


CANTHO UNIVERSITY 

Cây Tìm Kiểm Nhị Phân

Cấu trúc dữ liệu - CT177

Bộ môn Công Nghệ Phần Mềm

Khoa CNTT&TT - Đại học Cần Thơ www.ctu.edu.vn

 **Mục tiêu**

- Trình bày các khái niệm và phép toán trên cây
- Trình bày cây tìm kiểm nhị phân
- Cài đặt cây tìm kiểm nhị phân (TKPN)
- Đánh giá sự hiệu quả của cây TKNP

www.ctu.edu.vn



CANTHO UNIVERSITY

NỘI DUNG

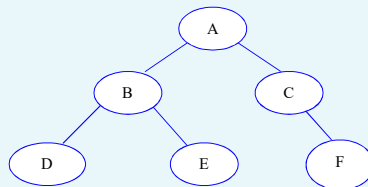
- CÁC THUẬT NGỮ CƠ BẢN
- CÁC PHÉP TOÁN CƠ BẢN
- CÁC PHƯƠNG PHÁP CÀI ĐẶT CÂY
- CÂY NHỊ PHÂN
- CÂY TÌM KIẾM NHỊ PHÂN

www.ctu.edu.vn

CANTHO UNIVERSITY

CÁC THUẬT NGỮ CƠ BẢN (1)

- Định nghĩa
 - Cây (tree): một tập hợp hữu hạn các phần tử gọi là các nút (nodes) và tập hợp hữu hạn các cạnh nối các cặp nút lại với nhau mà không tạo thành chu trình. Nói cách khác, cây là 1 đồ thị không có chu trình.
 - Ví dụ

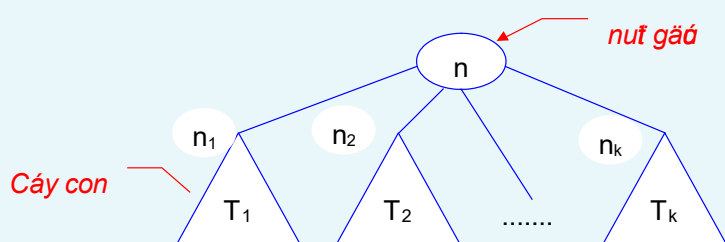
www.ctu.edu.vn



CANTHO UNIVERSITY

CÁC THUẬT NGỮ CƠ BẢN (2)

- Ta có thể định nghĩa cây 1 cách đệ qui như sau:
 - Một nút đơn độc là 1 cây, nút này cũng là nút gốc của cây.
 - Nút n là nút đơn độc và k cây riêng lẻ T_1, T_2, \dots, T_k có các nút gốc lần lượt là n_1, n_2, \dots, n_k . Khi đó ta có được 1 cây mới có nút gốc là nút n và các cây con của nó là T_1, T_2, \dots, T_k .
 - Mô hình



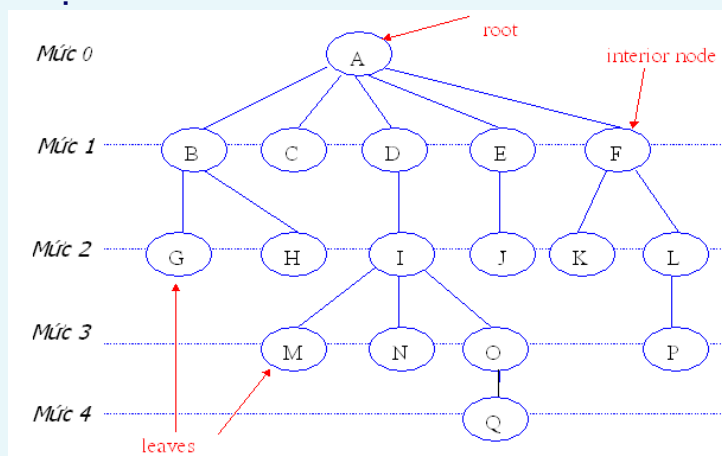
www.ctu.edu.vn




CANTHO UNIVERSITY

CÁC THUẬT NGỮ CƠ BẢN (3)

- Ví dụ

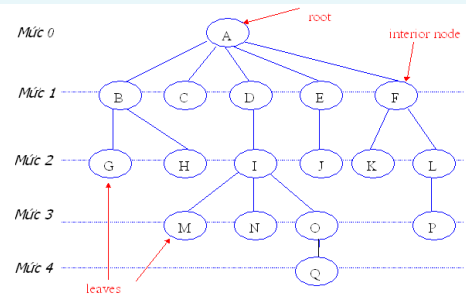


www.ctu.edu.vn




CANTHO UNIVERSITY

CÁC THUẬT NGŨ CƠ BẢN (4)



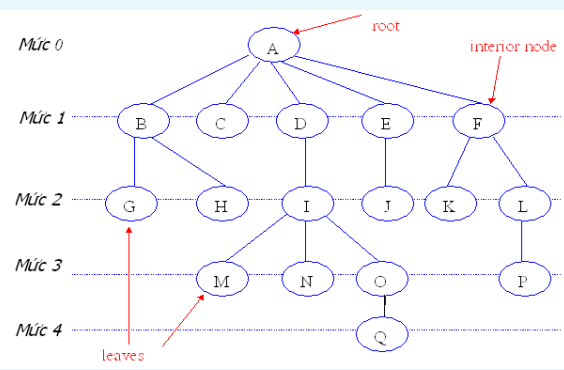
- Nút cha con: nút A là cha của nút B khi nút A ở mức i và nút B ở mức $i+1$, đồng thời giữa A và B có cạnh nối.
 - VD: Ở cây trên, nút B là cha của G và H. Nút I là con của D.
- Bậc của nút là số cây con của nút đó, bậc nút lá = 0.
 - VD: A có bậc 5, C có bậc 0, O có bậc 1.
- Bậc của cây là bậc lớn nhất của các nút trên cây.
 - VD: cây trên có bậc 5.
- Cây n-phân là cây có bậc n.
 - VD: Bậc của cây là 5 hay cây ngũ phân.

www.ctu.edu.vn




CANTHO UNIVERSITY

CÁC THUẬT NGŨ CƠ BẢN (5)



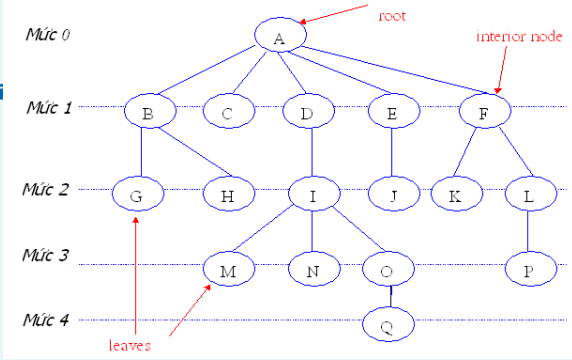
- Nút gốc (root) là nút không có cha.
 - VD: nút gốc A.
- Nút lá (leaf) là nút không có con.
 - VD: các nút C, G, H, J, K, M, N, P, Q.
- Nút trung gian (interior node): nút có bậc khác 0 và không phải là nút gốc.
 - VD: các nút B, D, E, F, I, L, O.

www.ctu.edu.vn




CANTHO UNIVERSITY

CÁC THUẬT NGŨ CƠ BẢN (6)



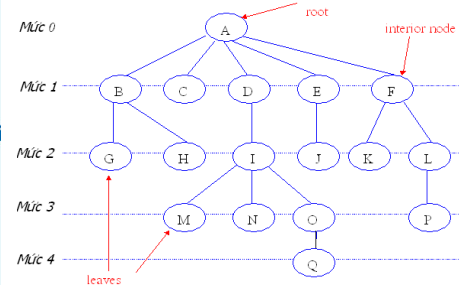
- Đường đi là một chuỗi các nút n_1, n_2, \dots, n_k trên cây sao cho n_i là nút cha của nút n_{i+1} ($i=1..k-1$).
VD: có đường đi A, D, I, O, Q.
- Độ dài đường đi bằng số nút trên đường đi trừ 1.
– VD: độ dài đường đi A, D, I, O, Q = 5 - 1 = 4.

www.ctu.edu.vn




CANTHO UNIVERSITY

CÁC THUẬT NGŨ CƠ BẢN (7)



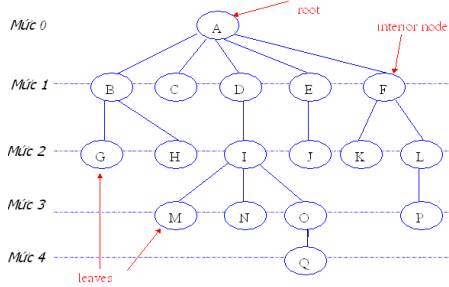
- Nút tiền bối (ancestor) & nút hậu duệ (descendant):
Nếu có đường đi từ nút a đến nút b thì nút a là tiền bối của b, còn b là hậu duệ của a.
– VD: D là tiền bối của Q, còn Q là hậu duệ của D.
- Cây con của 1 cây là 1 nút cùng với tất cả các hậu duệ của nó.
- Chiều cao của 1 nút là độ dài đường đi từ nút đó đến nút lá xa nhất.
– VD: nút B có chiều cao 1, nút D có chiều cao 3.
- Chiều cao của cây là chiều cao của nút gốc.
– VD: chiều cao của cây là 4

www.ctu.edu.vn

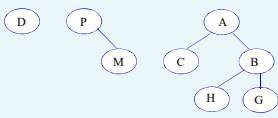


CANTHO UNIVERSITY

CÁC THUẬT NGŨ CƠ BẢN (8)



- Độ sâu của 1 nút là độ dài đường đi từ nút gốc đến nút đó, hay còn gọi là mức (level) của nút đó.
 - VD: I có độ sâu 2, E có độ sâu 1.
 - M, N, O, P có cùng mức 3.
- Nhãn của một nút không phải là tên mà là giá trị được lưu trữ tại nút đó.
- Rừng (forest) là một tập hợp nhiều cây.



www.ctu.edu.vn



CANTHO UNIVERSITY

CÁC THUẬT NGŨ CƠ BẢN (9)

- Cây có thứ tự
 - Nếu ta phân biệt thứ tự các nút trong cùng 1 cây thì ta gọi đó là có thứ tự. Ngược lại, gọi là cây không có thứ tự.
 - Trong cây có thứ tự, thứ tự qui ước từ trái sang phải.



www.ctu.edu.vn



C

18

- Các nút con cùng ruột (siblings).
- Mở rộng: nếu n_i và n_k thì ở bên trái nút n_k thì mọi hậu duệ của nút



CANTHO UNIVERSITY

CÁC THUẬT NGỮ CƠ BẢN (12)

- Định nghĩa theo đệ quy các phép duyệt
 - Cây rỗng hoặc cây chỉ có một nút: cả 3 biểu thức duyệt là rỗng hay chỉ có một nút tương ứng.
 - Ngược lại, giả sử cây T có nút gốc là n và các cây con là T_1, T_2, \dots, T_n thì:
 - Biểu thức duyệt tiền tự của cây T là nút n, kế tiếp là biểu thức duyệt tiền tự của các cây T_1, T_2, \dots, T_n theo thứ tự đó.
 - Biểu thức duyệt trung tự của cây T là biểu thức duyệt trung tự của cây T_1 , kế tiếp là nút n rồi đến biểu thức duyệt trung tự của các cây T_2, \dots, T_n theo thứ tự đó.
 - Biểu thức duyệt hậu tự của cây T là biểu thức duyệt hậu tự của các cây T_1, T_2, \dots, T_n theo thứ tự đó rồi đến nút n.

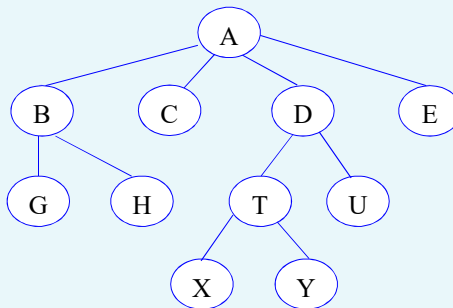
www.ctu.edu.vn



CANTHO UNIVERSITY

CÁC THUẬT NGỮ CƠ BẢN (13)

- Ví dụ



=> Các biểu thức duyệt:

- tiền tự: **A B G H C D T X Y U E**
- trung tự: **G B H A C X T Y D U E**
- hậu tự: **G H B C X Y T U D E A**

www.ctu.edu.vn



CANTHO UNIVERSITY

CÁC THUẬT NGỮ CƠ BẢN (14)

- Các giải thuật duyệt đệ qui

//Duyệt tiền tự

```
void PREORDER(node n){
    liệt kê nút n;
    for (mỗi cây con c của nút n theo thứ tự từ
        trái sang phải)
        PREORDER(c);
} //PREORDER
```

www.ctu.edu.vn



CANTHO UNIVERSITY

CÁC THUẬT NGỮ CƠ BẢN (15)

//Duyệt trung tự

```
void INORDER(node n){
    if (n là nút lá) liệt kê nút n
    else {INORDER(con trái nhất của n)
        Liệt kê nút n;
        for(mỗi cây con c của nút n, trừ cây
            con trái nhất, từ trái sang phải)
            INORDER(c);
    }
} //INORDER
```

www.ctu.edu.vn



CANTHO UNIVERSITY

CÁC THUẬT NGỮ CƠ BẢN (16)

//Duyệt hậu tự

```
void POSORDER(node n){
    if (n là nút lá) Liệt kê nút n
    else {
        for (mỗi nút con c của nút n từ trái sang phải)
            POSORDER(c);
        liệt kê nút n;
    }
}; //POSORDER
```

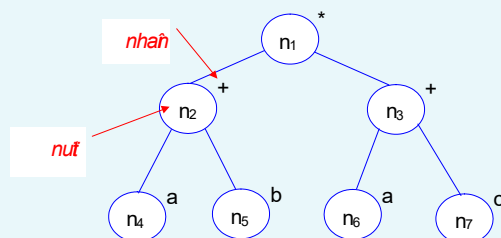
www.ctu.edu.vn



CANTHO UNIVERSITY

CÁC THUẬT NGỮ CƠ BẢN (17)

- Cây có nhãn và cây biểu thức
(Labeled trees and expression trees)




- Lưu trữ kết hợp một nhãn (label) hoặc một giá trị 1(value) với một nút trên cây.
- Nhãn: giá trị được lưu trữ tại nút đó, còn gọi là khóa của nút.
- VD: $(a+b)*(a+c)$

www.ctu.edu.vn

Tên phép toán/hàm	Diễn giải
MAKENULL(T)	Tạo cây T rỗng
EMPTY(T)	Kiểm tra xem cây T có rỗng không?
ROOT(T)	Trả về nút gốc của cây T
PARENT(n, T)	Trả về cha của nút n trên cây T
LEFTMOST_CHILD(n, T)	Trả về con trái nhất của nút n
RIGHT_SIBLING(n, T)	Trả về anh em ruột phải của nút n
LABEL(n, T)	Trả về nhãn của nút n
CREATEi(v, T ₁ , T ₂ , ..., T _i)	Tạo cây mới có nút gốc n nhãn là v, và có i cây con. Nếu n=0 thì cây chỉ có một nút n

www.ctu.edu.vn

 CANTHO UNIVERSITY	CÁC PHƯƠNG PHÁP CÀI ĐẶT CÂY
	<ul style="list-style-type: none"> • CÀI ĐẶT CÂY BẰNG MẢNG • CÀI ĐẶT CÂY BẰNG DANH SÁCH CÁC NÚT CON • CÀI ĐẶT CÂY THEO PHƯƠNG PHÁP CON TRÁI NHẤT VÀ ANH EM RUỘT PHẢI • CÀI ĐẶT CÂY BẰNG CON TRỞ
	<p>(Do giới hạn của thời gian giảng dạy, việc cài đặt cây tổng quát không được giới thiệu)</p>

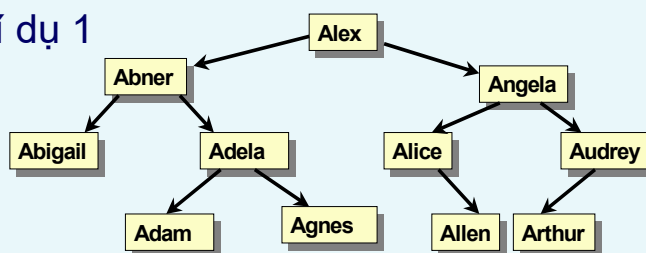
www.ctu.edu.vn



CANTHO UNIVERSITY

CÂY NHỊ PHÂN (1)

- Định nghĩa
 - Là cây rỗng hoặc có tối đa hai nút con.
 - Hai nút con có thứ tự phân biệt rõ ràng
 - Con trái (left child): nằm bên trái nút cha.
 - Con phải (right child): nằm bên phải nút cha.
- Ví dụ 1



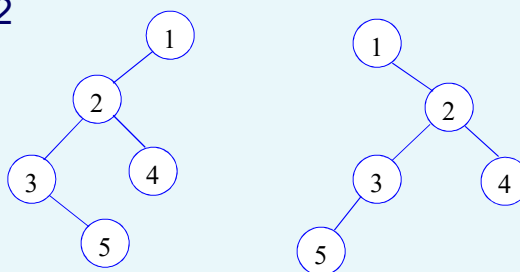
www.ctu.edu.vn



CANTHO UNIVERSITY

CÂY NHỊ PHÂN (2)

- Ví dụ 2



=> Là 2 cây nhị phân khác nhau

www.ctu.edu.vn



CANTHO UNIVERSITY

DUYỆT CÂY NHỊ PHÂN

- Các biểu thức duyệt: (N:Node, R:Right, L:Left)
 - Tiền tự (NLR): duyệt nút gốc, duyệt tiền tự con trái, duyệt tiền tự con phải.
 - Trung tự (LNR): duyệt trung tự con trái, duyệt nút gốc, duyệt trung tự con phải.
 - Hậu tự (LRN): duyệt hậu tự con trái, duyệt hậu tự con phải, duyệt nút gốc.

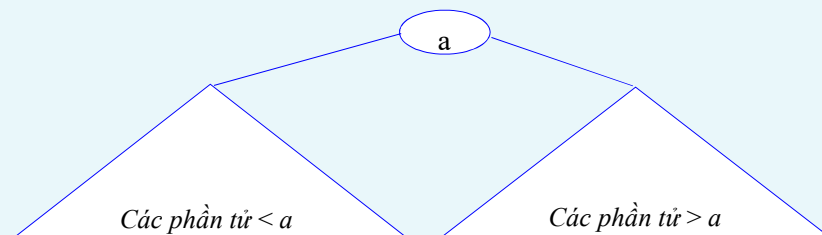
www.ctu.edu.vn



CANTHO UNIVERSITY

CÂY TÌM KIẾM NHỊ PHÂN (Binary search tree-BST)

- Định nghĩa
Cây TKNP là cây nhị phân mà nhãn tại mỗi nút lớn hơn nhãn của tất cả các nút thuộc cây con bên trái và nhỏ hơn nhãn của tất cả các nút thuộc cây con bên phải.
- Mô hình



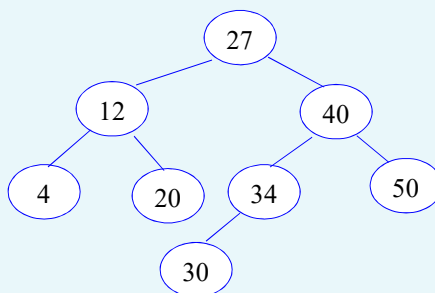
www.ctu.edu.vn



CANTHO UNIVERSITY

CÂY TÌM KIẾM NHỊ PHÂN

- Ví dụ



- Nhận xét

- Trên cây TKNP không có 2 nút trùng khóa.
- Cây con của 1 cây TKNP là 1 cây tìm kiếm nhị phân.
- Duyệt trung tự tạo thành dãy nhãn có giá trị tăng: 4, 12, 20, 27, 30, 34, 40, 50.

www.ctu.edu.vn



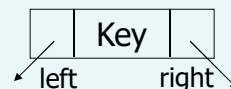
CANTHO UNIVERSITY

CÀI ĐẶT CÂY TKNP (1)

- Khai báo

```

typedef ... KeyType;
struct Node
{
    KeyType Key;
    struct Node* Left, Right;
}
typedef struct Node* Tree;
  
```



www.ctu.edu.vn



CANTHO UNIVERSITY

CÀI ĐẶT CÂY TKNP

- Tạo cây rỗng

```
void MakeNullTree (Tree *T)
{ (*T)=NULL; }
```

- Kiểm tra cây rỗng

```
int EmptyTree (Tree T)
{ return T==NULL; }
```

www.ctu.edu.vn



CANTHO UNIVERSITY

- Xác định con trái

```
Tree LeftChild (Tree T)
{ if (T!=NULL) return T->Left;
  else return NULL; }
```

- Xác định con phải

```
Tree RightChild (Tree T)
{ if (T!=NULL) return T->Right;
  else return NULL; }
```

- Kiểm tra xem một nút có phải là lá không?

```
int IsLeaf (Tree T)
{ if (T!=NULL)
  return (LeftChild (T)==NULL) && (RightChild (T)==NULL) ;
  else return 0; }
```

www.ctu.edu.vn



CANTHO UNIVERSITY

CÀI ĐẶT CÂY TKNP

- Duyệt tiền tự

```
void PreOrder(Tree T){  
    printf("%c ",T->Key);  
    if (LeftChild(T)!=NULL)  
        PreOrder(LeftChild(T));  
    if (RightChild(T)!=NULL)  
        PreOrder(RightChild(T));  
}
```

www.ctu.edu.vn

CANTHO UNIVERSITY

CÀI ĐẶT CÂY TKNP

- Duyệt trung tự

```
void InOrder(Tree T)  
{  
    if (LeftChild(T)!=NULL)  
        InOrder(LeftChild(T));  
    printf("%c ",T->Key);  
    if (RightChild(T)!=NULL)  
        InOrder(RightChild(T));  
}
```

www.ctu.edu.vn



CANTHO UNIVERSITY

CÀI ĐẶT CÂY TKNP

- Duyệt hậu tự

```
void PosOrder(TTree T)
{
    if(LeftChild(T) != NULL)
        PosOrder(LeftChild(T));
    if(RightChild(T) != NULL)
        PosOrder(RightChild(T));
    printf("%c ", T->Key);
}
```

www.ctu.edu.vn



CANTHO UNIVERSITY

CÀI ĐẶT CÂY TKNP

- Xác định số nút trong cây

```
int nb_nodes(TTree T)
{
    if(EmptyTree(T))
        return 0;
    else
        return 1 + nb_nodes(LeftChild(T))
                + nb_nodes(RightChild(T));
}
```

www.ctu.edu.vn



CANTHO UNIVERSITY

CÀI ĐẶT CÂY TKNP

- Tìm kiếm một nút có khoá x
 - Bắt đầu từ nút gốc ta tiến hành các bước sau:
 - Nếu nút gốc bằng NULL thì khoá x không có trên cây.
 - Nếu x bằng khoá nút gốc thì giải thuật dừng vì đã tìm gặp x trên cây.
 - Nếu x nhỏ hơn nhãn của nút hiện hành: tìm x trên cây con bên trái.
 - Nếu x lớn hơn nhãn của nút hiện hành: tìm x trên cây con bên phải.

www.ctu.edu.vn



CANTHO UNIVERSITY

CÀI ĐẶT CÂY TKNP

```

Tree Search(KeyType x, Tree Root) {
    if (Root == NULL) return NULL; // không tìm thấy x
    else if (Root->Key == x) // tìm thấy khoá x
        return Root;
    else if (Root->Key < x)
        // tìm tiếp trên cây bên phải
        return Search(x, Root->right);
    else // tìm tiếp trên cây bên trái
        return Search(x, Root->left);
}

```

www.ctu.edu.vn



CANTHO UNIVERSITY

CÀI ĐẶT CÂY TKNP

- Thêm một nút có khoá x vào cây

Muốn thêm 1 nút có khoá x vào cây TKNP, trước tiên ta phải tìm kiếm xem đã có x trên cây chưa.

Nếu có thì giải thuật kết thúc, nếu chưa ta mới thêm vào. Việc thêm vào không làm phá vỡ tính chất cây TKNP.

Giải thuật thêm vào như sau: Bắt đầu từ nút gốc ta tiến hành các bước sau:

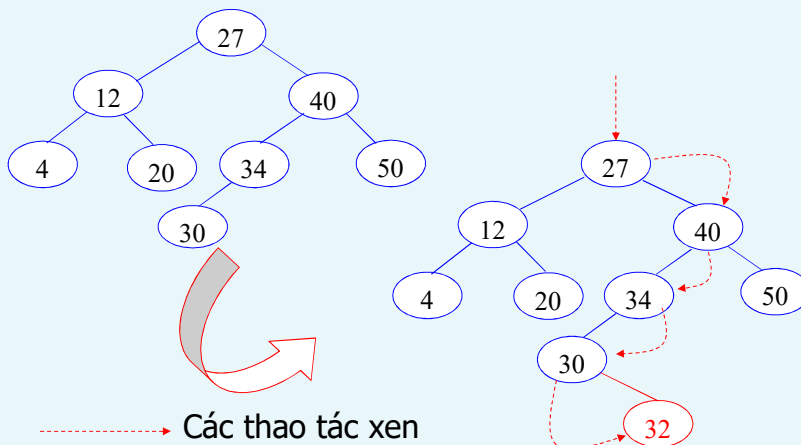
- Nếu nút gốc bằng NULL thì khoá x chưa có trên cây, do đó ta thêm 1 nút mới.
- Nếu x bằng khoá nút gốc thì giải thuật dừng vì x đã có trên cây.
- Nếu x nhỏ hơn nhãn của nút hiện hành: xen x vào cây con bên trái.
- Nếu x lớn hơn nhãn của nút hiện hành: xen x vào cây con bên phải.

www.ctu.edu.vn


CANTHO UNIVERSITY

CÀI ĐẶT CÂY TKNP

- Ví dụ: Xen nút có khoá 32


www.ctu.edu.vn



CANTHO UNIVERSITY

CÀI ĐẶT CÂY TKNP

```
void InsertNode(KeyType x, Tree *T){
    if ((*T) == NULL){
        (*T) = (struct Node*)malloc(sizeof(struct Node));
        (*T)->Key = x;
        (*T)->Left = NULL;
        (*T)->Right = NULL;
    }
    else
        if ((*T)->Key == x)
            printf("Da ton tai khoa x");
        else
            if ((*T)->Key > x)
                InsertNode(x, &(*T)->Left);
            else
                InsertNode(x, &(*T)->Right);
}
```

www.ctu.edu.vn



CANTHO UNIVERSITY

CÀI ĐẶT CÂY TKNP

- Xóa một nút khóa x khỏi cây
 - Muốn xóa 1 nút có khóa x trên cây TKNP.
Trước tiên ta phải tìm xem có x trên cây không.
 - Nếu không thì giải thuật kết thúc.
 - Nếu gặp nút N chứa khóa x, có 3 trường hợp xảy ra.

www.ctu.edu.vn

CÀI ĐẶT CÂY TKNP

- Trường hợp 1
 - N là nút lá: thay nút này bởi NULL
 - Ví dụ: Xóa nút nhãn 20

Nút cần xóa

www.ctu.edu.vn


CÀI ĐẶT CÂY TKNP

- Trường hợp 2
 - N có một cây con: thay nút này bởi cây con của nó.
 - Ví dụ: xóa nút có nhãn 34.

cây con

nút cần xóa

www.ctu.edu.vn




CÀI ĐẶT CÂY TKNP

CANTHO UNIVERSITY

- Trường hợp 3
 - N có hai cây con: thay nút này bởi
 - Nút có nhãn lớn nhất của cây con bên trái, hoặc
 - Nút có nhãn nhỏ nhất của cây con bên phải

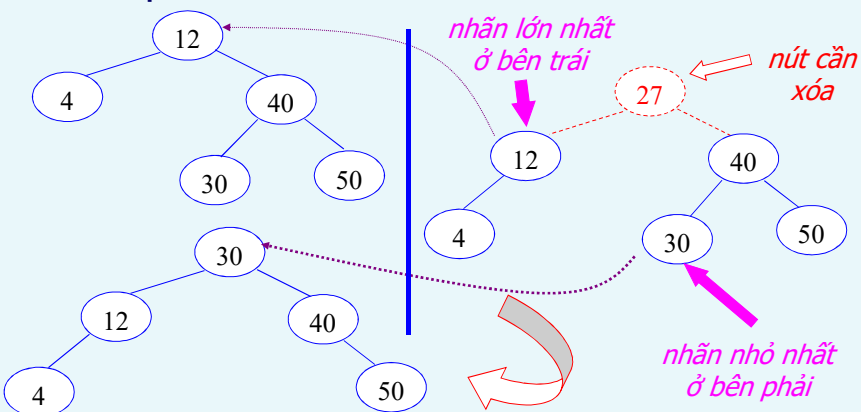
www.ctu.edu.vn



CÀI ĐẶT CÂY TKNP

CANTHO UNIVERSITY

- Ví dụ: Xoá nút có nhãn 27



www.ctu.edu.vn



CANTHO UNIVERSITY

CÀI ĐẶT CÂY TKNP

```

KeyType DeleteMin(Tree *T)
{
    KeyType k;
    if((*T)->Left == NULL)
    {
        k = (*T)->Key;
        (*T) = (*T)->Right;
        return k;
    }
    else return DeleteMin(&(*T)->Left);
}

```

www.ctu.edu.vn

```

void DeleteNode(KeyType x, Tree *T){
    if(*T != NULL) //Kiem tra cay khac rong
    {
        if(x < (*T)->Key) //Hy vong x nam ben trai cua nut
            DeleteNode(x, &(*T)->Left);
        else
        {
            if(x > (*T)->Key) //Hy vong x nam ben phai cua nut
                DeleteNode(x, &(*T)->Right);
            else // Tim thay khoa x tren cay
            {
                if((*T)->Left==NULL && (*T)->Right==NULL) //x la nut la
                    (*T)=NULL; // Xoa nut x
                else // x co it nhat mot con
                {
                    if((*T)->Left==NULL) //Chac chan co con phai
                        (*T) = (*T)->Right;
                    else
                    {
                        if((*T)->Right==NULL) //Chac chan co con trai
                            (*T) = (*T)->Left;
                        else // x co hai con
                            (*T)->Key = DeleteMin(&(*T)->Right);
                    }
                }
            }
        }
    }
}

```

www.ctu.edu.vn



CANTHO UNIVERSITY

KIẾN THỨC BỔ SUNG (1)

- Thời gian tìm kiếm một giá trị trên một cây TKNP có N nút là:
 - $O(\log N)$ nếu cây “cân bằng” (balanced)
 - $O(N)$ nếu cây “không cân bằng” (unbalanced)

Tại sao?


www.ctu.edu.vn

CANTHO UNIVERSITY

KIẾN THỨC BỔ SUNG (2)

Các slide kế tiếp giúp trả lời câu hỏi

www.ctu.edu.vn




KIẾN THỨC BỔ SUNG (3)


• Bên dưới là một cây TKNP “không cân bằng”



www.ctu.edu.vn



KIẾN THỨC BỔ SUNG (4)



• Tìm kiếm nút có khóa “Audrey” là trường hợp xấu nhất

- Ta phải duyệt qua hầu như toàn bộ N nút của cây.
- Thời gian tìm kiếm: $O(N)$.

www.ctu.edu.vn



CANTHO UNIVERSITY

CÂY NHỊ PHÂN ĐẦY ĐỦ (1) (Full binary tree)

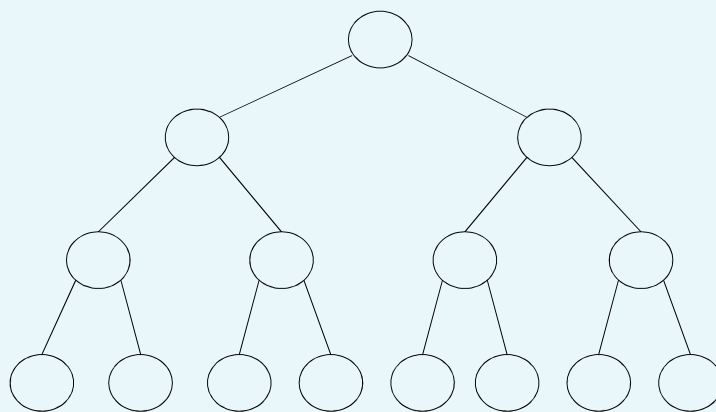
- Một cây nhị phân là “**cây nhị phân đầy đủ**” nếu và chỉ nếu
 - Mỗi nút không phải lá có chính xác 2 nút con.
 - Tất cả các nút lá có chiều cao bằng nhau.

www.ctu.edu.vn

CANTHO UNIVERSITY

CÂY NHỊ PHÂN ĐẦY ĐỦ (2)

- Ví dụ: Một cây nhị phân đầy đủ

www.ctu.edu.vn



CANTHO UNIVERSITY

CÂY NHỊ PHÂN ĐẦY ĐỦ (3)

- **Câu hỏi về cây nhị phân đầy đủ:**
 - Một cây nhị phân đầy đủ chiều cao h sẽ có bao nhiêu nút lá?
 - Một cây nhị phân đầy đủ chiều cao h sẽ có tất cả bao nhiêu nút?

www.ctu.edu.vn


CANTHO UNIVERSITY

CÂY NHỊ PHÂN ĐẦY ĐỦ (4)

- Một cây nhị phân đầy đủ chiều cao h sẽ có bao nhiêu nút lá?
 - 2^h
- Một cây nhị phân đầy đủ chiều cao h sẽ có tất cả bao nhiêu nút?
 - $2^{h+1} - 1$

www.ctu.edu.vn



CANTHO UNIVERSITY

CÂY NHỊ PHÂN HOÀN CHỈNH (1) (Complete binary tree)

- Một cây nhị phân hoàn chỉnh (về chiều cao) thỏa mãn các điều kiện sau:
 - Mức 0 đến $h-1$ là trình bày một cây nhị phân đầy đủ chiều cao $h-1$.
 - Một hoặc nhiều nút ở mức $h-1$ có thể có 0, hoặc 1 nút con.
 - Nếu j, k là các nút ở mức $h-1$, khi đó j có nhiều nút con hơn k nếu và chỉ nếu j ở bên trái của k .

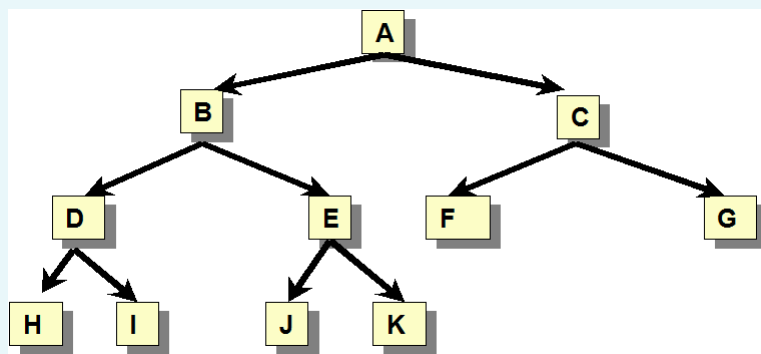
www.ctu.edu.vn



CANTHO UNIVERSITY

CÂY NHỊ PHÂN HOÀN CHỈNH (2)

- Ví dụ



Một cây nhị phân hoàn chỉnh

www.ctu.edu.vn



CANTHO UNIVERSITY

CÂY NHỊ PHÂN HOÀN CHÍNHH (3)

- Được cho một tập hợp N nút, một cây nhị phân hoàn chỉnh của những nút này cung cấp số nút lá nhiều nhất - với chiều cao trung bình của mỗi nút là nhỏ nhất.
- Cây hoàn chỉnh n nút phải chứa ít nhất một nút có chiều cao là $\lfloor \log N \rfloor$.

www.ctu.edu.vn


CANTHO UNIVERSITY

CÂY NHỊ PHÂN CÂN BẰNG VỀ CHIỀU CAO (Height-balanced Binary Tree) (1)

- Một cây nhị phân cân bằng về chiều cao là một cây nhị phân như sau:
 - Chiều cao của cây con trái và phải của bất kỳ nút nào khác nhau không quá một đơn vị.
 - Chú ý: mỗi cây nhị phân hoàn chỉnh là một cây cân bằng về chiều cao.

www.ctu.edu.vn

CANTHO UNIVERSITY

CÂY CÂN BẰNG VỀ CHIỀU CAO (2)

N M $N - M \leq 1$

Cân bằng về chiều cao là một thuộc tính cục bộ

www.ctu.edu.vn

CANTHO UNIVERSITY

ƯU ĐIỂM CỦA CÂY CÂN BẰNG (1)

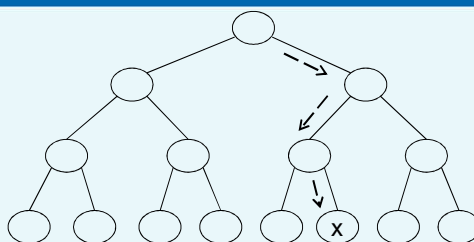
- Cây nhị phân cân bằng về chiều cao là cây “cân bằng”.
- Thời gian tìm kiếm một nút trên cây N nút là $O(\log N)$.

www.ctu.edu.vn



CANTHO UNIVERSITY

ƯU ĐIỂM CỦA CÂY CÂN BẰNG (2)



- Cây có N nút.
- Thời gian tìm x trong trường hợp xấu nhất khi x là lá
 - Ta phải đi qua đường đi độ dài h để tìm x (h là chiều cao cây)
 - $2^{(h+1)} - 1 = N$
 - $\Rightarrow h = \log(N+1) - 1$
 - $\Rightarrow O(\log(N+1) - 1) \approx O(\log(N))$

www.ctu.edu.vn




CANTHO UNIVERSITY

TỔNG KẾT

- Cây một tập hợp hữu hạn các phần tử gọi là các nút và tập hợp hữu hạn các cạnh nối các cặp nút lại với nhau mà không tạo thành chu trình.
- Cây nhị phân là cây rỗng hoặc có tối đa hai nút con.
- Cây TKNP là cây nhị phân mà nhãn tại mỗi nút lớn hơn nhãn của tất cả các nút thuộc cây con bên trái và nhỏ hơn nhãn của tất cả các nút thuộc cây con bên phải.
- Thời gian tìm kiếm một giá trị trên một cây TKNP có N nút là:
 - $O(\log N)$ nếu cây “cân bằng”.
 - $O(N)$ nếu cây “không cân bằng”.

www.ctu.edu.vn



Tài liệu

- Bài giảng Cấu trúc dữ liệu/Tập thể GV Khoa CNTT&TT – Đại học Cần Thơ, 2003.

www.ctu.edu.vn



Q&A?



www.ctu.edu.vn