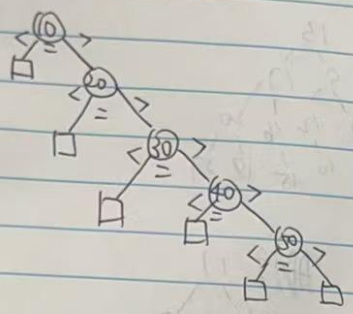
四.1



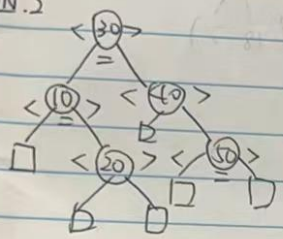
$ASL_{suc} = \frac{1}{5}(1+2+3+4+5)$

$= 3$

$ASL_{unsuc} = \frac{1}{6}(1+2+3+4+5+5)$

$= \frac{10}{3}$

四.2



$ASL_{suc} = \frac{1}{5}(1+2\times2+3\times2) = \frac{11}{5}$

$ASL_{unsuc} = \frac{1}{6}(2\times2+3\times4) = \frac{8}{3}$

四.3 (1)不同 有序: $ASL_{unsuc} = \frac{n(n+3)}{2(n+1)}$

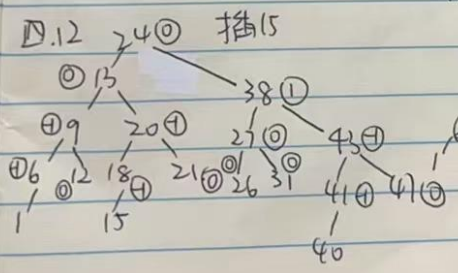
无序: $= n$

(2) 相同 $ASL_{suc} = \frac{n+1}{2}$

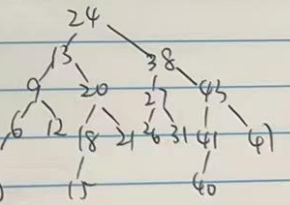
(3) 不同 无序: $ASL_{suc} = n$

有序 $= \frac{n+1}{2} + m$ (假设有m个关键字等于给定值的结点)

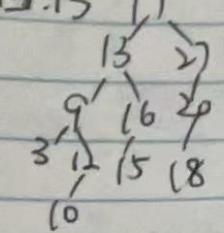
四.12 24① 插15



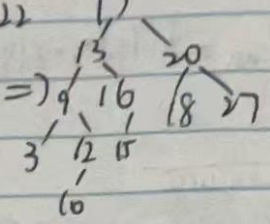
插36

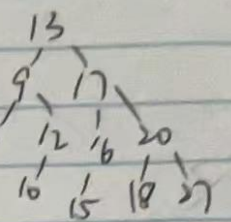


刪13

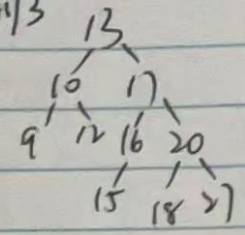


刪22

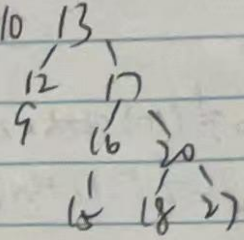




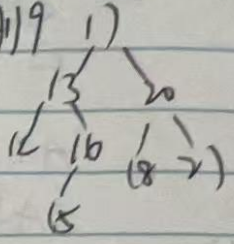
刪3



刪10



刪9



五/2

```
int find(T x) {
    int low = 0;
    int high = CurrentSize - 1;
    int mid;
    while (low <= high) {
        mid = (low + high) / 2;
        if (Element[mid].data == x)return Element[mid].data;
        if (Element[mid].data < x)low = mid + 1;
        else high = mid - 1;
    }
    if (mid == CurrentSize - 1 && Element[mid].data < x)return -1;
    if (Element[mid].data < x)return Element[mid + 1].data;
    else return Element[mid].data;
}
```

五/7

```
bool isBST(TreeNode* root, int min, int max) {
    if (root == NULL) return true;
    if (root->data <= min || root->data >= max) return false;
    else return isBST(root->left, min, root->data) && isBST(root->right, root->data, max);
}
```

五/16

```
bool isBST(TreeNode* root, int min, int max) {
    if (root == NULL) return true;
    if (root->data<min || root->data>max) return false;
    return isBST(root->left, min, root->data) && isBST(root->right, root->data, max);
}
```

```
int height(TreeNode* root) {
    if (root == NULL) return 0;
    int leftheight = height(root->left);
    int rightheight = height(root->right);
    if (leftheight == -1 || rightheight == -1 || abs(leftheight - rightheight) > 1) return -1;
    return max(leftheight, rightheight) + 1;
}
bool isAVL(TreeNode* root) {
    if (!isBST(root, -1000, 1000)) return false;
    if (height(root) == -1) return false;
    else return true;
}
```