1. Solve the following linear program using SIMPLEX:

Maximize $18x_1 + 12.5x_2$

Subject to $x_1 + x_2 \leq 20$

$x_1 \leq 12$

$x_2 \leq 16$

$x_1, x_2 \geq 0$

1. Firstly, we should convert linear program into slack form and introduce slack variables x_3, x_4, x_5. Then, the problem can be rewritten as:

Maximize: $Z = 18x_1 + 12.5x_2$

Subject to:

$$x_3 = 20 - x_1 - x_2 \quad \cdots ①$$
$$x_4 = 12 - x_1 \quad \cdots ②$$
$$x_5 = 16 - x_2 \quad \cdots ③$$
$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

Basic solution: $(\bar{x}_1, \bar{x}_2, \bar{x}_3, \bar{x}_4, \bar{x}_5) = (0, 0, 20, 12, 16)$

Since the ② constraint is the tightest constraint, we switch x_1 and x_4.

$$x_1 = 12 - x_4$$

The linear program is rewritten as:

Maximize: ~~Z = 18~~

$$Z = 18 \cdot (12 - x_4) + 12.5x_2 = 216 - 18x_4 + 12.5x_2$$
$$\cancel{x_1 = 12 - x_4}$$
$$x_3 = 8 + x_4 - x_2 \quad \cdots ①$$
$$x_1 = 12 - x_4 \quad \cdots ②$$
$$x_5 = 16 - x_2 \quad \cdots ③$$

New Basic Solution: $(\bar{x}_1, \bar{x}_2, \bar{x}_3, \bar{x}_4, \bar{x}_5) = (12, 0, 8, 0, 16)$

New object function: $Z = 216$

Since the ① constraint is the tightest constraint, we switch x_2 and x_3

$$x_2 = 8 + x_4 - x_3$$

The linear program is rewritten as:

Maximize: $Z = 316 - 5.5x_4 - 12.5x_3$

$$x_2 = 8 + x_4 - x_3$$
$$x_1 = 12 - x_4$$
$$x_5 = 8 - x_4 + x_3$$

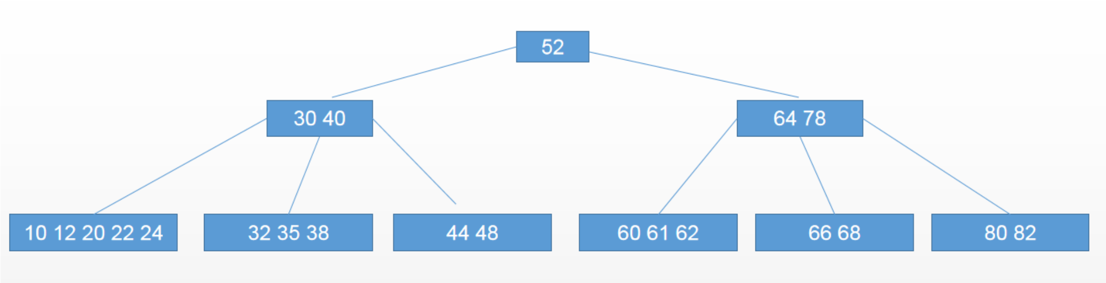
All coefficients in the objective function are negative, ~~the~~ as means the basic solution is the optimal solution.

Basic solution: $(\bar{x}_1, \bar{x}_2, \bar{x}_3, \bar{x}_4, \bar{x}_5) = (12, 8, 0, 0, 8)$

Objective function: $Z = 316$

2.

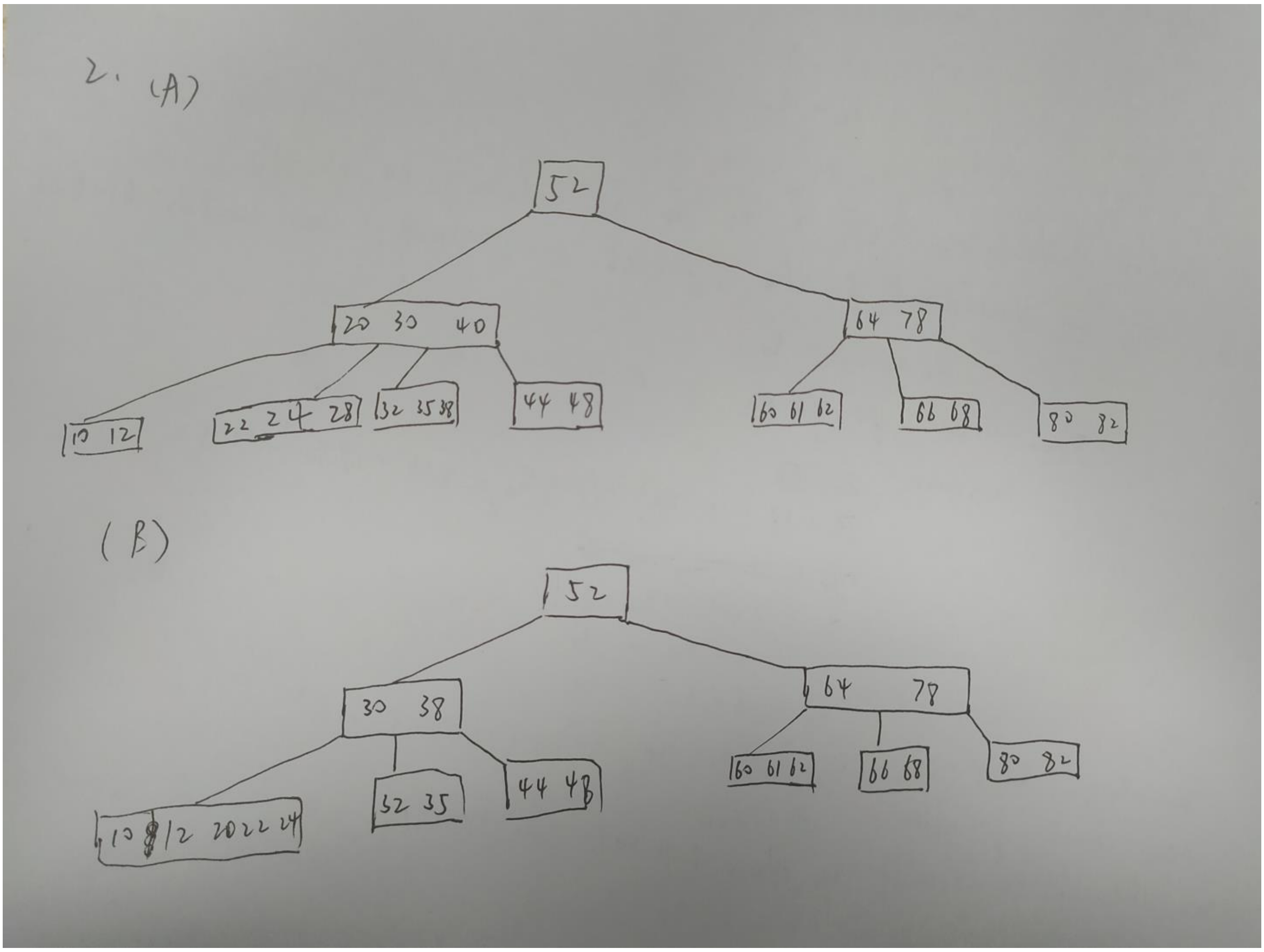
B tree ,minimum degree t=3



Draw the figure to show

(A).Insert 28

(B) Delete 40



3.The interval-tree builds the red-black tree according to the preceding segment of the interval, with each node of the red-black tree appended with an x.max, which is the maximum value of the endpoints of all intervals of the x-rooted subtree.

(A) The interval-tree has a new operation, INTERVAL-SEARCH(T, i), which is used to find the node in the tree that overlaps the interval i. If no node in the tree overlaps with interval i, T.nil is returned. Write the pseudocode for this operation.

(B) With interval set {[0, 3], [5,8], [6, 10], [8, 9], [15, 23], [16, 21], [17,19], [19, 20], [25, 30], [26, 26]}, please build an interval tree (write simple process in drawing.)

3.

(A).

```
INTERVAL-SEARCH(T, i)
    x = T.root;
    while x ≠ T.nail and x does not overlap i
        if x.left ≠ T.nail and x.left.max ≥ i.int.low
            x = x.left;
        else
            x = x.right;
    return x;
```

N 为新插入节点，P 为 N 的父节点，U 为 N 的叔叔节点，G 为 P 的父节点。

case1:若是根节点，直接涂为黑；

case2: P 为黑，直接插入新节点 N；

case3: P,U 都为红，G,P,U 切换颜色，并对 G 从 case1 进行递归检测；

case4-1: P 红，U 黑，N 是 P 的右节点，P 是 G 的左节点，以 P 为中心左旋，并对 P 进行 case5-1 检查;

case4-2: P 红，U 黑，N 是 P 的左节点，P 是 G 的右节点，以 P 为中心右旋，并对 P 进行 case5-2 检查;

case5-1: P 红，U 黑，N 是 P 的左节点，P 是 G 的左节点，以 G 为中心右旋，并切换 G、P 的颜色。

case5-2: P 红，U 黑，N 是 P 的右节点，P 是 G 的右节点，以 G 为中心左旋，并切换 G、P 的颜色。

注意：插入的新节点默认涂为红色，方便递归操作；

eg. 依次插入0, 5, 6, 8, 15, 16 构造一棵红黑树

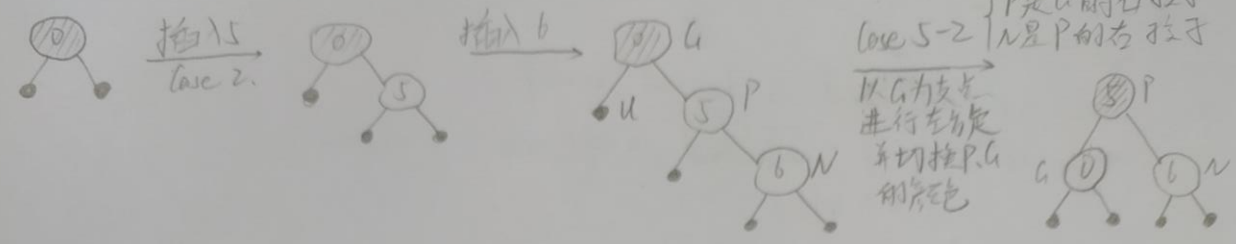
— black node　　— red node　　— leaf node

P是红
U缺或黑
P是G的右孩子
N是P的右孩子

插入5 case 2.

插入6

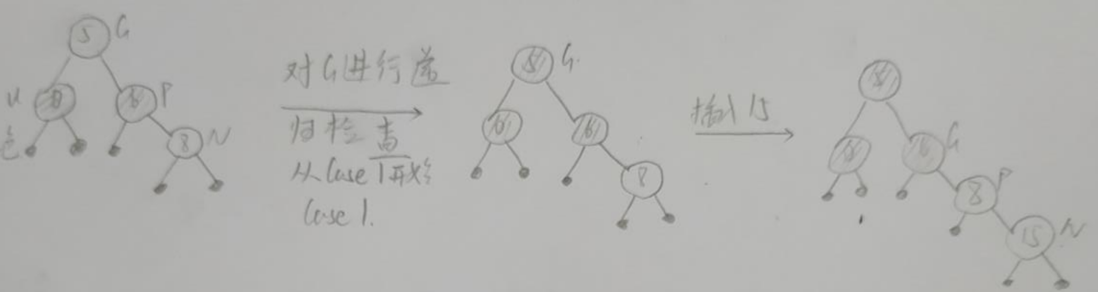
case 5-2 以G为支点进行左旋 并切换P,G的颜色



插入8 case 3 将U,P涂为红色

对G进行递归检查 从case 1开始 case 1.

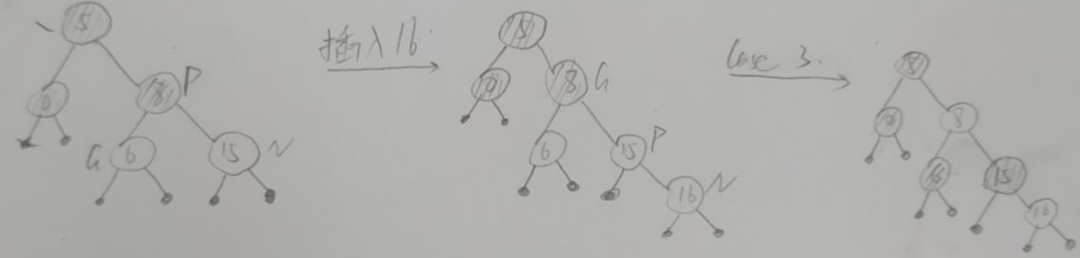
插入15



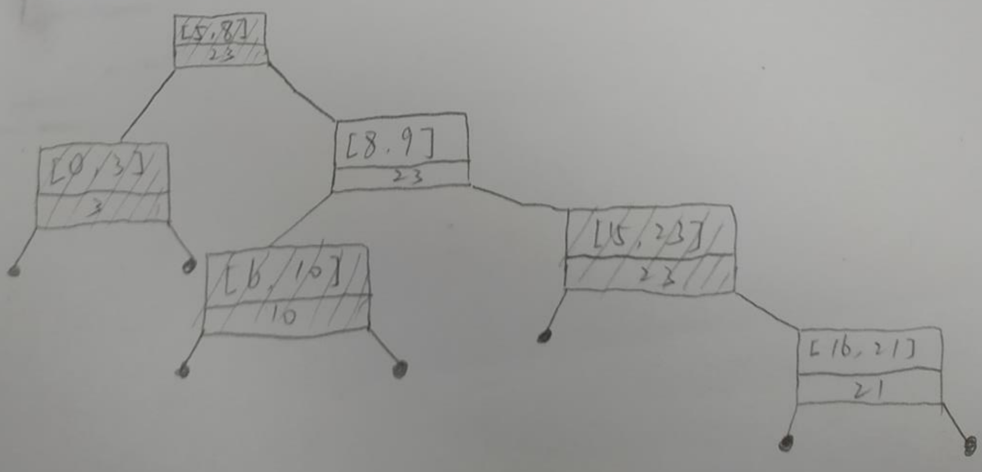
case 5-2 以G为支点进行左旋, 并切换P,G 的颜色

插入16

case 3.



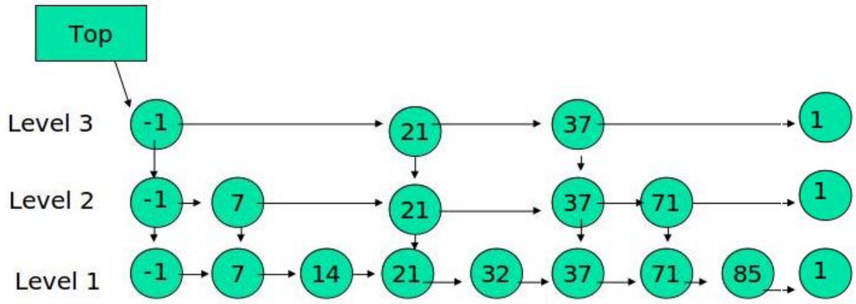
⇒ 按顺序插入前6个结点的结果:



4. Skip List（-1 represents flag of begin, 1 represent flag of end）

(A) Delete (x) is an algorithm for delete element x in a skip list. Write its pseudocode.

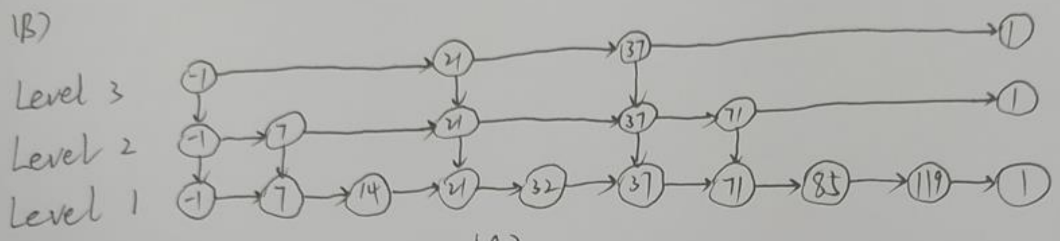
(B) There is a skip list as shown in the following figure. Insert element 119 in the skip list at level 4. (write simple process in drawing.)



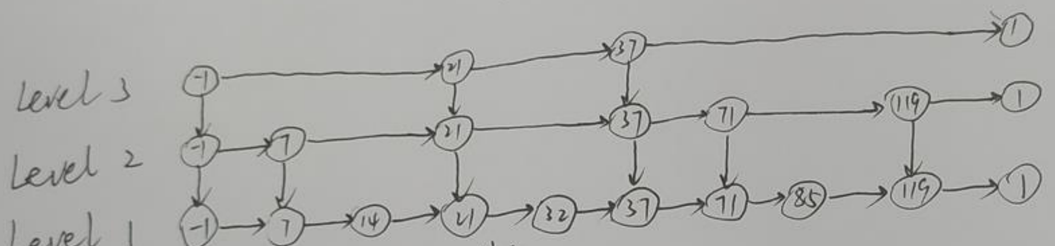
4. (A)

```
Delete (x)
{  i=1;
   while ( level i is not the highest and x in level i)
   {  x.left.right = x.right;
      free x;
      i = i+1;
   }
}
```

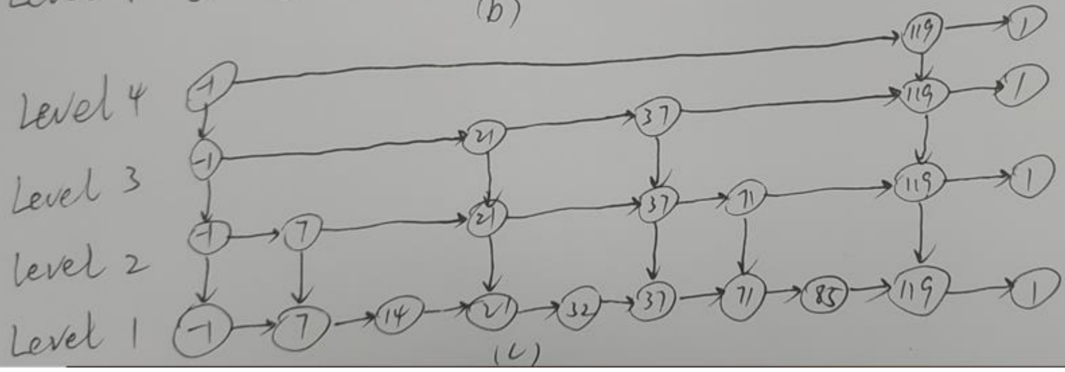
(B)



(a)



(b)



(c)

5. Problem X: Does the bool sequence $\{x_1, x_2, \dots, x_n\}$ have at least one value x_i is false.
Problem Y: Integer sequence $\{y_1, y_2, \dots, y_n\}$. Is the minimum value y_i negative.
What is the Construct function T to reduce problem X to problem Y.

5.

```
function ( X[1,..., n], Y[1, ..., n])
{ for ( i = 1 : n)
    { if xi = false
            yi = 0 ;
      else
            yi = 1 ;
    }
}
```