

Cấu trúc dữ liệu và giải thuật

TS. Phạm Tuấn Minh

Khoa Công nghệ Thông tin, Đại học Phenikaa

minh.phamtuan@phenikaa-uni.edu.vn

<https://sites.google.com/site/phamtuanminh/>

Chương 3: Cây và bảng băm

- ❑ Các khái niệm cây
- ❑ Cây nhị phân tìm kiếm
- ❑ Cây AVL
- ❑ Bảng băm

Duyệt cây nhị phân

□ Thứ tự duyệt cây

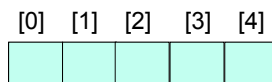
- Thứ tự trước
- Thứ tự giữa
- Thứ tự sau

□ Ví dụ ứng dụng

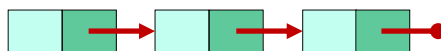
- Đếm số nút trên cây nhị phân
- Tìm nút cháu
- Tính chiều cao của nút

1-3

Duyệt mảng và danh sách liên kết



$i = i + 1;$



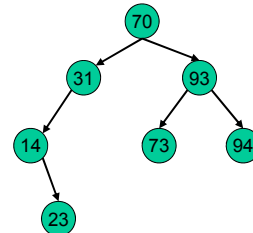
$cur = cur \rightarrow next;$

1-4

Duyệt cây

□ Duyệt cây:

- Thăm mọi nút theo một thủ tục xác định các bước rõ ràng và thủ tục có thể thực hiện lặp lại trên cây
- Không thăm lặp lại các nút



1-5

Giải thích lặp đệ quy

```
Void tellStory(){  
    printf("Once upon a time there was a mountain. ");  
    printf("On the mountain, there was a temple. ");  
    printf("In the temple was an monk, telling a story. ");  
    printf("What is the story? It is: ");  
    tellStory();  
}
```

Không kết thúc!



1-6

Giải thích lặp đệ quy

```
Void tellStory(){  
    printf("Once upon a time there was a mountain. ");  
    printf("On the mountain, there was a temple. ");  
    printf("In the temple was an monk, telling a story. ");  
    printf("What is the story? It is: ");  
    tellStory();  
    printf("The monk asked: do you like it?\n");  
}
```

Không bao giờ hiện "The monk asked..."!



1-7

Giải thích lặp đệ quy

```
Void tellStory(int i){  
    if (i==0) { printf("nothing!\n"); return;}  
    printf("Once upon a time there was a mountain. ");  
    printf("On the mountain, there was a temple. ");  
    printf("In the temple was an monk, telling a story. ");  
    printf("What is the story? It is: ");  
    tellStory(i-1);  
    printf("The monk asked: do you like it?\n");  
}
```

Nếu gọi tellStory(5),
thì câu chuyện kể bao nhiêu lần?



1-8

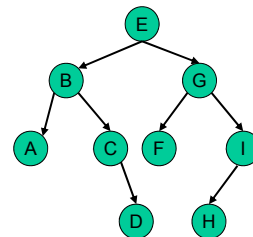
Giải thích lặp đệ quy

```
Void tellStory(int i){  
    if (i==0) { printf("nothing!\n"); return;}  
    printf("Once upon a time there was a mountain. ");  
    printf("On the mountain, there was a temple. ");  
    printf("In the temple was an monk %d, telling a story. ", i);  
    printf("What is the story? It is: ");  
    tellStory(i-1);  
    printf("The monk %d asked: do you like it?\n", i);  
}  
  
tellStory(5);
```



Ba cách cơ bản để duyệt cây

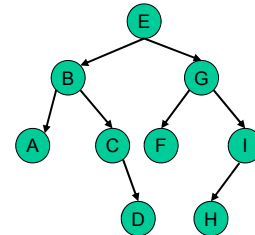
- ❑ Thứ tự trước (pre-order)
 - Xử lý dữ liệu của nút hiện tại
 - Thăm nút con trái trên cây con
 - Thăm nút con phải trên cây con
- ❑ Thứ tự giữa (in-order)
 - Thăm nút con trái trên cây con
 - Xử lý dữ liệu của nút hiện tại
 - Thăm nút con phải trên cây con
- ❑ Thứ tự sau (post-order)
 - Thăm nút con trái trên cây con
 - Thăm nút con phải trên cây con
 - Xử lý dữ liệu của nút hiện tại



1-10

Duyệt cây theo thứ tự trước

```
void TreeTraversal(BTNode *cur) {  
    if (cur == NULL)  
        return;  
  
    printf("%c", cur->item);  
  
    TreeTraversal(cur->left); //Thăm nút con trái  
    TreeTraversal(cur->right); //Thăm nút con phải  
}
```

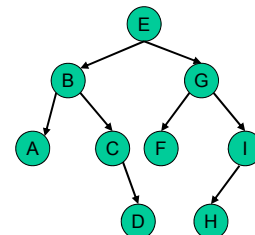


Kết quả:
E B A C D G F I H

1-11

Duyệt cây theo thứ tự giữa

```
void TreeTraversal(BTNode *cur) {  
    if (cur == NULL)  
        return;  
  
    TreeTraversal(cur->left); //Thăm nút con trái  
    printf("%c", cur->item);  
    TreeTraversal(cur->right); //Thăm nút con phải  
}
```

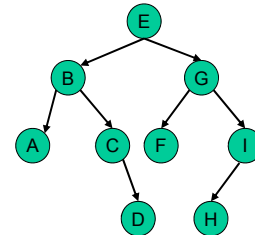


Kết quả:
A B C D F G H I

1-12

Duyệt cây theo thứ tự sau

```
void TreeTraversal(BTNode *cur) {  
    if (cur == NULL)  
        return;  
  
    TreeTraversal(cur->left); //Thăm nút con trái  
    TreeTraversal(cur->right); //Thăm nút con phải  
    printf("%c", cur->item);  
}
```



Kết quả:
A D C B F H I E

1-13

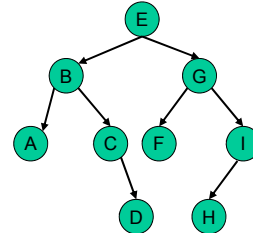
Duyệt cây nhị phân

- Thứ tự duyệt cây
 - Thứ tự trước
 - Thứ tự giữa
 - Thứ tự sau
- Ví dụ ứng dụng
 - Đếm số nút trên cây nhị phân
 - Tìm nút cháu
 - Tính chiều cao của nút

1-14

Đếm số nút trên cây nhị phân

- Định nghĩa đệ quy:
 - Số nút trên cây = 1 + số nút trên cây con trái + số nút trên cây con phải
- Mỗi nút trả về số nút trên cây con của nó
- Nút lá trả về 1



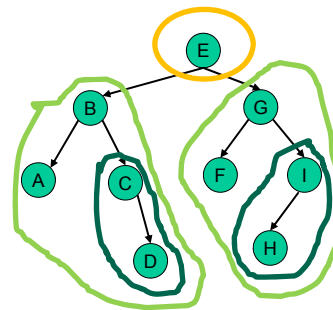
1-15

countNode()

- Trả về kích thước của cây con của nó cho nút cha
- Nút lá trả về 1 cho nút cha
- Nút gốc trả về kích thước của toàn bộ cây

```
void TreeTraversal(BTNode *cur) {  
    if (cur == NULL)  
        return;  
    // Có thể thực hiện một số thao tác  
    TreeTraversal(cur->left); //Thăm nút con trái  
    // Có thể thực hiện một số thao tác  
    TreeTraversal(cur->right); //Thăm nút con phải  
    // Có thể thực hiện một số thao tác  
}
```

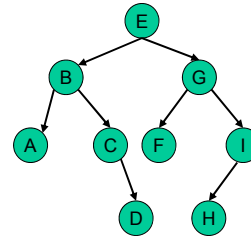
```
int countNode(BTNode*cur) {  
    if (cur == NULL)  
        return ???;  
    countNode(cur->left);  
    countNode(cur->right);  
    ??? //tính tổng  
}
```



1-16

countNode()

- ❑ Nút lá trả về 1 cho nút cha
- ❑ Nút Null trả về 0

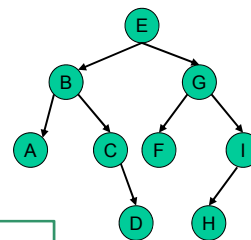


```
int countNode(BTNode*cur) {  
    if (cur == NULL)  
        return 0;  
    int l = countNode(cur->left);  
    int r = countNode(cur->right);  
    return l+r+1; //tính tổng  
}
```

1-17

countNode()

- ❑ Nút lá trả về 1 cho nút cha
- ❑ Nút Null trả về 0



```
int countNode(BTNode*cur) {  
    if (cur == NULL)  
        return 0;  
    return (countNode(cur->left) + countNode(cur->right) + 1);  
}
```

1-18

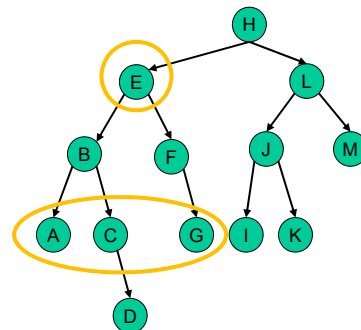
Duyệt cây nhị phân

- Thứ tự duyệt cây
 - Thứ tự trước
 - Thứ tự giữa
 - Thứ tự sau
- **Ví dụ ứng dụng**
 - Đếm số nút trên cây nhị phân
 - **Tìm nút cháu**
 - Tính chiều cao của nút

1-19

Tìm nút cháu

- Cho nút X, tìm tất các nút cháu của X
- Ví dụ: Cho nút E thì sẽ trả về các nút cháu là A, C và G
- Tìm các nút cháu trong **mức k**
 - Cần lưu vết về số mức đã đi qua



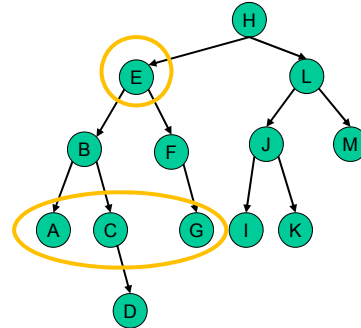
- Các nút cháu ở mức 2:
 - X->left->left
 - X->left->right
 - X->right->left
 - X->right->right

1-20

Tìm nút cháu

- Muốn đi xuống mức k: Sử dụng biến đếm counter để xác định độ sâu

```
void TreeTraversal(BTNode *cur) {
    if (cur == NULL)
        return;
    // Kiểm tra biến đếm counter
    TreeTraversal (cur->left);
    TreeTraversal(cur->right);
}
```



1-21

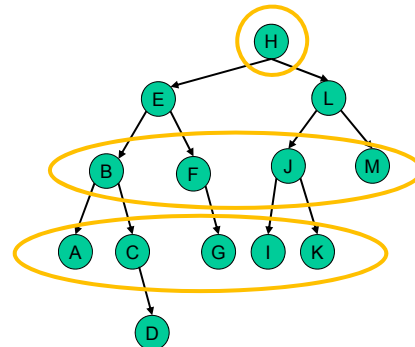
Tìm nút cháu

```
void main() { ...
    if ( X == null) return;
    findgrandchildren(X, 0);
}
```

```
void findgrandchildren(BTNode *cur, int c) {
    if (cur == NULL) return;

    if (c == k) {
        printf("%c ", cur->item);
        return;
    }

    findgrandchildren(cur->left, c+1);
    findgrandchildren(cur->right, c+1);
}
```



- Nếu k = 2, gọi findgrandchildren(H,0), kết quả là gì?
- Nếu k = 3, gọi findgrandchildren(H,0), kết quả là gì?
- Nếu k = 2, gọi findgrandchildren(H,1), kết quả là gì?

1-22

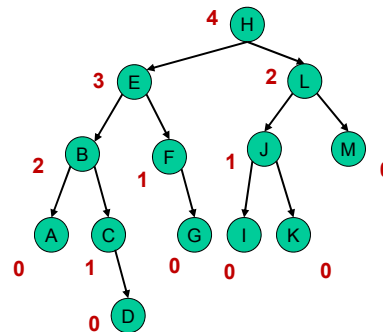
Duyệt cây nhị phân

- Thứ tự duyệt cây
 - Thứ tự trước
 - Thứ tự giữa
 - Thứ tự sau
- **Ví dụ ứng dụng**
 - Đếm số nút trên cây nhị phân
 - Tìm nút cháu
 - **Tính chiều cao của nút**

1-23

Tính chiều cao của một nút

- Chiều cao của một nút = số liên kết từ nút đó tới nút lá sâu nhất
- Chiều cao của nút D, C, H?
- Cách tính chiều cao của một nút?



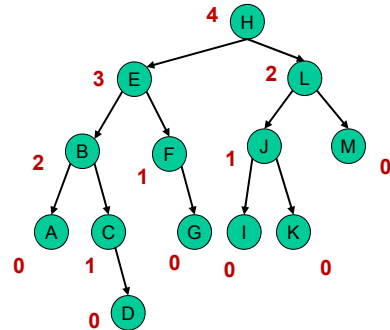
- leaf.height = 0
- Non-leaf node X: $X.height = \max(X.left.height, X.right.height) + 1$

1-24

Tính chiều cao của một nút

- ❑ Mỗi nút tính chiều cao của nó:
- ❑ Nút lá trả về chiều cao bằng 0
- ❑ Các nút NULL trả về -1

```
int TreeTraversal(BTNode *cur) {  
    if (cur == NULL)  
        return -1;  
  
    int l = TreeTraversal(cur->left);  
    int r = TreeTraversal(cur->right);  
  
    int c = max(l, r) + 1;  
  
    return c;  
}
```



1-25

Cấu trúc dữ liệu và giải thuật

- ❑ Nội dung bài giảng được biên soạn bởi TS. Phạm Tuấn Minh.

1-26