2018

第一讲 线性结构

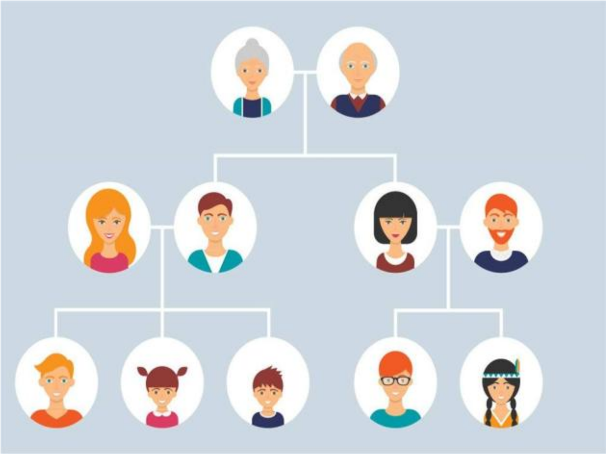
时间：2018年10月20

目录

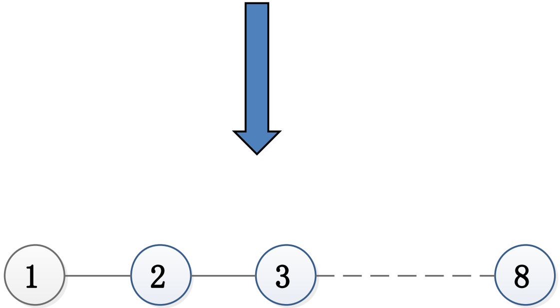
CONTENTS

前 言

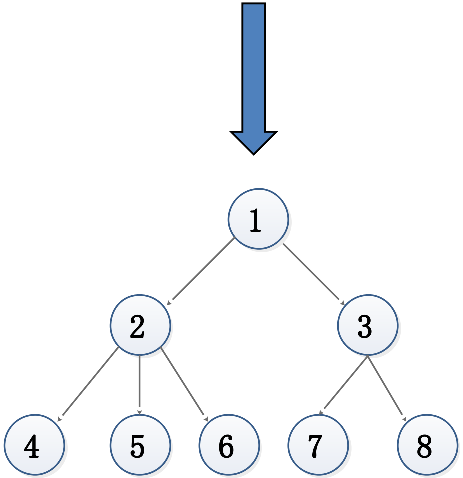
什么是数据结构



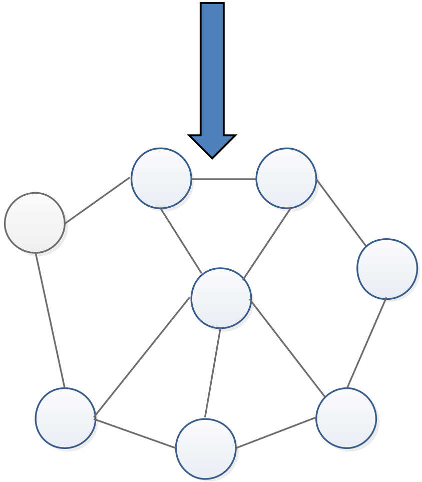
站队



家谱



城市交通



什么是数据结构

- 数据结构是计算机存储数据、组织数据的方式。
- 数据结构是指相互之间存在一种或多种关系的数据元素的集合以及该集合中数据元素之间的关系。

  记为：Data_Struct=(D,R)

  D是数据元素的集合；

  R是数据元素之间的关系的有限集合。

如：家谱

D={1,2,3,4,5,6,7,8}

R={<1,2>,<1,3>,<2,4>,<2,5>,<2,6>,<3,7>,<3,8>}

什么是数据结构

- 数据结构分为**逻辑结构**和**物理结构**
- **逻辑结构**：

反映数据元素之间的逻辑关系（先后关系），与在计算机中的存储位置无关。

包括：集合，线性结构，树形结构，图形结构。

通常所说的数据结构指的是逻辑结构

什么是数据结构

①集合结构：元素之间没有关系。
②线性结构：元素是一对一的关系。
③树形结构：元素存在着一对多的关系。
④图形结构：元素是多对多的关系。

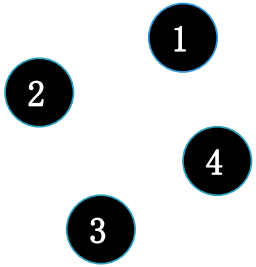


图1

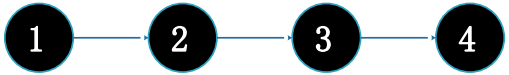


图2

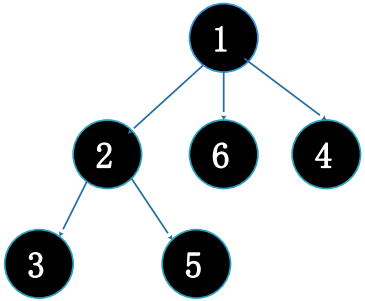


图3

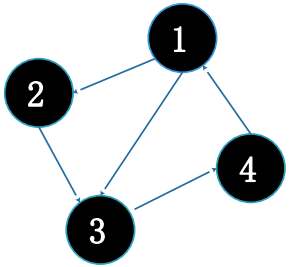


图4

什么是数据结构

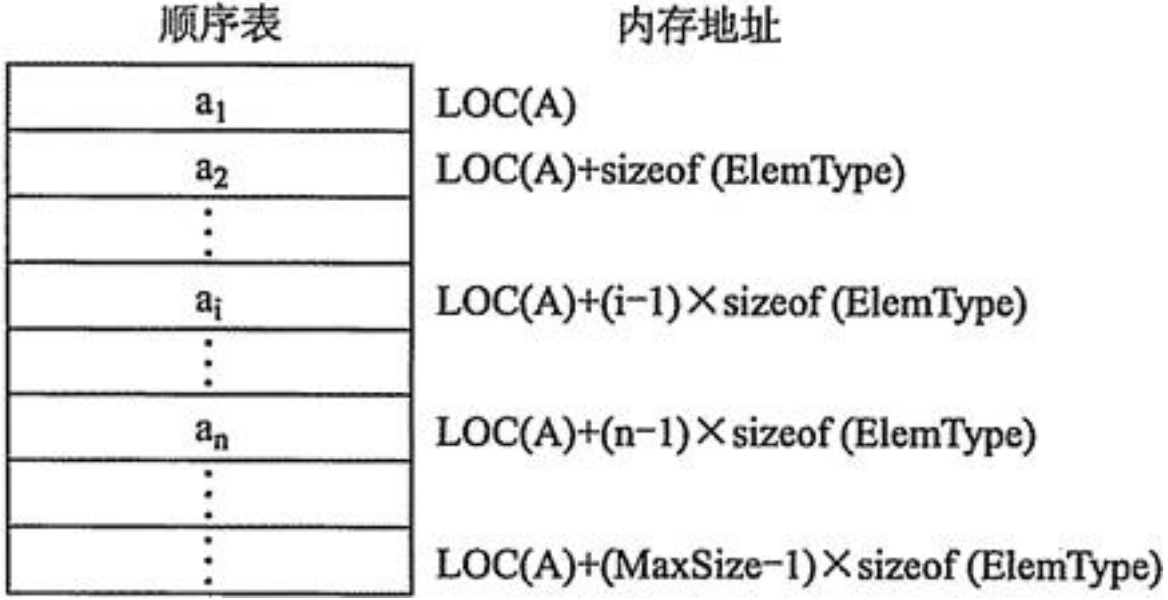
- **物理结构**：

数据的逻辑结构在计算机存储空间中的存放形式：顺序结构和链式结构。

- 顺序存储结构：是把数据元素放在地址连续的存储单元里，其数据间的逻辑关系和物理关系是一致的。
- 链式存储结构：把数据元素放在任意的存储单元了，这组存储单元可以是连续的，也可以是不连续的。

什么是数据结构

- **物理结构**：



顺序存储结构

什么是数据结构

- **物理结构**：

867

500	a_4	710
…	…	…
560	a_2	930
…	…	…
710	a_5	855
…	…	…
855	a_6	null
…	…	…
867	a_1	560
…	…	…
930	a_3	500

链式存储结构

第一部分

线性结构—队列

线性结构（线性表）是最基本、最简单、最常用的数据结构。所有元素排成一排。除了第一个元素都有“前一个元素”，除了最后一个都有“后一个元素”。

常用的线性表：栈，队列，链表

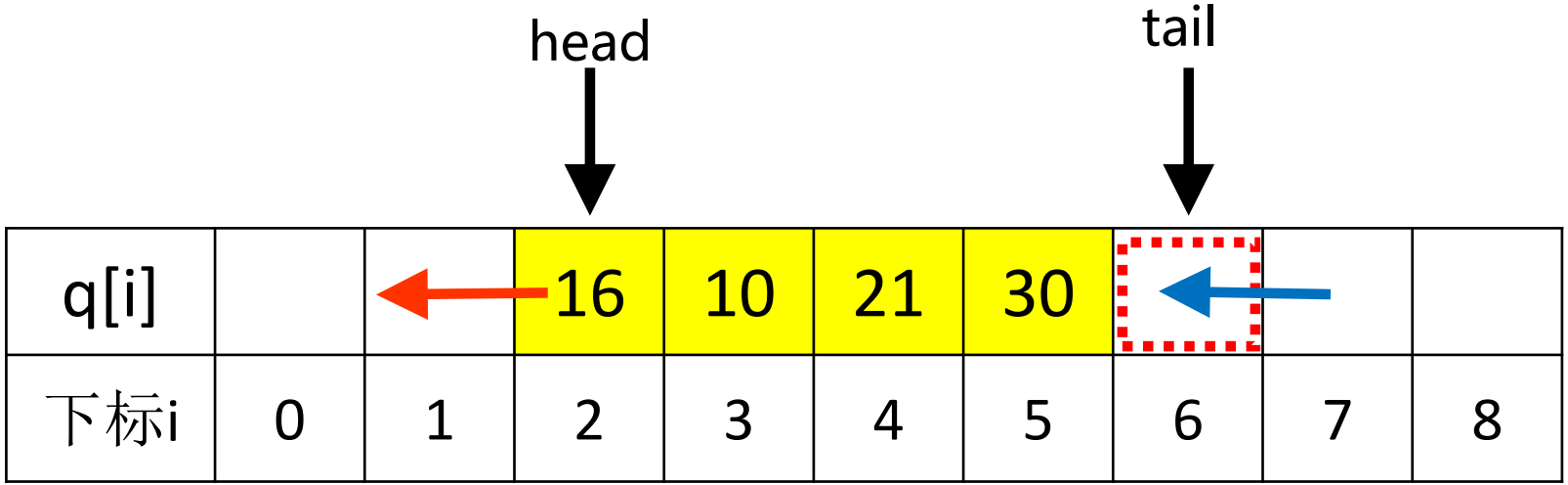
队 列（queue）

队列是一种运算受限的线性表：在一端删除（队首head），另一端插入（队尾tail）
操作原则：先进先出FIFO

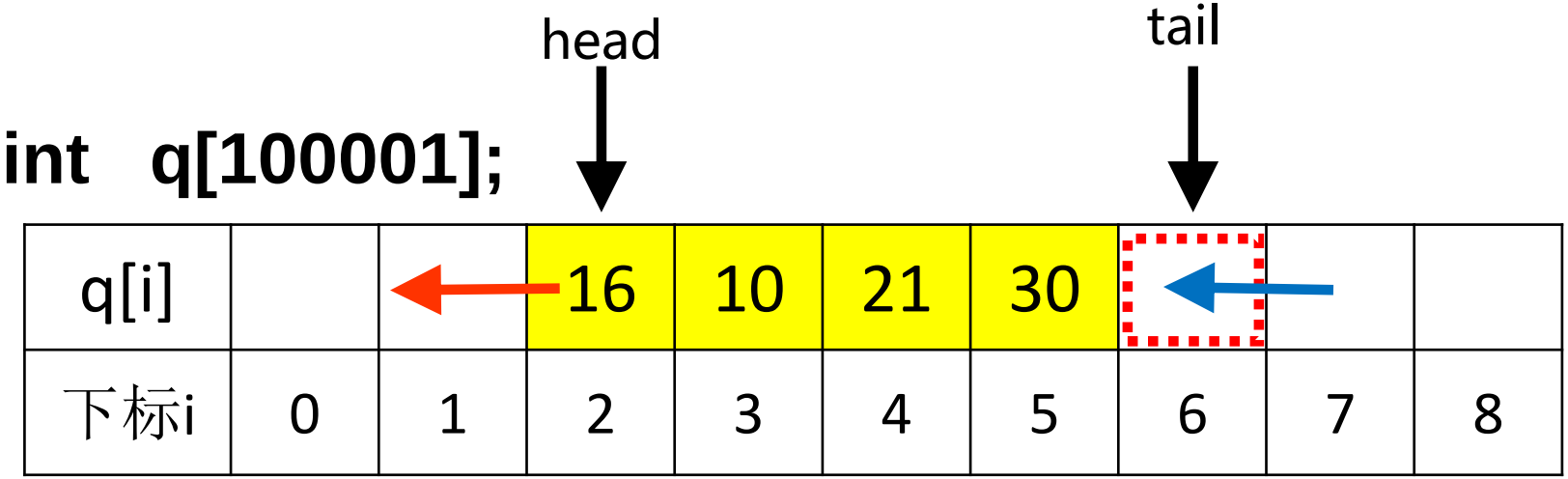
队列的两个操作：出队与入队

队首：head （习惯指向队首的前一位置，空的位置）

队尾：tail （指向队尾，最后一个元素）



int q[100001];



出队列(注意顺序)：

t=q[head];

head=head+1;

或：t=q[head++]

入队列(注意顺序)：

q[tail]=x;

tail++;

或：q[tail++]=x

队列非空：

Head<tail

【例1】卡片游戏

桌上有一叠牌，从第一张牌（即位于顶面的牌）开始从上往下依次编号为1~n；当至少还剩两张牌时进行以下操作：把第一张牌扔掉，然后把新的第一张放到整叠牌的最后。输入n，输出每次扔掉的牌。n<=54

样例输入：

7

样例输出：

1 3 5 7 4 2 6

```cpp
#include<cstdio>
#include<iostream>
using namespace std;
int q[110];
int main(){
    int n,head,tail;
    cin>>n;
    for(int i=0;i<n;i++)q[i]=i+1;
    head=0;tail=n;
    while(head<tail){
        cout<<q[head++]<<" ";
        q[tail++]=q[head++];
    }
    return 0;
}
```

思考：

队列使用了多少？
真的出队列了吗？
入队列了吗？
队列的对象可以是多个数据（结构体）

2.【例2-1】周末舞会

【题目描述】

假设在周末舞会上，男士们和女士们进入舞厅时，各自排成一队。跳舞开始时，依次从男队和女队的队头上各出一人配成舞伴。规定每个舞曲能有一对跳舞者。若两队初始人数不相同，则较长的那一队中未配对者等待下一轮舞曲。现要求写一个程序，模拟上述舞伴配对问题。

【输入】

第一行两队的人数;

第二行舞曲的数目。

【输出】配对情况。

【输入样例】

```
4 6
7
```

【输出样例】

```
1 1
2 2
3 3
4 4
1 5
2 6
3 1
```

3：1334【例2-3】围圈报数

【题目描述】
有 n 个人依次围成一圈，从第 1 个人开始报数，数到第m个人出列，然后从出列的下一个人开始报数，数到第m个人又出列，…，如此反复到所有的人全部出列为止。设 n 个人的编号分别为1，2，…，n，打印出列的顺序。
【输入】
n和m。
【输出】
出列的顺序。
【输入样例】
4 17
【输出样例】
1 3 4 2

4.1333【例2-2】Blah数集

【题目描述】

大数学家高斯小时候偶然间发现一种有趣的自然数集合Blah，对于以a为基的集合Ba定义如下：

(1)a是集合Ba的基，且a是Ba的第一个元素；

(2)如果x在集合Ba中，则2x+1和3x+1也都在集合Ba中；

(3)没有其他元素在集合Ba中了。

现在小高斯想知道如果将集合Ba中元素按照升序排列，第N个元素会是多少？

【输入】

输入包括很多行，每行输入包括两个数字，集合的基a(1≤a≤50))以及所求元素序号n(1≤n≤1000000)。

【输出】

对于每个输入，输出集合Ba的第n个元素值。

【输入样例】	【输出样例】
1 100	418
28 5437	900585

5.1335【例2-4】连通块

【题目描述】

一个n * m的方格图，一些格子被涂成了黑色，在方格图中被标为1，白色格子标为0。问有多少个四连通的黑色格子连通块。四连通的黑色格子连通块指的是一片由黑色格子组成的区域，其中的每个黑色格子能通过四连通的走法（上下左右），只走黑色格子，到达该联通块中的其它黑色格子。

【输入】

第一行两个整数n,m(1≤n,m≤100)，表示一个n * m的方格图。

接下来n行，每行m个整数，分别为0或1，表示这个格子是黑色还是白色。

【输出】

一行一个整数ans，表示图中有ans个黑色格子连通块。

【输入样例】

```
3 3
1 1 1
0 1 0
1 0 1
```

【输出样例】

```
3
```

【例2】家族

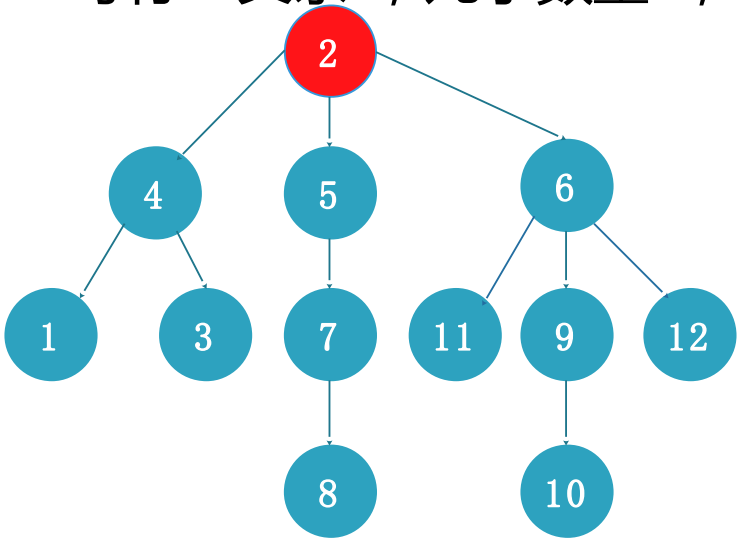
一个家族，有唯一的祖先，告诉成员之间的关系，然后按照辈份（按层）输出成员(最大100人)。

样例输入：

第一行：n；

以下n行：

每行：父亲f，儿子数量c，c个儿子




```
12
2 3 4 5
6
4 2 1 3
1 0
3 0
5 1 7
7 1 8
8 0
6 3 9 11
```

```
2 4 5 6 1 3 7 9 11 12 8 10
```

```
2 4 1 3 5 7 8 6 9 10 11 12
```

```
12 0
```

1. 保存父子关系

   儿子记住父亲；

   父亲记住儿子；

2. 确定祖先

算法描述：

- 读入数据，记录结点的父亲；
- 找出祖先root（没有父亲的结点）；
- Root进入队列
- While队列不空{
  - 出队列u；
  - 输出u；
  - 找出u的所有儿子v；
    - v进入队列
- }

```
int head=0,tail=1;
while(head<tail){
    int f=q[head++];
    cout<<f<<" ";
    for(int i=1;i<=n;i++)
        if(father[i]==f) q[tail++]=i;
}
```

STL 队列：

C++队列是一种容器适配器
#include<queue>

1. empty() 如果队列空则返回真
2. front() 返回第一个元素
3. pop() 删除第一个元素
4. push() 在末尾加入一个元素
5. size() 返回队列中元素的个数
6. back() 返回最后一个元素