



(http://hoclaptrinh.vn)

[Home \(http://hoclaptrinh.vn\)](http://hoclaptrinh.vn) / [Tutorial \(http://hoclaptrinh.vn/tutorial\)](http://hoclaptrinh.vn/tutorial)/ [Cấu trúc dữ liệu & giải thuật \(55 bài\) \(http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai\)](http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai)/ [Cây tìm kiếm nhị phân - Binary Search Tree](#)[Tất cả \(http://hoclaptrinh.vn\)](http://hoclaptrinh.vn)[Đã giải quyết \(http://hoclaptrinh.vn/da-giai-quyet\)](http://hoclaptrinh.vn/da-giai-quyet)[Html \(http://hoclaptrinh.vn/tags/html\)](http://hoclaptrinh.vn/tags/html)[Javascript \(http://hoclaptrinh.vn/tags/javascript\)](http://hoclaptrinh.vn/tags/javascript)[Laravel \(http://hoclaptrinh.vn/tags/laravel\)](http://hoclaptrinh.vn/tags/laravel)[NodeJS \(http://hoclaptrinh.vn/tags/nodejs\)](http://hoclaptrinh.vn/tags/nodejs)[MySQL \(http://hoclaptrinh.vn/tags/mysql\)](http://hoclaptrinh.vn/tags/mysql)[MongoDB \(http://hoclaptrinh.vn/tags/mongodb\)](http://hoclaptrinh.vn/tags/mongodb)[My Feed \(http://hoclaptrinh.vn/chu-de-toi-quan-tam\)](http://hoclaptrinh.vn/chu-de-toi-quan-tam)[Danh mục \(http://hoclaptrinh.vn/tags\)](http://hoclaptrinh.vn/tags)

Danh mục của bạn

[CSS \(http://hoclaptrinh.vn/tags/css\)](http://hoclaptrinh.vn/tags/css)[PHP \(http://hoclaptrinh.vn/tags/php\)](http://hoclaptrinh.vn/tags/php)[AngularJS \(http://hoclaptrinh.vn/tags/angularjs\)](http://hoclaptrinh.vn/tags/angularjs)[VueJS \(http://hoclaptrinh.vn/tags/vuejs\)](http://hoclaptrinh.vn/tags/vuejs)[C \(http://hoclaptrinh.vn/tags/c\)](http://hoclaptrinh.vn/tags/c)[Docker \(http://hoclaptrinh.vn/tags/docker\)](http://hoclaptrinh.vn/tags/docker)

« Previous Lesson

<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/duyet-cay-tree-traversal>

Next Lesson »

<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cay-avl-avl-tree>

## Cây tìm kiếm nhị phân - Binary Search Tree

### Cây tìm kiếm nhị phân là gì ?

Một cây tìm kiếm nhị phân (Binary Search Tree – viết tắt là BST) là một cây mà trong đó tất cả các nút đều có các đặc điểm sau:

- Cây con bên trái của một nút có khóa (key) nhỏ hơn hoặc bằng giá trị khóa của nút cha (của cây con này).
- Cây con bên phải của một nút có khóa lớn hơn hoặc bằng giá trị khóa của nút cha (của cây con này).

Vì thế có thể nói rằng, một cây tìm kiếm nhị phân (BST) phân chia tất cả các cây con của nó thành hai phần: *cây con bên trái* và *cây con bên phải* và có thể được định nghĩa như sau:



left\_subtree (keys) <big></big> node (http://hoclaptrinh.vn) <big></big> right\_subtree (keys)

```Biểu diễn cây tìm kiếm nhị phân (BST)

Cây tìm kiếm nhị phân (BST) là một tập hợp bao gồm các nút được sắp xếp theo cách để chúng có thể duy trì hoặc tuân theo các đặc điểm của cây tìm kiếm nhị phân. Mỗi một nút thì đều có một khóa và giá trị liên kết với nó. Trong khi tìm kiếm, khóa cần tìm được so sánh với các khóa trong cây tìm kiếm nhị phân (BST) và nếu tìm thấy, giá trị liên kết sẽ được thu nhận.

Ví dụ một cây tìm kiếm nhị phân (BST):

![Cây tìm kiếm nhị phân (Binary Search Tree)](../cau-truc-du-lieu-va-giai-thuat/images/binary\_search\_tree.jpg) Từ hình ví dụ minh họa trên ta thấy rằng, khóa của nút gốc có giá trị 27 và tất cả khóa bên trái của cây con bên trái đều có giá trị nhỏ hơn 27 này và tất cả các khóa bên phải của cây con bên phải đều có giá trị lớn hơn 27.

Hoạt động cơ bản trên cây tìm kiếm nhị phân

Dưới đây là một số hoạt động cơ bản có thể được thực hiện trên cây tìm kiếm nhị phân:

- **Hoạt động tìm kiếm**: tìm kiếm một phần tử trong một cây.
- **Hoạt động chèn**: chèn một phần tử vào trong một cây.
- **Hoạt động duyệt tiền thứ tự**: duyệt một cây theo cách thức duyệt tiền thứ tự.
- **Hoạt động duyệt trung thứ tự**: duyệt một cây theo cách thức duyệt trung thứ tự.
- **Hoạt động duyệt hậu thứ tự**: duyệt một cây theo cách thức duyệt hậu thứ tự.

Nút (Node) trong cây tìm kiếm nhị phân

Một nút có một vài dữ liệu, tham chiếu tới các nút con bên trái và nút con bên phải của nó.

```
struct node { int data;
struct node leftChild; struct node rightChild; };
```

## ```Hoạt động tìm kiếm trong cây tìm kiếm nhị phân

Mỗi khi một phần tử được tìm kiếm: bắt đầu tìm kiếm từ nút gốc, sau đó nếu dữ liệu là nhỏ hơn giá trị khóa (key), thì sẽ tìm phần tử ở cây con bên trái; nếu lớn hơn thì sẽ tìm phần tử ở cây con bên phải. Dưới đây là giải thuật cho mỗi nút:



```

struct node* search(int data){
    struct node *current = root;
    printf("Truy cap cac phan tu: ");

    while(current->data != data){

        if(current != NULL) {
            printf("%d ",current->data);

            //tới cây con bên trái
            if(current->data > data){
                current = current->leftChild;
            }//else tới cây con bên phải
            else {
                current = current->rightChild;
            }

            //không tìm thấy
            if(current == NULL){
                return NULL;
            }
        }
    }
    return current;
}

```

```Hoạt động chèn trong cây tìm kiếm nhị phân

Mỗi khi một phần tử được chèn: đầu tiên chúng ta cần xác định vị trí chính xác của phần tử này. Bắt đầu tìm kiếm từ nút gốc, sau đó nếu dữ liệu là nhỏ hơn giá trị khóa (key), thì tìm kiếm vị trí còn trống ở cây con bên trái và chèn dữ liệu vào đó; nếu dữ liệu là nhỏ hơn thì tìm kiếm vị trí còn trống ở cây con bên phải và chèn dữ liệu vào đó.

```

void insert(int data){ struct node tempNode = (struct node) malloc(sizeof(struct node)); struct
node current; struct node parent;

```

```

tempNode->data = data; tempNode->leftChild = NULL; tempNode->rightChild = NULL;

```

```

//Nếu cây là trống if(root == NULL){ root = tempNode; }else { current = root; parent = NULL;

```

```

while(1){
    parent = current;

    //tới cây con bên trái
    if(data < parent->data){
        current = current->leftChild;
        //chèn dữ liệu vào cây con bên trái

        if(current == NULL){
            parent->leftChild = tempNode;
            return;
        }
    } //tới cây con bên phải
    else{
        current = current->rightChild;
        //chèn dữ liệu vào cây con bên phải
        if(current == NULL){
            parent->rightChild = tempNode;
            return;
        }
    }
}
}

```

}}

“Loạt bài hướng dẫn **Cấu trúc dữ liệu và giải thuật** của chúng tôi dựa trên nguồn tài liệu của trang: Tutorialspoint

↑ 0    0



« Previous Lesson (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/duyet-cay-tree-traversal>)

Next Lesson » (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cay-avl-avl-tree>)

🗨️ Viết câu trả lời

👁️ Preview

Drop Images

**B** *I* **H** ~~S~~ / | </> “ ” ⋮ 1/2/3 / | 🔗 🖼️ 📄 / | 👁️ 📄 ✂️ /




(<https://simplemde.com/markdown-guide>)

Nội dung câu hỏi.. [Markdown Supported]

lines: 1   words: 0   0:0   1 Keystrokes



 Danh mục bài học (<http://hoclaptrinh.vn>)

## ➔ CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT ()

➔ Mở đầu (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai>)

➔ Cấu trúc dữ liệu là gì ? (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cau-truc-du-lieu-la-gi>)

➔ Cài đặt môi trường (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cai-dat-moi-truong>)

## ➔ MỘT SỐ KHÁI NIỆM VỀ GIẢI THUẬT ()

➔ Giải thuật là gì ? (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/giai-thuat-la-gi>)

➔ Giải thuật tiệm cận - Asymptotic Algorithms (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/giai-thuat-tiem-can-asymptotic-algorithms>)

➔ Giải thuật tham lam - Greedy Algorithms (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/giai-thuat-tham-lam-greedy-algorithms>)

➔ Giải thuật chia để trị - Divide and Conquer (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/giai-thuat-chia-de-tri-divide-and-conquer>)

➔ Giải thuật qui hoạch động - Dynamic Programming (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/giai-thuat-qui-hoach-dong-dynamic-programming>)

➔ Giải thuật định lý thợ - Master Theorem (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/giai-thuat-dinh-ly-tho-master-theorem>)

## ➔ CẤU TRÚC DỮ LIỆU MẢNG (ARRAY) ()

➔ Cấu trúc dữ liệu mảng (Array) (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cau-truc-du-lieu-mang-array>)

## ➔ DANH SÁCH LIÊN KẾT - LINKED LISTS ()

➔ Danh sách liên kết - Linked List (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/danh-sach-lien-ket-linked-list>)

➔ Danh sách liên kết đôi - Doubly Linked List (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/danh-sach-lien-ket-doi-doubly-linked-list>)

➔ Danh sách liên kết vòng - Circular Linked List (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/danh-sach-lien-ket-vong-circular-linked-list>)

## ➔ NGĂN XẾP & HÀNG ĐỢI ()

➔ Cấu trúc dữ liệu ngăn xếp - Stack (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cau-truc-du-lieu-ngan-xep-stack>)

➔ Cấu trúc dữ liệu hàng đợi - Queue (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cau-truc-du-lieu-hang-doi-queue>)

**MỘT SỐ GIẢI THUẬT TÌM KIẾM ()**

(http://hoclaptrinh.vn)



- ➔ Tìm kiếm tuyến tính - Linear Search (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/tim-kiem-tuyen-tinh-linear-search>)
- ➔ Tìm kiếm nhị phân - Binary Search (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/tim-kiem-nhi-phan-binary-search>)
- ➔ Tìm kiếm nội suy - Interpolation Search (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/tim-kiem-noi-suy-interpolation-search>)
- ➔ Cấu trúc dữ liệu Hash Table (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cau-truc-du-lieu-hash-table>)

**MỘT SỐ GIẢI THUẬT SẮP XẾP ()**

- ➔ Giải thuật sắp xếp (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/giai-thuat-sap-xep>)
- ➔ Sắp xếp nổi bọt - Bubble Sort (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/sap-xep-noi-bot-bubble-sort>)
- ➔ Sắp xếp chèn - Insertion Sort (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/sap-xep-chen-insertion-sort>)
- ➔ Sắp xếp chọn - Selection Sort (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/sap-xep-chon-selection-sort>)
- ➔ Sắp xếp trộn - Merge Sort (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/sap-xep-tron-merge-sort>)
- ➔ Giải thuật Shell Sort (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/giai-thuat-shell-sort>)
- ➔ Sắp xếp nhanh - Quick Sort (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/sap-xep-nhanh-quick-sort>)

- ➔ Quay lui - Back Tracking (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/quay-lui-back-tracking>)

**CẤU TRÚC DỮ LIỆU ĐỒ THỊ (GRAPH) ()**

- ➔ Cấu trúc dữ liệu đồ thị (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cau-truc-du-lieu-do-thi>)
- ➔ Tìm kiếm theo chiều sâu - Depth First Traversal (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/tim-kiem-theo-chieu-sau-depth-first-traversal>)

- ➔ Tìm kiếm theo chiều rộng - Breadth First Traversal (<http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/tim-kiem-theo-chieu-rong-breadth-first-traversal>)

**CẤU TRÚC DỮ LIỆU CÂY ()**

➔ Cấu trúc dữ liệu cây (http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cau-truc-du-lieu-cay) (http://hoclaptrinh.vn)



➔ Duyệt cây - Tree Traversal (http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/duyet-cay-tree-traversal)

➔ Cây tìm kiếm nhị phân - Binary Search Tree (http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cay-tim-kiem-nhi-phan-binary-search-tree)

➔ Cây AVL - AVL Tree (http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cay-avl-avl-tree)

➔ Cây Splay - splay Tree (http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cay-splay-splay-tree)

➔ Giải thuật Cây khung - Spanning Tree (http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/giai-thuat-cay-khung-spanning-tree)

➔ Cấu trúc dữ liệu Heap (http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/cau-truc-du-lieu-heap)

➔ ĐỆ QUI (RECURSION) ()

➔ Khái niệm cơ bản về Đệ qui (http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/khai-niem-co-ban-ve-de-qui)

➔ Bài toán Tháp Hà Nội - Tower of Hanoi (http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/bai-toan-thap-ha-noi-tower-of-hanoi)

➔ Dãy Fibonacci (http://hoclaptrinh.vn/tutorial/cau-truc-du-lieu-amp-giai-thuat-55-bai/day-fibonacci-cau-truc-du-lieu-amp-giai-thuat)

Hoclaptrinh.vn © 2017

From Coder With ♥ ()

