kiện cho người dùng tương tác dễ dàng và linh hoạt với đồ thị.

1.2 Trải Nghiệm Người Dùng Tốt :

- Giao Diện Thân Thiện: Giao diện người dùng được thiết kế sao cho dễ sử dụng và trực quan, giúp người dùng tìm hiểu và tương tác với ứng dụng một cách dễ dàng.

- Phản Hồi Nhanh Chóng: Ứng dụng cung cấp kết quả vẽ đồ thị một cách nhanh chóng sau mỗi thao tác, tạo cảm giác thoải mái và hứng thú cho người dùng.

1.3 Quản Lý Dữ Liệu Tốt :

- Lưu Trữ và Tái Sử Dụng: Khả năng lưu và nạp các hàm số đã nhập giúp người dùng có thể lưu trữ dữ liệu và tái sử dụng chúng một cách dễ dàng.

1.4 Tính Khả dụng và Thực Tiễn :

- Phù Hợp và Thực Tế: Ứng dụng có thể xử lý nhiều hàm số cùng một lúc và hiển thị đồ thị với độ chính xác cao, phục vụ cho nhu cầu thực tế của người dùng.

1.5 Khả Năng Mở Rộng và Phát Triển :

- Khả Năng Mở Rộng: Kiến trúc ứng dụng linh hoạt, dễ mở rộng và cải thiện theo yêu cầu cụ thể của người dùng.

2. Hạn chế :

2.1 Độ Phức Tạp Của Chức Năng :

- Hạn Chế Của Chức Năng: Có thể chưa thực hiện được một số chức năng phức tạp như tìm kiếm đồ thị đi qua một điểm cụ thể hoặc tính toán các giá trị đạo hàm, tích phân của hàm số.

2.2 Giao Diện Người Dùng :

- Khả năng Tương Tác: Giao diện có thể cần được cải thiện để tăng tính tương tác hoặc cung cấp thông tin hỗ trợ cho người dùng không chuyên.

2.3 Hiệu Suất và Tính Ổn Định :

- Hiệu Suất và Độ Ổn Định: Ứng dụng có thể gặp vấn đề với hiệu suất hoặc độ ổn định khi xử lý nhiều hàm số phức tạp hoặc trong điều kiện hoạt động tải cao.

2.4. Quản Lý Dữ Liệu :

- Quản Lý Dữ Liệu: Cần cải thiện khả năng quản lý dữ liệu, nhất là khi số lượng hàm số lớn.

3. Hướng phát triển :

3.1 Tích Hợp Chức Năng Nâng Cao :

- Tính Toán Chuyên Sâu: Bổ sung các chức năng tính toán cao cấp như đạo hàm, tích phân, và tìm cực trị, giúp người dùng phân tích chi tiết hơn về đồ thị hàm số.

- Tính Năng Tìm Kiếm Đồ Thị: Phát triển tính năng tìm kiếm đồ thị đi qua một điểm cụ thể, tạo điều kiện cho người dùng phân tích và khám phá thông tin đồ thị một cách dễ dàng.

3.2 Tối Ưu Hiệu Suất và Tương Tác :

- Tối Ưu Hóa Hiệu Suất: Cải thiện hiệu suất và tối ưu hóa để ứng dụng có thể xử lý đồ thị phức tạp và lớn một cách nhanh chóng và mượt mà hơn.

- Giao Diện Tương Tác: Nâng cấp giao diện người dùng, cung cấp thông tin hỗ trợ chi tiết, hướng dẫn sử dụng, và tương tác linh hoạt hơn.

3.3 Mở Rộng Quản Lý Dữ Liệu và Bảo Mật :

- Quản Lý Dữ Liệu Tốt Hơn: Tăng cường khả năng quản lý dữ liệu, cho phép người dùng tự do lưu trữ và quản lý một lượng lớn hàm số.

- Tăng Cường Bảo Mật: Đảm bảo tính bảo mật cao hơn cho dữ liệu người dùng thông qua các biện pháp bảo mật tiên tiến.

3.4 Hỗ Trợ Đa Nền Tảng và Tích Hợp :

- Đa Nền Tảng: Phát triển phiên bản ứng dụng đồ thị cho nhiều nền tảng, bao gồm cả web và di động để tăng khả năng tiếp cận cho người dùng.

- Tích Hợp Kết Nối Ngoại Vi: Hỗ trợ tích hợp với các thiết bị ngoại vi hoặc công cụ thống kê, phân tích số liệu để tạo ra ứng dụng đa năng hơn.

4. Tài liệu tham khảo :

- Oxy plot:https://oxyplot.readthedocs.io/en/latest/getting-started/hello-wpf-xaml.html

- Material design : https://github.com/MaterialDesignInXAML/MaterialDesignInXamlToolkit

- WPF Quickstart : https://scottplot.net/quickstart/wpf/