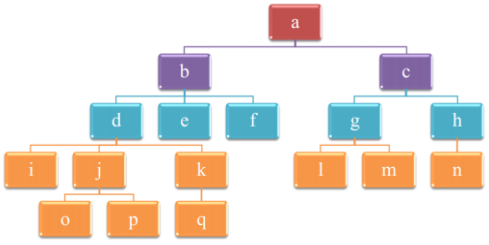
UIT
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

Data Structures & Algorithms

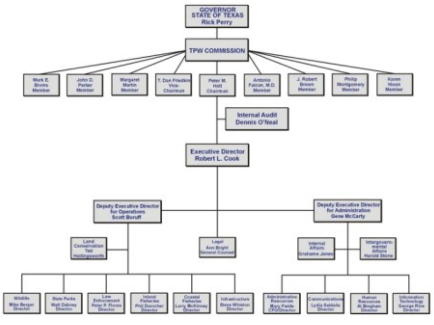
CÂY - TREE

CÂY



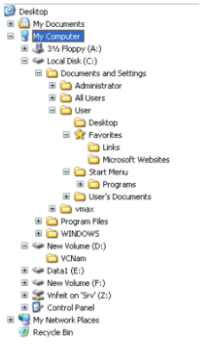
2

CÂY



3

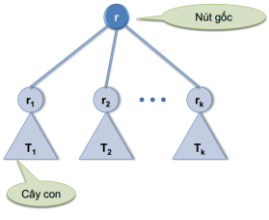
CÂY



4

CÂY – Khái Niệm

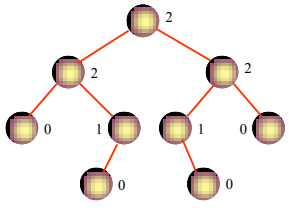
- Cây là một tập hợp T **các phần tử (gọi là nút của cây)**, trong đó có một nút đặc biệt gọi **là nút gốc**, các nút còn lại được chia thành những tập rời nhau T_1, T_2, ...,T_n theo quan hệ phân cấp, **trong đó T_i cũng là 1 cây**. Mỗi nút ở cấp i sẽ quản lý một số nút ở cấp i+1. Quan hệ này người ta gọi là quan hệ cha – con.



CÂY – Một số khái Niệm

Bậc – Degree/Oder

- **Bậc của một nút**: là **số cây con** của nút đó



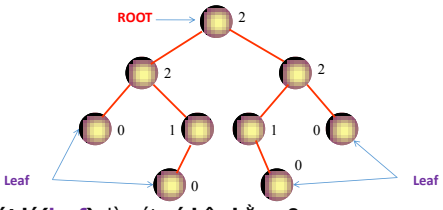
- **Bậc của một cây**: là **bậc lớn nhất** của các nút trong cây

➔ Cây có bậc n gọi là cây n-phân

6

CÂY – Một số khái Niệm

➢**Nút gốc (root):** là nút **không có nút cha**.



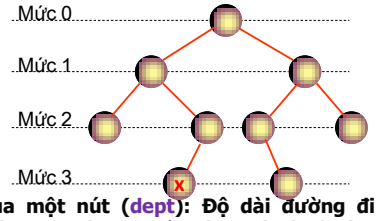
➢**Nút lá(leaf):** là nút **có bậc bằng 0** .

➢**Nút nhánh (branch/internal)**: là nút có **bậc khác 0** và **không phải là gốc**

CÂY – Một số khái Niệm

➢**Mức của một nút (level):**
- ➢Mức (gốc (T)) = 0.
- ➢Gọi T1, T2, T3, ... , Tn là các cây con của T0 :

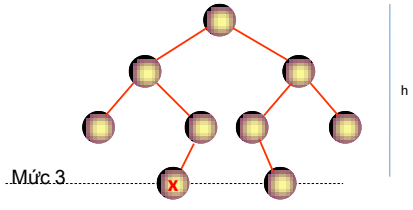
Mức (T1) = Mức (T2) = . . . = Mức (Tn) = Mức (T0) + 1.



➢**Độ sâu của một nút (dept): Độ dài đường đi giữa node gốc và node đó. Node ở mức i thì có độ dài i**

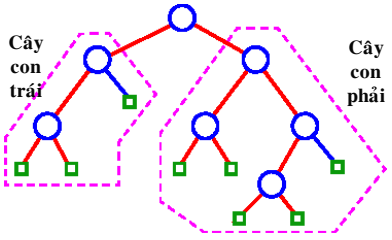
CÂY – Một số khái Niệm

➢**Chiều cao của cây (height):**
- ➢**Cây rỗng** = 0.
- ➢Cây **khác rỗng**: **Mức lớn nhất** giữa các node trên cây



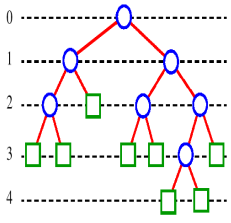
Cây Nhị Phân (Binary Tree)

- **Mỗi nút có tối đa 2 cây con**



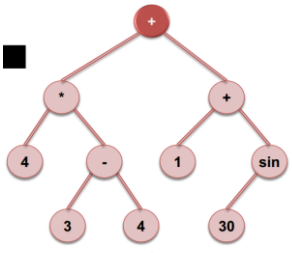
Tính chất Binary Tree

- **Số nút nằm ở mức i ≤ 2i.**
- **Số nút lá ≤ 2h-1, với h là chiều cao của cây.**
- **Chiều cao của cây h ≥ log2(N)**
  - **N = số nút trong cây**
- **Số nút trong cây ≤ 2h-1.**



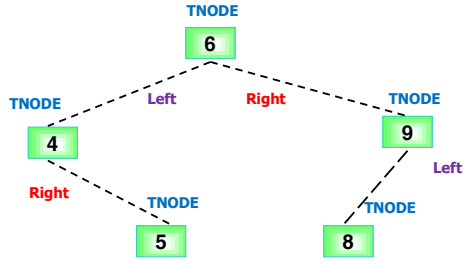
Một số ứng dụng Binary Tree

- **Cây tổ chức thi đấu.**
- **Cây biểu thức số học.**
- **Lưu trữ và tìm kiếm thông tin.**

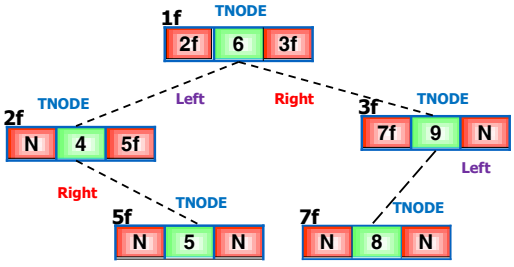


Cây biểu thức:
*4 * (3 – 4) + (1 + sin(30))*

Tổ chức Binary Tree

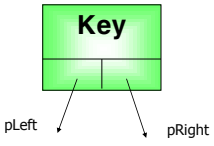


Tổ chức bộ trớ Binary Tree



Cấu trúc dữ liệu Binary Tree

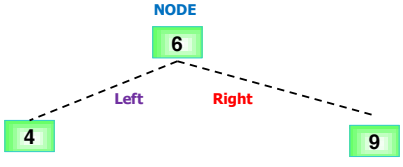
```
struct node
{
  KDL Key;
  struct node *pLeft;
  struct node *pRight;
};
typedef struct node TNode;

typedef TNode* TREE;
```



Duyệt Cây Nhị Phân

- Đảm bảo đến **mỗi node trên cây chính xác một lần** một cách **có hệ thống**
- Nhiều thao tác xử lý trên cây **cần phải sử dụng đến phép duyệt cây**
- Các phép cơ bản dựa trên **trình tự thăm gốc**
  - **Duyệt trước** (pre-oder)
  - **Duyệt giữa** (in-oder)
  - **Duyệt sau** (post-oder)

Duyệt Cây Nhị Phân



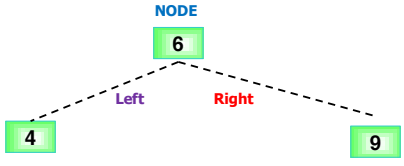
- **Duyệt trước** (pre-oder) – In thông tin node gốc trước
  - **NLR: 6 → 4 →9**
  - **NRL: 6 → 9 →4**

Duyệt Cây Nhị Phân

- **Duyệt trước** (pre-oder) – In thông tin node gốc trước

```
void  NLR(TREE Root)
{
 if (Root != NULL)
 {
     <Xử lý Root>; //Xử lý tương ứng theo nhu cầu
     NLR(Root->pLeft);
     NLR (Root->pRight);
 }
}
```

Duyệt Cây Nhị Phân



- **Duyệt giữa**(in-oder) – In thông tin node gốc thứ 2
  - **LNR: 4 → 6 →9**
  - **RNL: 9 → 6→4**

19

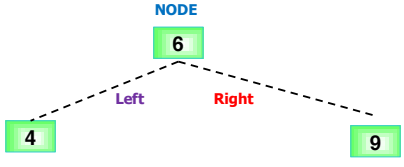
Duyệt Cây Nhị Phân

- **Duyệt giữa**(in-oder) – In thông tin node gốc thứ 2

```
void  LNR(TREE Root)
{
 if (Root != NULL)
 {
     LNR(Root->pLeft);
     <Xử lý Root>; //Xử lý tương ứng theo nhu cầu
     LNR (Root->pRight);
 }
}
```

20

Duyệt Cây Nhị Phân



- **Duyệt sau**(post-oder) – In thông tin node gốc sau
  - **LRN: 4→ 9 →6**
  - **RLN: 9 → 4→6**

21

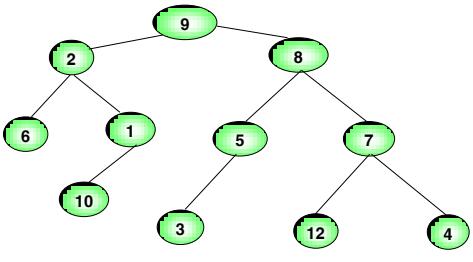
Duyệt Cây Nhị Phân

- **Duyệt sau**(post-oder) – In thông tin node gốc cuối

```
void  LRN (TREE Root)
{
 if (Root != NULL)
 {
     LRN (Root->pLeft);
     LRN (Root->pRight);
     <Xử lý Root>; //Xử lý tương ứng theo nhu cầu
 }
}
```

22

Duyệt Cây Nhị Phân – Ví Dụ

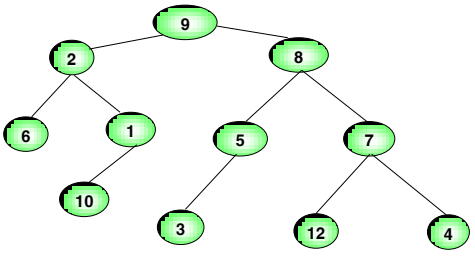


Cho biết kết quả của phép duyệt cây theo thứ tự sau

NLR, NLR, LNR, RNL, LRN, RLN

23

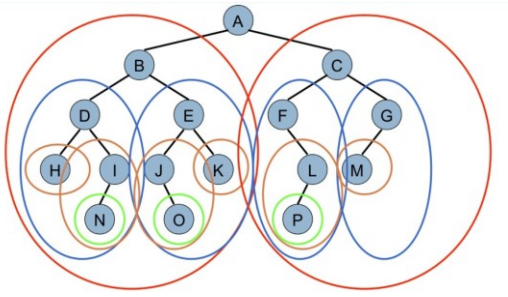
Duyệt Cây Nhị Phân – Ví Dụ



- **NLR**: 9, 2, 6, 1, 10, 8, 5, 3, 7, 12, 4.
- **LNR** : 6, 2, 10, 1, 9, 3, 5, 8, 12, 7, 4.

24

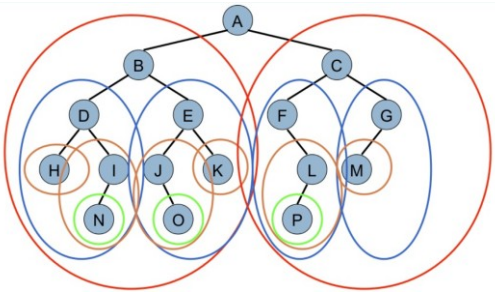
Duyệt Cây Nhị Phân – Ví Dụ



NLR: A B D H I N E J O K C F L P G M

25

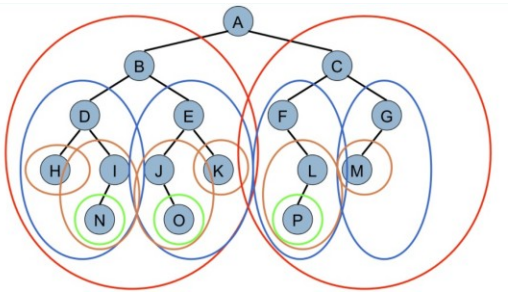
Duyệt Cây Nhị Phân – Ví Dụ



LNR: H D N I B J O E K A F P L C M G

26

Duyệt Cây Nhị Phân – Ví Dụ

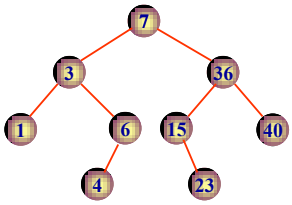


LRN: H N I D O J K E B P L F M G C A

27

Đếm số node trên cây

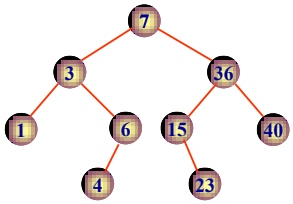
- Nếu **Tree rỗng**, numberNode (Tree) =0
- **Ngược lại**, numberNode(Tree) = 1+ numberNode(cây con trái) + numberNode(cây con phải)



28

Đếm số node lá

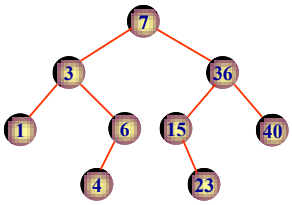
- Nếu **Tree rỗng**, numberLeafNode (Tree) =0
- **Nếu Tree là nút lá**, numberLeafNode (Tree) += 1
- **Nếu Tree không là nút lá**, numberLeafNode (Tree) = numberLeafNode (cây con trái) + numberLeafNode (cây con phải)



29

Tính chiều cao

- Nếu **Tree rỗng**, height (Tree) =0
- **Ngược lại**, height (Tree) = 1 + max(height (cây con trái) ,height (cây con phải))



30

Slide được tham khảo từ

- **Slide được tham khảo từ:**
  - Slide CTDL GT, Khoa Khoa Học Máy Tính, ĐHCNTT
  - Congdongcviet.com
  - Cplusplus.com