数据结构

【基础知识】

一．数据结构

数据结构是计算机存储、组织数据的方式。数据结构是指相互之间存在一种或多种特定关系的数据元素的集合。通常情况下，精心选择的数据结构可以带来更高的运行或者存储效率。数据结构往往同高效的检索算法和索引技术有关。

数据(data) 是对客观事物的符号的表示。例如数值、图像、声音都属于数据的范畴。

数据元素(data element) 是数据的基本单位

数据对象（data object）是性质相同的数据元素的集合，是数据的一个子集。

数据结构（data structure）是相互之间存在一种或多种特定关系的数据元素的集合。

二．数据结构的分类

◆ 逻辑结构：指各数据元素之间的逻辑关系。

◆ 存储结构：就是数据的逻辑结构用计算机语言的实现。

◆ 线性结构：数据逻辑结构中的一类，它的特征是若结构为非空集，则该结构有且只有一个开始结点和一个终端结点，并且所有结点都最多只有一个直接前趋和一个直接后继。线性表就是一个典型的线性结构。

◆ 非线性结构：数据逻辑结构中的另一大类，它的逻辑特征是一个结点可能有多个直接前趋和直接后继。

三．数据结构的内涵

数据结构的含义：数据，关系，操作

例子：数组

数据：a[1], a[2], …, a[n]

关系：前驱/后继

操作：随机存取，插入，删除…

数据结构为算法服务：根据算法对数据的操作要求，设计合适的数据结构；实现同一套操作，可以用多种数据结构；如何降低时空复杂度，又方便实现，这是要注意的问题。

四．如何学习数据结构

⑴了解常见的抽象数据类型

⑵对每种 ADT，了解常见的逻辑结构

⑶对给定的逻辑结构，自己设计物理结构

⑷特殊算法需要自己归纳出 ADT 并设计逻辑结构

五．学习数据结构注意方面

1．把握数据结构的基本概念，要求领会“数据”和“结构”的内涵

2．对问题不盲目地套某种数据结构，要学会根据数据的特点构造出自己的结构

3．数据结构和算法是紧密联系，没有离开算法的数据结构

4．广泛地吸取新的知识点，掌握不同结构构造后的时空效率及其他特点

【学习要求】

了解常见的抽象数据类型；对每种 ADT，了解常见的逻辑结构；对给定的逻辑结构，自己设计物理结构；*特殊算法需要自己归纳出 ADT 并设计逻辑结构（PQ 树，后缀树）。其中，设计物理结构一般只是涉及数组和链结构，数组可以随机访问（设计下标计算公式），经典例子：哈希表，二叉堆，并查集，线段树；链结构应该根据元素间关系（链接）进行“移动”，经典例子：伸展树，二项堆，跳跃表。

【常见试题分类解析】

【例 1】求两个一元多项式的和。输入多项式方式为，多项式项数，每项系数和指数，按指数从大到小的顺序输入。

【分析】

多项式的算术运算是表处理的一个经典问题。建立两张表 a 、b 分别存放两个多项式的内容，建立表指针 ta、tb，指向表 a 和表 b 的元素，根据表 a 、b 元素中的指数大小合并输出。

1 、比较 ta、tb 指向元素的大小，若 ta 的指数大于 tb 的指数，输出 ta 元素，改变指针 ta；

2、若 ta 的指数小于 tb 的指数，输出 tb 元素，改变指针 tb；

3、若 ta 的指数等于 tb 的指数，ta、tb 元素的系数相加输出，同时改变指针 ta 和 tb；

4、若有一表取空，则输出另一表剩余的内容。

【源程序】

```
program ex11_5a;
type
node=record
zhi,xi:integer;
end;
ar=array[1..1000] of node;
var
a,b:ar;
ta,tb,n:integer;
begin
write('One : '); readln(n);{输入第一个多项式的系数和指数}
for ta:=n downto 1 do readln(a[ta].xi,a[ta].zhi);
ta:=n;
write('Two : '); readln(n);{输入第二个多项式的系数和指数}
for tb:=n downto 1 do readln(b[tb].xi,b[tb].zhi);
tb:=n;
write('Result is ');
while (ta>0) and (tb>0) do {当两个表均不空时}
begin {比较两表指针指向的项指数,输出指数小的项系数和指数, 同时改变该表指针}
if a[ta].zhi>b[tb].zhi then
```

```
begin
if a[ta].xi<0 then write(#8' '#8);
write(a[ta].xi,'x',a[ta].zhi,'+');
dec(ta);
end
else
if a[ta].zhi<b[tb].zhi then
begin
if b[tb].xi<0 then write(#8' '#8);
write(b[tb].xi,'x',b[tb].zhi,'+');
dec(tb);
end
else
begin {若两表指针指向的项指数相等,则两系数相加输出, 两表指针同时改变}
if b[tb].xi+a[ta].xi<>0 then
begin
if b[tb].xi+a[ta].xi<0 then write(#8' '#8);
write(b[tb].xi+a[ta].xi,'x',b[tb].zhi,'+');
end;
dec(ta);
dec(tb);
end;
end;
while ta>0 do {若有一表空,则输出另一表的剩余项}
begin
if a[ta].xi<0 then write(#8' '#8);
write(a[ta].xi,'x',a[ta].zhi,'+');
dec(ta);
end;
while tb>0 do
begin
if b[tb].xi<0 then write(#8' '#8);
write(b[tb].xi,'x',b[tb].zhi,'+');
dec(tb);
end;
writeln(#8' '#8);
readln;
end.
```

【例 2】设有 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，打印出出列的顺序。

【分析】本题我们可以用数组建立标志位等方法求解，但如果用上数据结构中循环链的思想，则更贴切题意，解题效率更高。n 人围成一圈，把一人看成一个结点，n 人之间的关系采用

链接方式，即每一结点有一个前继结点和一个后继结点，每一个结点有一个指针指向下一个结点，最后一个结点指针指向第一个结点。这就是单循环链的数据结构。当m人出列时，将m结点的前继结点指针指向m结点的后继结点指针，即把m结点驱出循环链。

1、 建立循环链表。当用数组实现本题链式结构时，数组 a[i]作为"指针"变量来使用，a[i]存放下一个结点的位置。设立指针 j 指向当前结点，则移动结点过程为 j:=a[j]，当数到 m 时，m 结点出链，则 a[j]:=a[a[j]]。　　当直接用链来实现时，则比较直观，每个结点有两个域：一个数值域，一个指针域，当数到 m 时，m 出链，将 m 结点的前继结点指针指向其后继结点；

2、 设立指针，指向当前结点，设立计数器，计数数到多少人；

3、 沿链移动指针，每移动一个结点，计数器值加 1 ，当计数器值为m时，则m结点出链。计数器值置为 1 ；

4、 重复 3 、直到 n 个结点均出链为止。

```
program ex11-6a;
  const n=14;m=4;{设有 10 个人,报到 4 的人出列}
    var a:array[1..n] of integer;
      i,j,k,p:integer;
    begin
      for i:=1 to n-1 do a[i]:=i+1;{建立链表}
      a[n]:=1;j:=n;k:=1;p:=0;{第 n 人指向第 1 人,并置初始}
      repeat
        j:=a[j];k:=k+1;{报数,计数器加 1}
        if k=m then {数到 m,m 人出队,计数器置 1}
          begin
            write(a[j]:4);p:=p+1;a[j]:=a[a[j]];k:=1;
          end
        until p=n;{直到 n 个人均出队为止}
    end.
```

【例 3】

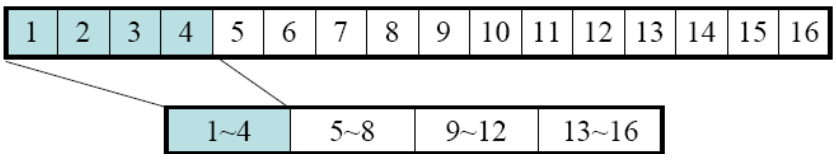
实现一个 n 个元素的线性表 A。每次可以修改其中一个元素，也可以询问闭区间[p, q]中元素的最小值。1<=n,m<=100000

【分析】

利用二级检索的思想，设块长为 L, 则一共有 n/L 个块；维护数组 B, 保存每个块的最小值

Modify(x, y)

- A[x] = y O(1)
- 重算 x 所在块的最小值(更新 B) O(L)



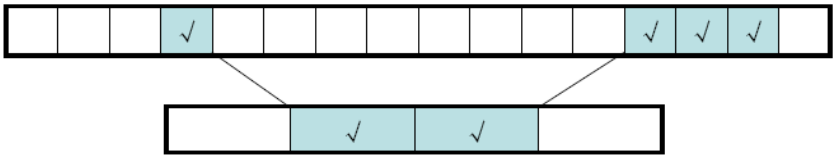
【解答】

Min(a, b)

- 把区间[a, b]分成若干部分
- 完整块: 一共最多 n/L 个块 O(n/L)
- 非完整块: 首尾各最多 L-1 个元素 O(L)

每次操作时间复杂度: O(n/L+L)

- 设 $L=O^{(n1/2)}$则渐进时间复杂度为 $O^{(n1/2)}$



【例 4】

k 路归并问题：把 k 个有序表合并成一个有序表.，元素共有 n 个。

【分析】

每个表的元素都是从左到右移入新表；

把每个表的当前元素放入二叉堆中, 每次删除最小值并放入新表中, 然后加入此序列的下一个元素；

每次操作需要 logk 时间, 因此总共需要 nlogk 的时间。

【例 5】

丑数问题：素因子都在集合{2, 3, 5, 7}的数称为 ugly number，求第 n 大的丑数

【分析】

初始：把 1 放入优先队列中；

每次从优先队列中取出一个元素 k，把 2k, 3k, 5k, 7k 放入优先队列中；

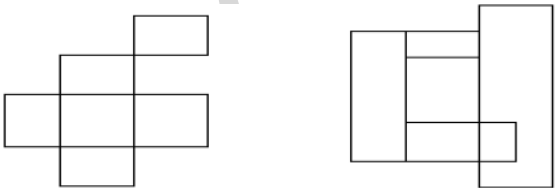
从 2 开始算，取出的第 n 个元素就是第 n 大的丑数；

每取出一个数，插入 4 个数，因此任何堆里的

【例 6】

在平面上画了 N 个长方形，每个长方形的边平行于坐标轴并且顶点坐标为整数。我们用以下方式定义印版：每个长方形是一个印版；如果两个印版有公共的边或内部，那么它们组成新的印版，否则这些印版是分离的。

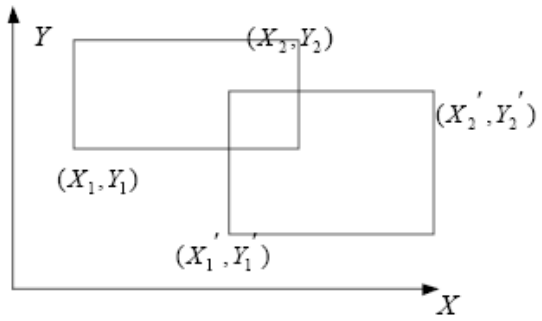
数出印版的个数. 左图有两个，右图只有一个



【分析】

把矩形看作点，有公共边的矩形连边，问题转化为求连通分量的个数。

判断方法如下图：



【例 7】

离线最大值问题：设计一个集合, 初始为空，每次可以插入一个 1~n 的数(1~n 各恰好被插入一次),也可以删除最大值, 要求 m 次操作的总时间尽量小。

【分析】

在最后加入 n-m 次虚拟的 MAX 操作, 并记第 i 个 MAX 操作为 Mi,记 M1 之前的插入序列为 S1, Mi-1(1<i<=m) 和 Mi 之间的插入序列为 Si。

如果 n 在 Sj 中被插入，则 Mj 的输出一定是 n. 然后删除 Mj，即把 Sj 合并到 Sj+1 中，然后再查找 n-1 所在的序列 Sk，则 Mk 的输出为 n-1…如此下去，从 n 到 1 依次查找每个数所在序列，就可以得到它后面的 MAX 操作的结果, 并把它和紧随其后的序列合并。