

C

15 2 30

CBCAD CADBC DCABD

, × 10 1 10

× × × × ×

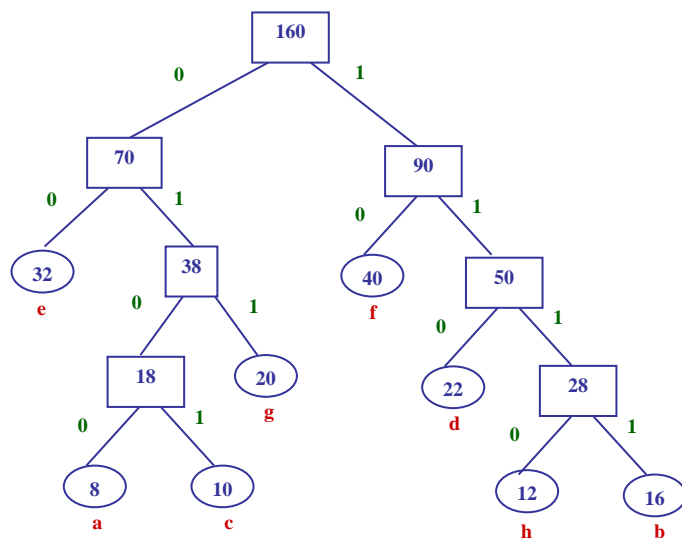
8 2 16

1.  $i > L.length + 1$
2.  $L.data[j + 1] = L.data[j]$
3.  $L.length++$
4.  $p \rightarrow data = e$
5.  $Q.rear \rightarrow next = p$
6.  $Q.rear = p$
7.  $high = mid \times 2 - 1$
8.  $low = mid + 1$

4 6 24 )

1.

(1) (3 )



(2) (2 )

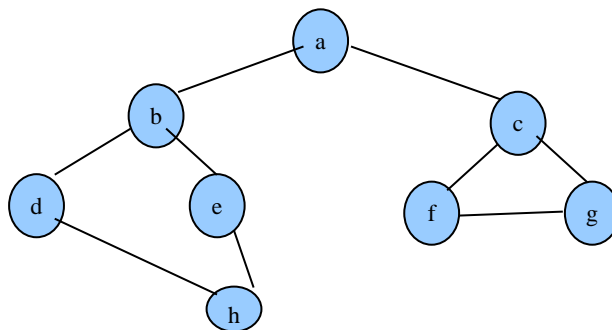
a: 0100, b: 1111, c: 0101, d: 110, e: 00, f: 10, g: 011, h: 1110

(3) (1 )

$$\begin{aligned}
 &= 4 \cdot 8 + 4 \cdot 16 + 4 \cdot 10 + 3 \cdot 22 + 2 \cdot 32 + 2 \cdot 40 + 3 \cdot 20 + 4 \cdot 12 \\
 &= 4 \cdot (8+16+10+12) + 3 \cdot (22+20) + 2 \cdot (32+40) \\
 &= 4 \cdot 46 + 3 \cdot 42 + 2 \cdot 72 = 454
 \end{aligned}$$

2.

(1) (2 )



(2) (2 )

0	1	1	0	0	0	0	0
1	0	0	1	1	0	0	0
1	0	0	0	0	1	1	0
0	1	0	0	0	0	0	1
0	1	0	0	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	0	0	1	0	0
0	0	0	1	1	0	0	0

(3) (1 )

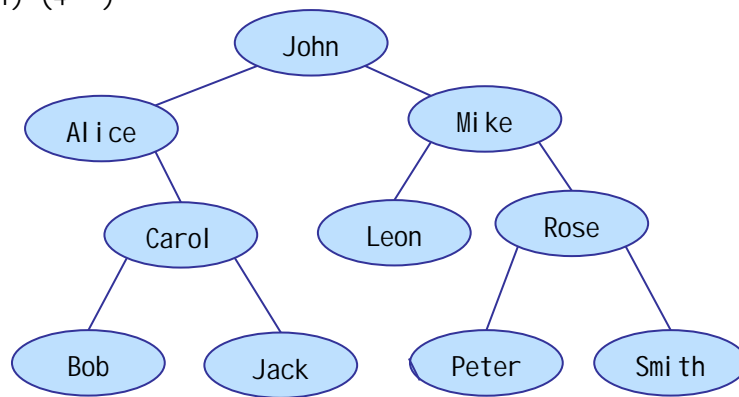
abdhecfg

(4) (1 )

abcdefgh

3.

(1) (4 )



(2) (2 )

$$= (1 \cdot 1 + 2 \cdot 2 + 3 \cdot 3 + 4 \cdot 4) / 10 = 30 / 10 = 3$$

4.

(1) (4 )

	0	1	2	3	4	5	6	7	8	9	10	11	12
		53		29	42	44	58	32	73	31	98	37	
		1		1	2	1	1	2	1	5	4	1	

(3) (2 )

$$ASL = (1 + 1 + 2 + 1 + 1 + 2 + \quad + 5 + 4 + 1) / 10 = 1.9$$

(2            10            20    )

1.

(1) (5    )

```
void Vote(LinkList L, int vote_id)
{
    LinkList p;
    for (p = L->next; p != NULL; p = p->next)
    {
        if (vote_id == p->id)
        {
            p->count++;
            return;
        }
    }
    printf("            ");
}
```

(2) (5    )

```
void SortLinkList(LinkList L)
{
    LinkList p, q, min;
    int temp_id, temp_count;

    for (p = L->next; p != NULL; p = p->next)
    {
        min = p;
        for (q = p; q != NULL; q = q->next)
            if (q->count < min->count)
                min = q;

        temp_id = min->id;
        temp_count = min->count;
        min->id = p->id;
        min->count = p->count;
        p->id = temp_id;
        p->count = temp_count;
    }
}
```

2.

(1) (5 )

```
int Statistics(BiTree T)
{
    //
    if (!T)
        return 0;

    //
    if (T->lchild == NULL && T->rchild == NULL)
    {
        if (T->count == 0)
            return 1;
        else
            return 0;
    }

    //
    return Statistics(T->lchild) + Statistics(T->rchild);
}
```

(2) (5 )

```
void Calculate(BiTree T)
{
    //
    if (!T)
        return;

    //
    if (T->lchild == NULL && T->rchild == NULL) {
        T->money = T->count * 100;
        return;
    }

    //
    Calculate(T->lchild);
    Calculate(T->rchild);
    T->money = T->lchild->money + T->rchild->money + 1000;
    T->count = T->lchild->count + T->rchild->count;
}
```