2015年

2019年11月20日　2:38

数据结构

一、

1. D
2. D
3. C
4. C
5. B
6. A
7. A
8. C
9. C
10. B

二、

1. 一个好的散列函数会造成较少的冲突，并且将关键字均匀的分布在散列表上。

一个散列函数可能会将两个关键字映射到同一位置，这种现象称为冲突。

开放地址法，链地址法

2、

(1),(2) 是大堆

(4) 是小堆

(3) → 100, 98, 66, 85, 80, 60, 40, 82, 77, 10, 26

三、

1.

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

struct BTnode
{
        int data;
        BTnode *left;
        BTnode *right;
};

int BTHeight(BTnode *a)
{
        int height=0,lheight,rheight;
        if(a==NULL)
                return 0;
        else
        {
                height++;
                lheight=BTHeight(a->left);
                rheight=BTHeight(a->right);
                height+=lheight>rheight?lheight:rheight;
        }
        return height;
}

int main()
{

        BTnode *node17=(BTnode*)malloc(sizeof(BTnode));      node17->left=NULL;              node17->
        right=NULL;                  node17->data=7;
        BTnode *node16=(BTnode*)malloc(sizeof(BTnode));     node16->left=NULL;              node16->
        right=NULL;             node16->data=6;
        BTnode *node15=(BTnode*)malloc(sizeof(BTnode));     node15->left=NULL;              node15->
        right=NULL;             node15->data=5;
        BTnode *node14=(BTnode*)malloc(sizeof(BTnode));     node14->left=node16;  node14->
        right=node17;   node14->data=4;
        BTnode *node13=(BTnode*)malloc(sizeof(BTnode));     node13->left=NULL;              node13->
        right=NULL;             node13->data=3;
        BTnode *node12=(BTnode*)malloc(sizeof(BTnode));     node12->left=node14;  node12->
        right=node15;   node12->data=2;
        BTnode *node11=(BTnode*)malloc(sizeof(BTnode));     node11->left=node12;  node11->
        right=node13;   node11->data=1;
        /*
        BTnode *node15=(BTnode*)malloc(sizeof(BTnode));     node15->left=NULL;              node15->
        right=NULL;             node15->data=5;
        BTnode *node14=(BTnode*)malloc(sizeof(BTnode));     node14->left=node15;  node14->
        right=NULL;     node14->data=4;
        BTnode *node13=(BTnode*)malloc(sizeof(BTnode));     node13->left=NULL;              node13->
        right=NULL;             node13->data=3;
        BTnode *node12=(BTnode*)malloc(sizeof(BTnode));     node12->left=node14;  node12->
        right=NULL;     node12->data=2;
        BTnode *node11=(BTnode*)malloc(sizeof(BTnode));     node11->left=node12;  node11->
        right=node13;   node11->data=1;
        */
        cout<<BTHeight(node11);
}
```

2.

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

struct List
{
        int data;
        List *next;
};

List* Create()
{
        int tem;
        List *head=(List*)malloc(sizeof(List));
        List *temNode1=head;
        List *temNode2,*temNode3;
        while(cin>>tem,tem!=-1)//输入-1结束
        {

                temNode1->data=tem;
                temNode2=(List*)malloc(sizeof(List));
                temNode3=temNode1;
                temNode1->next=temNode2;
                temNode1=temNode2;
        }
        temNode3->next=NULL;

        return head;
}


void PrintList(List *a)
{
        while(a)
        {
                cout<<a->data<<" ";
                a=a->next;
        }
        cout<<endl;
}
List* Reverse(List *a)
{
        List *L1=NULL,*L2=NULL,*L3=NULL;
        while(a)
        {
                L1=a;
                L2=a->next;
                a=a->next;
                L1->next=L3;
                L3=L1;
        }
        return L3;
}

int main()
{
        List *a=Create();
        PrintList(a);
        a=Reverse(a);
        PrintList(a);
}
```

数据库部分

一、

1. A

2. B

3. B

4. B

5. C

二

1.

$\pi_{S\#,SNAME}(\sigma_{CNAME='C语言程序'\wedge GRADE>90}(S\bowtie SC\bowtie C))$

2.

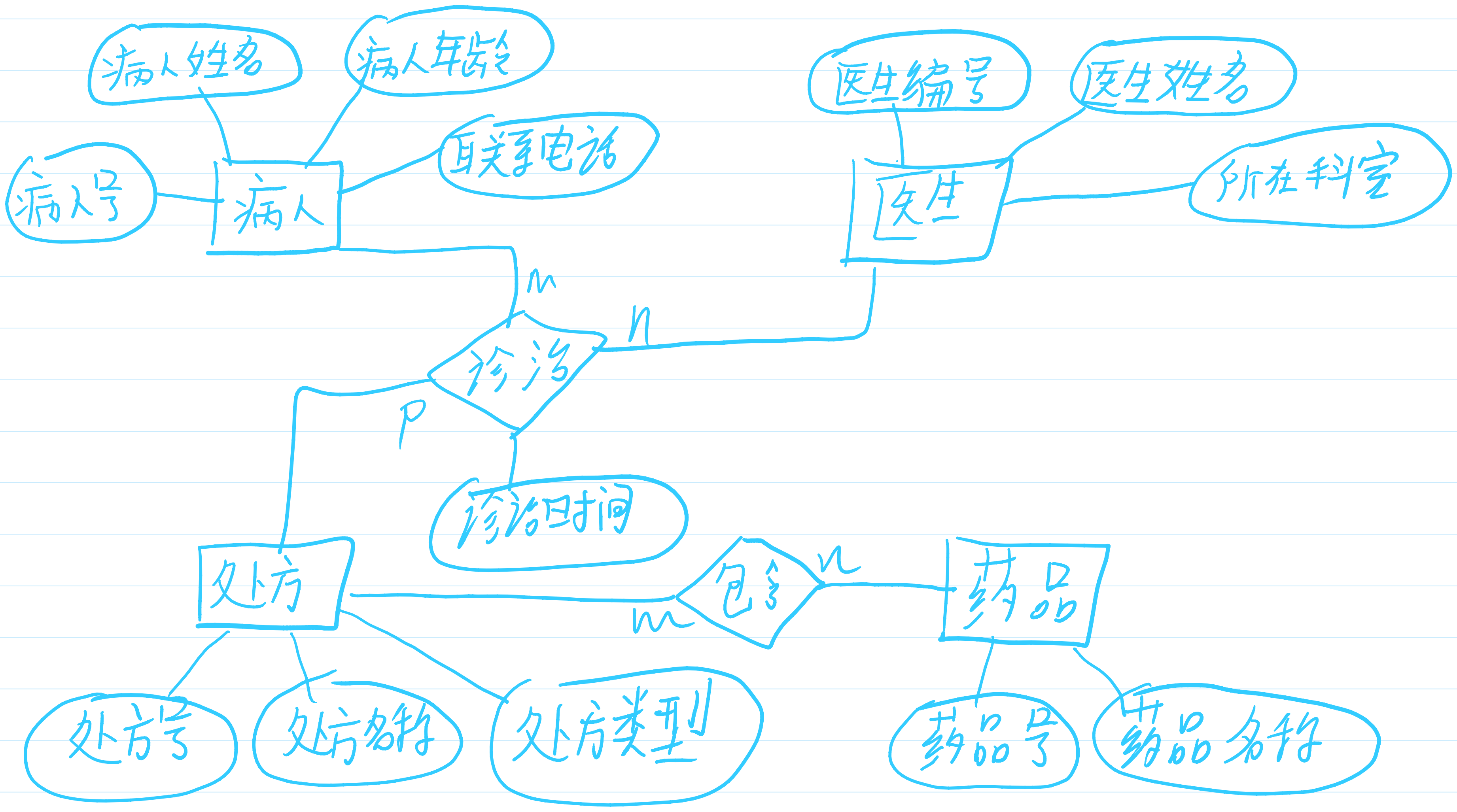
$\pi_{S\#,SNAME}(S\bowtie(\sigma_{2='C语言程序'\wedge 4='数据库系统原理'}(\pi_{S\#,CNAME}(SC\bowtie C)\underset{1=1}{\bowtie}$

$\pi_{S\#,CNAME}(SC \bowtie C)))$

3. select C.CNAME, AVG(GRADE) from SC, C where SC.C#=C.C# group by(CNAME) having AVG(GRADE)>85

三、

1.



2

医生（医生编号，医生姓名，所在科室）

病人（病人号，病人姓名，病人年龄，联系电话）

处方（处方号，处方名称，处方类型）

药品（药品号，药品名称）

R1（处方号，药品号）

R2（医生号，病人号，处方号，诊治时间）