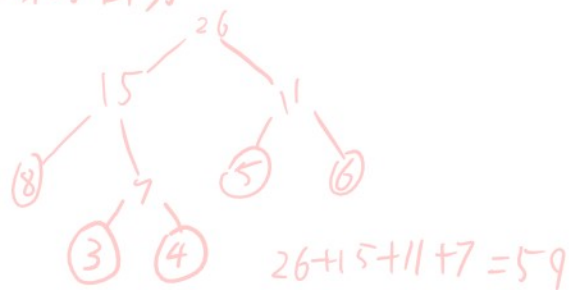


## 数据结构部分

1. C



2. D 20王道 P64, 18题

3. C 20王道 P30, 5. 插入结点操作

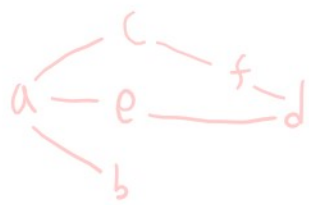
4. C 20王道 P266, 6.4.4 散列查找及性能分析

5. C 20王道 P109, 3. 二叉树的性质, 1)

6. A 20王道 P183 7. 连通、连通图和连通分量  
若图有  $n$  个顶点, 且边数小于  $n-1$ , 则此图必是非连通图

⇒ 若图有  $n$  个顶点, 且是连通图, 则边数大于等于  $n-1$

7. D



A: a b e c f d f c

B: a c f e b d d e b

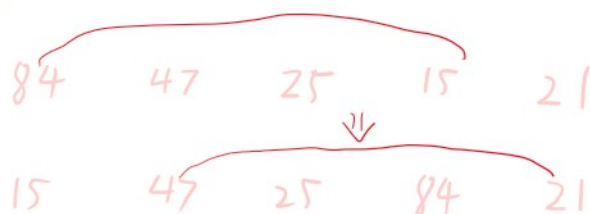
C: a e b c f d d f c b

8. B

空间复杂度:

冒泡:  $O(1)$  20王道 P300选择排序:  $O(1)$  20王道 P310插入排序:  $O(1)$  20王道 P293希尔排序:  $O(1)$  20王道 P295快速排序: 平均  $O(\log_2 n)$ , 最坏  $O(n)$  20王道 P301堆排序: 是一种选择排序,  $O(1)$  20王道 P312

9. B



↓

15 21 25 84 47

15是L[1...5]中最小的

21是L[2...5]中最小的, 符合简单选择排序描述

20王道 P309, 7.4.1 简单选择排序

10. C

20王道 P110 堆排序

$$\lfloor n/2 \rfloor = 5$$

$$L[5] = 60 \quad (n=10)$$

二.

1.

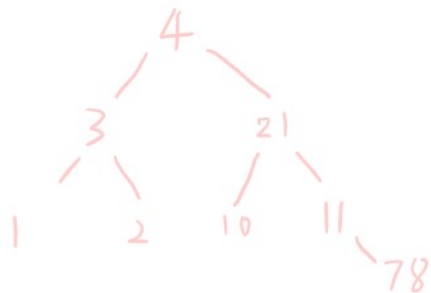
(1) 顺序查找:

0	1	2	3	4	5	6	7	顺序表
11	78	10	1	3	2	4	21	

$$ASL = \frac{(1+8) \times 8}{2} \cdot \frac{1}{8} = 4.5$$

(2) 二分查找:

20王道 P242, 6.2.2 折半查找



$$ASL = \frac{1}{8} (1 \times 1 + 2 \times 2 + 3 \times 4 + 4)$$

$$\approx 2.63$$

(3) 哈希查找:

0	1	2	3	4	5	6	7	8	9	10
11	78	1	3	2	4	21				10

$$ASL = (1 + 1 + 1 + 2 + 1 + 3 + 2 + 8) / 8$$

$$= \frac{19}{8} \approx 2.38$$

2.

前  
后

J F D B A C E H X I k  
A C B E D X I H F k J



FD B A / F H X I J



```

        treeNode->cousin;
    };

int LeafCount(treeNode *node)
{
    if(node->child==NULL)
        return 1;
    else
    {
        int sum=0;
        for(treeNode *temNode=node->child; temNode; temNode=temNode->cousin)
            sum+=LeafCount(temNode);
        return sum;
    }
}

int main()
{
    /*
    treeNode *node7=(treeNode*)malloc(sizeof(treeNode));node7->data=7; node7->child=NULL;
    node7->cousin=NULL;
    treeNode *node6=(treeNode*)malloc(sizeof(treeNode));node6->data=6; node6->child=NULL;
    node6->cousin=node7;
    treeNode *node5=(treeNode*)malloc(sizeof(treeNode));node5->data=5; node5->child=NULL;
    node5->cousin=NULL;
    treeNode *node4=(treeNode*)malloc(sizeof(treeNode));node4->data=4; node4->child=node6;
    node4->cousin=NULL;
    treeNode *node3=(treeNode*)malloc(sizeof(treeNode));node3->data=3; node3->child=NULL;
    node3->cousin=node4;
    treeNode *node2=(treeNode*)malloc(sizeof(treeNode));node2->data=2; node2->child=node5;
    node2->cousin=node3;
    treeNode *node1=(treeNode*)malloc(sizeof(treeNode));node1->data=1; node1->child=node2;
    node1->cousin=NULL;
    cout<<LeafCount(node1)<<endl;
    */
    treeNode *node11=(treeNode*)malloc(sizeof(treeNode));    node11->data=11; node11->
    child=NULL;    node11->cousin=NULL;
    treeNode *node10=(treeNode*)malloc(sizeof(treeNode));    node10->data=10; node10->
    child=NULL;    node10->cousin=node11;
    treeNode *node9=(treeNode*)malloc(sizeof(treeNode));    node9->data=9; node9->
    child=NULL;    node9->cousin=node10;
    treeNode *node8=(treeNode*)malloc(sizeof(treeNode));    node8->data=8; node8->
    child=node9;    node8->cousin=NULL;
    treeNode *node7=(treeNode*)malloc(sizeof(treeNode));    node7->data=7; node7->
    child=NULL;    node7->cousin=node8;
    treeNode *node6=(treeNode*)malloc(sizeof(treeNode));    node6->data=6; node6->
    child=NULL;    node6->cousin=NULL;
    treeNode *node5=(treeNode*)malloc(sizeof(treeNode));    node5->data=5; node5->
    child=node6;    node5->cousin=NULL;
    treeNode *node4=(treeNode*)malloc(sizeof(treeNode));    node4->data=4; node4->
    child=node7;    node4->cousin=NULL;
    treeNode *node3=(treeNode*)malloc(sizeof(treeNode));    node3->data=3; node3->
    child=NULL;    node3->cousin=node4;
    treeNode *node2=(treeNode*)malloc(sizeof(treeNode));    node2->data=2; node2->
    child=node5;    node2->cousin=node3;
    treeNode *node1=(treeNode*)malloc(sizeof(treeNode));    node1->data=1; node1->
    child=node2;    node1->cousin=NULL;
    cout<<LeafCount(node1)<<endl;
}

```

为算法部分  
其他为验证程序

2.

```

#include<iostream>
#include<stdlib.h>
using namespace std;

struct xnode
{
    int val;
    xnode *next;
};

xnode* Create()
{
    xnode *head=NULL,*tem1;
    int x;
    while(cin>>x,x!=1)
    {
        tem1=(xnode*)malloc(sizeof(xnode));
        tem1->val=x;
        tem1->next=head;
        head=tem1;
    }
    return head;
}

void PrintList(xnode *a)
{
    while(a)
    {
        cout<<a->val<<" ";
        a=a->next;
    }
    cout<<endl;
}

xnode* Partition(xnode *L1)
{
    xnode *L2=NULL,*L3,*L4,*tem=L1;
    int i=1;
    while(tem&&tem->next)
    {
        L3=tem->next;
        tem->next=tem->next->next;
        L3->next=L2;
        L2=L3;
        tem=tem->next;
    }
    return L2;
}

int main()
{
    xnode *L1=Create();
    PrintList(L1);
    xnode *L2=Partition(L1);
    PrintList(L1);
    PrintList(L2);
}

```

数据库部分

一.

1. B

教材 P25 4.7 高度的数据独立性

2. C

3. B

教材 P100, 3.5.1 视图的创建和撤销

4. C

教材 P224, 7.3. | 并发操作带来的 3 个问题

5. C

教材 P150, 5.1.4 逻辑设计阶段

1. 把概念模型转换成逻辑模型

二.

候选码为 CD

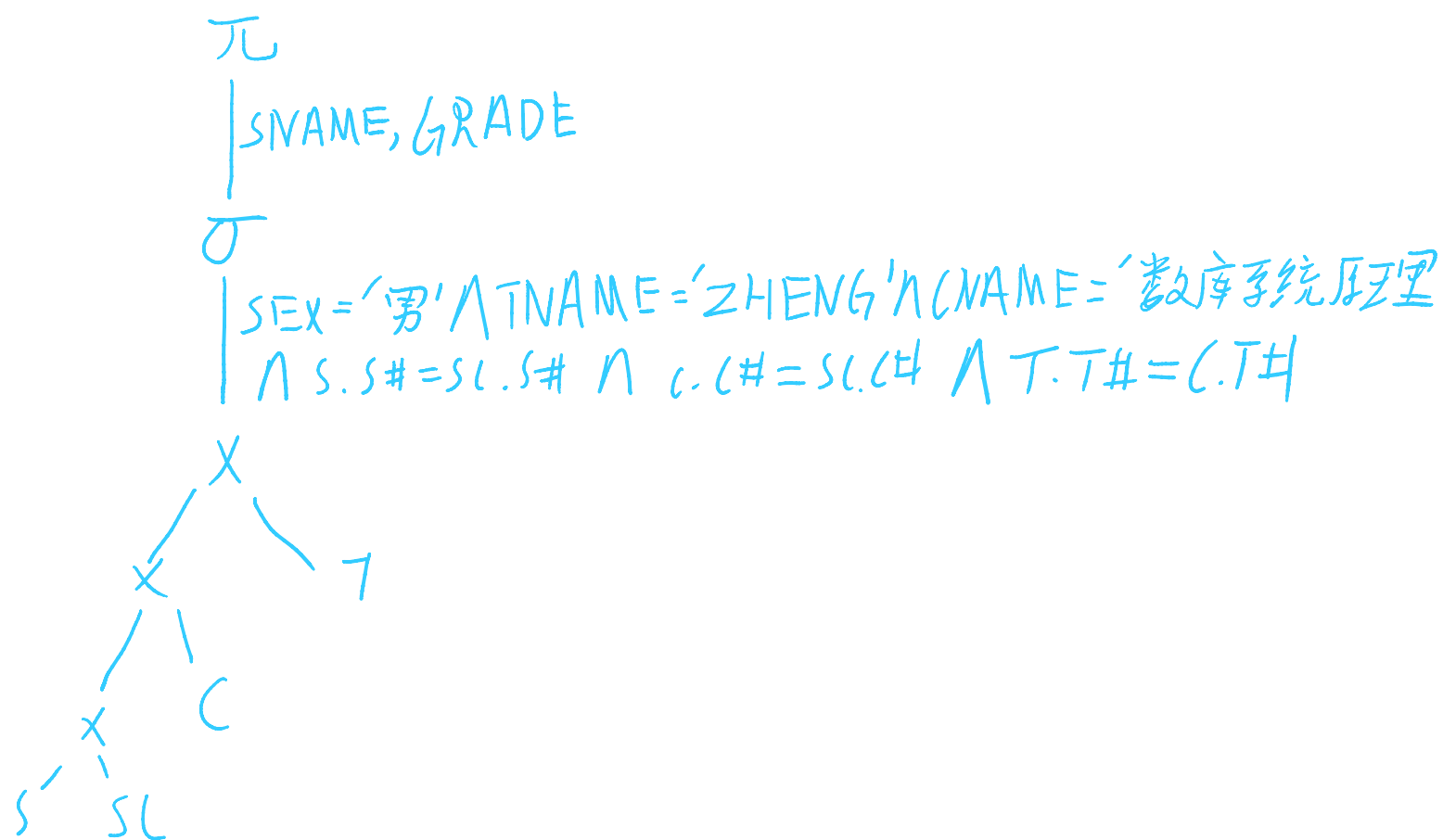
1.  $\because CD \rightarrow B \quad B \rightarrow A \Rightarrow CD \rightarrow A$  是传递依赖

2.  $R_1(CDB) \quad R_2(BA)$

3. 是否保持 FD, 是否无损连接 练习册 P80, 填空 22 题

三.

1.  $\pi_{2,9}(\sigma_{1=7 \wedge 6=10 \wedge 4=8 \wedge 11='ZHENG' \wedge 5='数据库系统原理' \wedge 3='男'}(S \times C \times S \times T))$



2.

