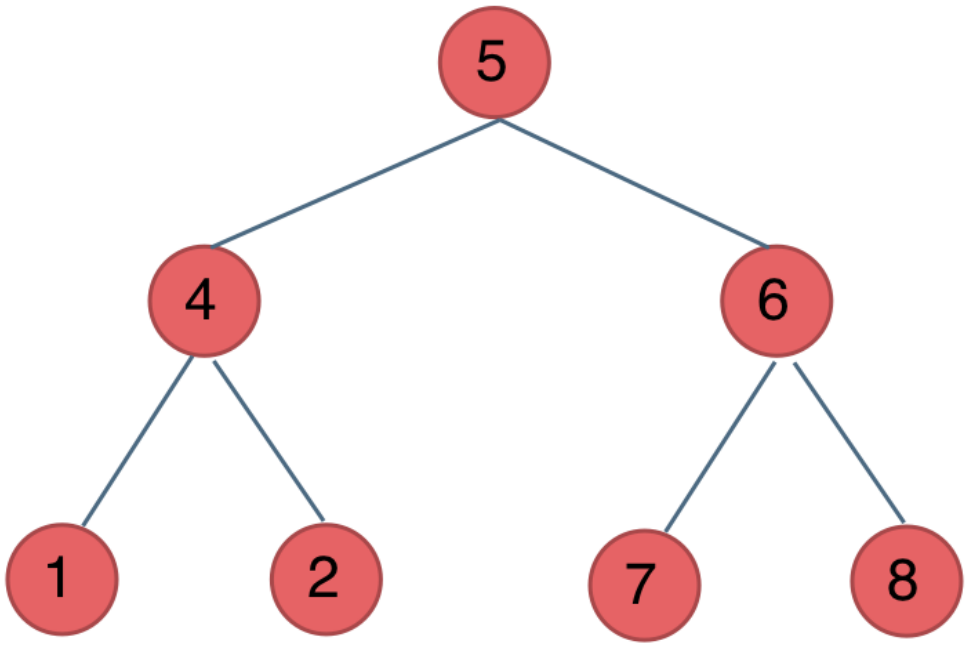
第二周 总结

平衡二叉搜索树：

（来自）
所有左节点的值小于根节点的值， 所有右节点的值大于根节点的值

```
struct  TreeNode{
  int val;
  TreeNode *left;
  TreeNode *right;
  TreeNode(int x): val(x), left(NULL), right(NULL){

  }
}
```



- 前序遍历（中左右）：5412678
- 中序遍历（左中右）：1425768
- 后序遍历（左右中）：1247865

1.递归算法

三要素

——（以前序遍历为例）

- 确定递归函数的参数和返回值
  确定哪些参数是递归的过程中需要处理的，那么就在递归函数里加上这个参数，并且还要明确每次递归的返回值是什么从而确定递归函数的返回条件

```
void traversal(TreeNode* cur, vector<int>& vec)
```

- 确定终止条件：
  防止栈溢出。

```
if (cur==NULL) return;
```

- 确定单层递归的逻辑：
  确定每一层递归需要处理的信息。重复调用自己

```
vec.push_back(cur->val); //中
traversal(cur->left, vec); //左
traversal(cur->right, vec); //右
```

前中后序遍历

——前序遍历

```
class Solution{
  public:
      void traversal(TreeNode* cur, vector<int>& vec){
        if(cur==NULL) return;
        vec.push_back(cur->val);    //中
        traversal(cur->left, vec);  //左
        traversal(cur->right, vec); //右
      }
      vector<int> preorderTraversal(TreeNode* root){
        vector<int> result;
        traversal(root, result);
        return result;
      }
};
```

——中序遍历：

```
void traversal(TreeNode* cur, vector<int>& vec){
  if(cur==NULL) return;
  traversal(cur->left, vec);  //左
  vec.push_back(cur->val);    //中
```

```
    traversal(cur->right, vec); //右
}
```

——后序遍历：

```
void traversal(TreeNode* cur, vector<int>& vec){
    if (cur==NULL) return;
    traversal(cur->left, vec); //左
    traversal(cur->right, vec);//右
    vec.push_back(cur->val);   //中
}
```

问题

- 迭代法到现在还没有掌握
- 对于 递归中 调用自己的语句不是很理解，

```
void traversal(TreeNode* cur, vector<int>& vec){
    if (cur==NULL) return;
    traversal(cur->left, vec); //左
    traversal(cur->right, vec);//右
    vec.push_back(cur->val);   //中
}
```

以此为例， 为什么，调用自己的语句，用 traversal(cur->left, vec), 函数不是被定义为void traversal(TreeNode* cur, vector<int>& vec)。

堆 heap

堆的实现方法， 最主要的是二叉堆，但是我没在例题中找到关于C++ 写的二叉堆， 反而是通过priority_queue来建立小顶堆

hash label

1.在c++中，没有haoshmap，但是有unordered_map: 可以通过key去调用相应value值。
常用函数：
.begin()/ .end() 起始位置/ 末尾位置
.size() 返回有效元素个数
.empty() 判断是否为空
.count() 返回匹配给定主键的元素的个数
（其他的与链表差不多）
2. 自己建立key的数组；
在有效的字母异位词中， 就是建立了一个数组记录字母个数。

为了赶作业，后面划水了。。。