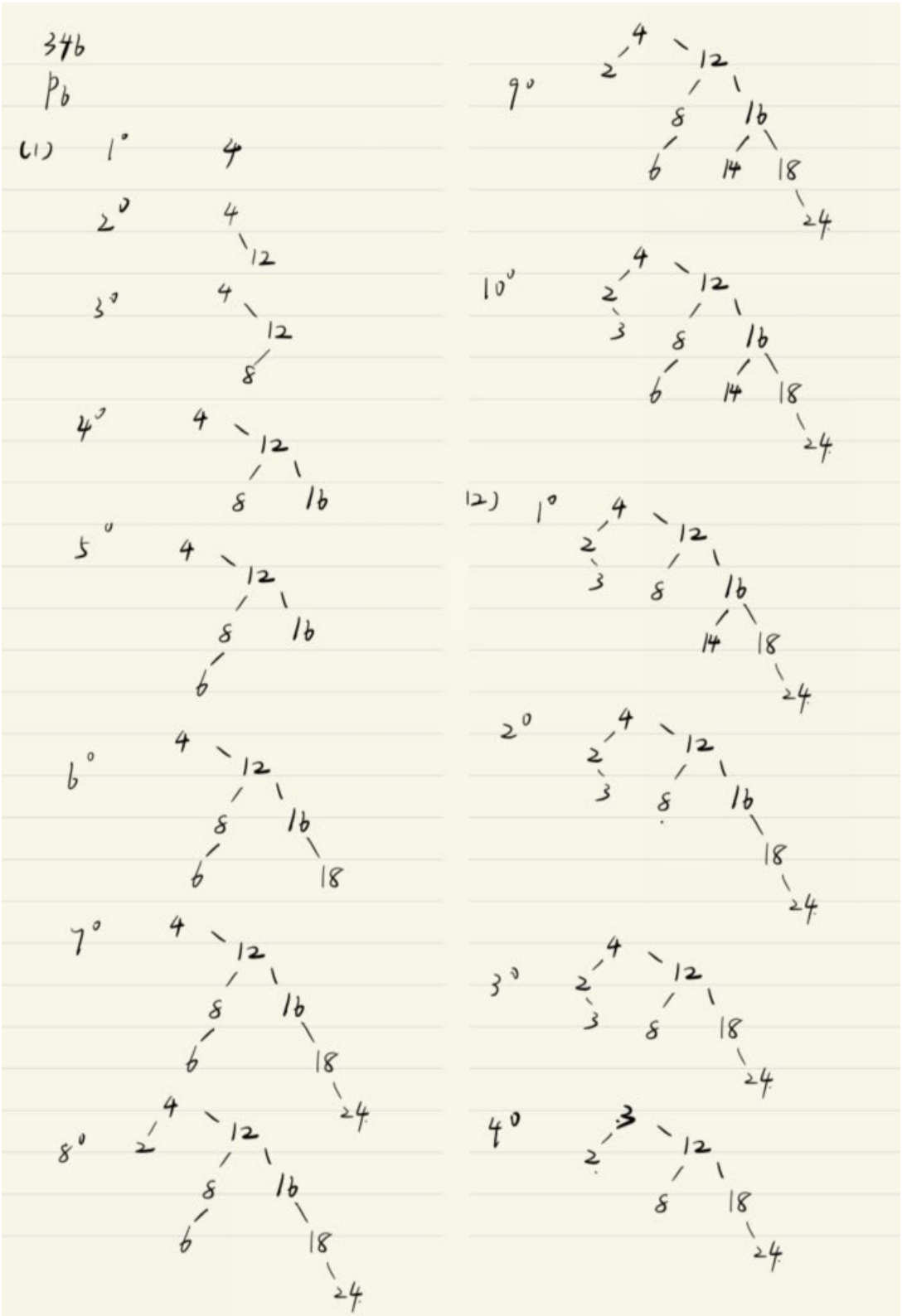
P346 T6



P346 T10

1.

```
#include"binarySearchTree.hpp"
const int N=100010;
```

```
int n;
int a[N];
int index=0;

template <class K, class E>
inline void BinarySearchTree<K, E>::BSSort(BinaryTreeNode<pair<K,E>>
*t)
{
    if(t!=NULL)
    {
        BSSort(t->leftChild);
        a[index++]=t->element.first;
        BSSort(t->rightChild);
    }
}

int main()
{
    BinarySearchTree<int,int>bsTree;
    cin>>n;
    for(int i=0;i<n;i++)
    {
        cin>>a[i];
        bsTree.insert(pair<int,int>(a[i],a[i]));
    }
    bsTree.BSSort(bsTree.root);
    for(int i=0;i<n;i++)
        cout<<a[i]<<" ";
}
```

测试输入

5

3 1 2 5 4

输出

1 2 3 4 5

2. 与插入排序和堆排序进行比较

二叉搜索树排序:

平均情况下，二叉搜索树排序的时间复杂度为 O(nlogn)。但是，如果树不平衡，最坏情况下时间复杂度可能会达到 O(n^2)。

插入排序：

对于随机数据，插入排序的平均运行时间为 O(n^2)。但在某些情况下，对于部分有序的输入，它可能表现得更好，平均时间可以更接近 O(n)。

堆排序：

堆排序的平均运行时间为 O(n log n)。它的性能相对稳定，不受输入数据的影响，保持在 O(n

log n)的时间复杂度。

总结

因此，在平均情况下，堆排序通常具有比较稳定且较快的运行时间。而二叉搜索树排序和插入排序可能会受到数据特性的影响，导致性能波动。