**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**

**------------------**



**BÁO CÁO THỰC TẬP CƠ SỞ**

**VIẾT CHƯƠNG TRÌNH MÔ PHỎNG THUẬT TOÁN**

**DUYỆT CÂY NHỊ PHÂN TRÊN MÔI TRƯỜNG ĐỒ HOẠ**

|  |  |  |
| --- | --- | --- |
| **Giảng viên hướng dẫn** | **:** | **TS. Nguyễn Đình Hưng** |
| **Sinh viên thực hiện** | **:** | **Nguyễn Quốc Khánh** |
| **Lớp học phần** | **:** | **62.CNTT-2** |
| **Mã sinh viên** | **:** | **62130853** |
|  |  |  |

Khánh Hoà – 2023

**MỤC LỤC**

[**LỜI CẢM ƠN** 4](#_Toc123769212)

[**CHƯƠNG I: GIỚI THIỆU** 5](#_Toc123769213)

[1. Tổng quan về Cây (Tree) 5](#_Toc123769214)

[2. Cây nhị phân (Binary Tree) 5](#_Toc123769215)

[3. Duyệt Cây nhị phân 5](#_Toc123769216)

[4. Yêu cầu của đề tài 7](#_Toc123769217)

[3.1. Ngôn ngữ lập trình C# 7](#_Toc123769221)

[3.2. Phần mềm Microsoft Visual Studio 2022 7](#_Toc123769222)

[3.3. Một số thư viện sử dụng trong đề tài 7](#_Toc123769223)

[**CHƯƠNG II: CƠ SỞ LÝ THUYẾT** 9](#_Toc123769224)

[1. Tìm hiều ngôn ngữ lập trình C# 9](#_Toc123769225)

[1.1. Cơ bản về ngôn ngữ lập trình C# 9](#_Toc123769226)

[1.2. Tính đóng gói trong C# 9](#_Toc123769227)

[1.3. Phương thức (Hàm) 9](#_Toc123769228)

[1.4. Lớp (Class) 9](#_Toc123769229)

[1.5. Kế thừa. 10](#_Toc123769230)

[2. Tìm hiểu Winform 10](#_Toc123769231)

[2.1. Tổng quan 10](#_Toc123769232)

[2.2. Cơ bản về Winform 10](#_Toc123769233)

[3. Giải thuật Duyệt Cây 11](#_Toc123769234)

[3.1. Hàm duyệt trước 11](#_Toc123769235)

[3.2. Hàm duyệt giữa 12](#_Toc123769236)

[3.3. Hàm duyệt sau 12](#_Toc123769237)

[**CHƯƠNG III: CÀI ĐẶT CHƯƠNG TRÌNH** 14](#_Toc123769238)

[1. Giao diện 14](#_Toc123769239)

[2. Các lớp bên trong 14](#_Toc123769240)

[2.1 Form1.cs 14](#_Toc123769241)

[2.2 Node\_Tree.cs 14](#_Toc123769242)

[2.3 B\_Tree.cs 14](#_Toc123769243)

[3. Mã Nguồn 14](#_Toc123769244)

[**CHƯƠNG 4: KẾT LUẬN** 15](#_Toc123769245)

[1. Đánh giá chung về đề tài 15](#_Toc123769246)

[2. Hướng phát triển của đề tài 15](#_Toc123769247)

[**TÀI LIỆU THAM KHẢO** 16](#_Toc123769248)

[**NHẬN XÉT CỦA GIẢNG VIÊN** 17](#_Toc123769249)

**MỤC LỤC HÌNH ẢNH**

[Hình 1: Cây Nhị Phân 5](file:///C:\Users\KHANH-NQ\Desktop\Word\NguyenQuocKhanh_62130853_TTCS_2022.docx#_Toc123769880)

[Hình 2: Hàm Duyệt Trước 11](file:///C:\Users\KHANH-NQ\Desktop\Word\NguyenQuocKhanh_62130853_TTCS_2022.docx#_Toc123769881)

[Hình 3: Hàm Duyệt Giữa 12](file:///C:\Users\KHANH-NQ\Desktop\Word\NguyenQuocKhanh_62130853_TTCS_2022.docx#_Toc123769882)

[Hình 4: Hàm Duyệt Sau 12](file:///C:\Users\KHANH-NQ\Desktop\Word\NguyenQuocKhanh_62130853_TTCS_2022.docx#_Toc123769883)

[Hình 5: Demo Giao Diện 14](#_Toc123769884)

**LỜI CẢM ƠN**

Trước hết, em muốn gửi lời cảm ơn chân thành và sâu sắc đến thầy Nguyễn Đình Hưng người đã trực tiếp quan tâm và hướng dẫn trong thời gian qua, cũng như các thầy cô Khoa Công nghệ thông tin của Trường đại học Nha Trang đã tạo điều kiện, hỗ trợ và giúp đỡ chúng em trong quá trình học tập và nghiên cứu đề tài này.

Do kiến thức còn hạn chế và thời gian thực hiện ngắn nên bài báo cáo của em còn nhiều thiếu sót. Kính mong sự góp ý của quý thầy cô.

Em xin chân thành cám ơn.

**CHƯƠNG I: GIỚI THIỆU**

## **Tổng quan về Cây (Tree)**

Cây là một cấu trúc dữ liệu gồm một tập hữu hạn các nút, giữa các nút có một quan hệ phân cấp gọi là quan hệ “cha – con”. Có một nút đặc biệt gọi là gốc (root).

Không giống như Mảng (Array) hay Danh Sách Liên Kết (Linked List), là các cấu trúc dữ liệu tuyến tính, Cây (Tree) là cấu trúc dữ liệu phi tuyến tính (dạng phân cấp).

Cấu trúc Cây (Tree) thường được dụng để lưu trữ các thông tin dưới dạng một hệ thống phân cấp tự nhiên.

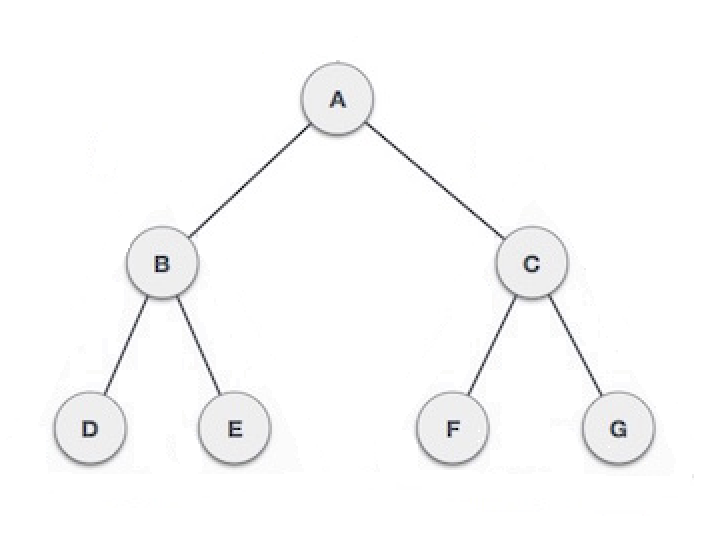
Ví dụ: Cấu trúc thư mục trên đĩa cũng có cấu trúc cây, thư mục gốc có thể coi là gốc của cây đó với các cây con là các thư mục con và tệp nằm trên thư mục gốc.

## **Cây nhị phân (Binary Tree)**

Cây nhị phân là một dạng quan trọng của cấu trúc cây. Nó có đặc điểm là mọi nút trên cây chỉ có tối đa hai nhánh con. Với một nút thì người ta cũng phân biệt cây con trái và cây con phải của nút đó. Cây nhị phân là cây có tính đến thứ tự của các nhánh con.

## **Duyệt Cây nhị phân**

Duyệt cây là một tiến trình để truy cập tất cả các nút của một cây và cũng có thể in các giá trị của các nút này. Bởi vì tất cả các nút được kết nối thông qua các cạnh (hoặc các liên kết), nên chúng ta luôn luôn bắt đầu truy cập từ nút gốc. Do đó, chúng ta không thể truy cập ngẫu nhiên bất kỳ nút nào trong cây.

Có ba phương thức có thể sử dụng để duyệt một cây:

Hình 1: Cây Nhị Phân

* **Duyệt trước (Pre-order Traversal)** : nút gốc được duyệt đầu tiên, sau đó sẽ duyệt nhánh cây con bên trái đến khi không còn nút bên trái và cuối cùng sẽ duyệt nhánh cây con bên phải đến hết

Ví dụ: Ở hình 1, A là nút gốc nên A được duyệt đầu tiên, tiếp đến di chuyển đến nhánh cây con bên trái A là B. Ở đây B trở thành nút gốc tạm thời, duyệt B sau đó tiếp tục di chuyển đến nhánh trái của nút gốc tạm thời B là D. Ở đây D trở thành nút gốc tạm thời, duyệt D. Nút D không có nút trái và phải, di chuyển trở về B sau đó di chuyển về phía bên nhánh phải của B là E, E trở thành nút gốc tạm thời và được duyệt. Tương tự như vậy với nhánh bên phải của A ta thu được kết quả duyệt như sau: A->B->D->E->C->F->G.

* **Duyệt giữa (In-order Traversal)**: nhánh cây con bên trái được duyệt đầu tiên, tiếp theo sau đó là nút gốc và cuối cùng là nhánh cây con bên phải:

Ví dụ: Ở hình 1, A là nút gốc, đầu tiên từ A di chuyển đến nhánh cây con bên trái A là B. Ở đây B trở thành nút gốc tạm thời, sau đó tiếp tục di chuyển về nhánh trái của B là D. Ở đây D trở thành nút gốc tạm thời, vì D không có nhánh trái lẫn nhánh phải nên duyệt D. Sau đó di chuyển về B và duyệt B, tiếp đó di chuyển về phía nhánh phải của B là E, E trở thành nút gốc tạm thời, giống như D, E không có nhánh trái và phải nên E được duyệt. Sau khi đã duyệt hết nhánh bên trái thì A là nút gốc được duyệt. Tương tự như vậy với nhánh bên phải của A ta thu được kết quả duyệt như sau: D->B->E->A->F->C->G

* **Duyệt sau (Post-order Traversal)**: đầu tiên duyệt nhánh cây con bên trái, sau đó sẽ duyệt đến nhánh cây con bên phải và cuối cùng là duyệt nút gốc:

Ví dụ: Ở hình 1, A là nút gốc, đầu tiên từ A di chuyển đến nhánh cây con bên trái A là B. Ở đây B trở thành nút gốc tạm thời, sau đó tiếp tục di chuyển về nhánh trái của B là D. Ở đây D trở thành nút gốc tạm thời, vì D không có nhánh trái lẫn nhánh phải nên duyệt D. Sau đó di chuyển về B và đó di chuyển về phía nhánh phải của B là E, E trở thành nút gốc tạm thời, giống như D, E không có nhánh trái và phải nên E được duyệt. Sau khi đã duyệt hết cả hai nhánh thì quay trở lại nút gốc tạm thời B và duyệt B. Tương tự như vậy với nhánh bên phải của A. Và khi cả hai nhánh của nút gốc A đều đã duyệt xong thì mới duyệt nút gốc A. Qua đó ta thu được kết quả duyệt như sau: D->E->B->F->G->C->A

## **Yêu cầu của đề tài**



### Ngôn ngữ lập trình C#

Ngôn ngữ C# là một được phát triển bởi đội ngũ kỹ sư của Microsoft vào năm 2000. C# là ngôn ngữ lập trình hiện đại, hướng đối tượng và kết thừa một số tính năng của C++ và Java như cú pháp lập trình, đa hình tĩnh, đa hình động, tính đóng gói….

C# với sự hỗ trợ mạnh mẽ của .NET Framework giúp cho việc tạo một ứng dụng Windows Forms hay WPF (Windows Presentation Foundation), phát triển game, ứng dụng Web, ứng dụng Mobile trở nên rất dễ dàng.

C# hiện là một trong những công cụ lập trình phổ biến nhất thế giới, có cộng đồng phát triển lớn (cho phép tái sử dụng mã nguồn, giảm chi phí phát triển).

### Phần mềm Microsoft Visual Studio 2022

Phần mềm Microsoft Visual Studio là một môi trường phát triển tích hợp (IDE) từ Microsoft. Microsoft Visual Studio còn được gọi là "Trình soạn thảo mã nhiều người sử dụng nhất thế giới ", được dùng để lập trình C++ và C# là chính.

Nó được sử dụng để phát triển chương trình máy tính cho Microsoft Windows, cũng như các trang web, các ứng dụng web và các dịch vụ web. Visual Studio sử dụng nền tảng phát triển phần mềm của Microsoft như Windows API, Windows Forms, Windows Presentation Foundation, Windows Store và Microsoft Silverlight. Nó có thể sản xuất cả hai ngôn ngữ máy và mã số quản lý.

### Một số thư viện sử dụng trong đề tài

#### **System.Windows.Forms**

Windows Forms (hay WinForms) là thư viện đồ họa (GUI) mã nguồn mở và miễn phí được bao gồm như một phần của Microsoft.NET Framework hoặc Mono Framework, cung cấp nền tảng để viết các ứng dụng khách phong phú cho máy tính để bàn, máy tính xách tay và máy tính bảng.

#### **System.Drawing**

Lớp đồ họa cung cấp các phương pháp để vẽ đến thiết bị hiển thị. Các lớp như Hình chữ nhật và điểm đóng gói các nguyên thủy GDI +. Lớp Pen được sử dụng để vẽ các đường thẳng, trong khi các lớp có nguồn gốc từ lớp trừu tượng Brush được sử dụng để lấp đầy nội thất của các hình dạng.

#### **System.Threading**

System.Threading trong .NET Framework cung cấp các lớp và các đối tượng cho việc tạo và quản lý các luồng (thread) trong ứng dụng của bạn. Luồng là một đoạn mã chạy song song với các đoạn mã khác trong chương trình của bạn.

**CHƯƠNG II: CƠ SỞ LÝ THUYẾT**

## **Tìm hiều ngôn ngữ lập trình C#**

### Cơ bản về ngôn ngữ lập trình C#

C# là ngôn ngữ hiện đại

C# là một ngôn ngữ lập trình thuần hướng đối tượng

C# là một ngôn ngữ ít từ khóa

C# kế thừa một số ưu điểm của C++ và Java

### Tính đóng gói trong C#

Tính đóng gói (Encapsulation) trong C# được định nghĩa là "tiến trình đóng gói một hoặc nhiều mục bên trong một gói logic hoặc vật lý". Tính đóng gói, trong phương pháp lập trình hướng đối tượng, ngăn cản việc truy cập tới chi tiết của trình trình triển khai

### Phương thức (Hàm)

Một phương thức là một nhóm lệnh cùng nhau thực hiện một tác vụ. Mỗi chương trình C# có ít nhất một lớp với một phương thức là Main.

Để sử dụng một phương thức trong C#, bạn cần:

Định nghĩa phương thức

Gọi phương thức

**Định nghĩa phương thức**: Trong C# khi bạn định nghĩa một phương thức, về cơ bản, bạn khai báo các phần tử của cấu trúc của nó. Cú pháp để định nghĩa một phương thức trong C# là như sau:

<Access Specifier> <Kiểu\_trả\_về> <tên\_phương\_thức>(danh\_sách\_tham\_số)

{//phần thân phương thức}

### Lớp (Class)

Class trong C# là một loại kiểu dữ liệu đặc biệt chứa định nghĩa những thuộc tính (thông tin) và phương thức (hành vi), dùng để mô tả chung cho một nhóm những thực thể cùng loại.

Trong C#, mỗi class có thể chứa:

Biến thành viên (field): Lưu trữ các thông tin mô tả về đối tượng hay trạng thái của đối tượng;

Thuộc tính (property): có vai trò lưu trữ thông tin tương tự như biến thành viên nhưng có khả năng kiểm soát dữ liệu xuất nhập;

Phương thức (method): Dùng để cập nhật, tính toán, cung cấp và xử lý thông tin;

Sự kiện (delegate/event): Gửi thông báo về sự thay đổi trạng thái của đối tượng ra bên ngoài.

Ngoài ra, trong class còn có thể chứa định nghĩa của kiểu dữ liệu khác, gọi là kiểu thành viên. Class có thể chứa định nghĩa của bất kỳ nhóm kiểu nào mà bạn đã biết.

### Kế thừa.

Tính kế thừa (Inheritance) trong C# là một trong những khái niệm quan trọng nhất trong lập trình hướng đối tượng C#.

Tính kế thừa cho phép chúng ta định nghĩa một lớp trong điều kiện một lớp khác, mà làm cho nó dễ dàng hơn để tạo và duy trì một ứng dụng. Điều này cũng cung cấp một cơ hội để tái sử dụng tính năng code, tiện hơn và thời gian thực thi nhanh hơn.

## **Tìm hiểu Winform**

### 2.1 Tổng quan

Winform là thuật ngữ mô tả một ứng dụng được viết dùng .NET Framework và có giao diện người dùng Windows Forms.

Mỗi màn hình windows cung cấp một giao diện giúp người dùng giao tiếp với ứng dụng. Giao diện này được gọi là giao diện đồ họa (GUI) của ứng dụng.

### Cơ bản về Winform

Form: Dùng để thiết kế giao diện cho người sử dụng.

Button: Dùng để người dùng tương tác với giao diện khi nhấn sẽ kích hoạt sự kiện thiết lập trong ứng dụng.

Label: Hiển thị văn bản cố định trên form.

Textbox: Dùng để hiển thị dữ liệu hoặc để người dùng nhập dữ liệu vào ứng dụng.

MessageBox: Hiển thị thông báo khi các sự kiện trong ứng dụng được kích hoạt.

Panel: Là thành phần hiển thị một khối nội dung bố trí trên một form. Nó có thể chứa các điều khiển khác như các nút, hộp văn bản, và các điều khiển khác, và có thể được sử dụng để tạo các khu vực cho phép người dùng thao tác với các điều khiển bên trong nó.

## **Giải thuật Duyệt Cây**

### 3.1 Hàm duyệt trước

Hình 2: Hàm Duyệt Trước

Duyệt trước là cách duyệt câu nhị phân theo thứ tự là duyệt nút gốc, duyệt nhánh con bên trái và sau cùng là nhánh con bên phải.

Trong hình 2 là hàm dùng để duyệt cây nhị phân theo cách duyệt trước có tên là “preOrder”. Trước hết hàm sẽ kiểm tra xem nút có tồn tại hay không bằng cách gọi hàm “existe()”. Nếu nút có tồn tại thì gán giá trị của nút vào biến “text” (nối vào sau nếu biến “text” không rỗng). Tiếp theo gọi lại hàm “preOrder” với nút con trái của nút hiện tại làm đối số. Sau đó tiếp tục gọi lại hàm “preOrder” với nút con phải của nút hiện tại làm đối số.

Việc gọi lại hàm “preOrder” với các nút con đều được bao quanh bởi điều kiện "if (existe(node))" sẽ giúp hàm sẽ duyệt qua tất cả các nút có trong cây theo thứ tự nút gốc sau đó đến nút trái và cuối cùng là nút phải bởi vì hàm sẽ chỉ thực hiện khi nút con tồn tại.

### Hàm duyệt giữa

Hình 3: Hàm Duyệt Giữa

Duyệt giữa là cách duyệt câu nhị phân theo thứ tự là duyệt nhánh con bên trái sau đó đến nút gốc và sau cùng là nhánh con bên phải.

Trong hình 3 là hàm dùng để duyệt cây nhị phân theo cách duyệt giữa có tên là “inOrder”. Trước hết hàm sẽ kiểm tra xem nút có tồn tại hay không bằng cách gọi hàm “existe()”. Nếu nút có tồn tại thì gọi lại hàm “inOrder” với nút con trái của nút hiện tại làm đối số. Tiếp theo là gán giá trị của nút hiện tại vào biến “text” (nối vào sau nếu biến “text” không rỗng). Sau đó tiếp tục gọi lại hàm “inOrder” một lần nữa với nút con phải của nút hiện tại làm đối số.

Việc gọi lại hàm “inOrder” với các nút con đều được bao quanh bởi điều kiện "if (existe(node))" sẽ giúp hàm sẽ duyệt qua tất cả các nút có trong cây theo thứ tự nút trái sau đó đến nút gốc và cuối cùng là nút phải bởi vì hàm sẽ chỉ thực hiện khi nút con tồn tại.

### Hàm duyệt sau

Hình 4: Hàm Duyệt Sau

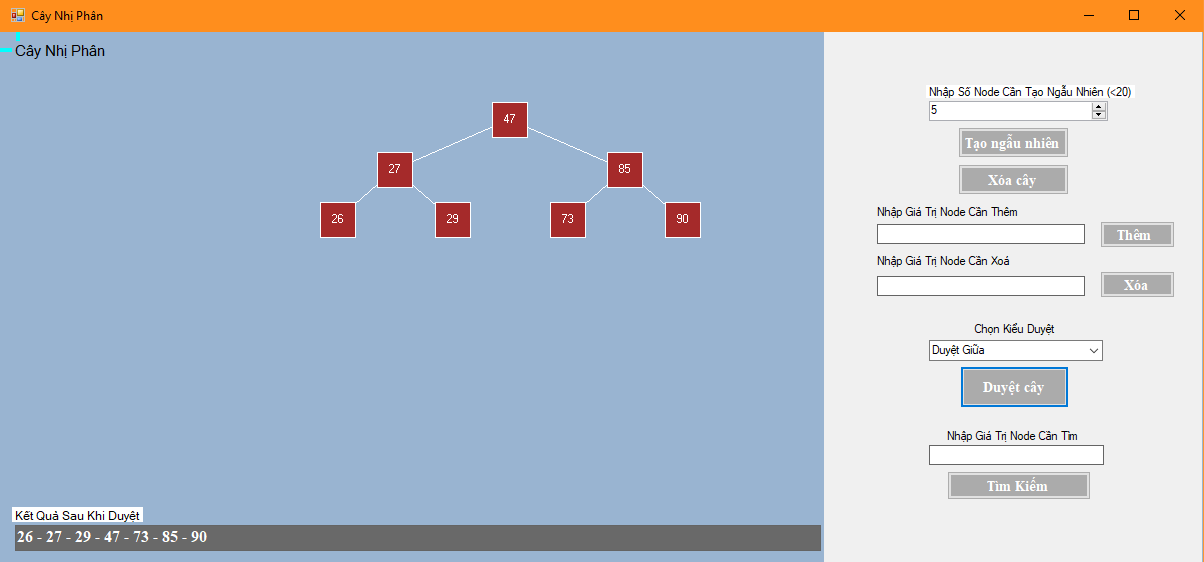
Duyệt sau là cách duyệt câu nhị phân theo thứ tự là duyệt nhánh con bên trái sau đó đến nhánh con bên phải và cuối cùng là nút gốc.

Trong hình 4 là hàm dùng để duyệt cây nhị phân theo cách duyệt sau có tên là “postOrder”. Trước hết hàm sẽ kiểm tra xem nút có tồn tại hay không bằng cách gọi hàm “existe()”. Nếu nút có tồn tại thì gọi lại hàm “inOrder” với nút con trái của nút hiện tại làm đối số. Tiếp theo, tiếp tục gọi lại hàm “inOrder” một lần nữa với nút con phải của nút hiện tại làm đối số. Cuối cùng là gán giá trị của nút hiện tại vào biến “text” (nối vào sau nếu biến “text” không rỗng).

Việc gọi lại hàm “postOrder” với các nút con đều được bao quanh bởi điều kiện "if (existe(node))" sẽ giúp hàm sẽ duyệt qua tất cả các nút có trong cây theo thứ tự nút trái sau đó đến nút phải và cuối cùng là nút gốc bởi vì hàm sẽ chỉ thực hiện khi nút con tồn tại.

**CHƯƠNG III: CÀI ĐẶT CHƯƠNG TRÌNH**

## **Giao diện**



Hình 5: Demo Giao Diện

## **Các lớp bên trong**

### 2.1 Form1.cs

Quản lý giao diện đồ họa (GUI) trên Windows Form. Chương trình này dùng để tạo ra một cây nhị phân và có thể thực hiện các thao tác như thêm, xóa, tìm kiếm, in kết quả duyệt trước, giữa, sau và vẽ cây ra màn hình.

### 2.2 Node\_Tree.cs

Dùng để lưu trữ dữ liệu của nút, các liên kết với các nút con và các liên kết với nút cha của nó, và một số thuộc tính khác để hỗ trợ việc vẽ cây và thao tác.

### 2.3 B\_Tree.cs

Chức năng: Sử dụng để tạo ra một cây nhị phân và thực hiện các thao tác trên cây như: thêm, xóa, tìm, vẽ cây, và duyệt cây.

## **Mã Nguồn**

Đường dẫn: <https://github.com/Qkay3107/Ma_Nguon_Bao_Cao-Nguyen_Quoc_Khanh-62130853.git>

**CHƯƠNG 4: KẾT LUẬN**

## **Đánh giá chung về đề tài**

* 1. **Kết quả đạt được**
  + Tạo được cây nhị phân.
  + Duyệt cây theo 3 cách: duyệt trước, duyệt giữa và duyệt sau.
  + Có các chức năng thêm, xoá, tìm kiếm, tạo cây ngẫu nhiên và xoá cây.
  + Thông báo khi người dùng nhập sai hoặc thao tác sai.
  1. **Hạn chế**
  + Mô phỏng khi duyệt cây còn hạn chế.
  + Giao diện chưa tối ưu và bắt mắt

## **Hướng phát triển của đề tài**

* + Thêm các chuyển động để mô phỏng rõ hơn khi duyệt cây.
  + Cải thiện giao diện

**TÀI LIỆU THAM KHẢO**

[1] Microsoft Learn : [C# Documentation](https://learn.microsoft.com/en-us/dotnet/csharp/)

[2] Microsoft Learn : [Winform](https://learn.microsoft.com/vi-vn/dotnet/desktop/winforms/?view=netframeworkdesktop-4.8)

**NHẬN XÉT CỦA GIẢNG VIÊN**

**Nhận xét của Giảng viên hướng dẫn**

|  |
| --- |
| Khánh Hoà, Ngày……Tháng…….Năm……. |
| **Giảng viên hướng dẫn** |
| *(Ký tên)* |

**Nhận xét của Giảng viên phản biện**

|  |
| --- |
| Khánh Hoà, Ngày……Tháng…….Năm……. |
| **Giảng viên phản biện** |
| *(Ký tên)* |