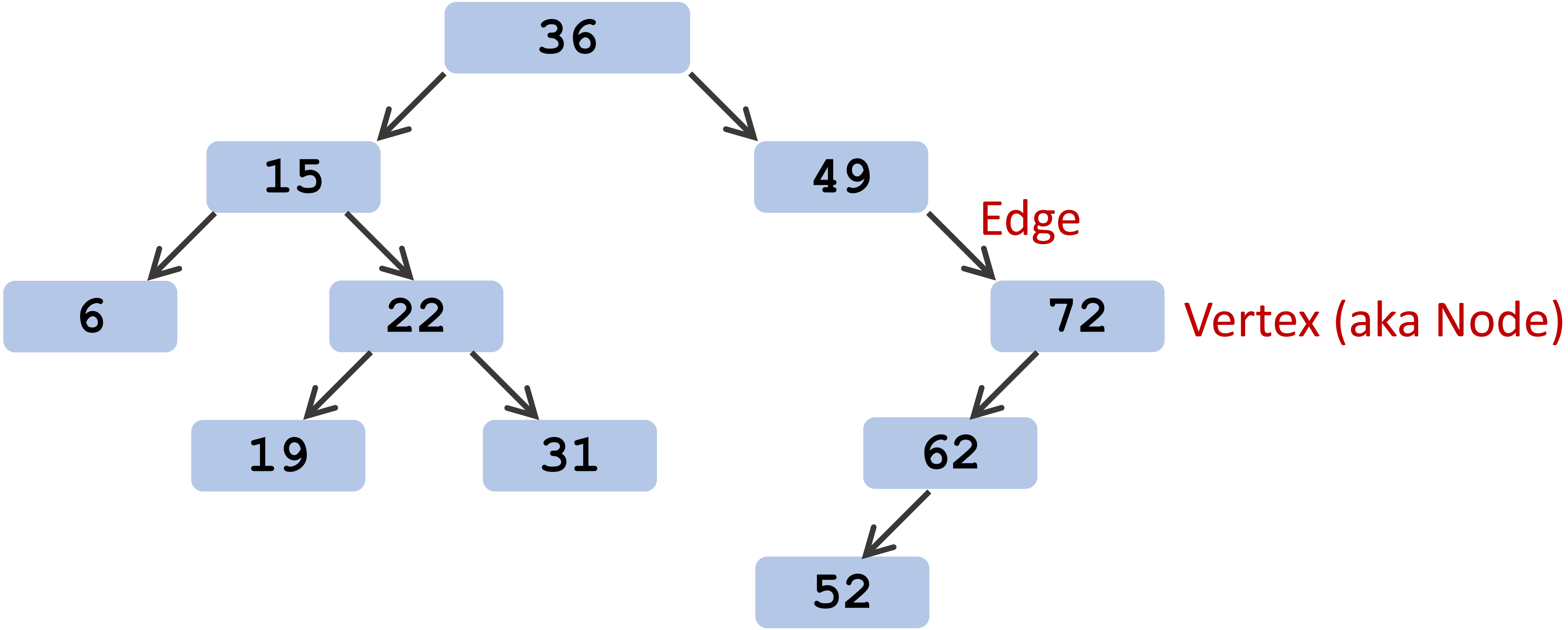
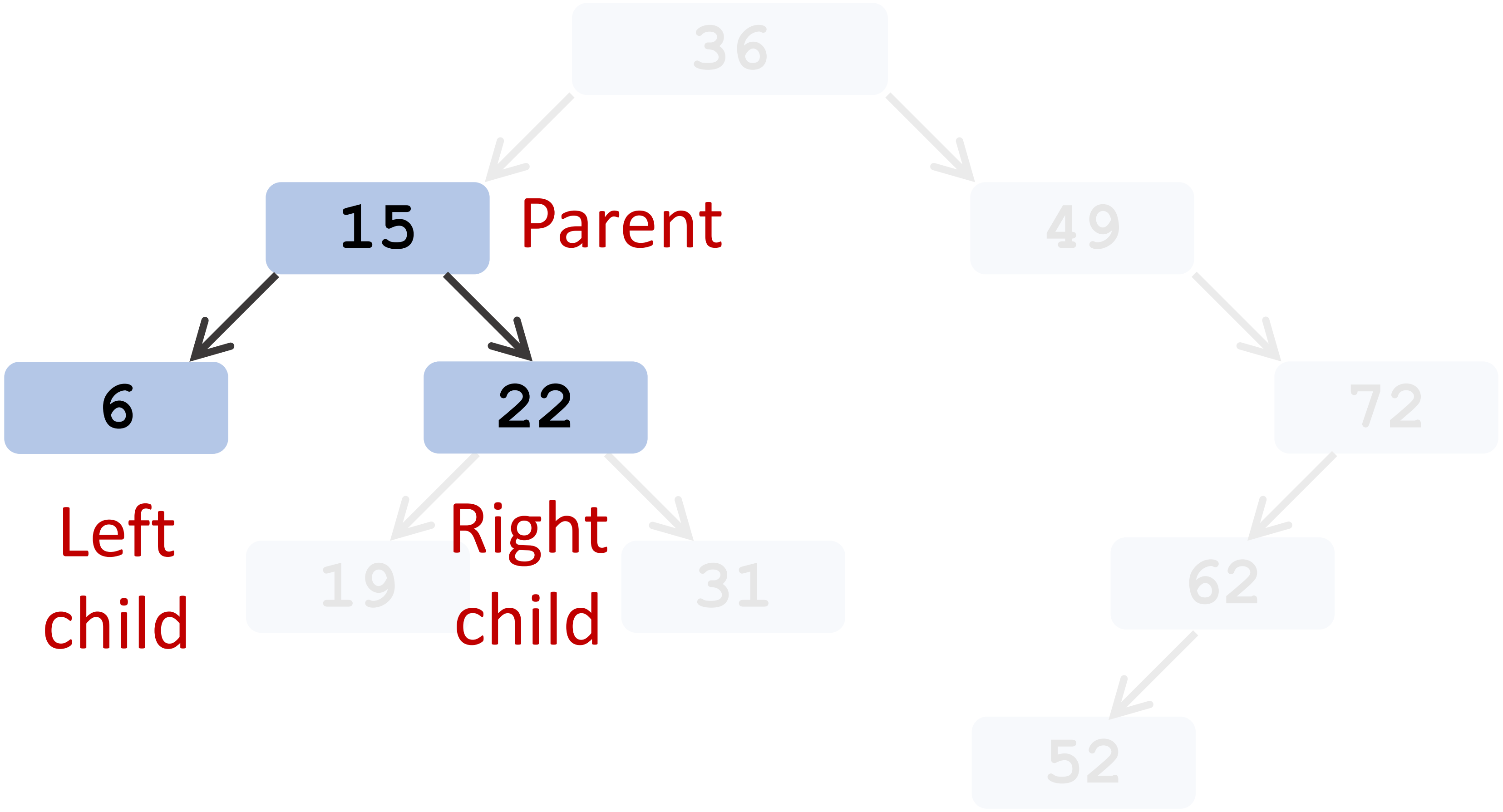
Binary Tree

Shusen Wang

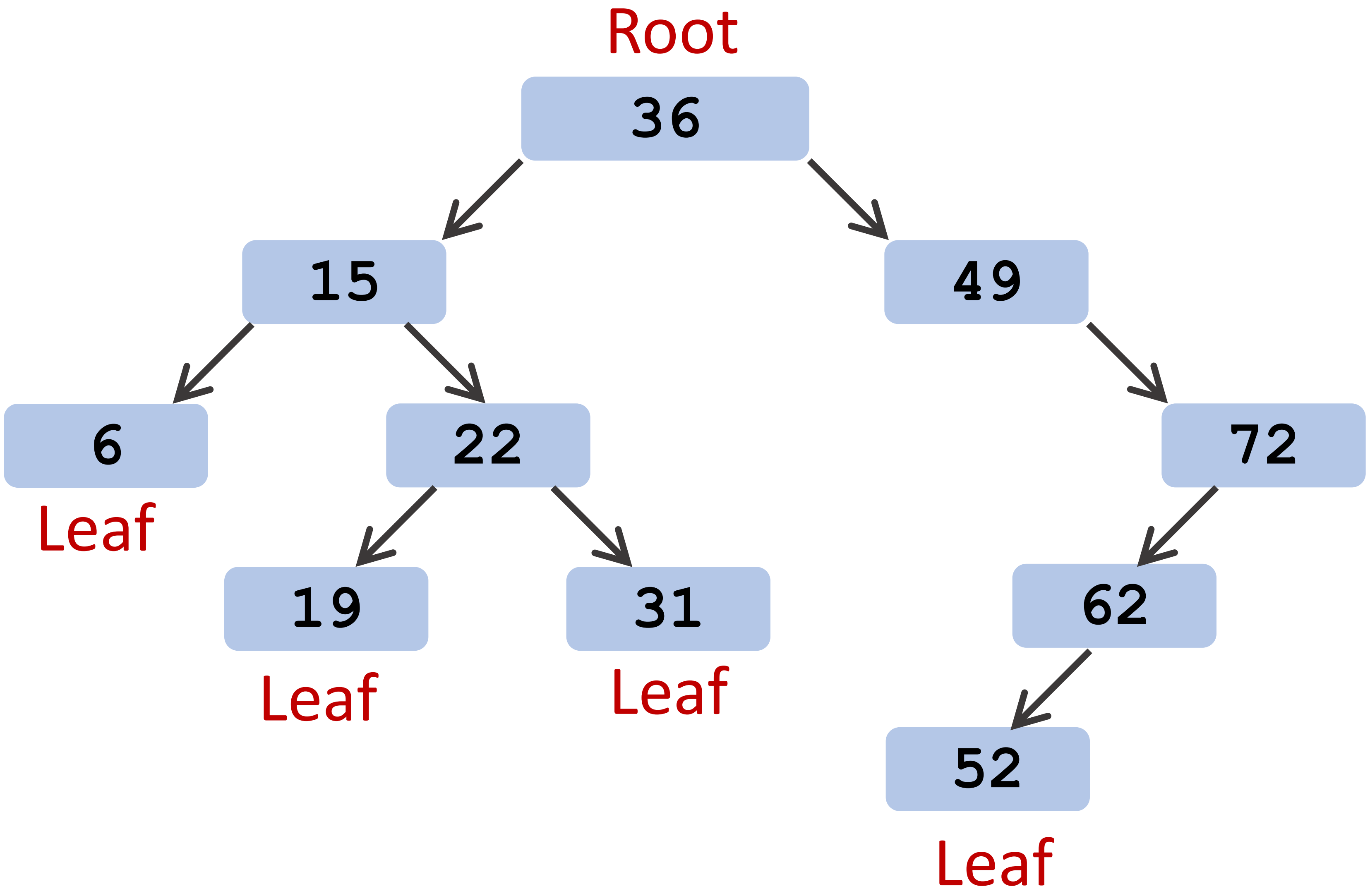
Binary Tree



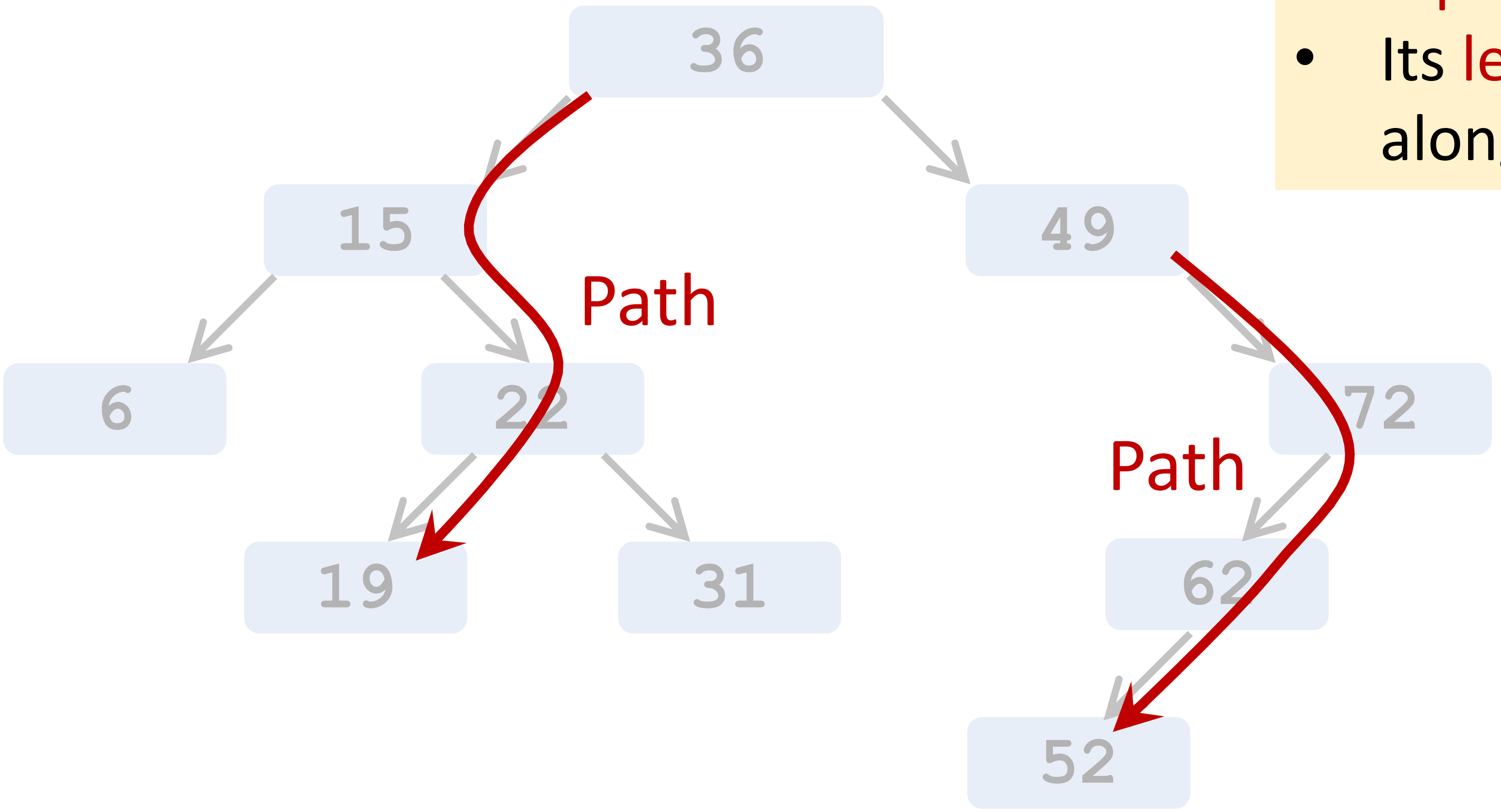
Parent and Children



Root and Leaves

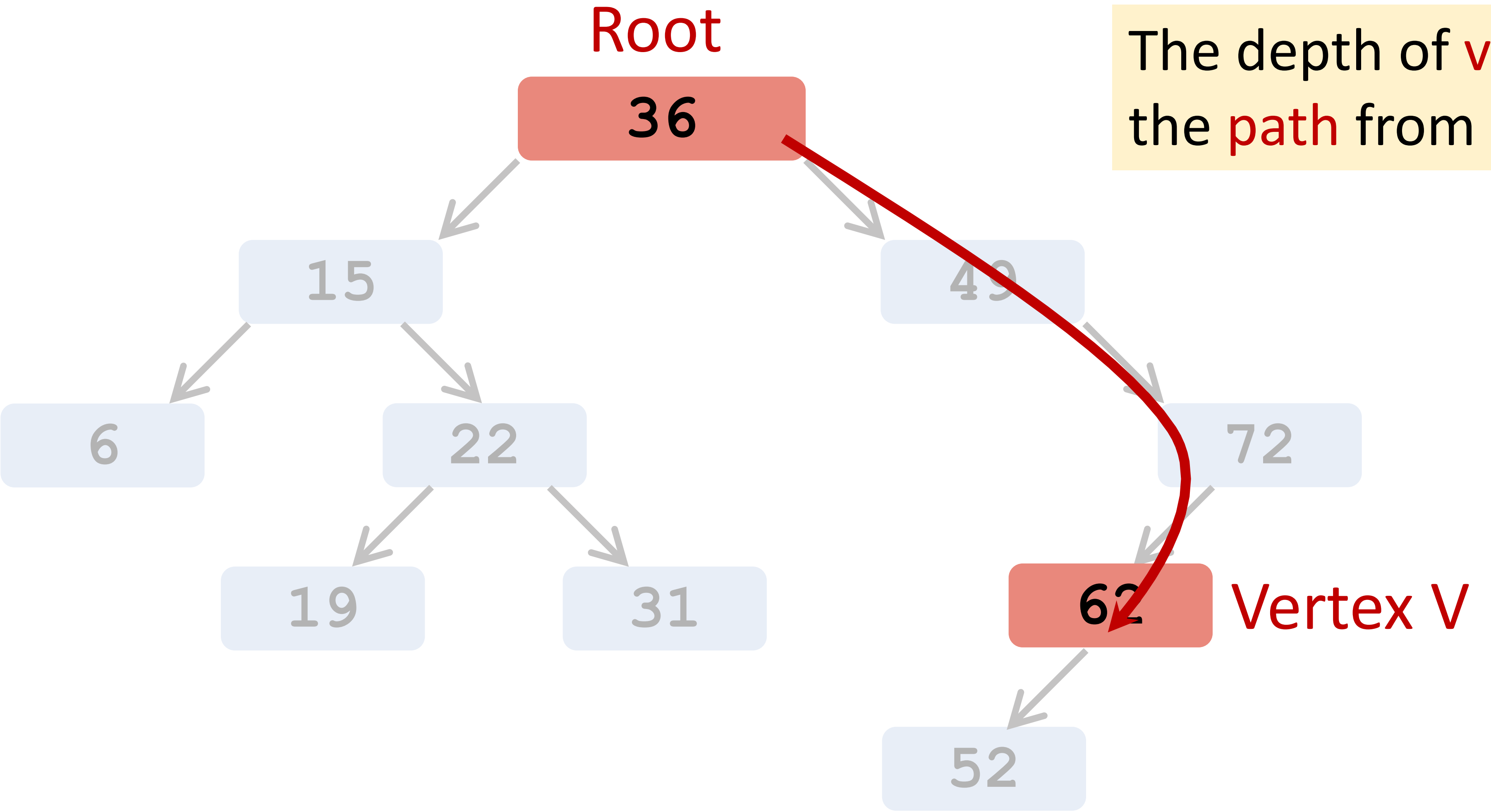


Path



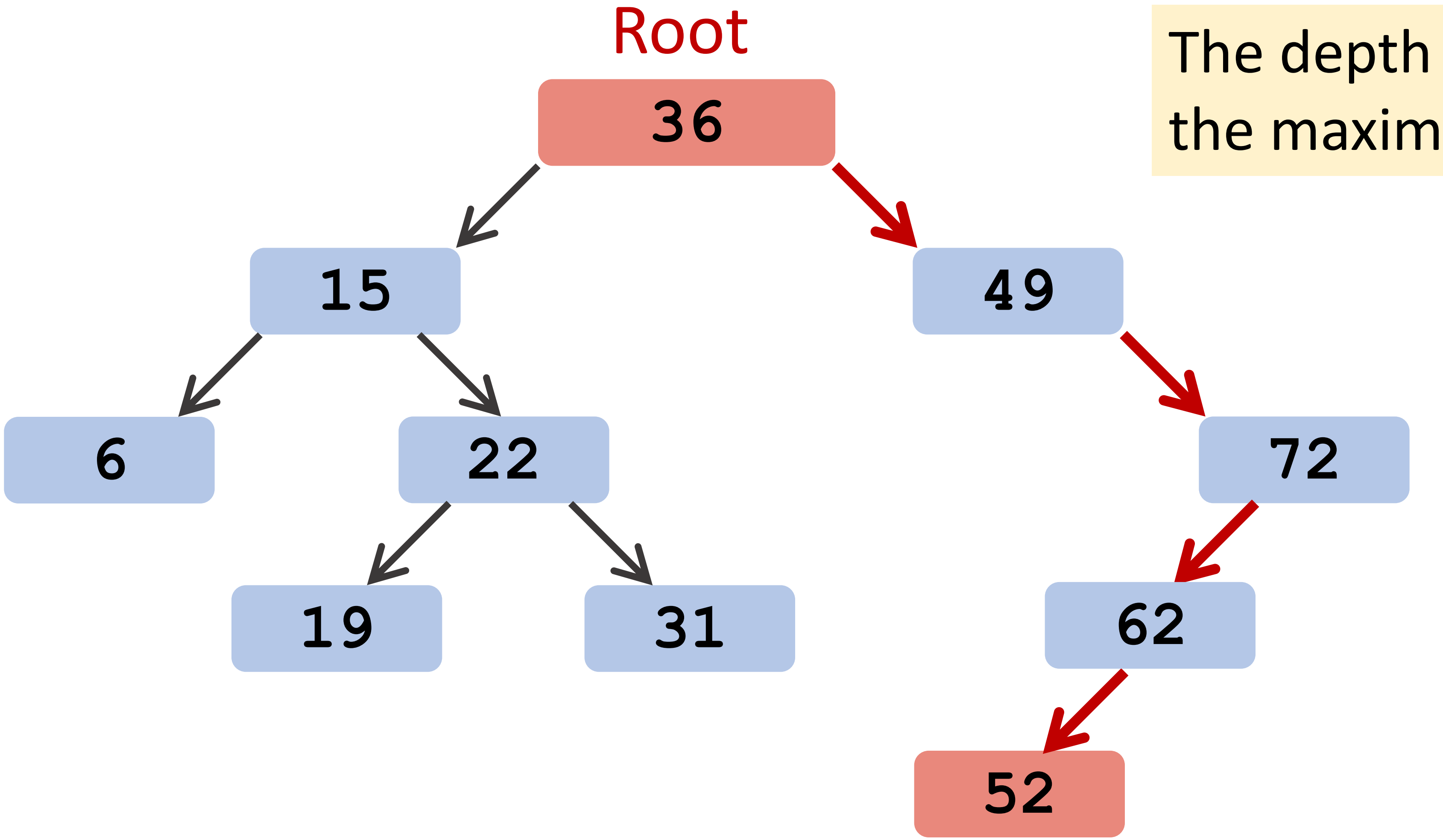
- A path connects two vertices.
- Its length is the number of edges along the path.

Depth of a vertex



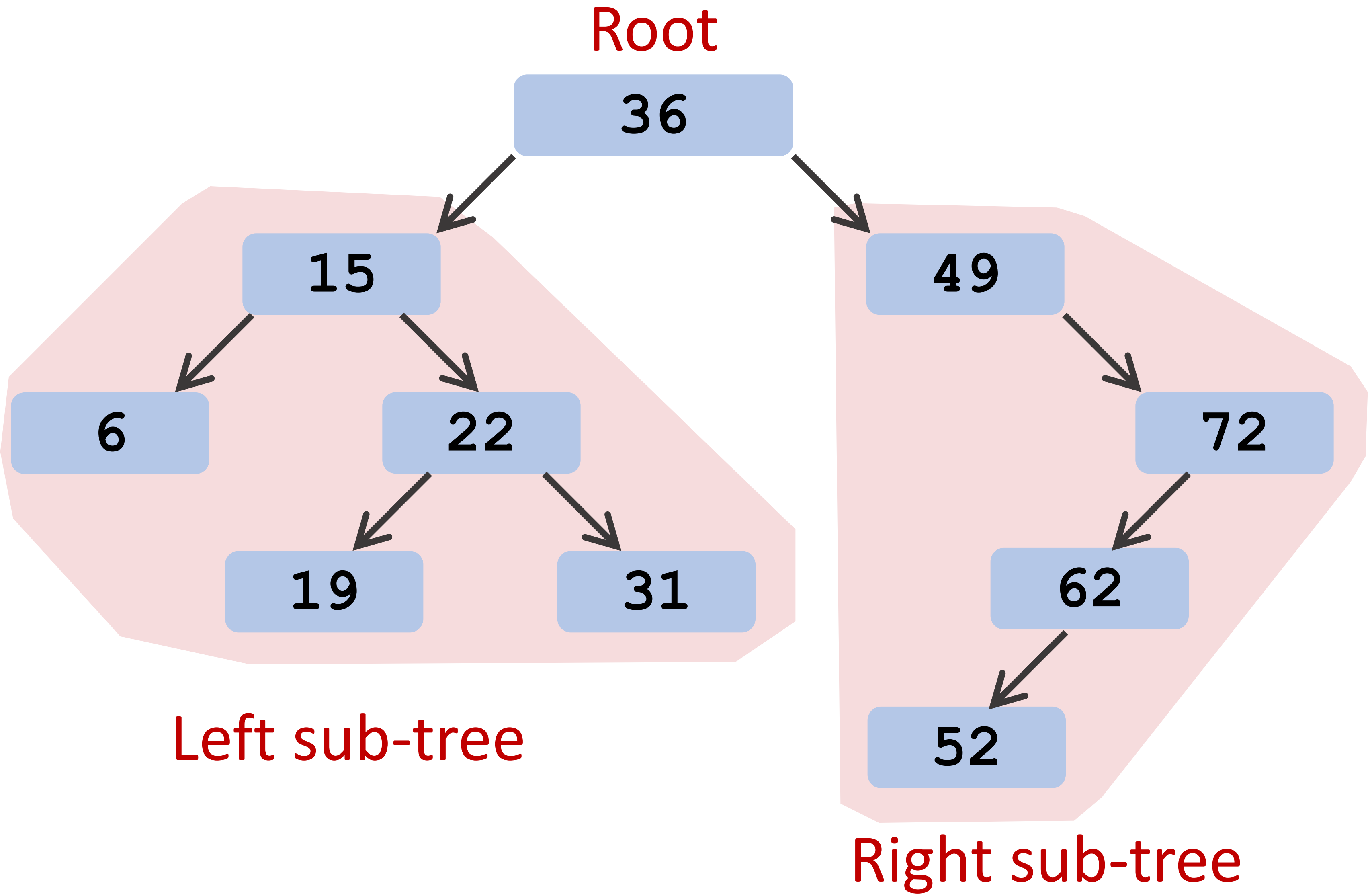
The depth of vertex V is the length of the path from the root to V.

Depth (or height) of the tree



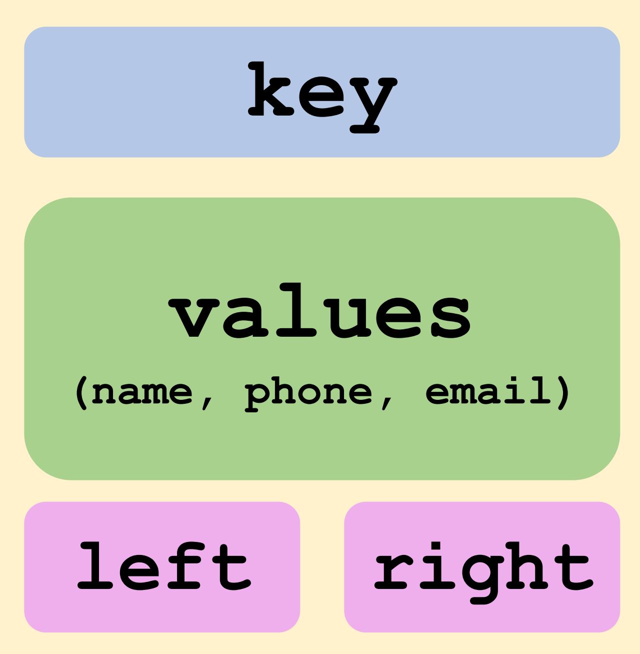
The depth (or height) of the tree is the maximum depth of the vertices.

Sub-trees



Binary Tree Data Structure

Vertex:



key

values

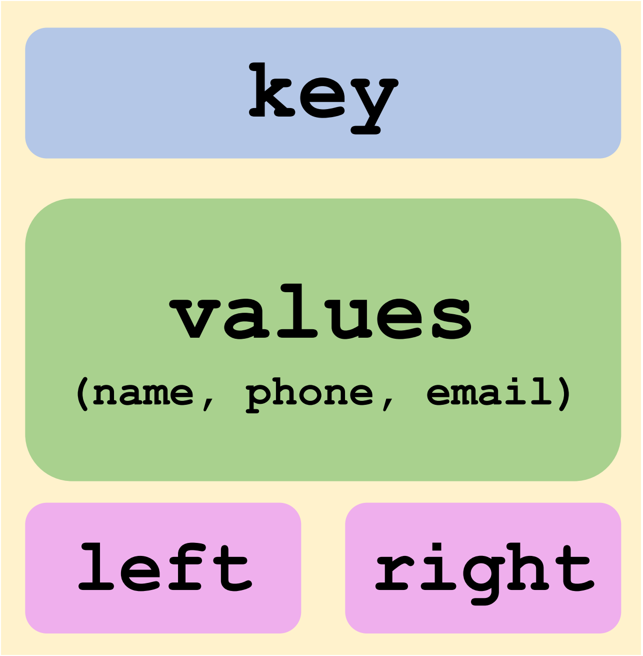
(name, phone, email)

left

right

Binary Tree Data Structure

Vertex:



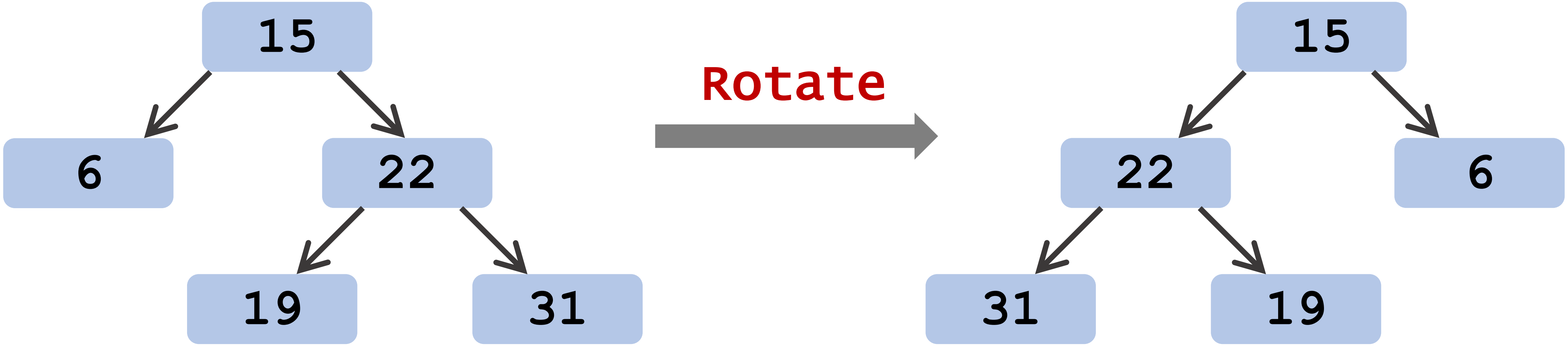
```
struct vertex {
    int key;
    // declare values (optional)
    struct vertex* left;
    struct vertex* right;
};
```

Binary Tree Data Structure

Function for creating a new node.

```
struct vertex* newVertex(int key) {
    struct vertex* v = new vertex;
    v->key = key;
    v->left = NULL;
    v->right = NULL;
    return v;
};
```

Rotate a binary tree



Rotate a binary tree

```
void rotate(struct vertex* root) {
    // swap left and right pointers
    vertex* ptr = root->left;
    root->left = root->right;
    root->right = ptr;
    // recursively rotate the children
    if (root->left != NULL) rotate(root->left);
    if (root->right != NULL) rotate(root->right);
}
```

Thank You!