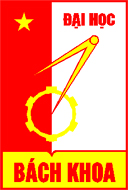
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

──────── \* ───────



**BÀI TẬP LỚN**

HỌC PHẦN: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

*(Mã học phần: IT3103)*

*Đề tài số 2:*

**Hình ảnh hóa các hoạt động cấu trúc dữ liệu cây**

**Sinh viên thực hiện: NHÓM 11**

**Nguyễn Mạnh Hùng 20225630**

**Bùi Ngọc Hợp 20225721**

**Nguyễn Minh Hoàng 20225720**

**Nông Phi Hùng 20194581**

**Trịnh Quốc Hoàng 20225629**

**Lớp: 151965**

**Giảng viên hướng dẫn: Nguyễn Thị Thu Trang**

***Hà Nội, tháng 12 năm 2024***

**PHÂN CÔNG THÀNH VIÊN TRONG NHÓM**

| STT | Họ và tên | MSSV | Email | Công việc | Mức độ hoàn thành | Ghi chú |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Nguyễn Mạnh Hùng | 20225630 | hung.nm225630@sis.hust.edu.vn | * Trưởng nhóm, code chính * Thiết kế giao diện người dùng | Tốt | 30% |
| 2 | Nguyễn Minh Hoàng | 20225720 | hoang.nm225720@sis.hust.edu.vn | * Code phụ * Viết báo cáo * Chỉnh sửa giao diện | Tốt | 15% |
| 3 | Bùi Ngọc Hợp | 20225721 | hop.bn225721@sis.hust.edu.vn | * Code phụ * Viết báo cáo * Kiểm thử phần mềm * Chỉnh sửa giao diện | Tốt | 15% |
| 4 | Nông Phi Hùng | 20194581 | hung.np194581@sis.hust.edu.vn | * Code phụ * Viết báo cáo * Kiểm thử phần mềm | Tốt | 20% |
| 5 | Trịnh Quốc Hoàng | 20225629 | hoang.tq225629@sis.hust.edu.vn | * Code phụ * Làm slide | Tốt | 20% |

# **MỤC LỤC**

[MỤC LỤC 3](#)

[LỜI NÓI ĐẦU 4](#)

[**CHƯƠNG 1. MÔ TẢ BÀI TOÁN 4**](#_2et92p0)

[Mô tả yêu cầu bài toán 5](#)

[Giao diện (GUI) 5](#_7fcoz4g540y)

[**Chương 2. Phân tích bài toán thành các biểu đồ 6**](#_fdhh638fpdzk)

[2.1 Biểu đồ Use Case 6](#_2ve5l447oltg)

[2.2 Biểu đồ Class 7](#_iiql9b1uvcdk)

[Node 7](#_t57iwws7qiyw)

[Tree (Abstract class) 8](#_au4y98o3816n)

[GenericTree (extends Tree) 8](#_p712i7cfl62c)

[BinaryTree 8](#_nyfw4eq7e30m)

[BalancedTree 8](#_wvvxdzc4lft1)

[BalancedBinaryTree 9](#_ehr6og5jx40j)

[TreeVisualizerController 9](#_jsugxkp3hnv)

[Tree Dialog 10](#_rblvznk3kenc)

[CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH MINH HOẠ 11](#)

[3.1 Giao diện khởi động 11](#_4ks8y2rdo771)

[Giao diện chọn cây 12](#_jb9wgja8nf50)

[Giao diện tạo cây 13](#_u78n0h5iogs1)

[**Giao diện thêm mới 1 nút 15**](#_vx1227)

[Giao diện xóa 1 nút 17](#_13jjep1gsera)

[Giao diện cập nhật giá trị 1 nút 19](#_hl2k0y16nend)

[Giao diện duyệt cây 21](#_wj64bbrx7jtl)

[Giao diện tìm kiếm 1 nút 23](#_4srrf8xx2f8a)

# **LỜI NÓI ĐẦU**

Trong lĩnh vực khoa học máy tính, cây dữ liệu là một cấu trúc quan trọng, được ứng dụng rộng rãi trong các bài toán xử lý dữ liệu, tìm kiếm, và tổ chức thông tin. Từ việc quản lý các hệ thống tệp trong máy tính, xây dựng cấu trúc cơ sở dữ liệu, cho đến các thuật toán nâng cao trong trí tuệ nhân tạo, cây dữ liệu đóng vai trò như một công cụ hiệu quả để giải quyết những vấn đề phức tạp.

Dự án **"Ứng dụng Thao tác trên Cây Dữ liệu"** được xây dựng với mục tiêu cung cấp một công cụ trực quan và dễ sử dụng, giúp người dùng khám phá, học tập, và thực hành các thao tác trên nhiều loại cây dữ liệu khác nhau. Ứng dụng không chỉ hỗ trợ các thao tác cơ bản như tạo mới, chèn, xóa, cập nhật, duyệt cây, và tìm kiếm, mà còn cung cấp giao diện đồ họa hiện đại, giúp hiển thị rõ ràng từng bước thực hiện.

Với giao diện người dùng thân thiện và các tính năng như hiển thị mã giả, thanh tiến trình, và khả năng Undo/Redo, ứng dụng không chỉ hỗ trợ học tập, mà còn thúc đẩy tư duy thuật toán và sự sáng tạo trong việc giải quyết vấn đề. Đặc biệt, người dùng có thể lựa chọn giữa các loại cây phổ biến như cây tổng quát, cây nhị phân, cây cân bằng, và cây nhị phân cân bằng để thực hành và nghiên cứu chuyên sâu.

Mặc dù đã cố gắng hoàn thiện sản phẩm nhưng không thể tránh khỏi những thiếu hụt về kiến thức và sai sót trong kiểm thử. Chúng em mong muốn nhận được những nhận xét thẳng thắn, chi tiết đến từ cô và các bạn để tiếp tục hoàn thiện hơn nữa. Cuối cùng, nhóm chúng em xin được gửi lời cảm ơn cô Nguyễn Thị Thu Trang đã hướng dẫn bọn em trong suốt quá trình hoàn thiện bài tập lớn. Xin trân trọng cảm ơn cô.

# **CHƯƠNG 1. MÔ TẢ BÀI TOÁN**

## **Mô tả yêu cầu bài toán**

Ứng dụng sẽ giúp người dùng trực quan hóa và thực hiện các thao tác trên bốn loại cây dữ liệu: Cây tổng quát, Cây nhị phân, Cây cân bằng và Cây nhị phân cân bằng. Các thao tác bao gồm:

1. **Tạo cây mới**: Khởi tạo cây rỗng.
2. **Chèn**: Thêm nút mới vào dưới nút cha.
3. **Xóa**: Xóa nút và các con của nó.
4. **Cập nhật**: Cập nhật giá trị nút.
5. **Duyệt cây**: Duyệt cây theo thuật toán DFS hoặc BFS.
6. **Tìm kiếm**: Tìm kiếm nút theo giá trị.

## **Giao diện (GUI)**

**Menu chính**: Chọn loại cây, menu trợ giúp, và nút thoát.

**Menu thao tác**: Chọn thao tác cần thực hiện, nhập tham số cần thiết.

**Hiển thị mã**: Hiển thị mã giả hoặc mã thực thi khi thao tác được thực hiện.

**Thanh tiến trình**: Hiển thị tiến độ thao tác, cho phép quay lại hoặc tiếp tục.

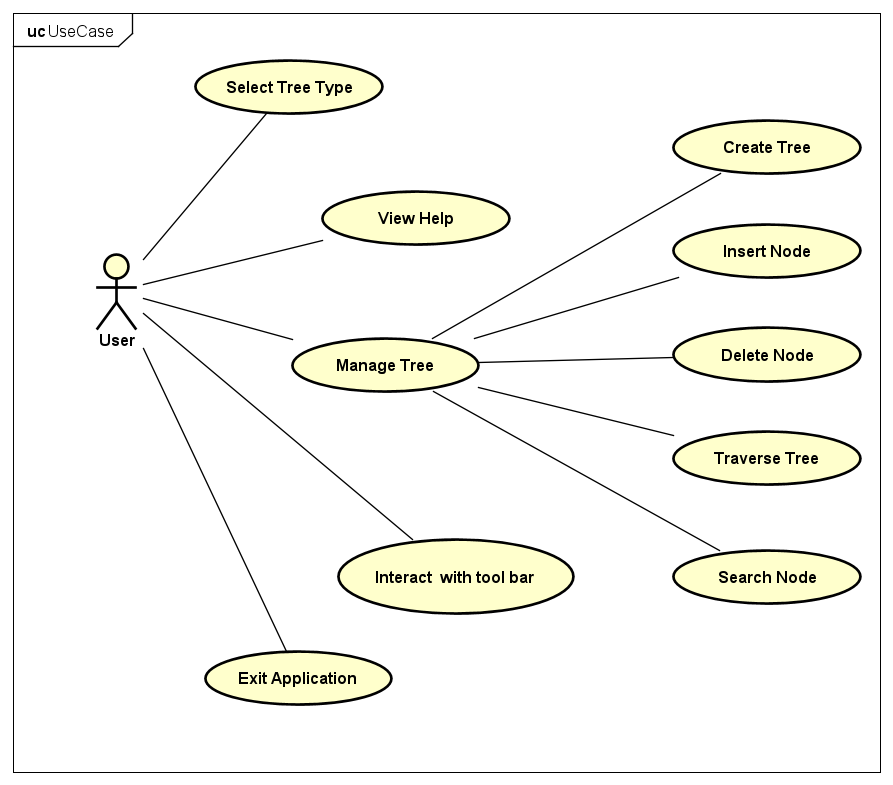
**Yêu cầu thêm**:

* Không có giá trị nút trùng lặp.
* Khoảng cách tối đa từ các nút lá đến gốc cây có thể được người dùng cấu hình cho cây cân bằng và cây nhị phân cân bằng.

Ứng dụng sẽ giúp người dùng hiểu rõ cách hoạt động của các thao tác trên cây dữ liệu thông qua giao diện trực quan và mã nguồn.

# **Chương 2. Phân tích bài toán thành các biểu đồ**

## **Biểu đồ Use Case**



Chương trình của nhóm có 3 use case chính, bao gồm:

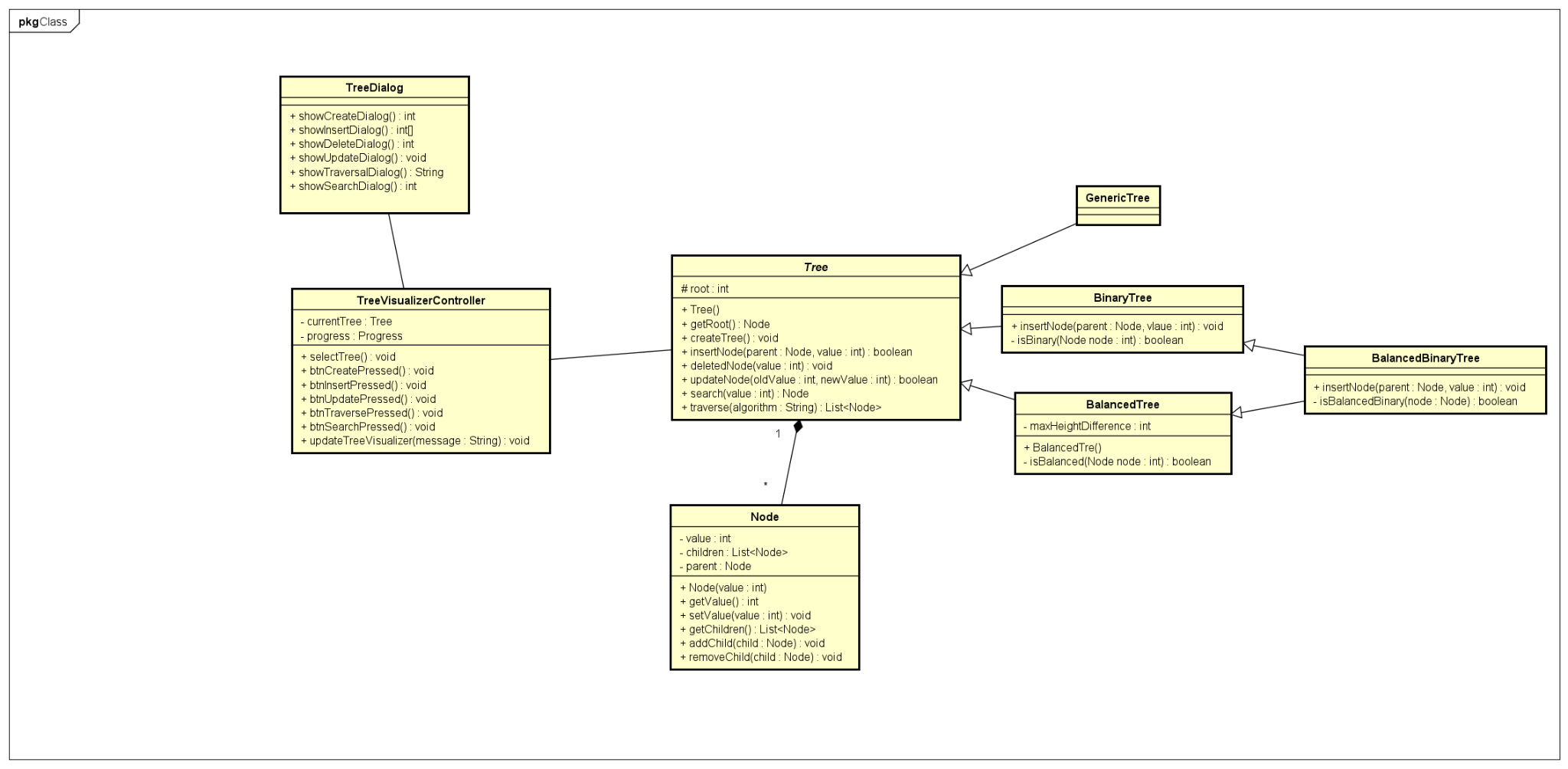
* Chọn loại cây (4 loại cây):
* Generic Tree
* Binary Tree
* Balanced Tree:
* Balanced Binary Tree

\* khi chọn Balanced Tree sẽ hiển thị ra 1 màn hình nhập maxHeight Different, mặc định nếu không nhập gì sẽ là 1

* Mở mục trợ giúp: Hiển thị ra màn hình dialog hướng dẫn sử dụng các thao tác của ứng dụng
* Chọn chức năng (Options):
* Tạo cây: hiển thị dialog để người nhập nút gốc
* Thêm node con: hiển thị dialog dialog nhập nút cha và nút con
* Xóa node con: hiển thị dialog nhập nút muốn xóa
* Duyệt cây: hiển thị dialog cho phép người chọn kiểu duyệt ( DFS hoặc BFS)
* Tìm kiếm 1 node: hiển thị dialog cho phép người dùng nhập nút muốn tìm
* Điều chỉnh thanh công cụ
* Tua nhanh/chậm tốc độ quá trình thực thi
* Tạm dừng quá trình thực thi
* Tiếp tục quá trình thực thi
* Tiến lên/lùi lại từng bước của quá trình thực thi
* Thoát ứng dụng:

Hiển thị dialog xác nhận thoát

## **Biểu đồ Class**



Phân tích Class:

#### Node

**Thuộc tính**

private int value: giá trị node lưu trữ

private List<Node> children: danh sách các nút con

private Node parent: tham chiếu đến node cha

**Phương thức**

public Node(int value) // Constructor

tạo 1 nút mới với giá trị được cung cấp

ban đầu children là danh sách rỗng, parent là null

public int getValue(): trả về giá trị nút

public void setValue(int value): cài giá trị cho nut

public List<Node> getChildren: trả về danh sách nút con

public void addChild(Node child): thêm một nút con

public void removeChild(Node child): xóa một nút con

**Quan hệ**

Node - Tree (composition): 1 cây bắt buộc cây cần có 1 node hoặc có thể có nhiều node

#### Tree (Abstract class)

**Thuộc tính**

protected Node root: nút gốc của cây

**Phương thức**

public Tree() // Constructor: khởi tạo cây với nút gốc là null

public Node getRoot(): trả về nút gốc

public void createTree(): tạo một cây mới bằng cách đặt nút gốc về null

public boolean insertNode(Node parent, int value): thêm 1 nút mới vào cây

public boolean deleteNode(int value): xóa 1 nút khỏi cây

public boolean updateNode(int oldValue, int newValue): cập nhật giá trị 1 nút

public Node search(int value): tìm kiếm và trả về nút có giá trị value

public List<Node> traverse(String algorithm): duyệt cây theo thuật toán được truyền vào (DFS hoặc BFS)

**Quan hệ**

Generic Tree, Binary Tree, Balance Tree kế thừa Tree

#### GenericTree (extends Tree)

Kế thừa tất cả thuộc tính và phương thức từ lớp Tree

#### BinaryTree

**Phương thức**

insertNode(Node parent, int value): Thêm nút vào cây nhị phân

(Override Tree).

isBinary(Node node): Kiểm tra tính hợp lệ của cây nhị phân

(mỗi nút có tối đa 2 con).

#### BalancedTree

**Thuộc tính**

private int maxHeightDifference : Khoảng cách tối đa giữa các lá để duy trì cân bằng

**Phương thức**

BalancedTree(int maxHeightDifference): Khởi tạo cây với khoảng cách tối đa giữa các lá.

isBalanced(Node node): Kiểm tra cây có cân bằng hay không.

#### BalancedBinaryTree

**Phương thức**

insertNode(Node parent, int value): Thêm nút vào cây nhị phân cân bằng

(Override BalancedTree).

isBalancedBinary(Node node): Kiểm tra cây nhị phân cân bằng.

#### TreeVisualizerController

**Thuộc tính**

private Pane treeVisualizer: Màn hình chính hiển thị

private Tree currentTree: Cây hiện tại đang được hiển thị.

private Pane pseudoCode (String): Mã giả cho các thao tác.

private Slider sliderSpeed: Thanh trượt điều chỉnh tốc độ hoạt ảnh minh hoạ

**Phương thức**

selectTree(): Lựa chọn các loại cây

void btnHelpPressed(ActionEvent event): Hiển thị hướng dẫn sử dụng cơ bản

(gắn với sự kiện ấn nút Help)

void btnQuitPressed(ActionEvent event): Thoát ứng dụng  
 (gắn với sự kiện ấn nút Quit)

void btnCreatePressed(ActionEvent event): Khởi tạo tree theo kiểu đã chọn  
 (gắn với sự kiện ấn nút Create)

void btnInsertPressed(ActionEvent event): Chèn thêm node mới vào tree, làm node con của node chỉ định (gắn với sự kiện ấn nút Insert)

void btnDeletePressed(ActionEvent event): Xoá node chỉ định khỏi tree

(gắn với sự kiện ấn nút Delete)

void btnUpdatePressed(ActionEvent event): Cập nhật giá trị mới của node  
chỉ định (gắn với sự kiện ấn nút Update)

void btnTraversePressed(ActionEvent event): Thực hiện duyệt tree (từ node gốc) theo thuật toán do người dùng chọn (DFS hoặc BFS) (gắn với sự kiện ấn nút Traverse)

void btnSearchPressed(ActionEvent event): Thực hiện tìm node chỉ định trên tree, bắt đầu từ node gốc, theo thuật toán do người dùng chọn (DFS hoặc BFS) (gắn với sự kiện ấn nút Search)

private void updateTreeVisualizer(String message): Cập nhật trạng thái của Pane treeVisualizer, giúp thể hiện hoạt ảnh của quá trình thao tác trên tree

**Quan hệ**

TreeVisualizerController - Tree (association): Giúp thực hiện và hiển thị các hành động khi thao tác trên cây

#### Tree Dialog

**Phương thức**

public static int showCreateDialog(): Hiển thị ra màn hình hộp thoại để người dùng nhập giá trị cho node gốc, từ đó khởi tạo tree mới

public static int[] showInsertDialog(): Hiển thị ra màn hình hộp thoại để người dùng nhập giá trị cho node muốn thêm vào tree, và giá trị của node cha của nó

public static int showDeleteDialog(): Hiển thị ra màn hình hộp thoại để người dùng nhập giá trị của node muốn xoá khỏi tree

public static int[] showUpdateDialog(): Hiển thị ra màn hình hộp thoại để người dùng nhập giá trị hiện tại của node cần cập nhật, và giá trị mới mà người dùng mong muốn cho node đó

public static String showTraversalDialog(): Hiển thị ra màn hình hộp thoại để người dùng chọn thuật toán để duyệt tree, ở đây sử dụng BFS và DFS

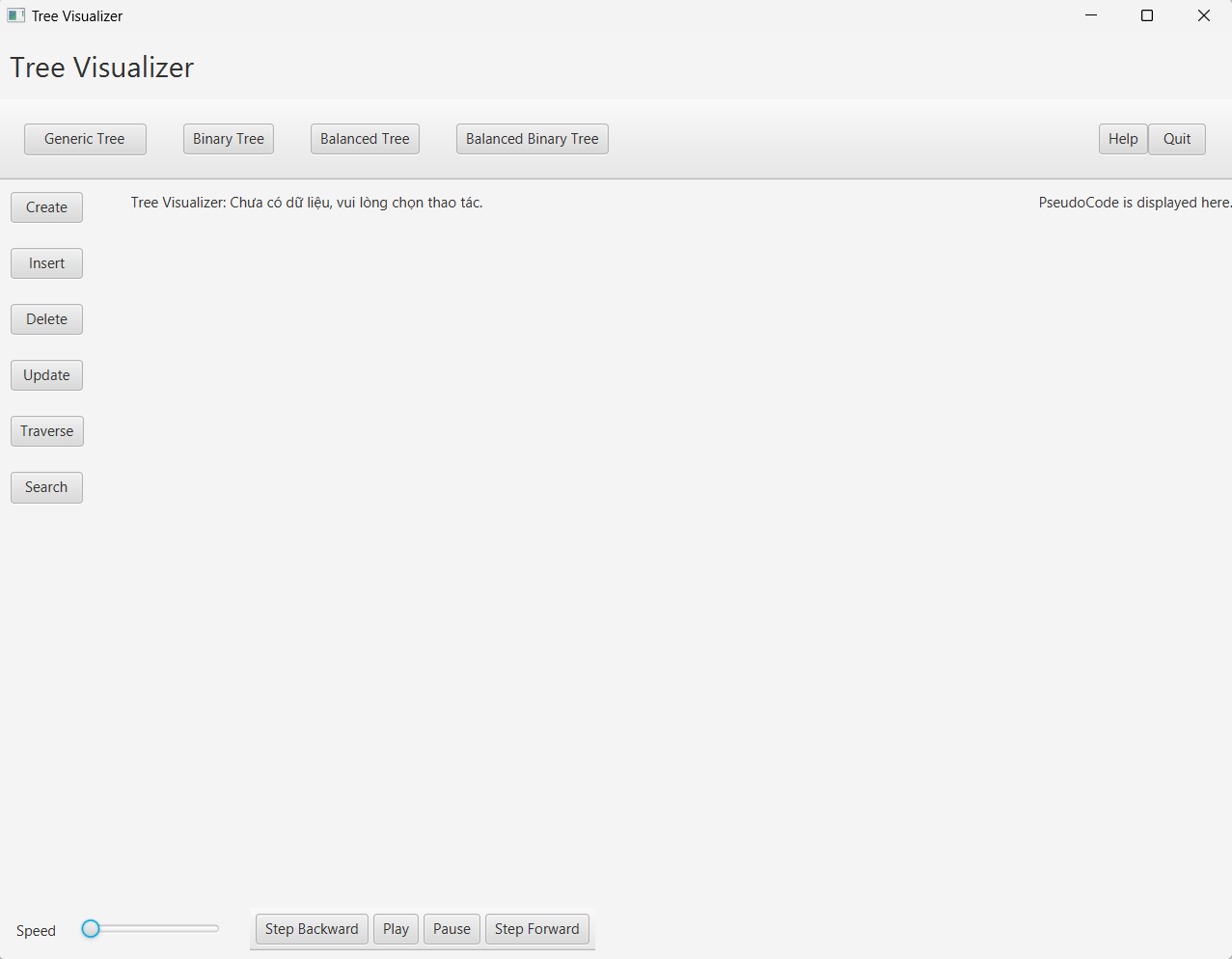
public static int showSearchDialog(): Hiển thị ra màn hình hộp thoại để người dùng nhập giá trị node muốn tìm, sau đó chọn thuật toán muốn sử dụng để thực hiện tìm kiếm (DFS hoặc BFS)

**Quan hệ**

TreeDialog - TreeVisualizerController (association): treeDialog hỗ trợ hiển thị dialog khi ấn các nút thao tác trên cây

# **CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH MINH HOẠ**

#### **Giao diện khởi động**

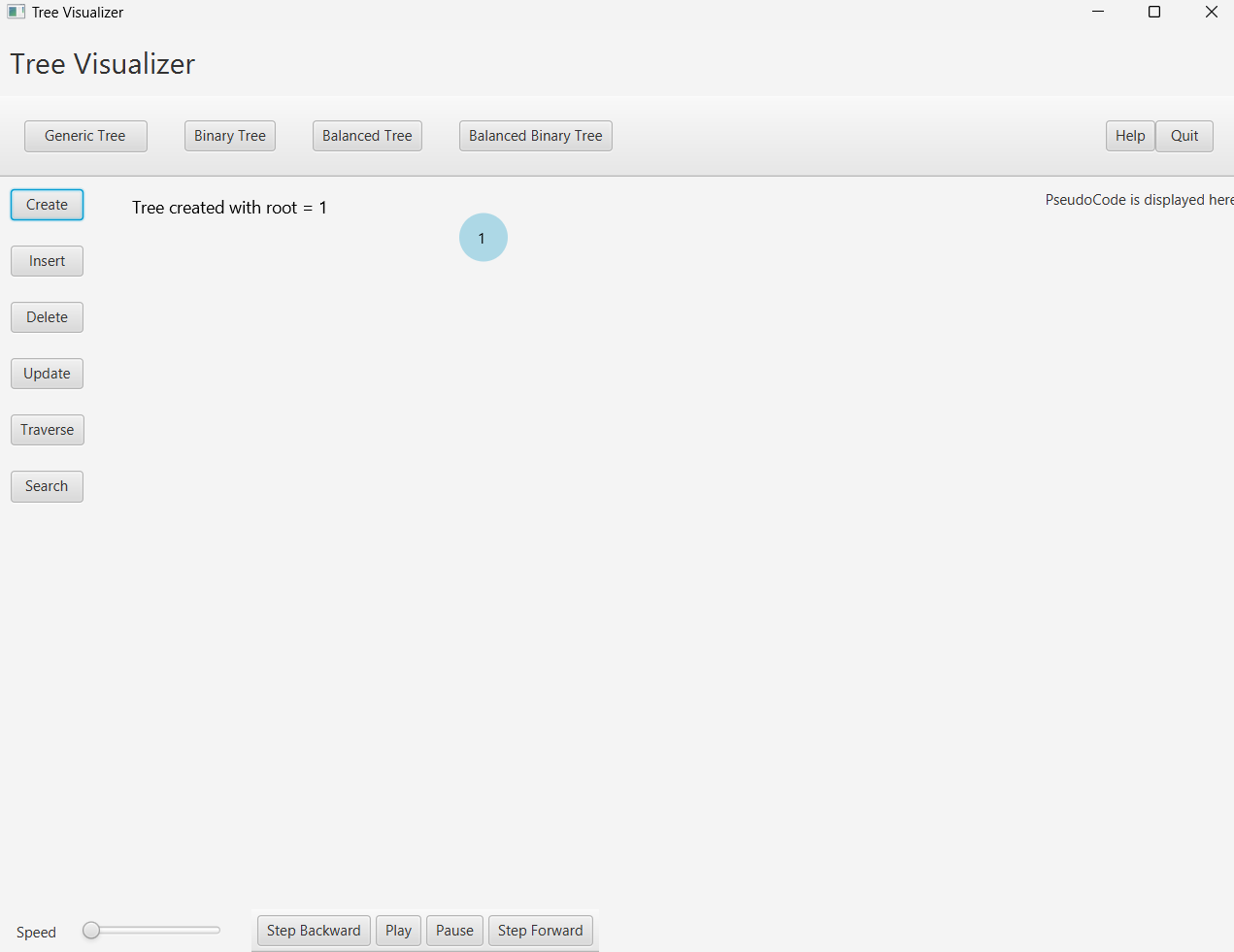
****

Đây là giao diện khởi đầu khi chạy ứng dụng. Để sử dụng, trước tiên cần chọn 1 trong 4 kiểu tree bằng các nút bấm tương ứng ở góc trên bên trái.

Sau khi đã chọn kiểu tree, có thể bắt đầu tạo tree bằng cách ấn nút Create. Một hộp thoại sẽ hiện ra, hỏi người dùng về giá trị của node gốc của tree như dưới đây

#### 

#### **Giao diện chọn cây**



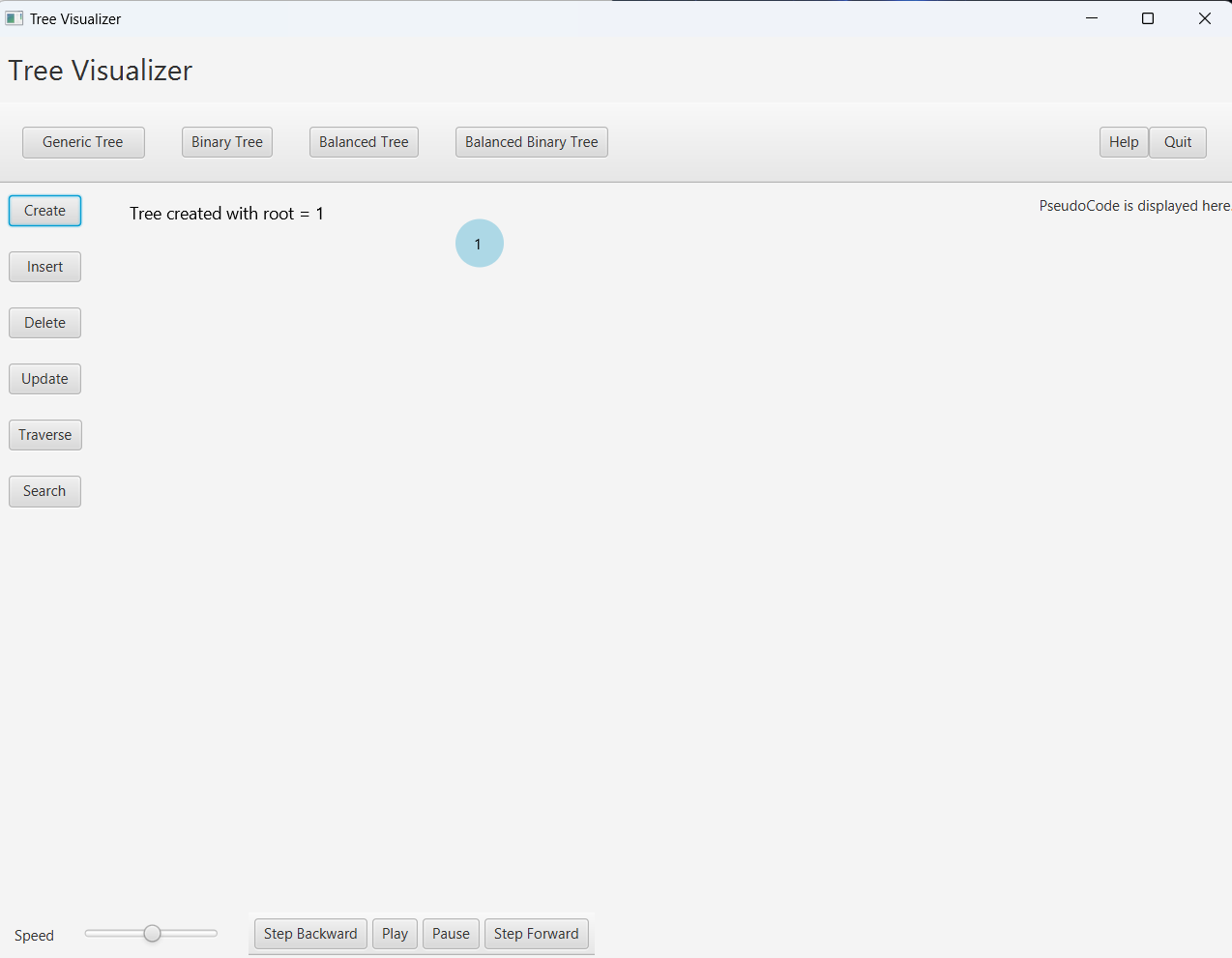
Sau khi tạo tree như trên, ta có thể bắt đầu thêm/xoá các node khỏi tree bằng cách ấn nút Insert/Delete tương ứng. Người dùng cũng có thể cập nhật trực tiếp giá trị mới cho một node bằng nút Update.

#### **Giao diện tạo cây**

#### 

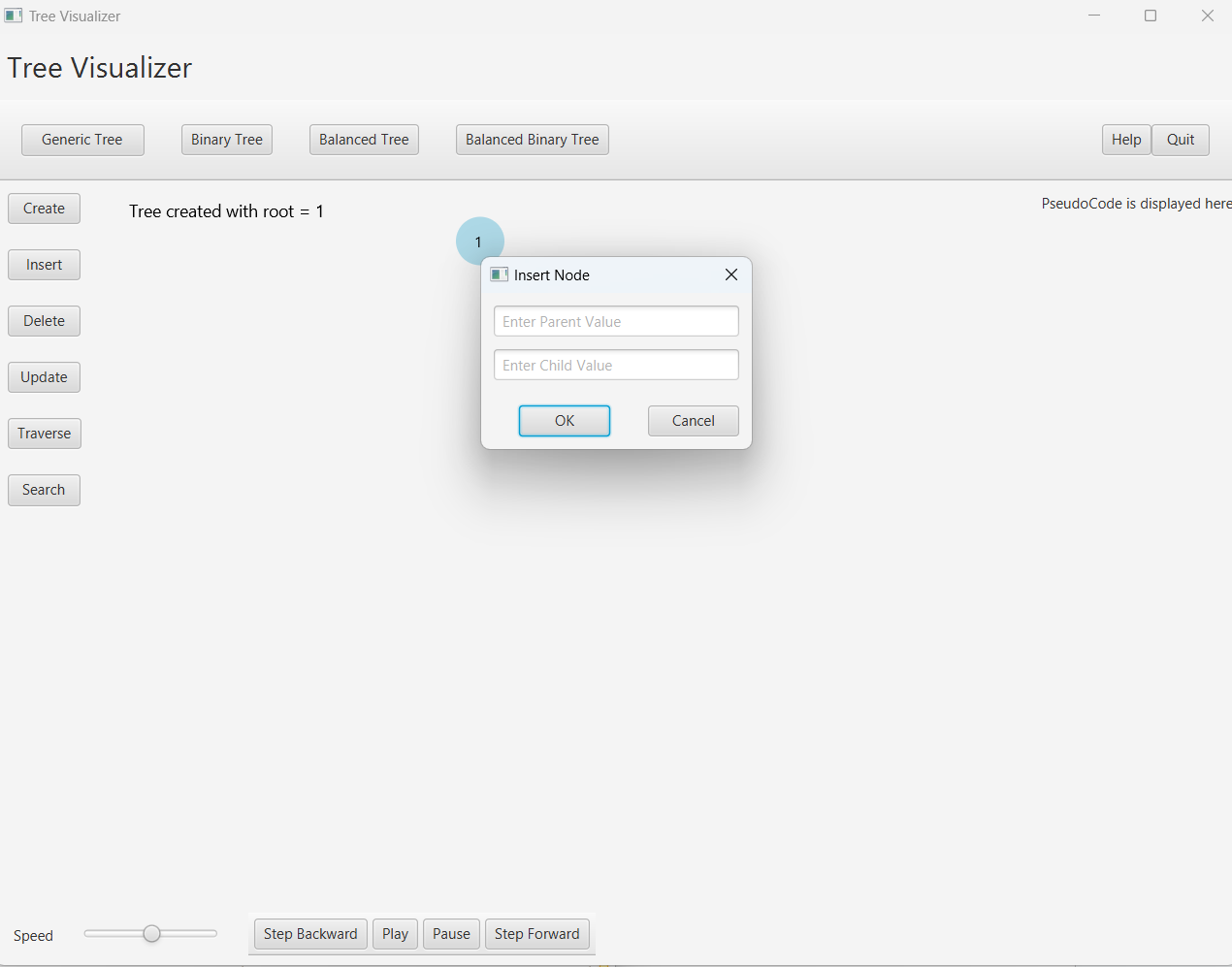
Đây là khi người dùng nhập nút “Create” màn hình sẽ hiện ra hộp thoại để khởi tạo nút gốc

#### 

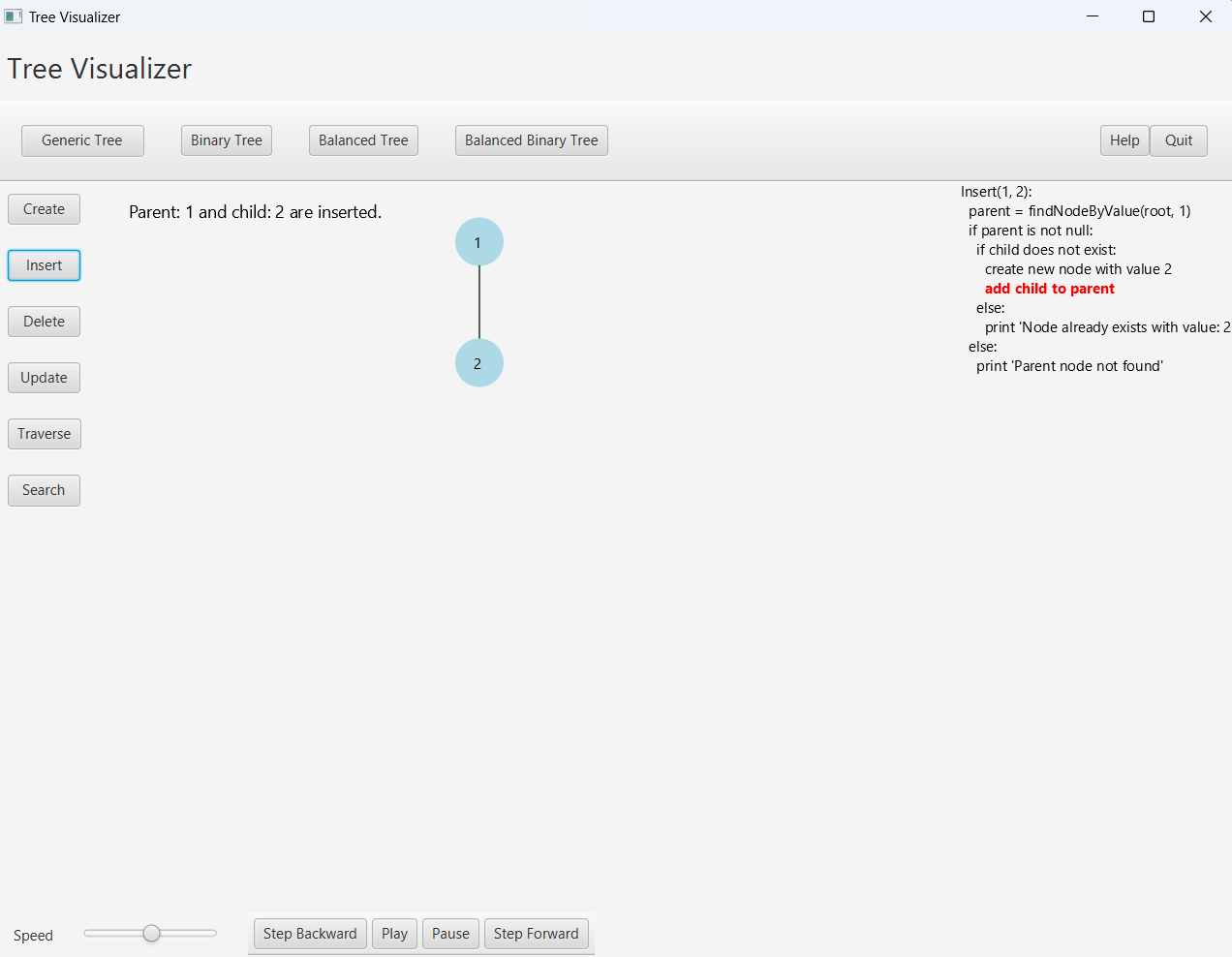


Sau khi người dùng create node gốc thì màn hình sẽ hiển thị ra cây với nốt đó cùng dòng Tree created with root = 1

#### **Giao diện thêm mới 1 nút**



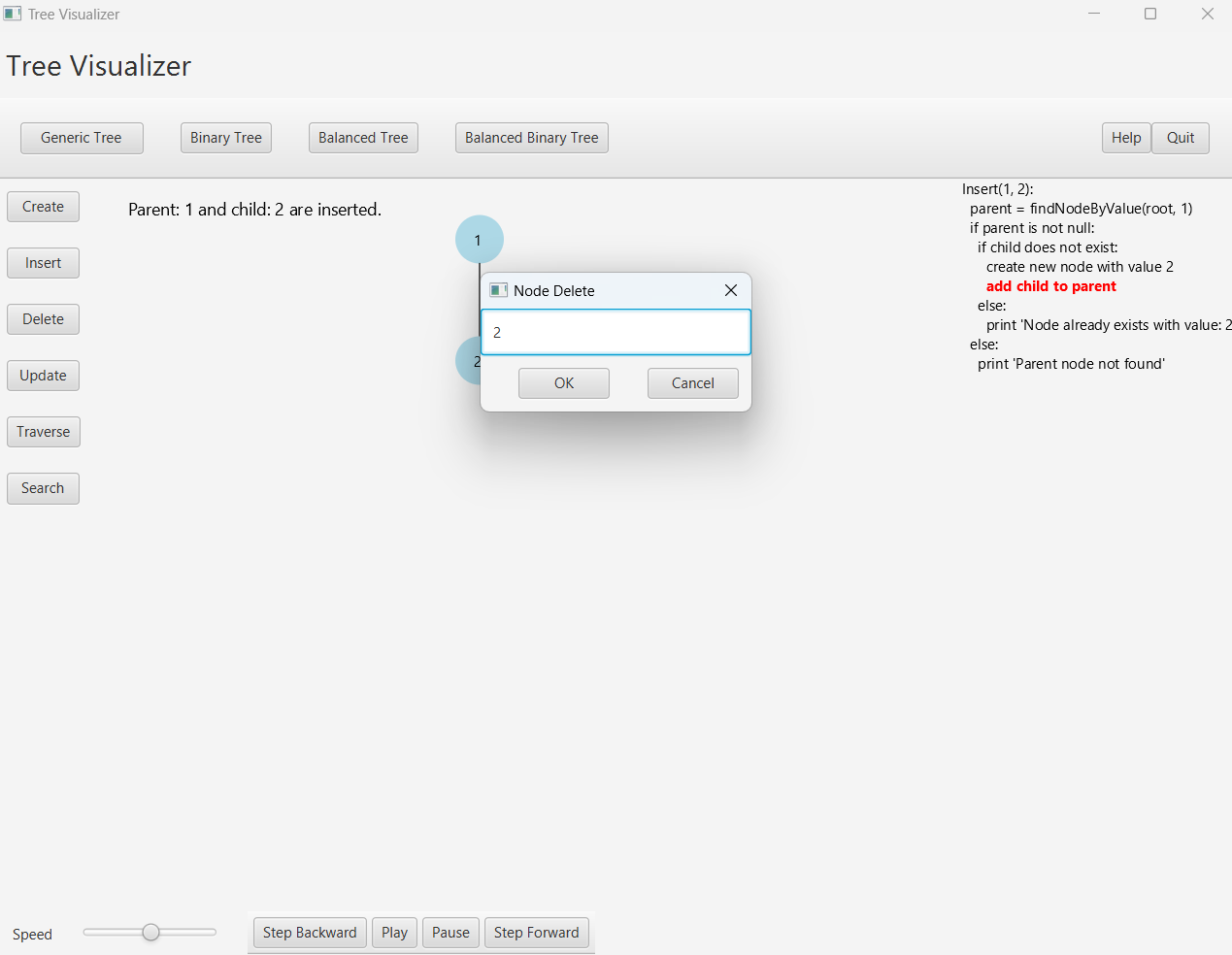
Dưới đây là hình ảnh khi người dùng nhập nút insert, sau khi nhập thì màn hình sẽ hiển thị ra thêm 1 cửa sổ để người dùng có thể nhập nút cha và nút con muốn insert.



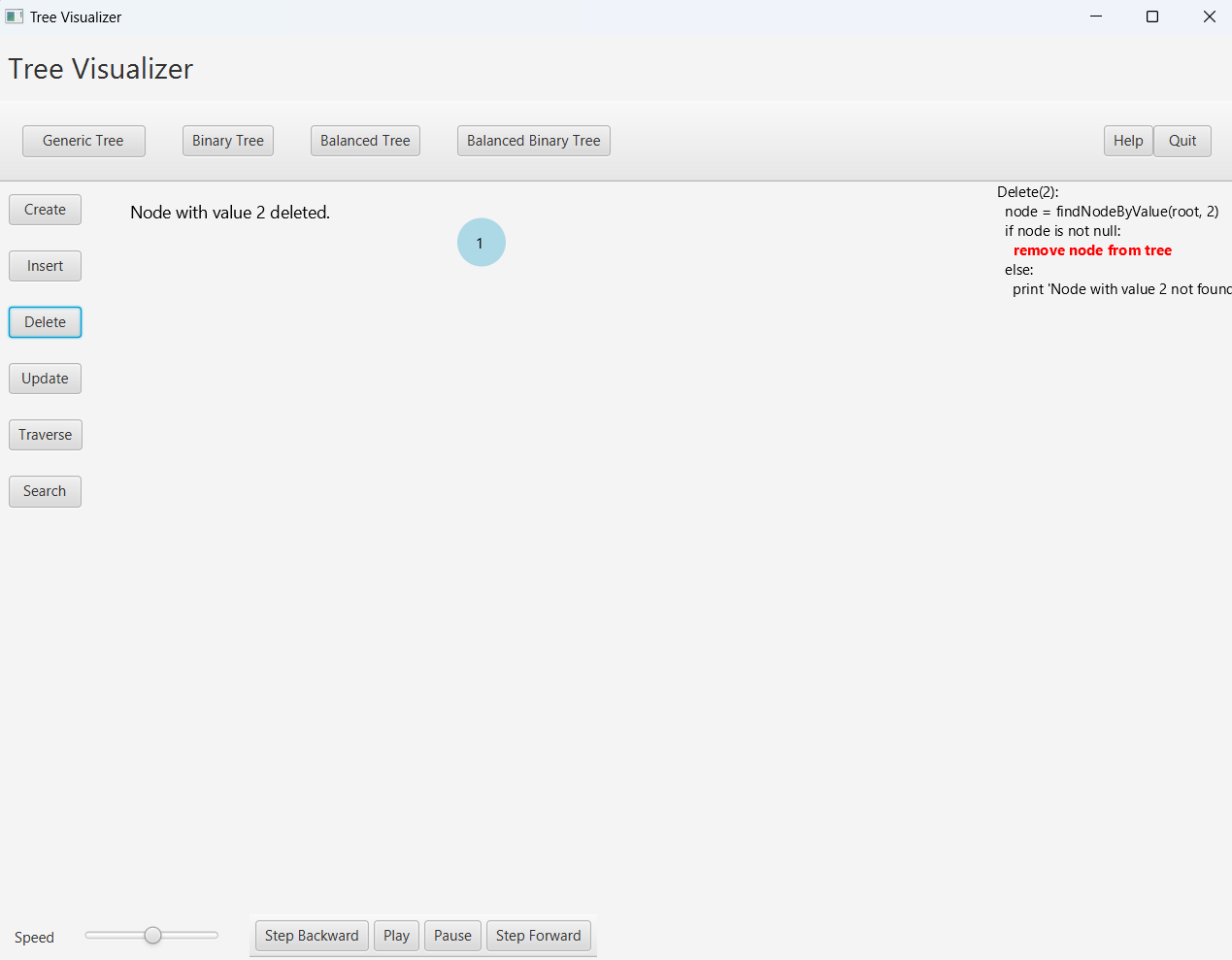
*Hình ảnh cây sau khi được thêm*

Sau khi đã nhập nút cha và con thì màn hình sẽ hiển thị ra hình ảnh cây hiện tại và thì màn hình sẽ hiển thị ra hình ảnh cây hiện tại và mã giả minh họa thuật toán sẽ bắt đầu chạy giúp người dùng dễ hình dung

#### **Giao diện xóa 1 nút**

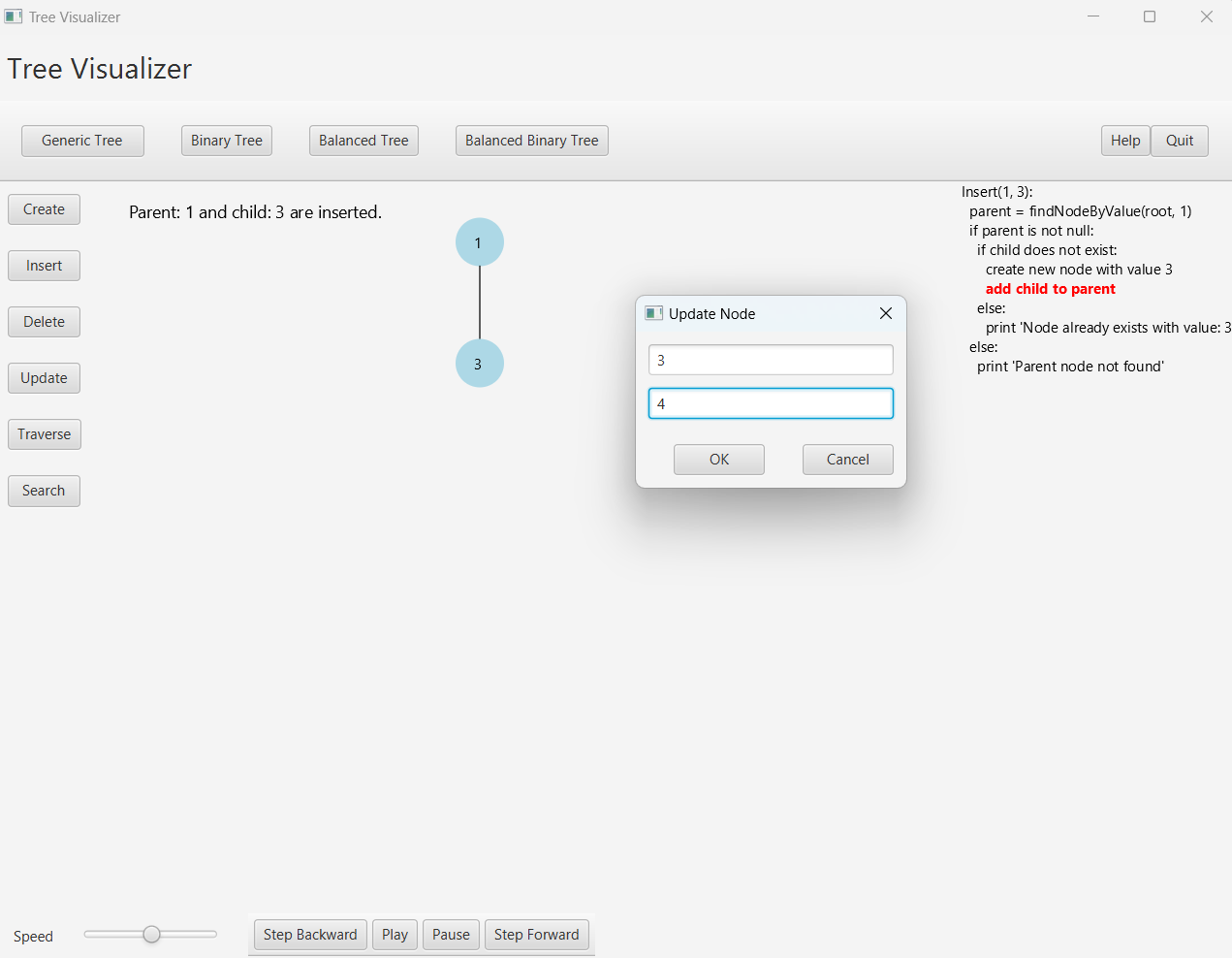


Khi người dùng nhập vào nút delete thì màn hình sẽ hiển thị cửa sổ để người dùng có thể nhập vào Note mà mình muốn delete

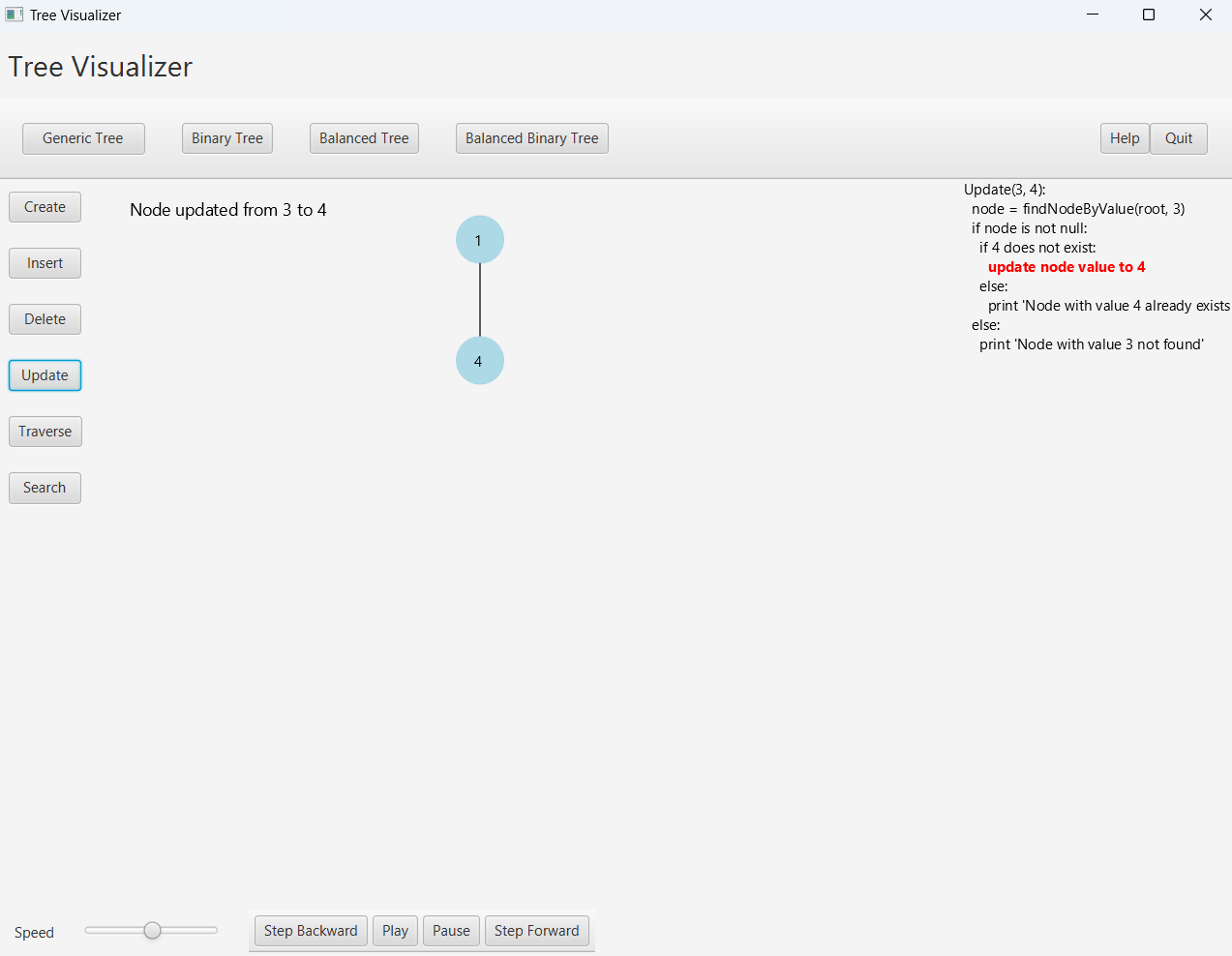


*Hình ảnh sau khi người dùng delete node 2*

#### **Giao diện cập nhật giá trị 1 nút**

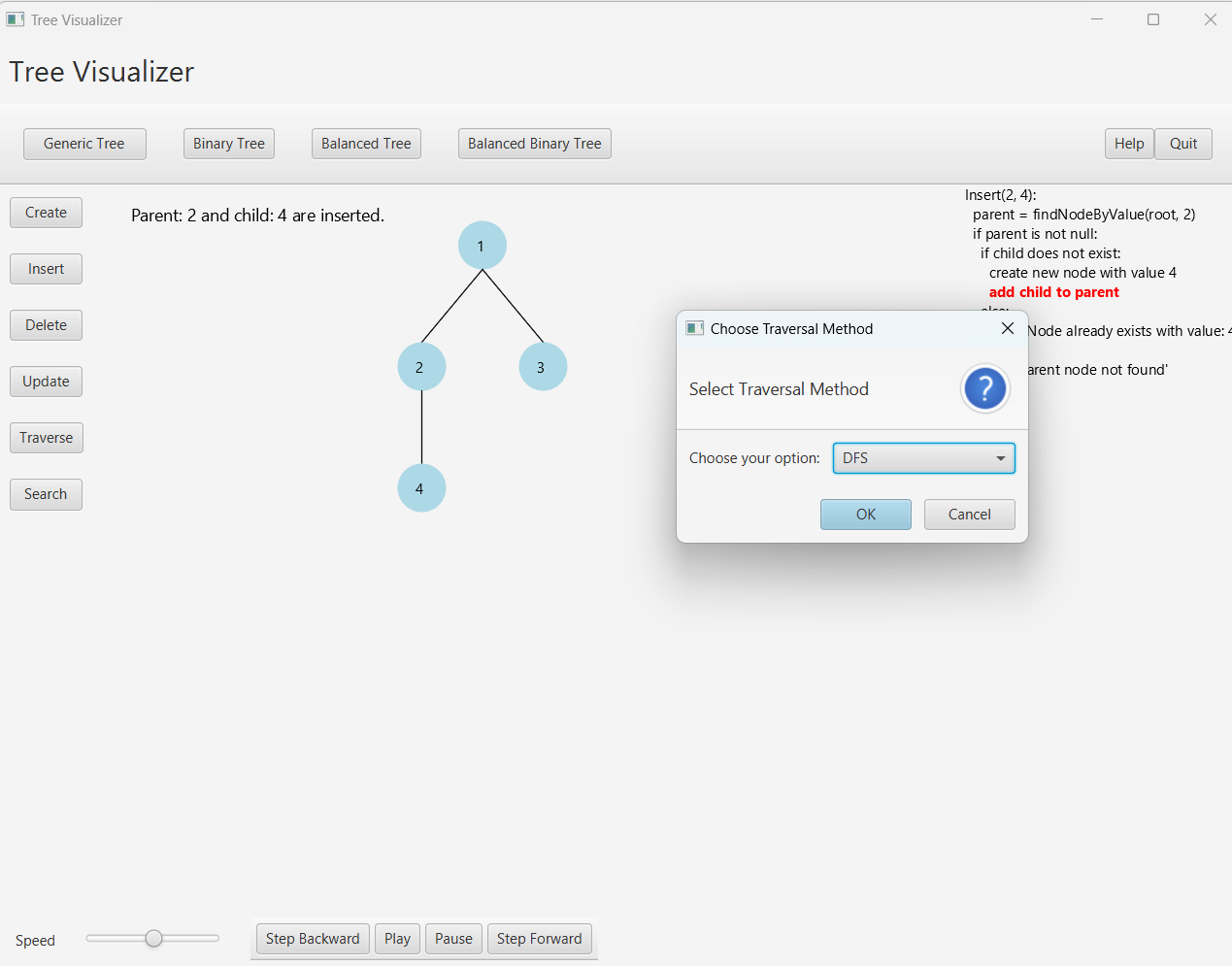


Khi người dùng muốn cập nhật một Node trong cây dữ liệu thì màn hình sẽ hiển thị ra một cửa sổ để người dùng có thể nhập giá trị của Node muốn thay thế và giá trị thay thế

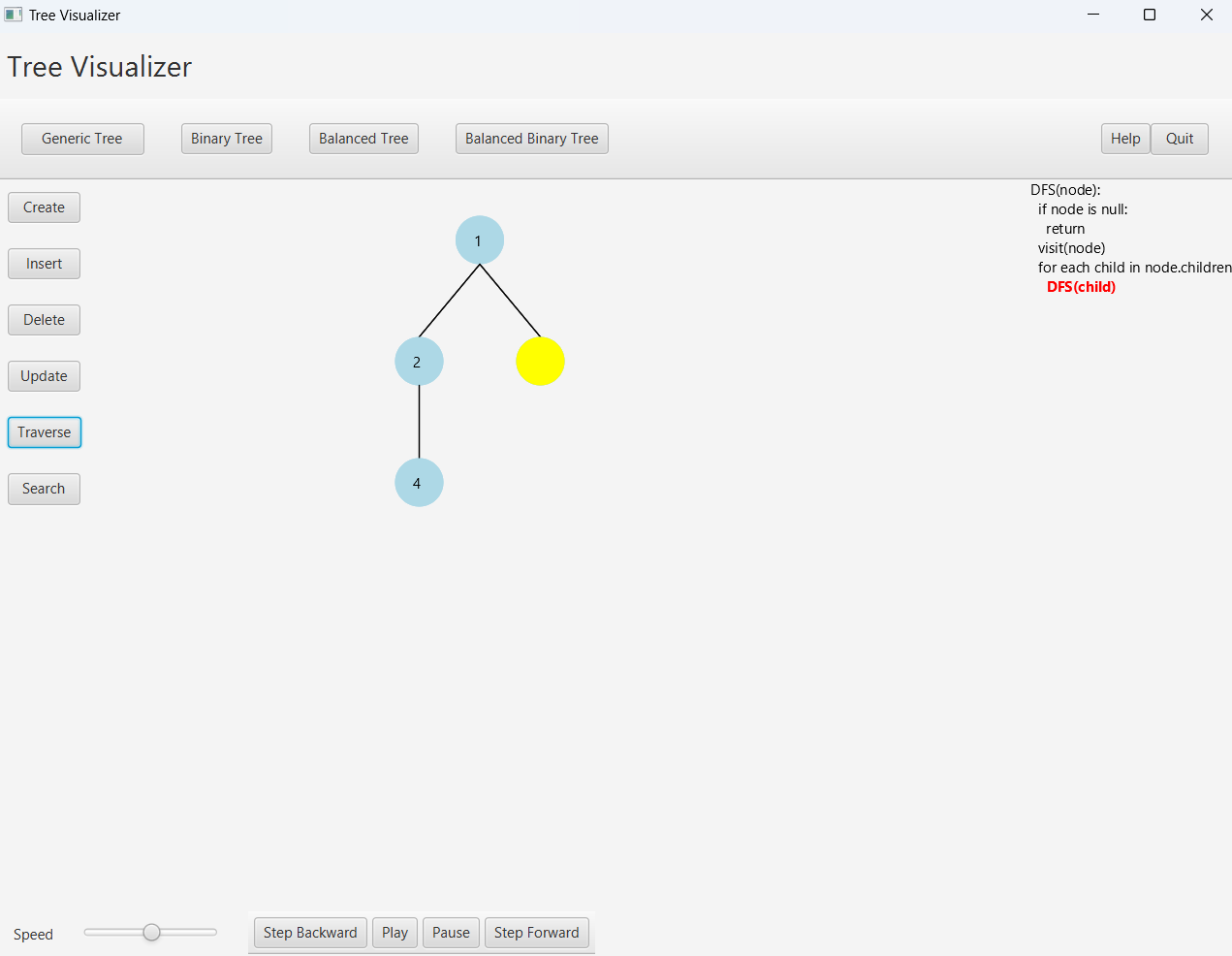


*Hình ảnh khi người dùng cập nhật Note 3 thành 4*

#### **Giao diện duyệt cây**

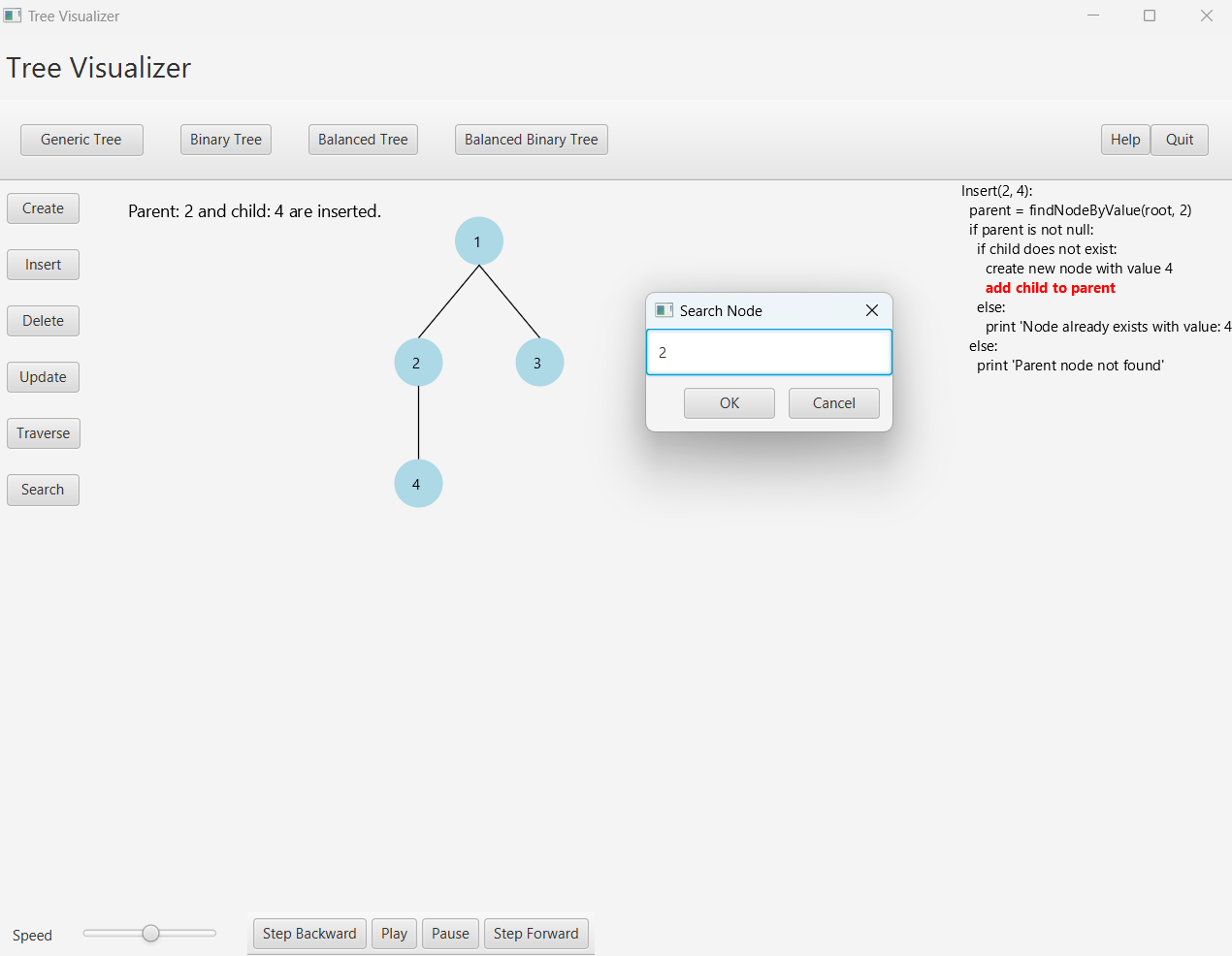


Khi người dùng nhập nút Traverse thì sẽ có cửa sổ hiển thị lên người dùng muốn sử dụng thuật toán gì để duyệt cây ở đây chúng ta có DFS và BFS

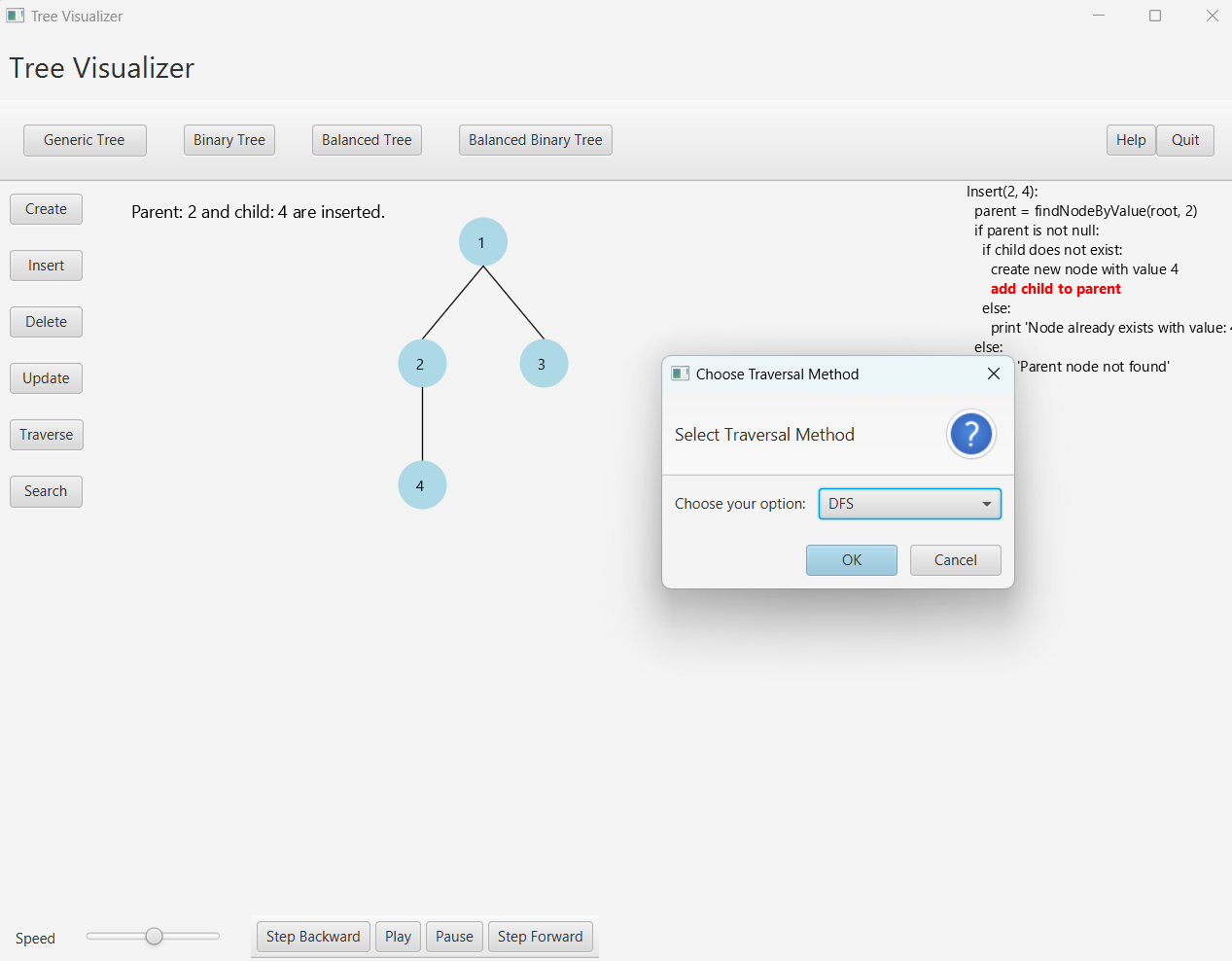


Khi người dùng chọn thuật toán để duyệt cây thì chương trình bắt đầu chạy và sẽ Highlight những Note mà nó duyệt đến cùng với mã giả ở bên tay phải sẽ bắt đầu chạy để người dùng có thể dễ hình dung ra thuật toán dễ dàng hơn

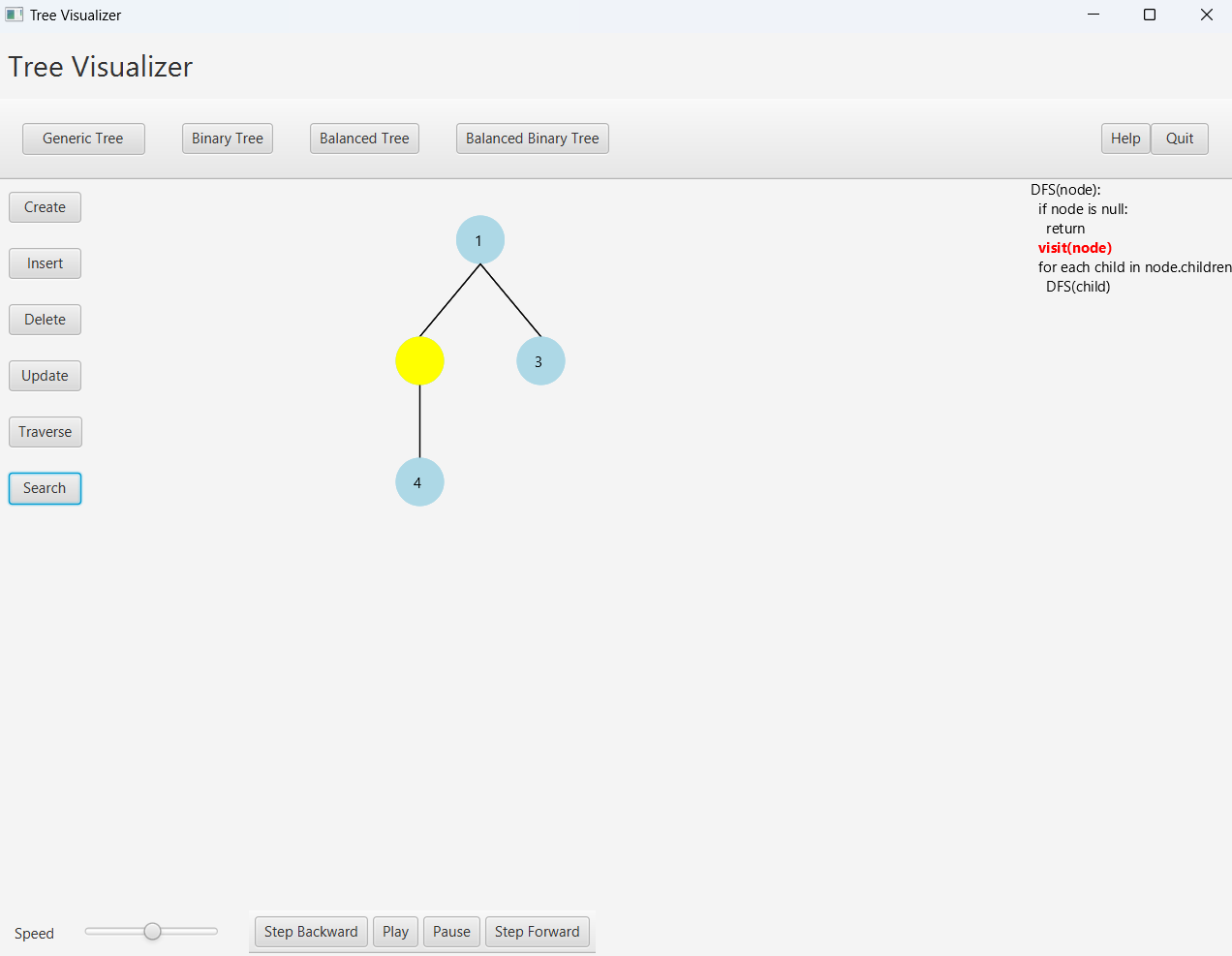
#### **Giao diện tìm kiếm 1 nút**



Trên đây hình ảnh khi dùng nhập vào nút search thì ứng dụng sẽ hiện ra cửa sổ để người dùng có thể tìm Node mình mong muốn.



Sau khi nhập Node tìm kiếm thì sẽ hiển thị ra cửa sổ để người dùng có thể chọn thuật toán mà mình muốn chọn ở đây là DFS và BFS.



Chương trình sẽ highlight Node duyệt đến cùng với đó là mã giả sẽ bắt đầu chạy để giúp người dùng hình dung dễ hơn về thuật toán