

**Binary Search Tree**

C:\Users\tdqua_000\Dropbox\SS-Slides\DeCuong-CDIO\Template CDIO v4.2\Templates\Hinh anh\LogoTruong.png

# C:\Users\tdqua_000\Desktop\checklist (1).jpg BINARY SEARCH TREE FRAMEWORK

## 

## Mô tả

**a. Tên**: Binary Search Tree Framework

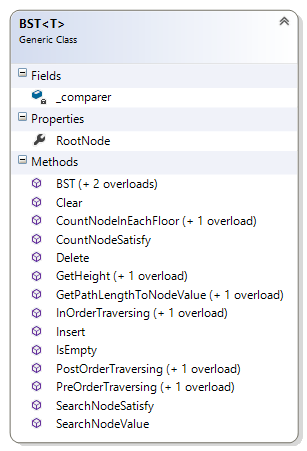
**b. Môi trường thực thi**: Ứng dụng yêu cầu chạy trên hệ điều hành Windows

**c. Mục tiêu của framework, chương trình**: Hỗ trợ việc lưu trữ, thêm, xóa, tìm kiếm, xem thông tin và nhiều thao tác khác trên Binary Search Tree, hỗ trợ giao diện đồ họa vẽ Binary Search Tree

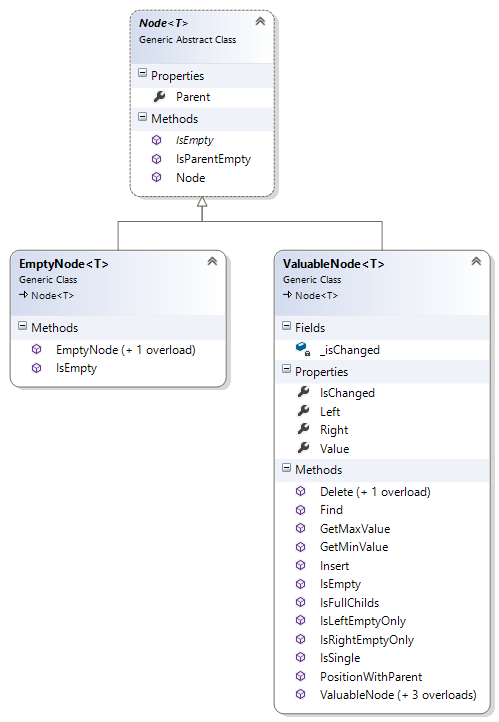
Để build được Project cần phải cài đặt thư viện DevExpress phiên bản 14.1 trở lên và tích hợp vào IDE

## Sơ đồ thiết kế framework

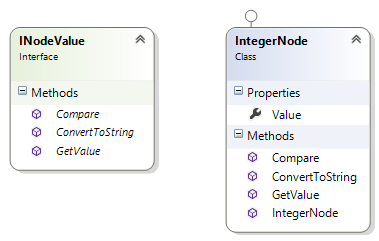
* **Class Binary Search Tree**



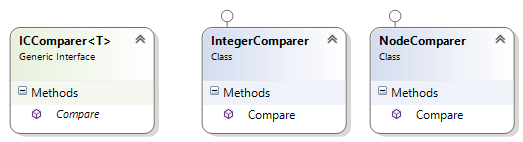
* **BST Node**



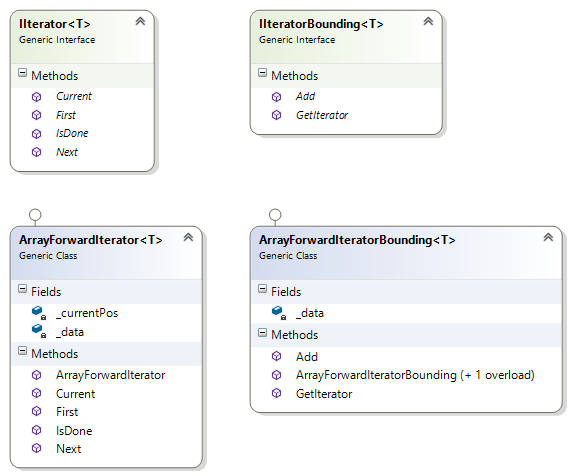
* **Node value**



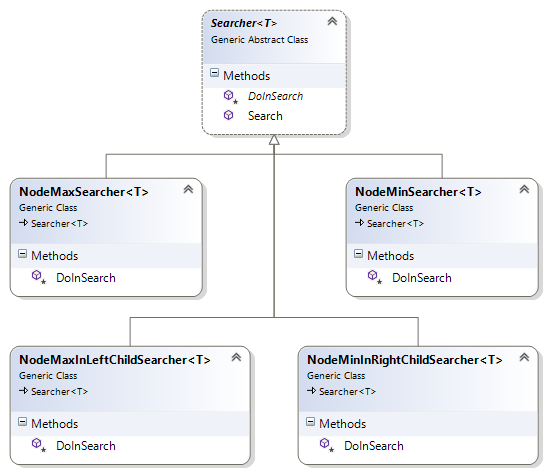
* **Comparer**



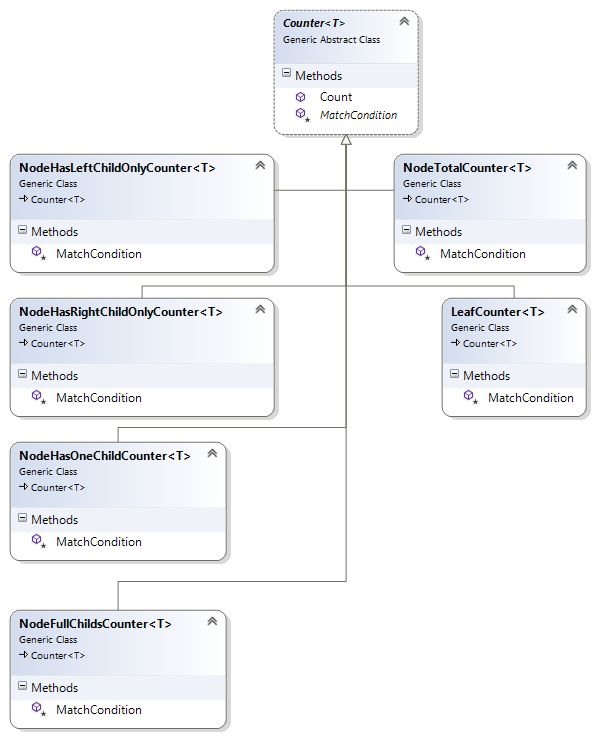
* **Hỗ trợ duyệt BST**



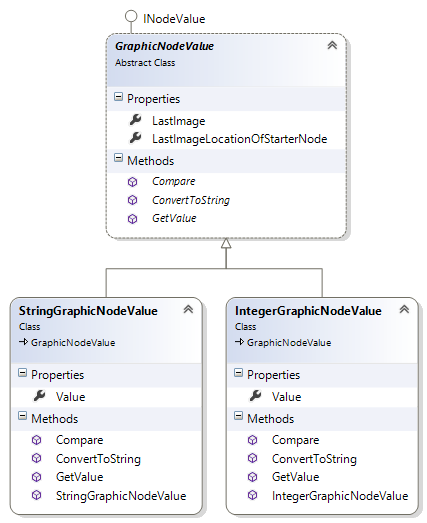
* **Hỗ trợ tìm kiếm trên BST**

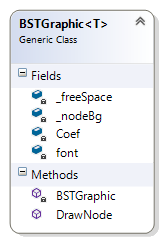


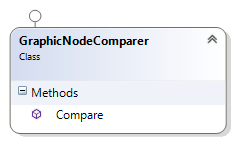
* **Hỗ trợ đếm trên BST**



* **Ứng dụng vào việc vẽ**







## Chức năng



### Xây dựng cây từ mảng số nguyên, chuỗi

* Framework cho phép tạo ra cây BST từ mảng số nguyên hoặc mảng các chuỗi:

### Duyệt cây

* Framework cung cấp 3 cách duyệt cây:
  + Left – Node – Right
  + Node – Left – Right
  + Left – Right – Node

### Xem thông tin cây

* Đềm số nút lá của cây
* Đếm số nút có một cây con
* Đếm số nút chỉ có một cây con phải
* Đếm số nút chỉ có một cây con trái
* Đếm số nút có hai cây con
* Xác định chiều cao của cây (Chiều cao của cây là tổng số nút trên đường đi từ nút gốc đến nút lá, cây rỗng có chiều cao là 0)
* Xác định số nút trên cây
* Xác định số nút trên từng mức của cây (nút gốc có mức là 0, các nút con của nút gốc có mức là 1, …)
* Xác định độ dài đường đi từ nút gốc đến nút có giá trị là x trên cây (trả về -1 nếu x không tồn tại trên cây).

### Tìm kiếm

* Tìm nút có giá trị x trên cây
* Tìm nút có giá trị nhỏ nhất trên cây
* Tìm giá trị lớn nhất trên cây
* Tìm nút có giá trị nhỏ nhất của cây con bên phải
* Tìm giá trị lớn nhất của cây con phía bên trái

## Các mẫu đã sử dụng

* Composite
* Iterator
* Strategy
* Template method
* Factory method

## Chi tiết áp dụng từng mẫu



### Composite

#### Sơ đồ thiết kế

* Mẫu được áp dụng cho việc tổ chức Node trong Binary Search Tree (BST), BST bao gồm 2 loại Node là EmptyNode và ValuableNode, cụ thể:
* Lớp abstract Node<T> : định nghĩa Node chung, với T được ràng buộc bởi interface INodeValue (ưu điểm của thiết kế này được giải thích tại mục 4. Những điểm đặc biệt trong đồ án)
* Lớp EmptyNode<T> : kế thừa Node<T>, đây là Node đặc biệt không chứa dữ liệu, tương đương với Null.
* Lớp ValuableNode<T> : kế thừa Node<T>, đây là 1 Node thực sự, chứa các Node con left và right.
* BST được biểu diễn bởi class BST<T>, và class này chứa chỉ 1 Root Node. Từ Root chúng ta dễ

dàng truy xuất đến các Node con khác trong cây (sơ đồ dưới, class này biểu diễn 1 số phương thức đại diện, xem chi tiết tại mục 2.1).



### Iterator

#### Sơ đồ thiết kế



### Factory method

#### Sơ đồ thiết kế



### Template method

#### Sơ đồ thiết kế

* Áp dụng cho việc tìm kiếm 1 Node thỏa điều kiện cho trước trong BST
* Tìm nút có giá trị nhỏ nhất trên cây
* Tìm giá trị lớn nhất trên cây
* Tìm nút có giá trị nhỏ nhất của cây con bên phải
* Tìm giá trị lớn nhất của cây con phía bên trái



* Áp dụng cho đếm số lượng Node thỏa điều kiện cho trước trong BST
* Đếm tổng số nút của cây
* Đếm số nút lá của cây
* Đếm số nút có một cây con
* Đếm số nút chỉ có một cây con phải
* Đếm số nút chỉ có một cây con trái
* Đếm số nút có hai cây con



### Strategy

#### Sơ đồ thiết kế

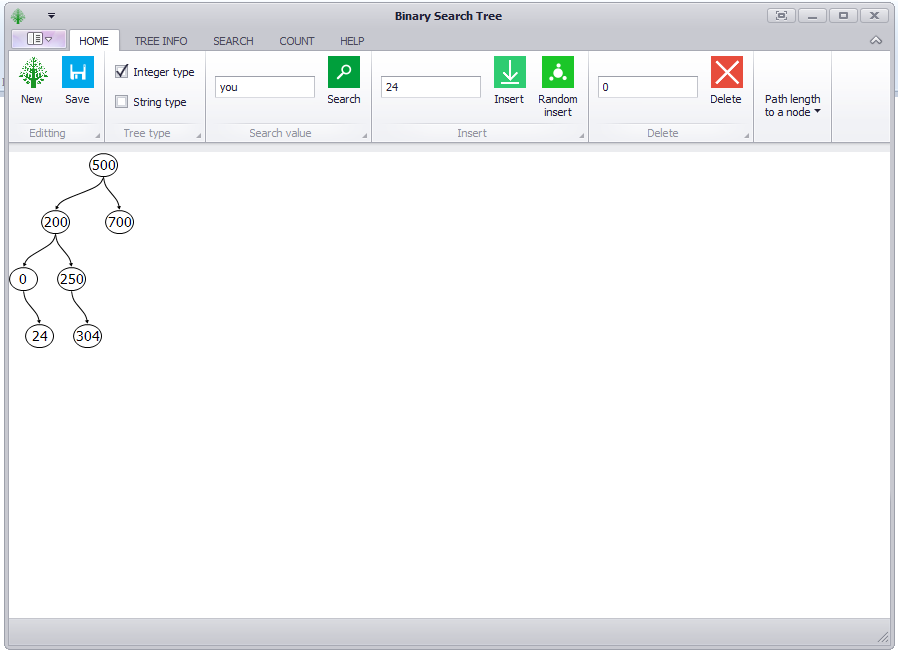


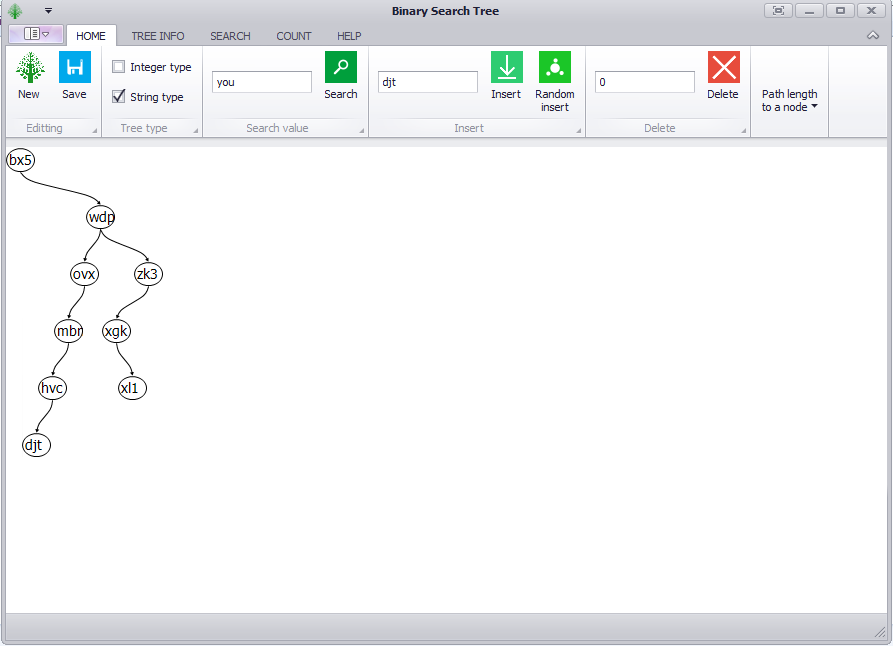


# C:\Users\tdqua_000\Desktop\checklist (1).jpg CÁC CHỨC NĂNG ỨNG DỤNG

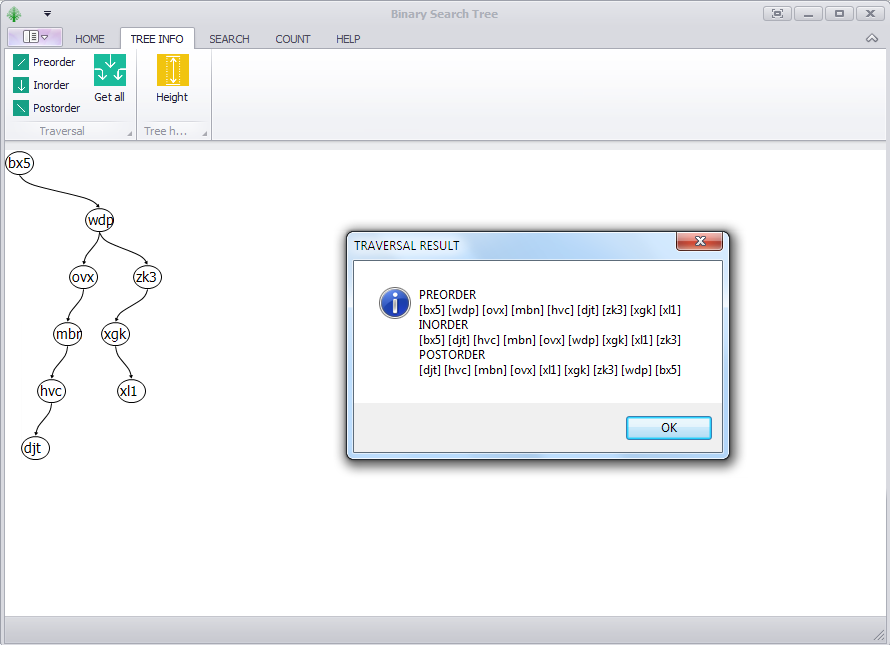
## Xây dựng cây từ mảng số nguyên, chuỗi

* Chương trình sẽ vẽ cây từ mảng số nguyên hoặc mảng các chuỗi:

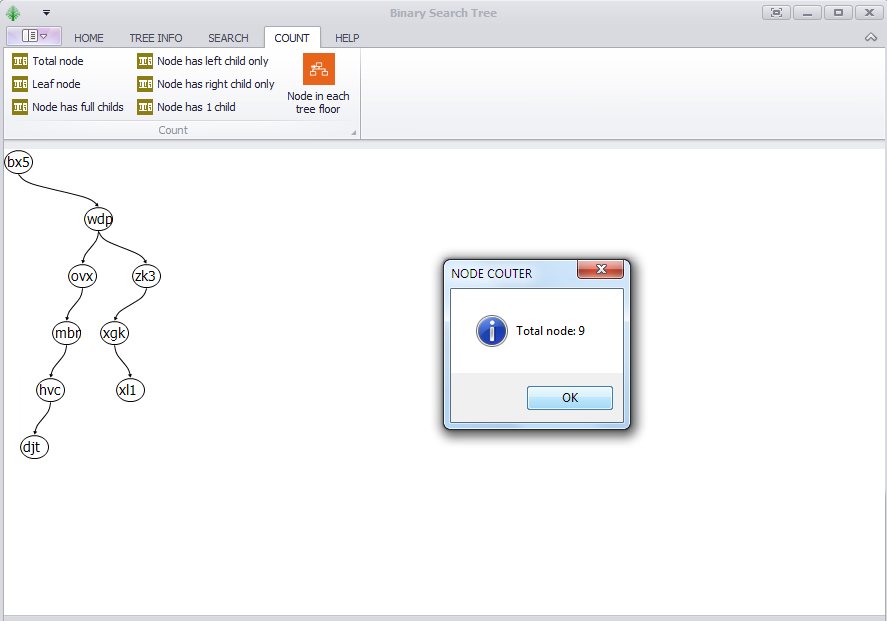
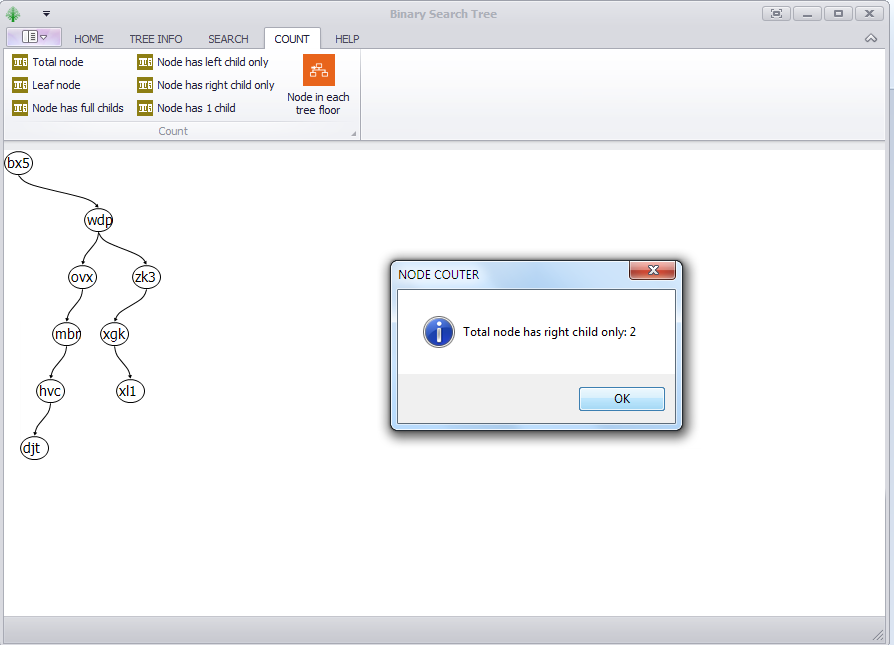


* 

## Duyệt cây

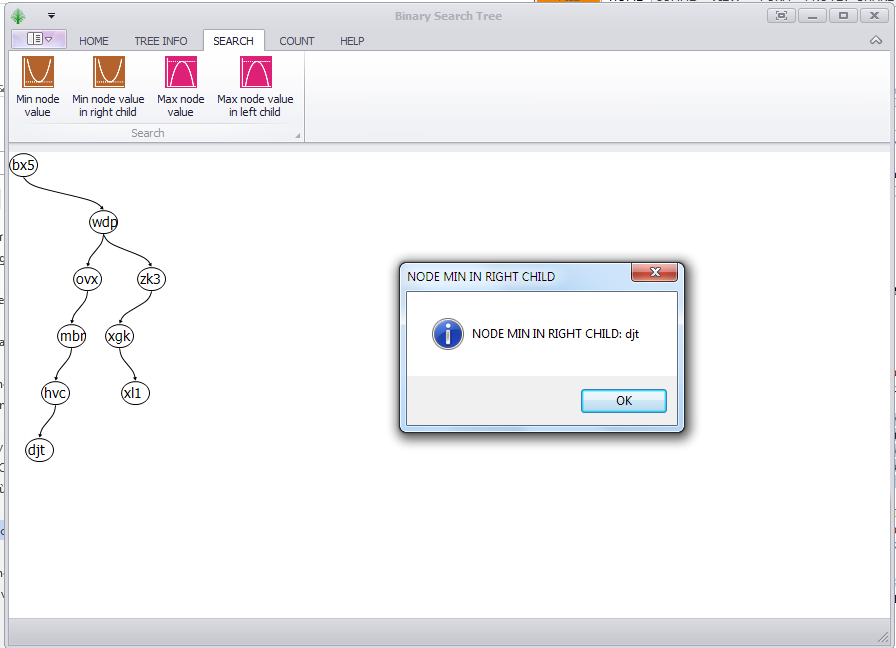
* Cung cấp 3 cách duyệt cây:
  + Left – Node – Right
  + Node – Left – Right
  + Left – Right – Node
* 

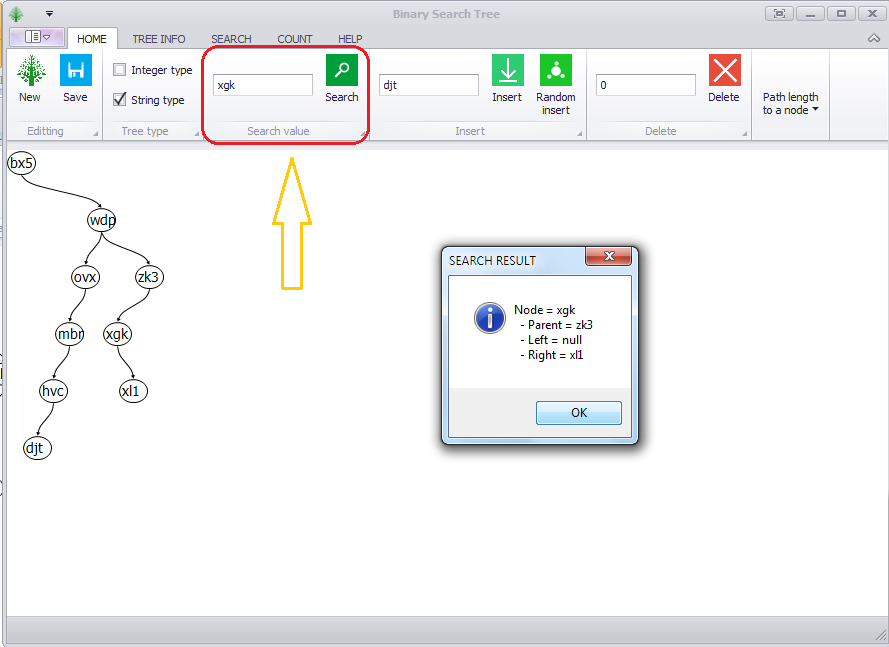
## Xem thông tin cây

* Đềm số nút lá của cây
* Đếm số nút có một cây con
* Đếm số nút chỉ có một cây con phải
* Đếm số nút chỉ có một cây con trái
* Đếm số nút có hai cây con
* Xác định chiều cao của cây (Chiều cao của cây là tổng số nút trên đường đi từ nút gốc đến nút lá, cây rỗng có chiều cao là 0)
* Xác định số nút trên cây
* Xác định số nút trên từng mức của cây (nút gốc có mức là 0, các nút con của nút gốc có mức là 1, …)
* Xác định độ dài đường đi từ nút gốc đến nút có giá trị là x trên cây (trả về -1 nếu x không tồn tại trên cây).
* 
* 

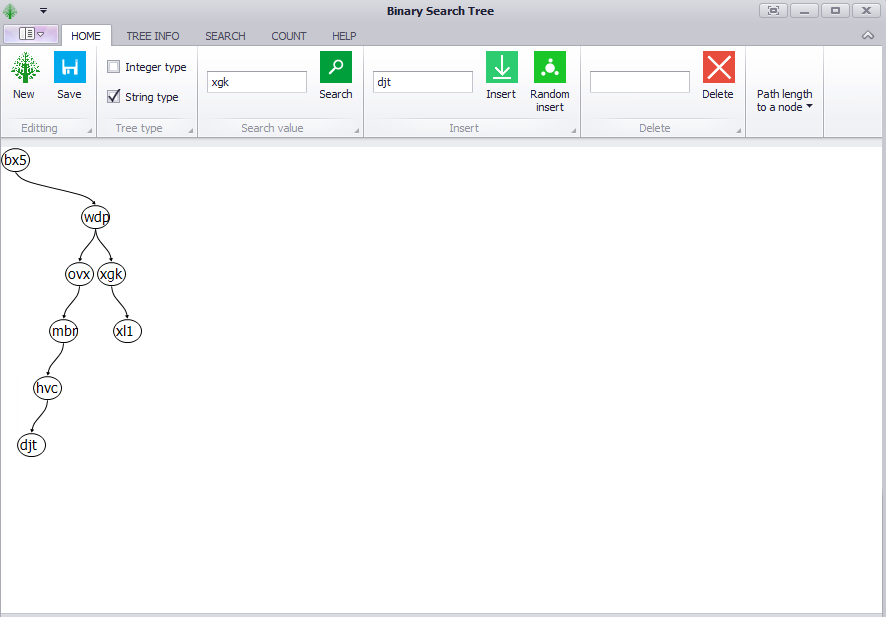
## Tìm kiếm

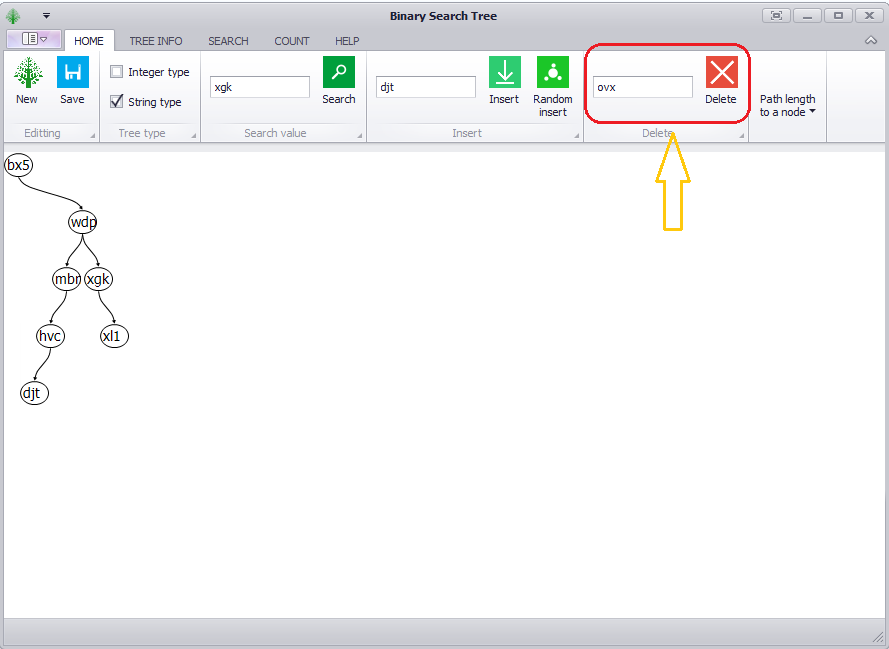
* Tìm nút có giá trị x trên cây
* Tìm nút có giá trị nhỏ nhất trên cây
* Tìm giá trị lớn nhất trên cây
* Tìm nút có giá trị nhỏ nhất của cây con bên phải
* Tìm giá trị lớn nhất của cây con phía bên trái



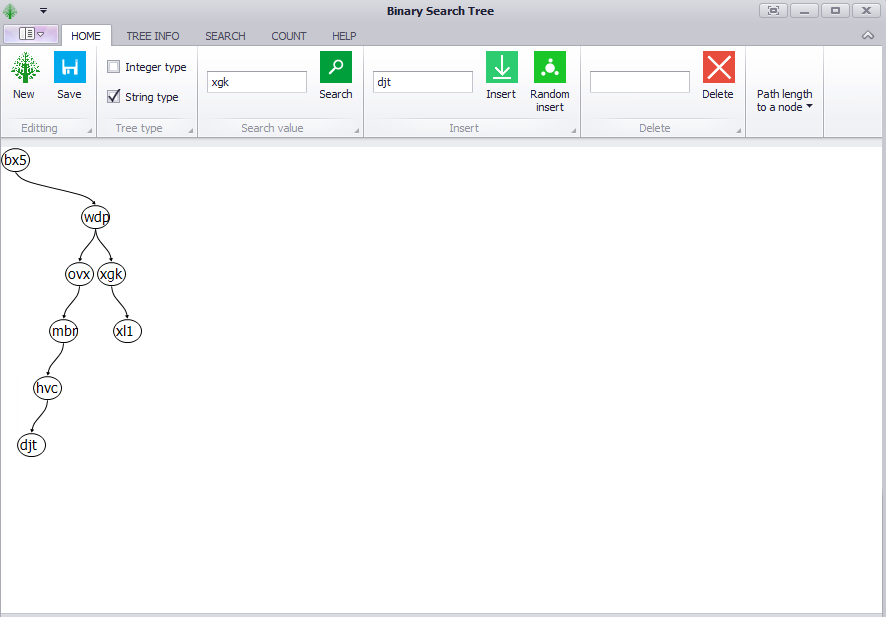
* 

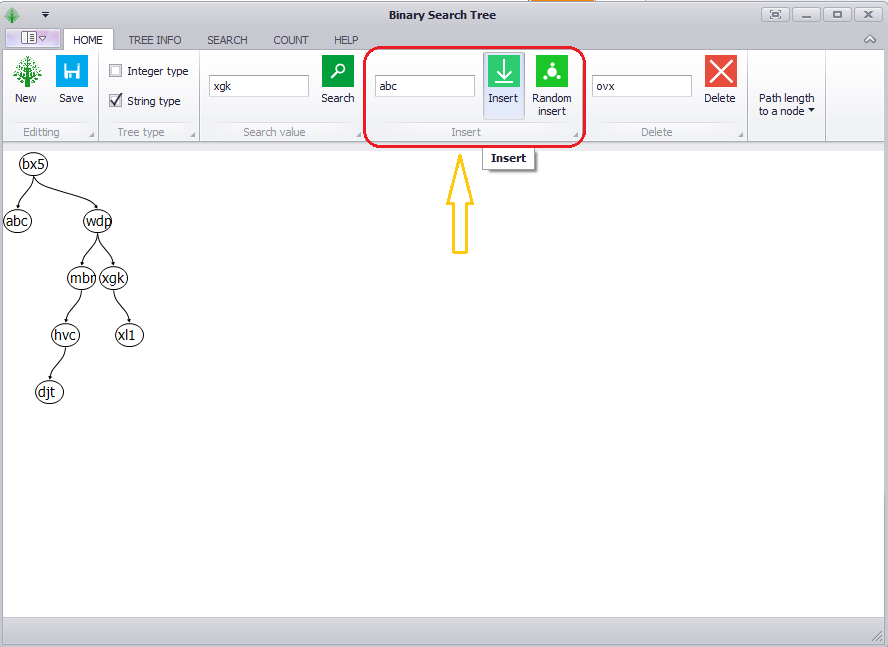
## Xóa một nút khỏi cây





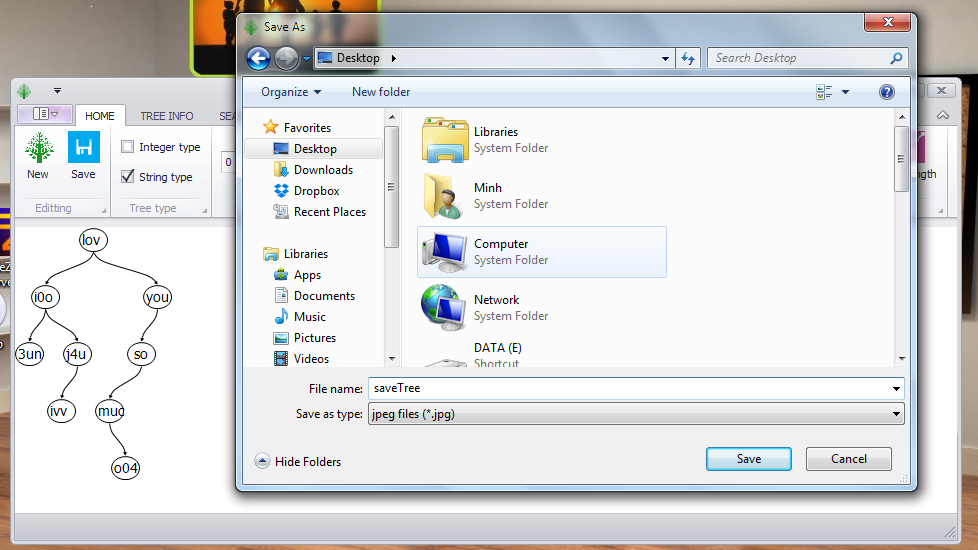
## Thêm một nút vào cây

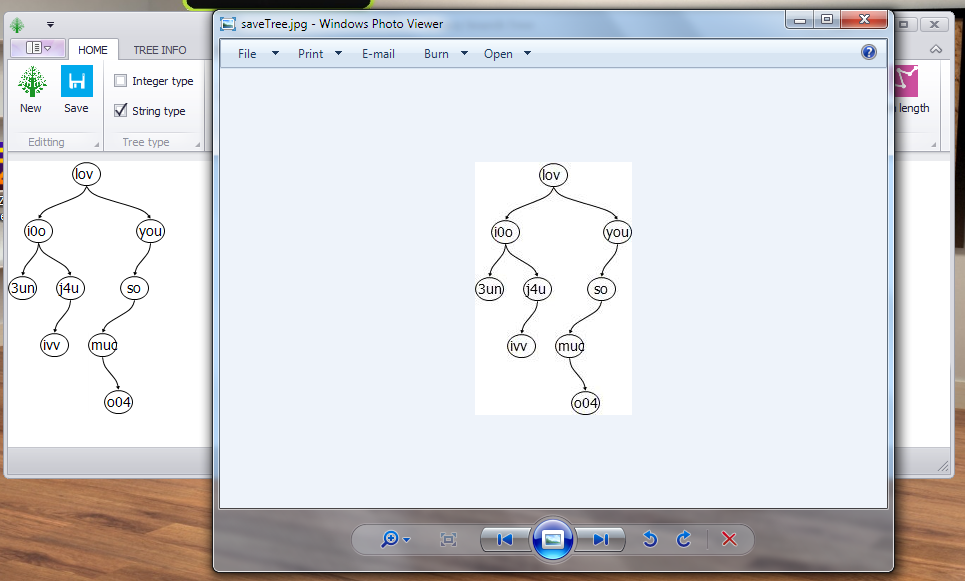
* Ứng dụng cung cấp 2 cách thêm nút vào cây:
  + Thêm bằng cách nhập giá trị của nút vào
  + Ứng dụng sẽ tự random giá trị của nút để thêm vào cây.
* 



## Lưu ảnh của cây

* Ứng dụng cho phép lưu ảnh của cây đang duyệt.





# C:\Users\tdqua_000\Desktop\exercise-clipart.jpg NHỮNG ĐIỂM ĐẶC BIỆT

## Xây dựng BST với nhiều kiểu dữ liệu khác nhau

* Khi xây dựng class BST<T>, chúng ta ràng buộc T phải có 1 số đặc tính nhất định (implement <<interface>> INodeValue), cụ thể:
* Lấy giá trị của T: GetValue().
* So sánh giá trị của T với đối tượng khác cùng kiểu : Compare(val2 : INodeValue).
* Chuyển đổi giá trị của T sang chuổi : ConvertToString().
* Với thiết kế này chúng ta sẽ dễ dàng mở rộng kiểu của BST bằng việc custom lại T, cụ thể ứng dụng đã xây dựng được BST kiểu integer và kiểu string, đồng thời cũng hỗ trợ luôn cho việc đồ họa (tạo ra class GraphicNodeValue)



## Ứng dụng cho phép lưu hình ảnh

Ứng dụng cho phép lưu BST đã vẽ dưới dạng hình ảnh (Xem thêm mục 3.7)

# C:\Users\tdqua_000\Desktop\119498595413718994help-books-aj.svg_aj_ash_01.svg.hi.png CÁC THAM KHẢO

**Mô tả**

Lý thuyết về Binary Search Tree

<https://en.wikipedia.org/wiki/Binary_search_tree>

Xây dựng cơ bản với Binary Search Tree

<https://msdn.microsoft.com/en-us/library/ms379572%28v=vs.80%29.aspx>

MỤC LỤC

[1 BINARY SEARCH TREE FRAMEWORK 2](#_Toc10995031)

[1.1 2](#_Toc10995032)

[Mô tả 2](#_Toc10995033)

[1.2 Sơ đồ thiết kế framework 2](#_Toc10995034)

[*1.3* Chức năng 10](#_Toc10995035)

[1.3.1 Xây dựng cây từ mảng số nguyên, chuỗi 10](#_Toc10995040)

[1.3.2 Duyệt cây 10](#_Toc10995041)

[1.3.3 Xem thông tin cây 10](#_Toc10995042)

[1.3.4 Tìm kiếm 11](#_Toc10995043)

[*1.4* Các mẫu đã sử dụng 11](#_Toc10995044)

[1.5 Chi tiết áp dụng từng mẫu 11](#_Toc10995045)

[1.5.1 Composite 11](#_Toc10995048)

[1.5.2 Iterator 13](#_Toc10995049)

[1.5.3 Factory method 13](#_Toc10995050)

[1.5.4 Template method 14](#_Toc10995051)

[1.5.5 Strategy 17](#_Toc10995052)

[2 CÁC CHỨC NĂNG ỨNG DỤNG 18](#_Toc10995058)

[2.1 Xây dựng cây từ mảng số nguyên, chuỗi 18](#_Toc10995059)

[2.2 Duyệt cây 20](#_Toc10995060)

[2.3 Xem thông tin cây 21](#_Toc10995061)

[2.4 Tìm kiếm 23](#_Toc10995062)

[2.5 Xóa một nút khỏi cây 26](#_Toc10995063)

[2.6 Thêm một nút vào cây 27](#_Toc10995064)

[2.7 Lưu ảnh của cây 29](#_Toc10995065)

[3 NHỮNG ĐIỂM ĐẶC BIỆT 31](#_Toc10995066)

[3.1 Xây dựng BST với nhiều kiểu dữ liệu khác nhau 31](#_Toc10995067)

[3.2 Ứng dụng cho phép lưu hình ảnh 32](#_Toc10995068)

[4 CÁC THAM KHẢO 33](#_Toc10995069)