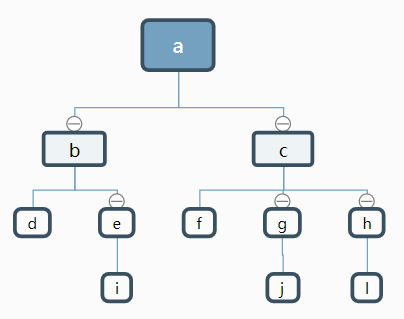
1.

T



1） 根节点： a

叶子节点： d，i，f，j，l

g的双亲： c

g的祖先： a，c

g的孩子： j

e的子孙： i

e的兄弟： d

f的兄弟： g，h

2） b的层次： 2

h的层次： 3

树的深度： 4

c子树深度： 3

2.

1） 1，k，k^2，…，k^(h - 1)

2）

假设i的层数为m，

则前m-1层共(k^(m - 1) – 1) / (k - 1)个元素，m-2层共(k^(m - 2) – 1) / (k - 1)个元素。

可得ans = ceil((I - (k^(m - 1) – 1) / (k - 1)) / k) + (k^(m - 2) – 1) / (k - 1) = ceil((I – 1) / k)

因为参考答案：(I - 2) / k + 1 = floor((I + k - 2) / k)

所以要证答案ceil((I – 1) / k)正确，需证ceil((I – 1) / k) = floor((I + k - 2) / k)

因为(I + k - 2) / k - (I – 1) / k = (I - 1) / k

所以当(I - 1) / k = ceil((I - 1) / k)时

floor((I + k – 2) / k) = floor(ceil((I - 1) / k) + (k - 1) / k) = ceil((I - 1) / k)，

当(I - 1) / k < ceil((I - 1) / k)时

所以设i0, i1使 (i0 – 1) / k = ceil((I - 1) / k) – 1, (i1 - 1) / k = ceil((I - 1) / k)

此时因为i为整数所以I = i0 + 1, i0 + 2, … i1

所以(I - 1) / k >= (I0 - 1) / k + 1 / k = ceil((I - 1) / k) – 1 + 1 / k = ceil((I - 1) / k) – (k - 1) / k

即(I + k – 2) / k >= ceil((I - k) / k)

因为(I + k – 2) / k = (I - 1) / k + (k - 1) / k <= ceil((I - 1) / k) + (k - 1) / k < ceil((I - 1) / k) + 1

所以floor((I + k - 2) / k) = ceil((I – 1) / k)

综上，答案为ceil((I – 1) / k)或(I - 2) / k + 1

3） 设i的第k个child的编号为n，n的层数为m

I = (n - (k^(m - 1) – 1) / (k - 1)) / k + (k^(m - 2) – 1) / (k - 1) = (n – 1) / k

所以n = I \* k + 1

所以i的第j个child的编号：ans = (l – 1) \* k + j + 1

4） 条件： logk(i \* (k - 1) + 1)不是整数

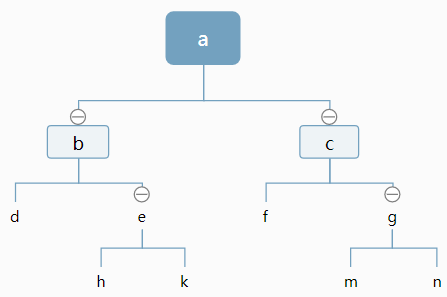
i + 1

3.

1） 顺序存储方法：

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Position | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Value | a | b | c | d | e | f | g | h | k | m | n |
| Parent | -1 | 0 | 0 | 1 | 1 | 2 | 2 | 4 | 4 | 6 | 6 |

链接存储方法：



T

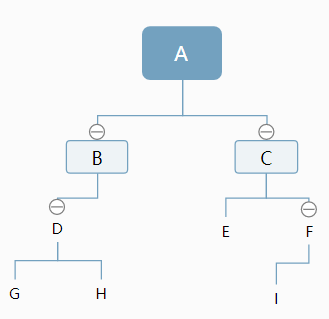
2） 先序：a，b，d，e，h，k，c，f，g，m，n

中序：d，b，h，e，k，a，f，c，m，g，n

后序：d，h，k，e，b，f，m，n，g，c，a

4.

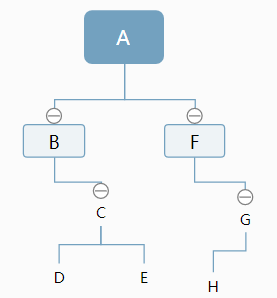
T



G，H，D，B，E，I，F，C，A

T

5.

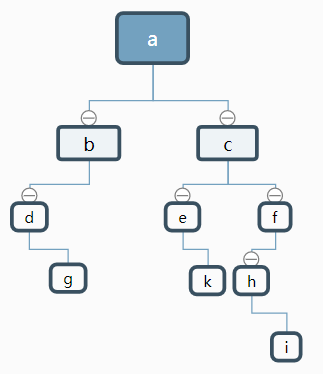


A，B，C，D，E，F，G，H

6.

T

中序线索树：

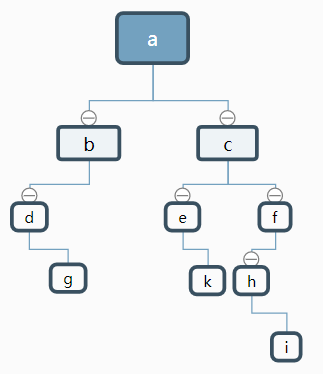


T

NULL

NULL

后序线索树：



NULL

7.

1） 双亲表示法：

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Position | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Value | a | b | c | m | d | e | h | k | g | f | n |
| Parent | -1 | 0 | 0 | 0 | 1 | 1 | 2 | 2 | 2 | 3 | 3 |

孩子表示法：

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Position | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Value | a | b | c | m | d | e | h | k | g | f | n |
| Children | 1,2,3 | 4,5 | 6,7,8 | 9,10 |  |  |  |  |  |  |  |

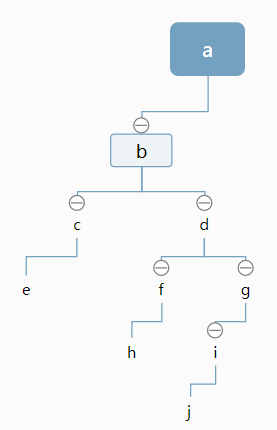
孩子兄弟表示法：

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Position | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Value | A | b | c | m | d | e | h | k | g | f | n |
| First\_child | 1 | 4 | 6 | 9 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| Next\_sibling | -1 | 2 | 3 | -1 | 5 | 6 | 7 | 8 | 9 | 10 | -1 |

8.

T

1）



2） 前序：a，b，c，e，d，f，h，g，i，j

中序：e，c，b，h，f，d，j，i，g，a

后序：e，c，h，f，j，i，g，d，b，a

9.

|  |  |  |
| --- | --- | --- |
| √ | √ | √ |
|  |  |  |
| √ | √ |  |
|  | √ | √ |

10.

stack1.push\_back(\*t);

while (!stack1.empty()) {

back = new node(stack1.back());

stack1.pop\_back();

stack2.push\_back(back->value);

if (back->l) {

stack1.push\_back(\*(back->l));

}

if (back->r) {

stack1.push\_back(\*(back->r));

}

}

while (!stack2.empty()) {

cout << stack2.back();

stack2.pop\_back();

}

11.

node\* find\_x\_parent (node\* t, int x) {

if (t->l) {

if (t->l->value == x) {

return t;

}

else {

node\* ans = find\_x\_parent(t->l, x);

if (ans) {

return ans;

}

}

}

if (t->r) {

if (t->r->value == x) {

return t;

}

else {

node\* ans = find\_x\_parent(t->r, x);

if (ans) {

return ans;

}

}

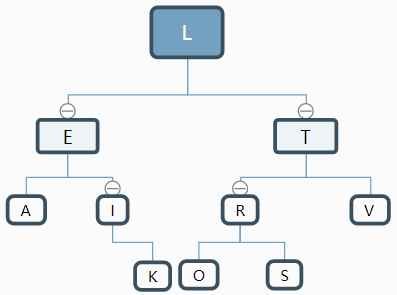
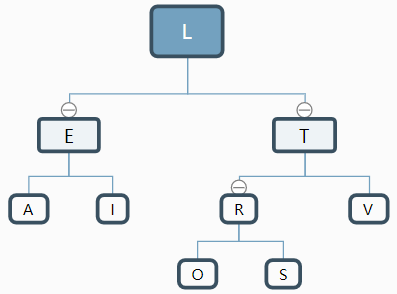
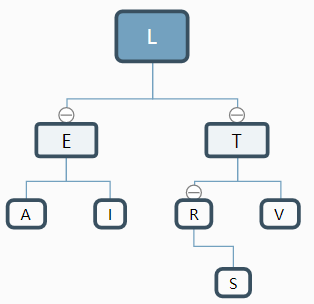
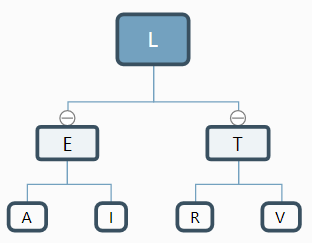
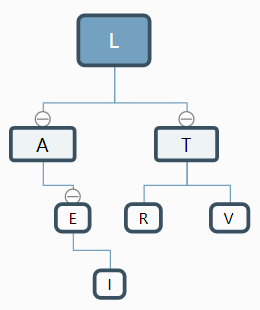
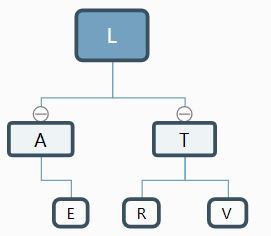
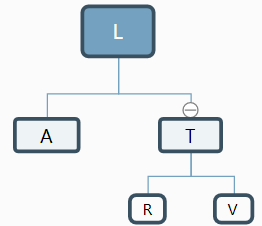
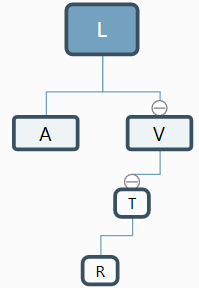
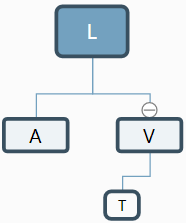
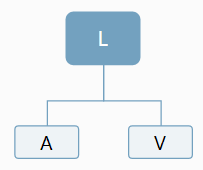
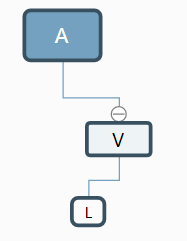
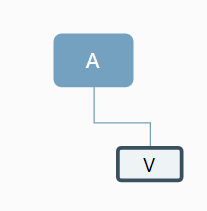
}

return nullptr;;

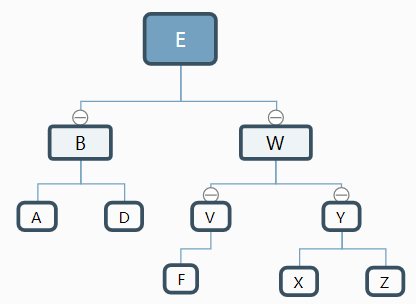
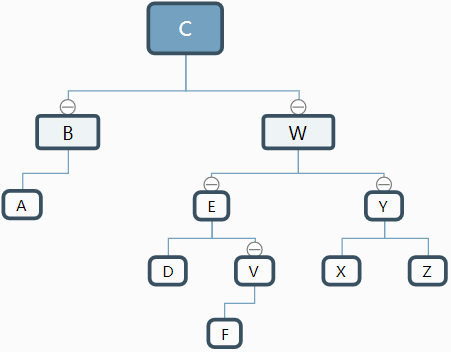
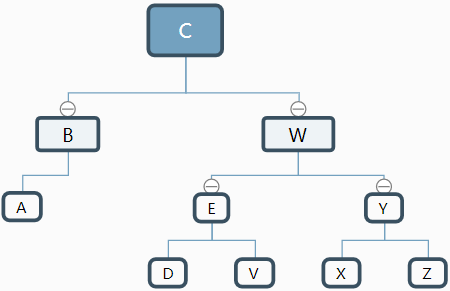
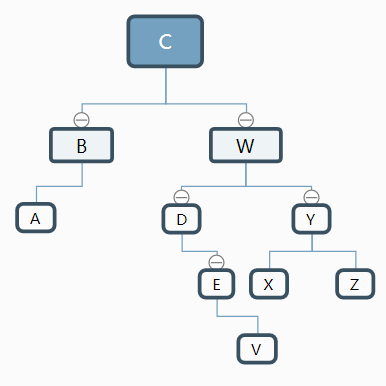
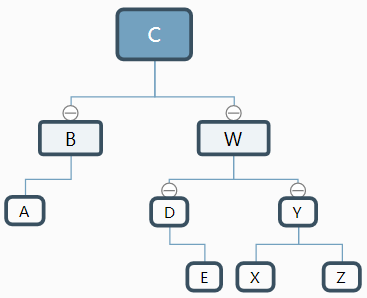
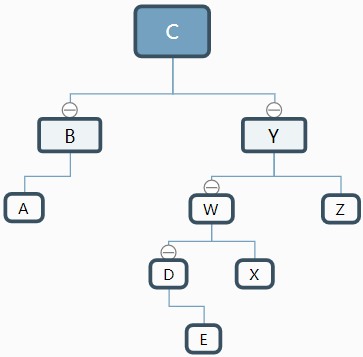
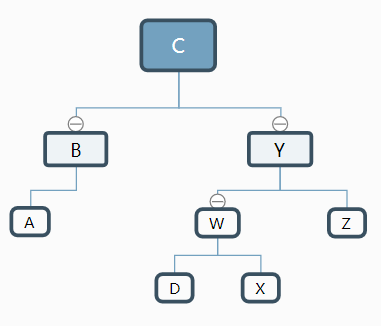
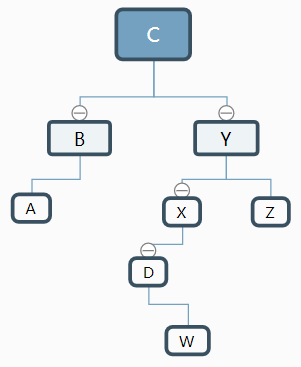
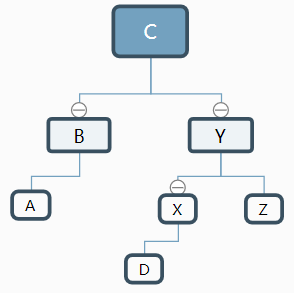
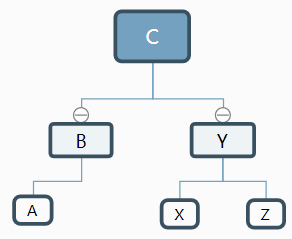
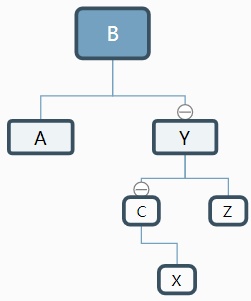
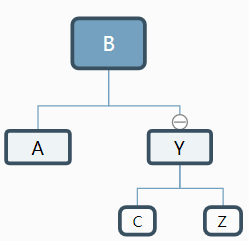
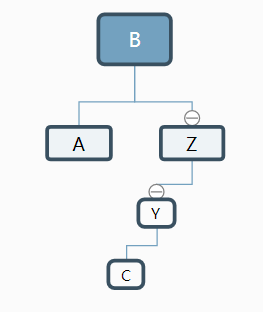
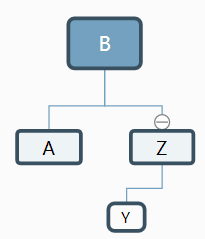
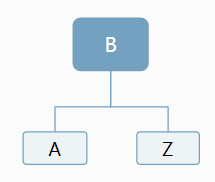
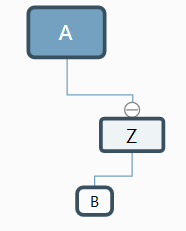
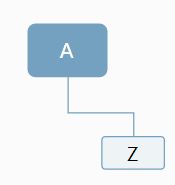
}

12.

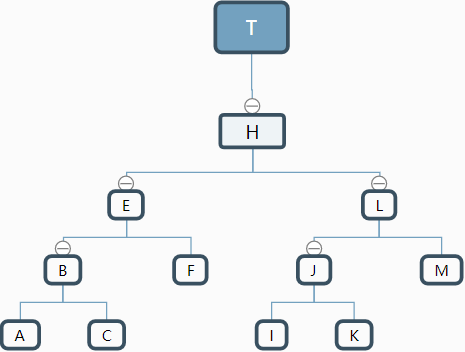
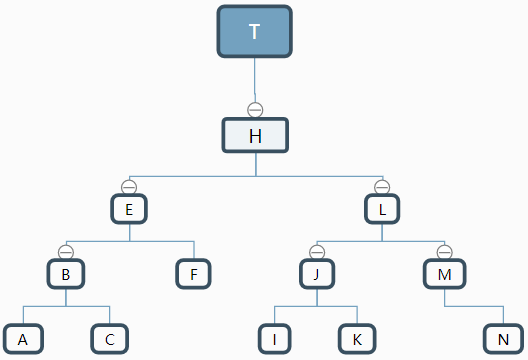
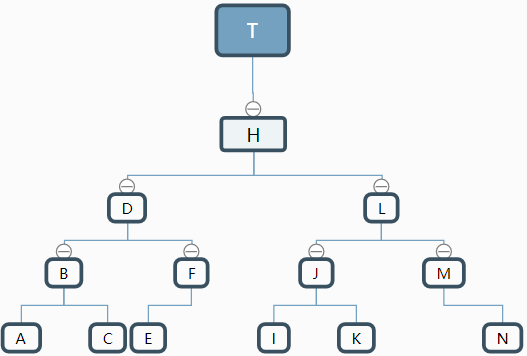
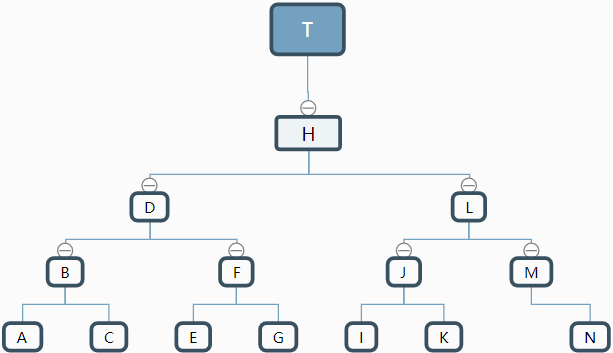
（1）



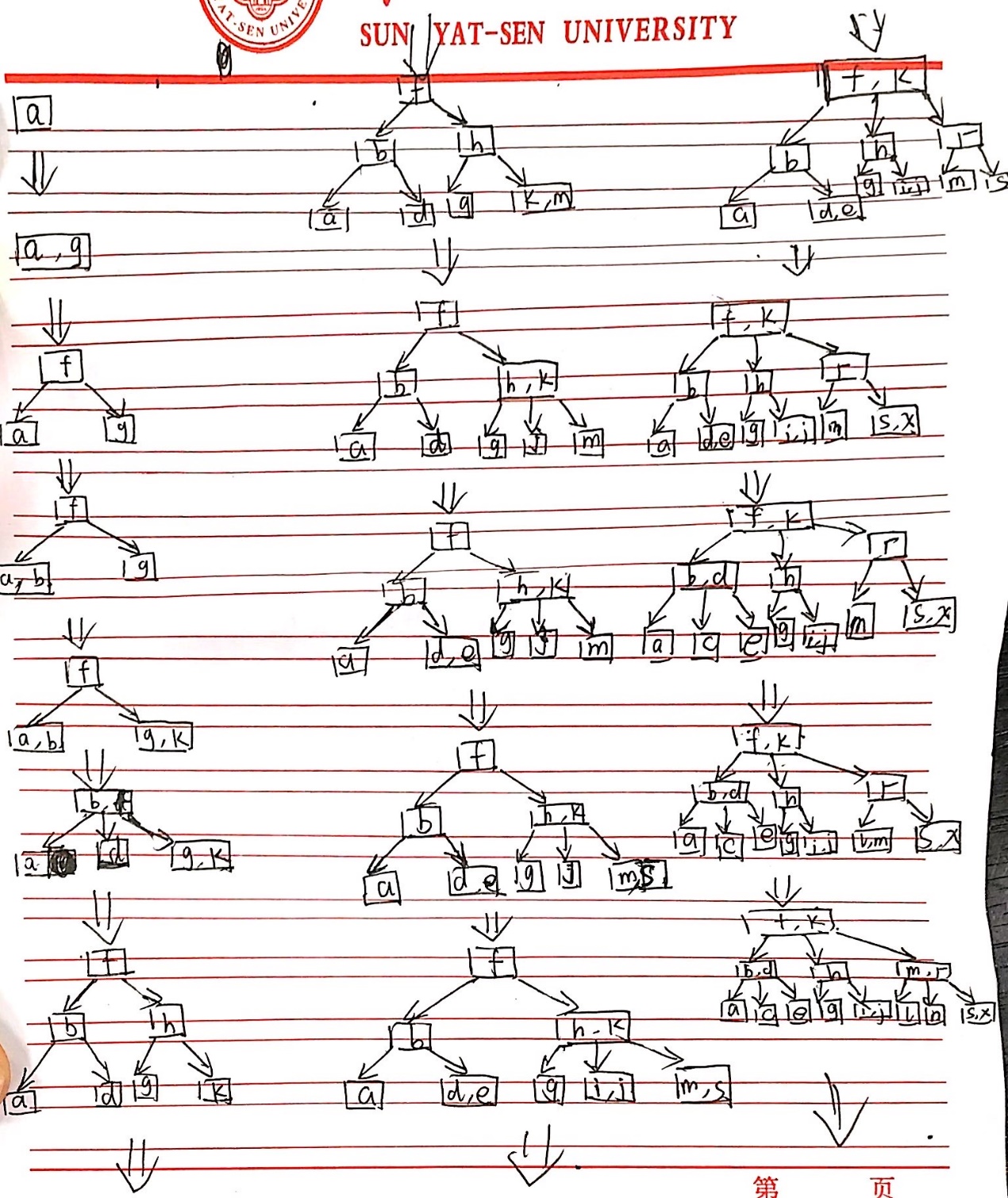
（2）

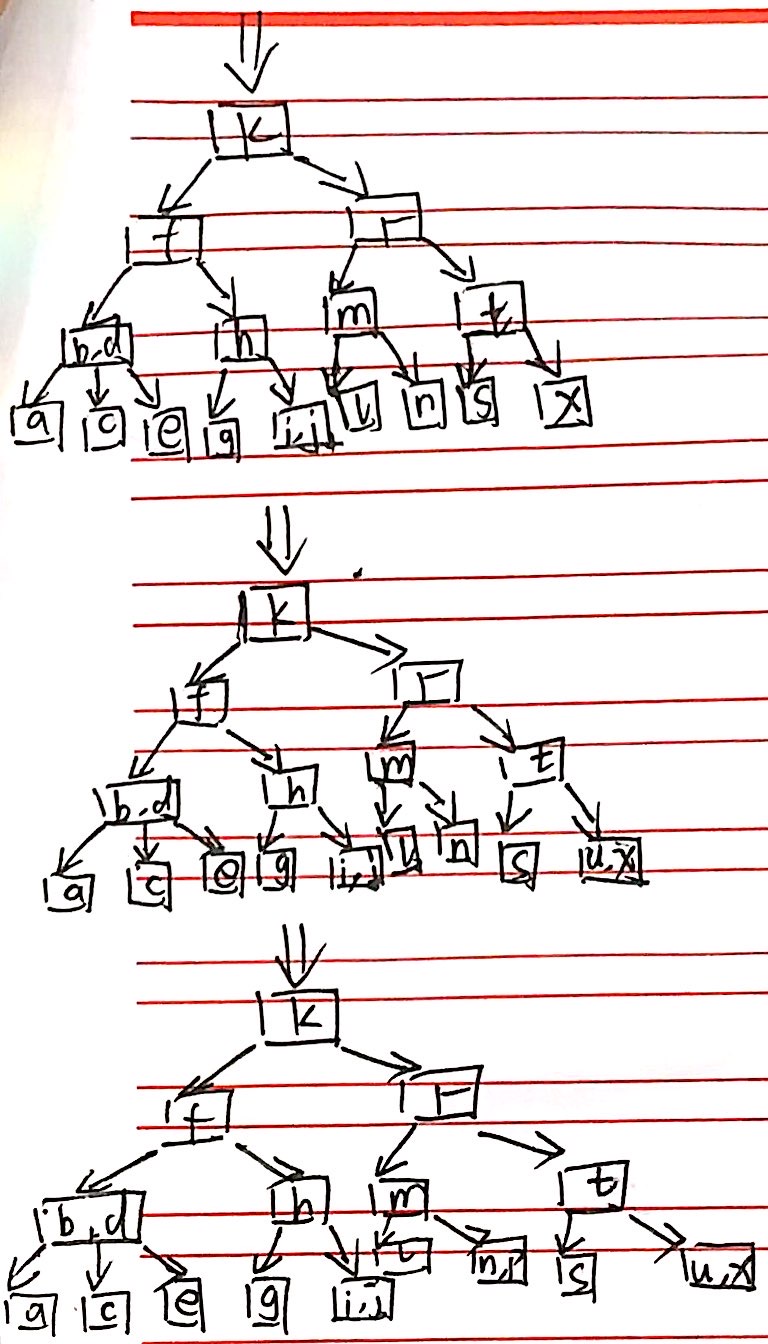


13.



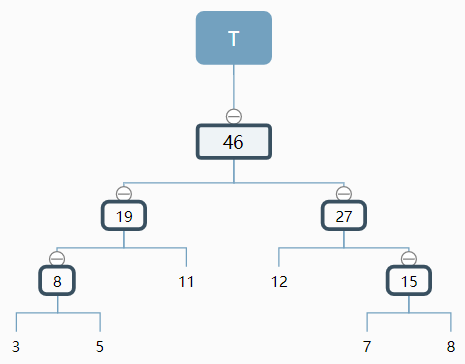
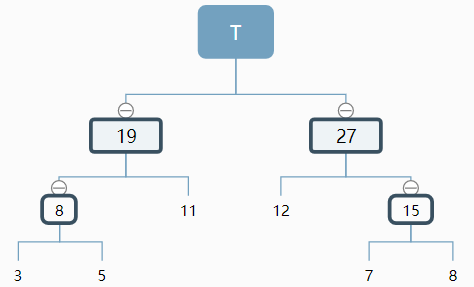
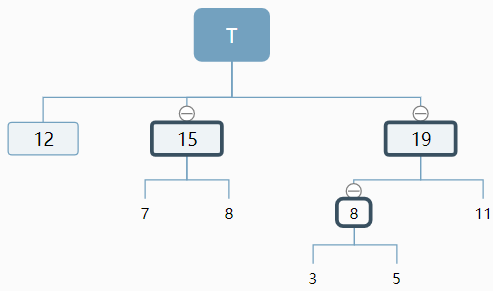
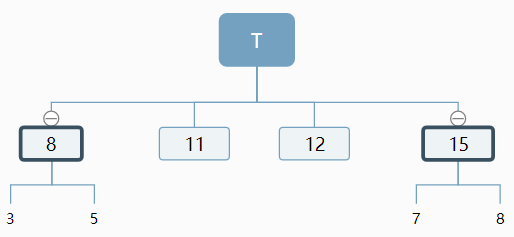
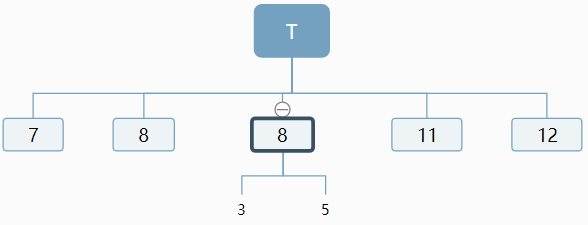
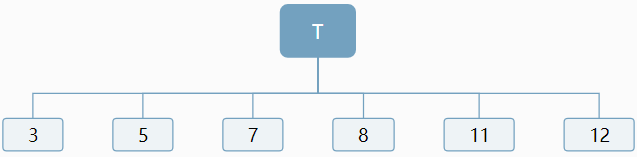
14.





15. 17

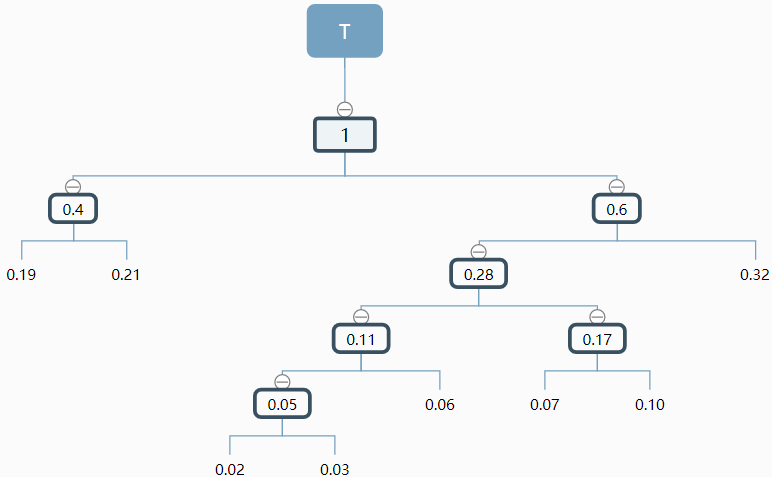
16.



WPL = (11 + 12) \* 2 + (3 + 5 + 7 + 8) \* 3 = 115

17.

1）



2） a：1010

b：00

c：10000

d：1001

e：11

f：10001

g：01

h：1011