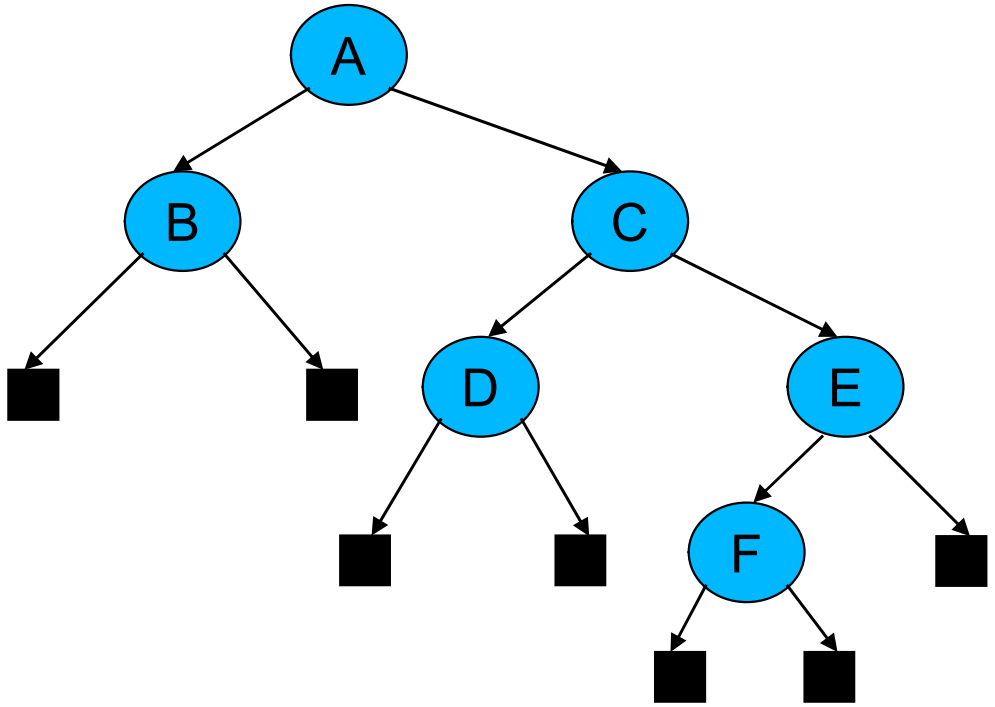
CÂY NHỊ PHÂN

❖ĐỊNH NGHĨA

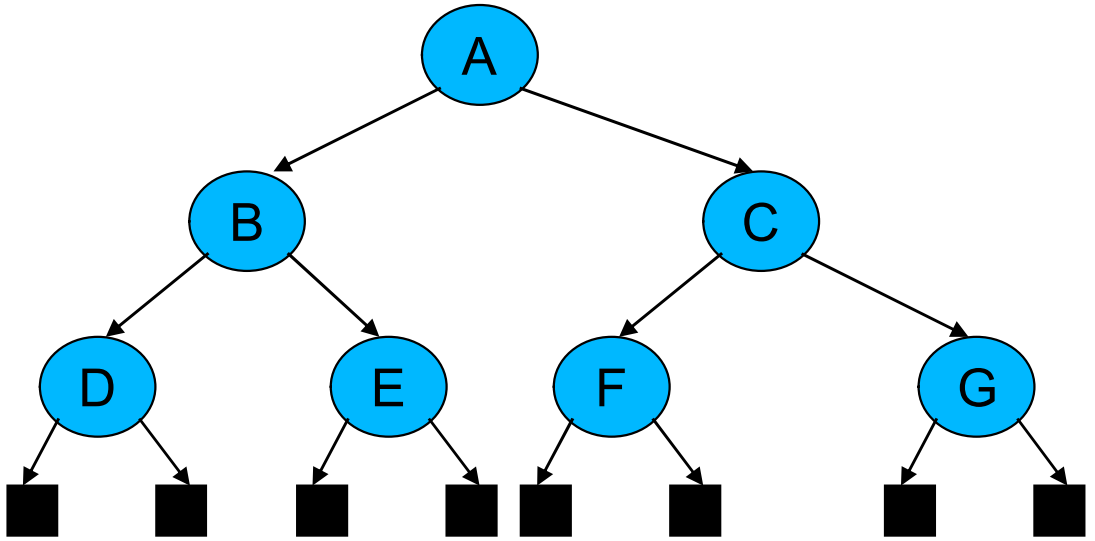
Cây nhị phân là cây mà mỗi node có tối đa hai cây con. Hai cây con này được đặt tên lần lượt là cây con trái và cây con phải.



CÂY NHỊ PHÂN

ĐỊNH NGHĨA

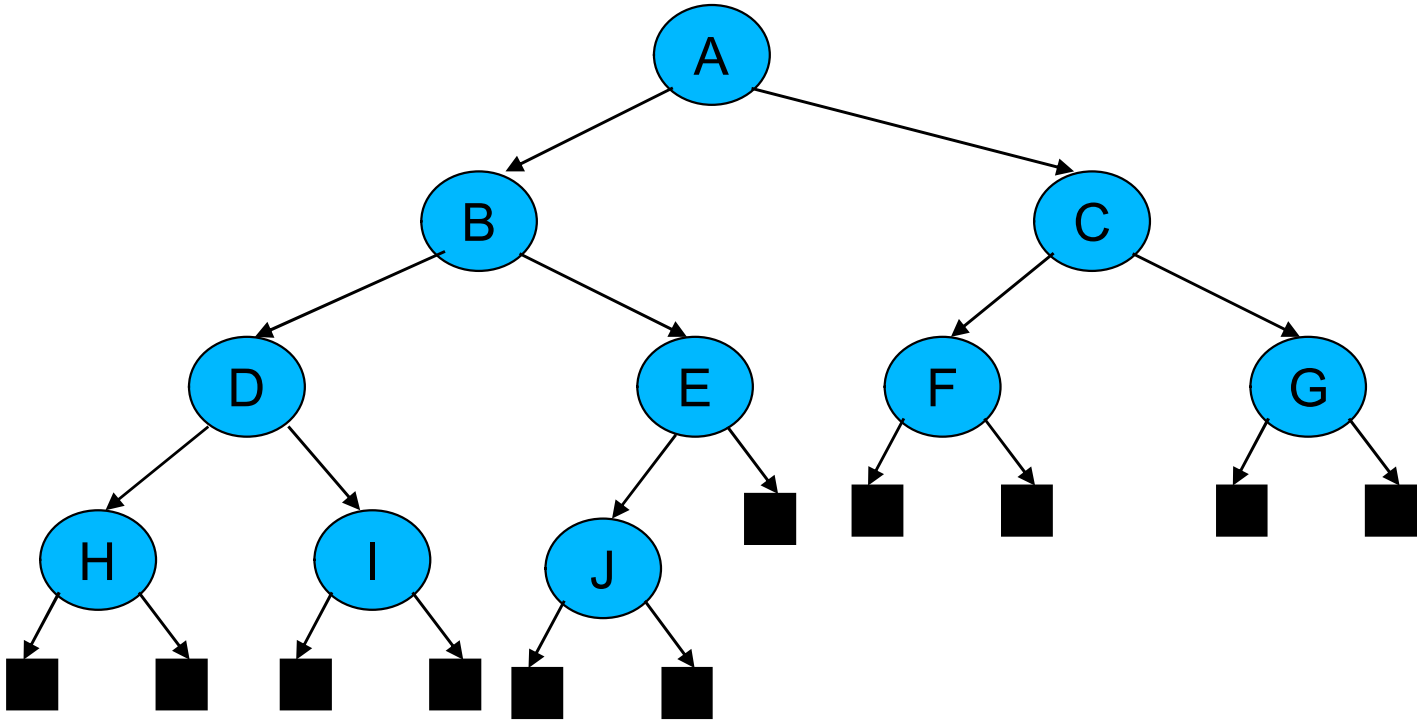
Cây nhị phân đầy đủ là cây nhị phân có các node lá có cùng mức và mỗi node trung gian có đúng 2 node con



CÂY NHỊ PHÂN

❖ĐỊNH NGHĨA

Cây nhị phân hoàn chỉnh có chiều cao n là cây nhị đầy đủ với chiều cao n – 1 và các nút lá lệch trái nhất tại mức n



CÂY NHỊ PHÂN

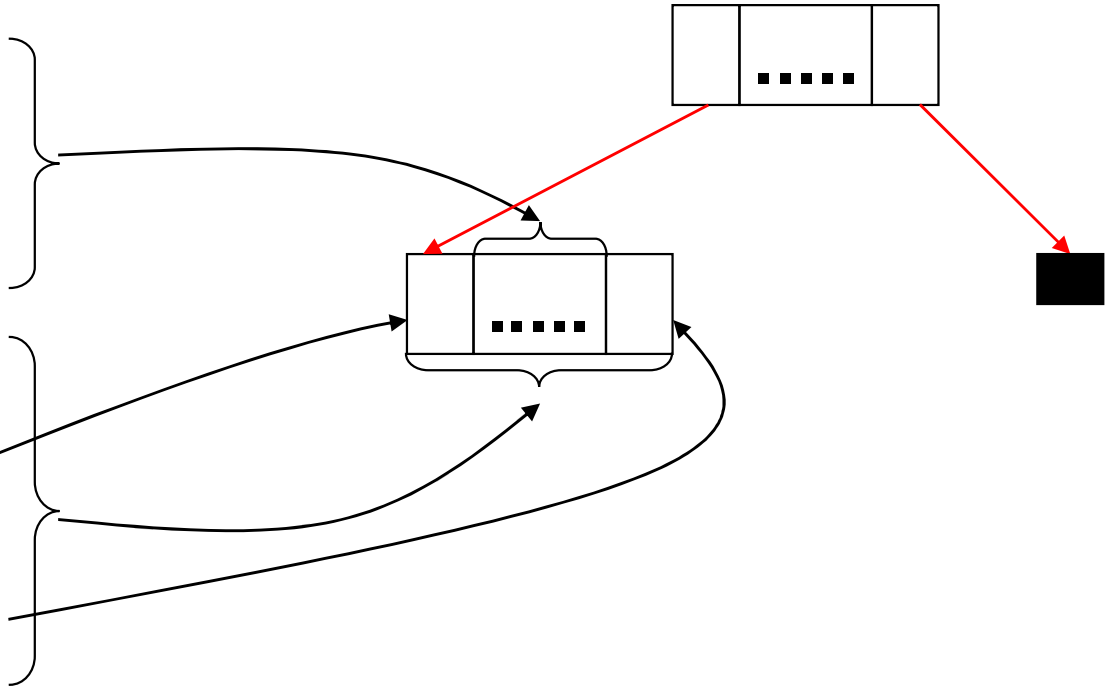
❖TÍNH CHẤT

- Số node tại mức i không quá 2^{i-1}
- Số node của cây T không quá 2^h-1 với h là chiều cao của cây T.
- Gọi n là số node của cây T, h là chiều cao của cây T, có $h \geq \log_2(n + 1)$

CÂY NHỊ PHÂN

❖TỔ CHỨC DỮ LIỆU

```
struct TenDulieu {
    // dữ liệu quản lý
};
struct Node {
    TenDulieu key;
    Node *pLeft, *pRight;
};
typedef Node * TREE;
```



CÂY NHỊ PHÂN

❖CÁC THAO TÁC CƠ BẢN

- Tạo cây rỗng
- Tạo một node có khóa x
- Duyệt cây
- Tạo cây từ kết quả duyệt.

CÂY NHỊ PHÂN

❖CÁC THAO TÁC CƠ BẢN

- Tạo cây rỗng

```
void CreateTree(TREE &root) {
    root = NULL;
}
```

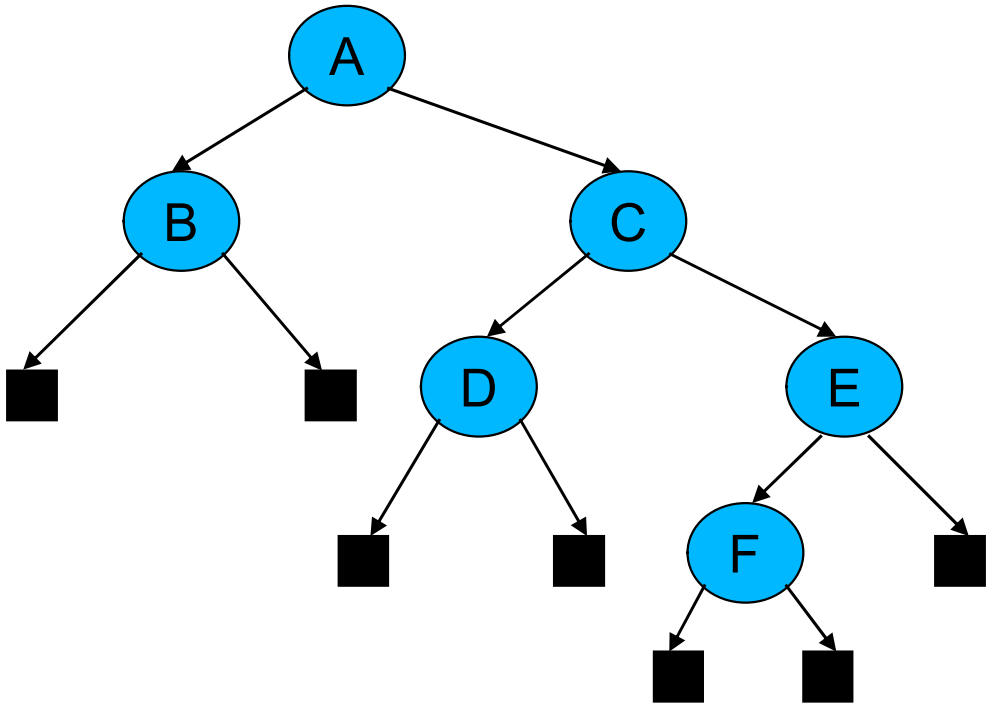
- Tạo node có khóa x

```
Node* CreateNode(TenDulieu x) {
    Node *p = new Node;
    if (p!=NULL)
    { p->key = x; p->pLeft = NULL; p->pRight = NULL;}
    return p;
}
```

CÂY NHỊ PHÂN

❖VÍ DỤ

- Viết chương trình tạo cây nhị phân như sau:



CÂY NHỊ PHÂN

```
struct Node {
    char key;
    Node *pLeft, *pRight;
};
typedef Node *TREE;
void CreateTree(TREE &root) {
    root = NULL;
}
```

CÂY NHỊ PHÂN

```
Node * CreateNode(char x) {
    Node *p = new Node;
    if (p != NULL) {
        p->key = x; p->pLeft = NULL; p->pRight = NULL;
    }
    return p;
}
```

CÂY NHỊ PHÂN

```
TREE CreateVDTree() {
    TREE root;
    CreateTree(root);
    Node *a, *b, *c, *d, *e, *f;
    a = CreateNode('A'); b = CreateNode('B');
    c = CreateNode('C'); d = CreateNode('D');
    e = CreateNode('E'); f = CreateNode('F');
    if ( a && b && c && d && e && f) {
        a->pLeft = b; a->pRight = c; c->pLeft = d;
        c->pRight = e; e->pLeft = f; root = a;
    }
    return root;
}
```

CÂY NHỊ PHÂN

❖CÁC THAO TÁC CƠ BẢN

- Duyệt cây: có 3 thứ tự duyệt dựa trên trình tự xử lý node gốc:
  - Duyệt tiền thứ tự (duyệt trước)
  - Duyệt trung thứ tự (duyệt giữa)
  - Duyệt hậu thứ tự (duyệt sau)

CÂY NHỊ PHÂN

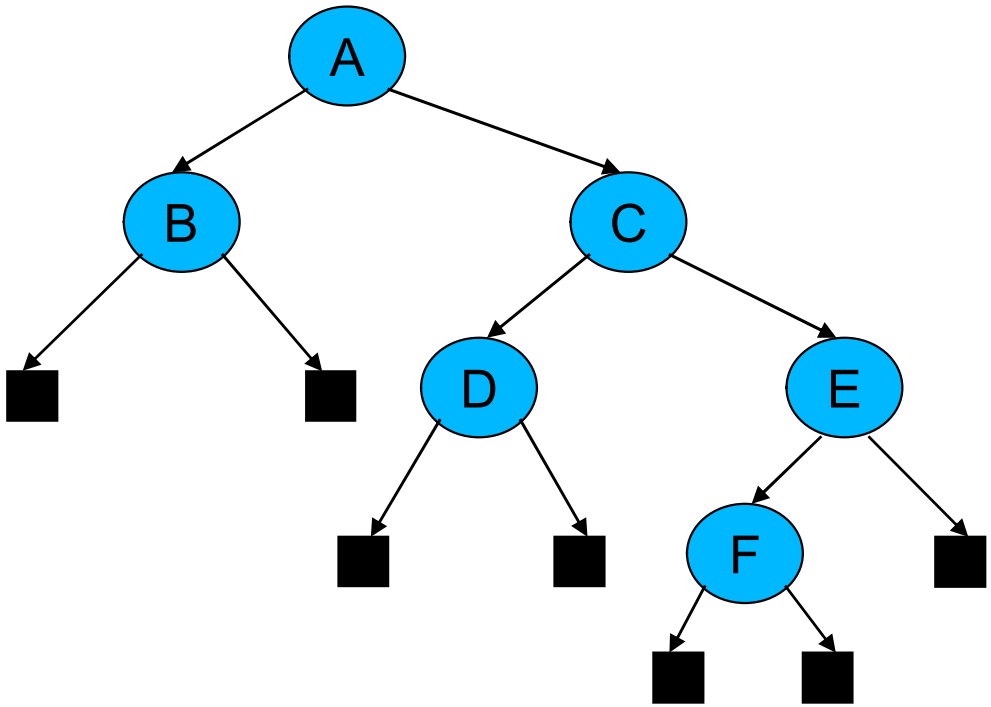
❖CÁC THAO TÁC CƠ BẢN

- Duyệt tiền thứ tự (duyệt trước - Pre-order): nguyên tắc là xử lý nút gốc, sau đó duyệt cây con bên trái rồi duyệt cây con bên phải

Ví dụ:

Kết quả duyệt tiền thứ tự

A B C D E F



CÂY NHỊ PHÂN

- CÁC THAO TÁC CƠ BẢN
  - Duyệt tiền thứ tự (duyệt trước - Pre-order):

```
void NLR(TREE root) {
   if (root) {
   // xử lý root
   NLR(root->pLeft);
   NLR(root->pRight);
   }
}
```

CÂY NHỊ PHÂN

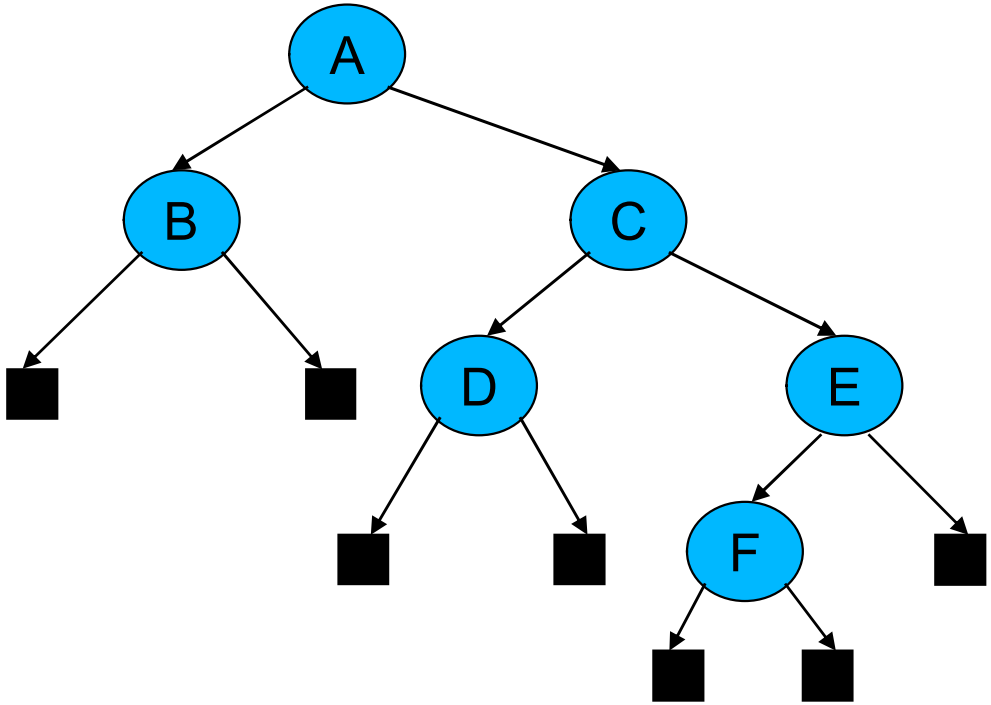
CÁC THAO TÁC CƠ BẢN

- Duyệt trung thứ tự (duyệt giữa - In-order): nguyên tắc là duyệt cây con bên trái, sau đó xử lý nút gốc, rồi duyệt cây con bên phải

Ví dụ:

Kết quả duyệt trung thứ tự

B A D C F E



CÂY NHỊ PHÂN

- CÁC THAO TÁC CƠ BẢN
  - Duyệt trung thứ tự (duyệt giữa - In-order):

```
void LNR(TREE root) {
   if (root) {
   LNR(root->pLeft);
   // xử lý root
   LNR(root->pRight);
   }
}
```

CÂY NHỊ PHÂN

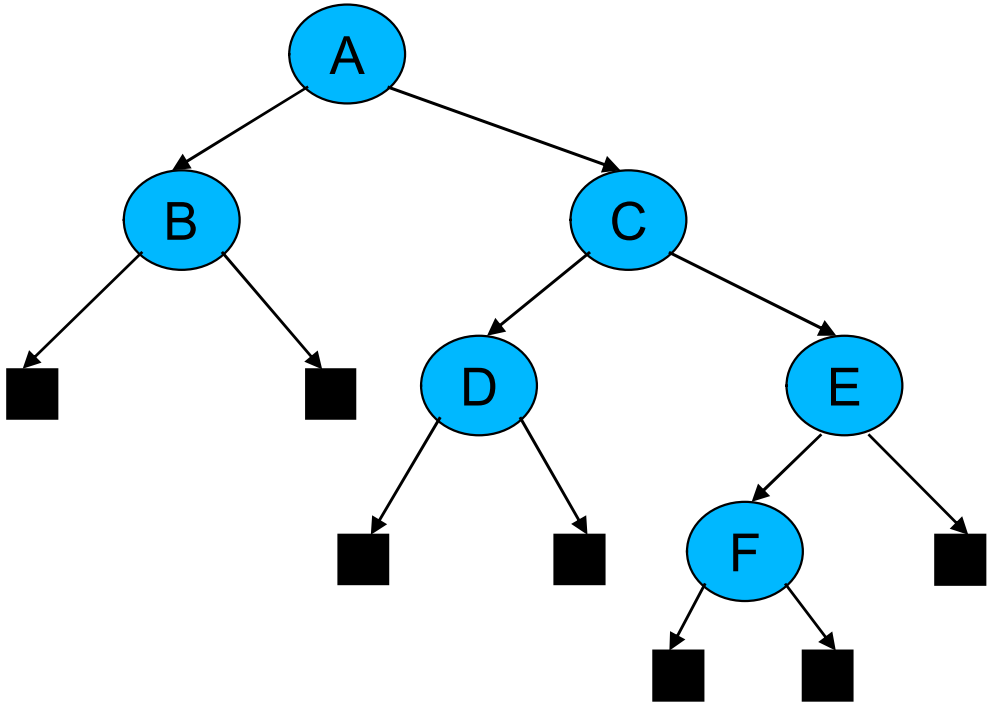
❖CÁC THAO TÁC CƠ BẢN

- Duyệt hậu thứ tự (duyệt sau - Post-order): nguyên tắc là duyệt cây con bên trái, sau đó duyệt cây con bên phải rồi xử lý nút gốc

Ví dụ:

Kết quả duyệt hậu thứ tự

B D F E C A



CÂY NHỊ PHÂN

- CÁC THAO TÁC CƠ BẢN
  - Duyệt hậu thứ tự (duyệt sau - Post-order):

```
void LRN(TREE root) {
   if (root) {
   LRN(root->pLeft);
   LRN(root->pRight);
   // xử lý root
   }
}
```

CÂY NHỊ PHÂN

❖CÁC THAO TÁC CƠ BẢN

- Tạo cây từ kết quả duyệt: có thể tạo cây từ các kết quả duyệt sau:
  - Tiền thứ tự và trung thứ tự
  - Hậu thứ tự và trung thứ tự

CÂY NHỊ PHÂN

❖CÁC THAO TÁC CƠ BẢN

- Tạo cây từ kết quả duyệt tiền thứ tự và trung thứ tự:

Nguyên tắc:

- Node đầu tiên X trong dãy tiền thứ tự là node gốc.
- Node X sẽ nằm trong dãy trung thứ tự, chia dãy này thành 2 dãy bên trái và bên phải. Dãy bên trái sẽ là cây con trái của X, dãy bên phải là cây con phải của X.
- Thực hiện tương tự cho dãy bên trái trước rồi đến dãy bên phải.

CÂY NHỊ PHÂN

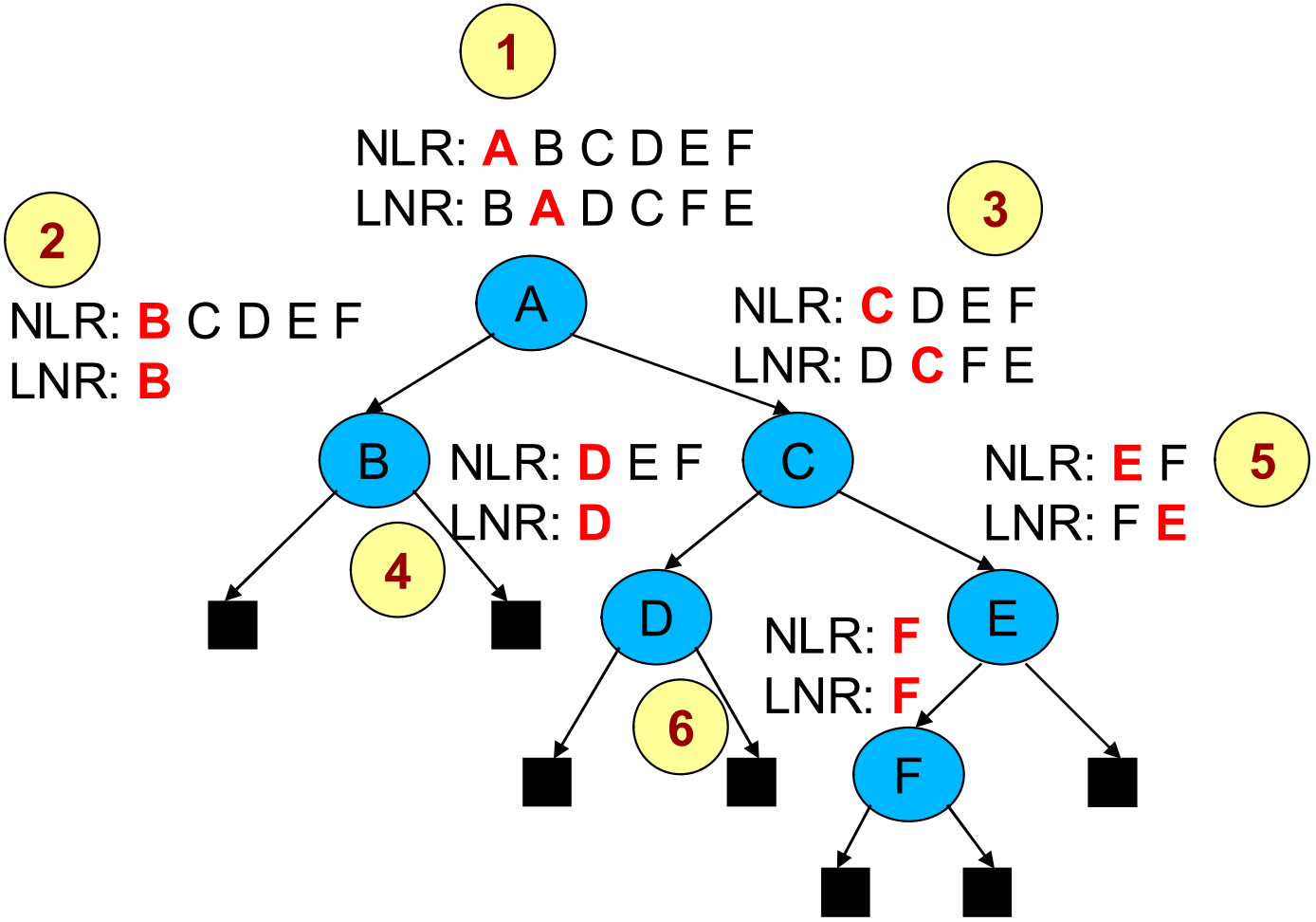
- CÁC THAO TÁC CƠ BẢN
  - Tạo cây từ kết quả duyệt tiền thứ tự và trung thứ tự:

Ví dụ: tạo cây từ kết quả duyệt:

NLR: A B C D E F

LNR: B A D C F E

CÂY NHỊ PHÂN


1
NLR: A B C D E F
LNR: B A D C F E
2
NLR: B C D E F
LNR: B
3
NLR: C D E F
LNR: D C F E
A
B
NLR: D E F
LNR: D
C
NLR: E F
LNR: F E
5
4
D
NLR: F
LNR: F
E
6
F

CÂY NHỊ PHÂN

❖CÁC THAO TÁC CƠ BẢN

- Tạo cây từ kết quả duyệt tiền thứ tự và trung thứ tự:

Giả sử kết quả duyệt là các số nguyên

- pre là kết quả duyệt tiền thứ tự
- in là kết quả duyệt trung thứ tự
- m, n là chỉ số đầu và cuối dãy tiền thứ tự cần xét
- k, l là chỉ số đầu và cuối dãy trung thứ tự cần xét.

CÂY NHỊ PHÂN

```
TREE CreateTree(int *pre, int *in, int m, int n, int k, int l) {
    int i;
    TREE root;
    if (l < k) return NULL;
    root = new Node;
    if (root != NULL) {
        root->key = pre[m];
        for (i = k; i <= l; i++)
            if (in[i] == pre[m]) break;
        root->pLeft = CreateTree(pre, in, m+1, n, k, i - 1);
        root->pRight = CreateTree(pre, in, m+i-k+1, n, i+1,l);
    } return root;
}
```

CÂY NHỊ PHÂN

❖CÁC THAO TÁC CƠ BẢN

- Tạo cây từ kết quả duyệt hậu thứ tự và trung thứ tự:

Nguyên tắc:

- Node cuối cùng X trong dãy hậu thứ tự là node gốc.
- Node X sẽ nằm trong dãy trung thứ tự, chia dãy này thành 2 dãy bên trái và bên phải. Dãy bên trái sẽ là cây con trái của X, dãy bên phải là cây con phải của X.
- Thực hiện tương tự cho dãy bên phải trước rồi đến dãy bên trái.

CÂY NHỊ PHÂN

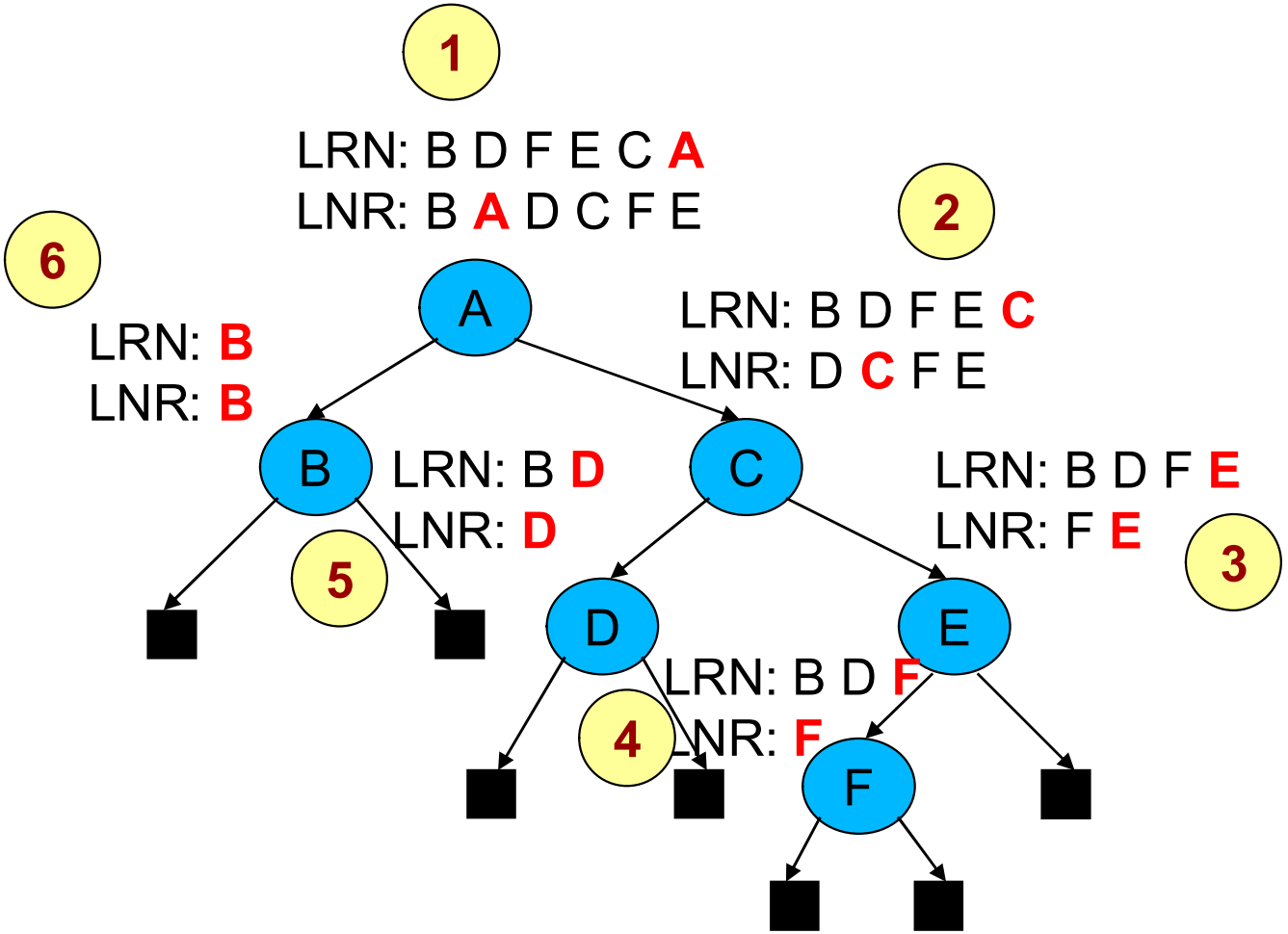
- CÁC THAO TÁC CƠ BẢN
  - Tạo cây từ kết quả duyệt hậu thứ tự và trung thứ tự:

Ví dụ: tạo cây từ kết quả duyệt:

LRN: B D F E C A

LNR: B A D C F E

CÂY NHỊ PHÂN


1
LRN: B D F E C A
LNR: B A D C F E
2
LRN: B D F E C
LNR: D C F E
6
LRN: B
LNR: B
A
B
LRN: B D
LNR: D
C
LRN: B D F E
LNR: F E
3
5
D
E
LRN: B D F
LNR: F
4
F

CÂY NHỊ PHÂN

❖CÁC THAO TÁC CƠ BẢN

- Tạo cây từ kết quả duyệt tiền thứ tự và trung thứ tự:

Giả sử kết quả duyệt là các số nguyên

- post là kết quả duyệt hậu thứ tự
- in là kết quả duyệt trung thứ tự
- m, n là chỉ số đầu và cuối dãy hậu thứ tự cần xét
- k, l là chỉ số đầu và cuối dãy trung thứ tự cần xét.

CÂY NHỊ PHÂN

```
TREE CreateTree(int *post, int *in, int m, int n, int k, int l) {
    int i;
    TREE root;
    if (l < k) return NULL;
    root = new Node;
    if (root != NULL) {
        root->key = post[n];
        for (i = k; i <= l; i++)
            if (in[i] == post[n]) break;
        root->pRight=CreateTree(post,in,m,n-1,i+1,l);
        root->pLeft=CreateTree(post,in,m,n+i-l-1,k,i-1);
    } return root;
}
```