

HUNAN UNIVERSITY

程序设计训练

报 告

学生姓名 侯茱元

学生学号 201926010214

专业班级 软件1902

指导老师 陆邵飞

助 教 汪国庆

2020 年7月13日

目录

[题1：小中大（ID：201903-1） 3](#_Toc45549680)

[一、 问题描述 3](#_Toc45549681)

[二、 问题分析及边界条件 3](#_Toc45549682)

[三、 算法设计 4](#_Toc45549683)

[四、 详细设计： 4](#_Toc45549684)

[五、 样例设计与测试 4](#_Toc45549685)

[六、 分析与总结： 5](#_Toc45549686)

[题2：公共钥匙盒（ID：201709-2） 7](#_Toc45549687)

[一、 问题描述 7](#_Toc45549688)

[二、 问题分析及边界条件： 8](#_Toc45549689)

[三、 算法设计： 8](#_Toc45549690)

[四、 详细设计： 8](#_Toc45549691)

[五、 样例设计与测试： 10](#_Toc45549692)

[六、 分析与总结： 12](#_Toc45549693)

[题3：化学方程式（ID：201912-3） 13](#_Toc45549694)

[一、 问题描述： 13](#_Toc45549695)

[二、 问题分析及边界条件： 14](#_Toc45549696)

[三、 算法设计： 14](#_Toc45549697)

[四、 详细设计： 15](#_Toc45549698)

[五、 样例设计与测试 17](#_Toc45549699)

[六、 分析与总结 18](#_Toc45549700)

[题4：学生排队（ID：201703-2） 19](#_Toc45549701)

[一、 问题描述： 19](#_Toc45549702)

[二、 问题分析与边界条件： 20](#_Toc45549703)

[三、 算法设计： 20](#_Toc45549704)

[四、 详细设计： 20](#_Toc45549705)

[五、 样例设计与测试 21](#_Toc45549706)

[六、 分析与总结 22](#_Toc45549707)

[题5：游戏（ID：201712-2） 23](#_Toc45549708)

[一、 问题描述 23](#_Toc45549709)

[二、 问题分析与边界条件： 24](#_Toc45549710)

[三、 算法设计： 24](#_Toc45549711)

[四、 详细设计： 24](#_Toc45549712)

[五、 样例设计与测试 25](#_Toc45549713)

[六、 分析与总结 26](#_Toc45549714)

题1：小中大（ID：201903-1）

1. 问题描述

[题目背景]

在数据分析中，最小值最大值以及中位数是常用的统计信息。

[题目描述]

老师给了你n个整数组成的测量数据，保证有序(可能为升序或降序)，可能存在重复的数据。请统计出这组测量数据中的最大值、中位数以及最小值，并按照从大到小的顺序输出这三个数。

[输入格式]

从标准输入读入数据。

第一行输入一个整数n,在第二行中存在n个有序的整数，表示测量数据，可能为升序或降序排列，可能存在连续多个整数相等，整数与整数之间使用空格隔开。

[输出格式]

输出到标准输出。

包含一行，包括最大值、中位数以及最小值共三个数，并按照从大到小的顺序输出。数据与数据之间使用空格隔开。对于整数请直接输出整数，对于可能出现的分数，请输出四舍五入保留1位小数的结果。

1. 问题分析及边界条件

要处理的问题：

1. 出现分数和整数的情况要分别处理
2. 按升序和降序输入的两种情况

边界条件：

出现全部相同的输入

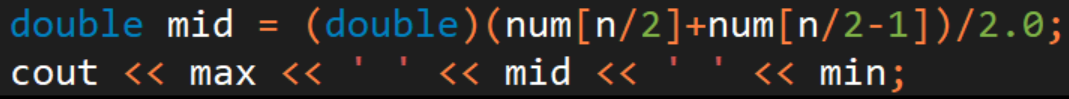
1. 算法设计

通过上面分析，采用以下方法解决：

1. 使用整形数组，比较数组中第一个与最后一个元素的大小，依次判断升序/降序输入。
2. 如果是整数可以直接输出：



如果是分数，则类型转化：



1. 升序和降序也可以考虑直接sort（）排序。
2. 判断为小数时也可以直接输出“.5”/控制格式输出。
3. 详细设计：
4. 程序主框架设计：

int main(){

……//输入n，数组num[n];

long long int min, max;//定义最大最小值

min = num[0]<=num[n-1]?num[0]:num[n-1];

max = num[0]>=num[n-1]?num[0]:num[n-1];//比较，以存入相应的

if (n%2==1)

……//整数情况1

else if (n%2==0)

{ if ((num[n/2]+num[n/2-1])%2==0)

……//整数情况2

else if ((num[n/2]+num[n/2-1])%2==1)

……//小数情况

}

return 0;

}

1. 样例设计与测试

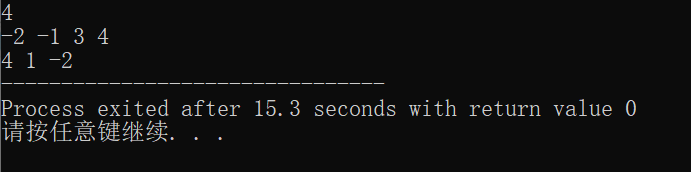
[样例1输入]

4

-2 -1 3 4

[样例1输出]

4 1 -2



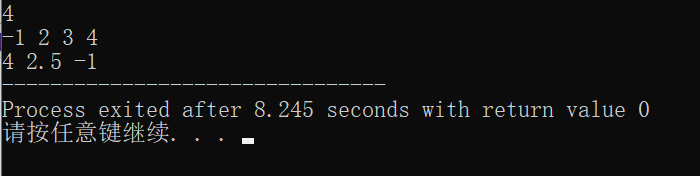
[样例2输入]

4

-1 2 3 4

[样例2输出]

4 2.5 -1



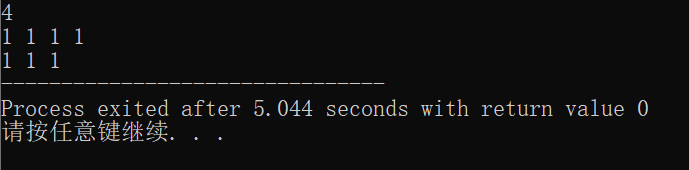
[样例3输入]

4

1 1 1 1

[样例3输出]

1 1 1



1. 分析与总结：

一定要认真审题，不然非常容易写到奔溃。（注：对于整数请直接输出整数，对于可能出现的分数，请输出四舍五入保留1位小数的结果。）

题2：公共钥匙盒（ID：201709-2）

1. 问题描述

[问题描述]

有一个学校的老师共用N个教室，按照规定，所有的钥匙都必须放在公共钥匙盒里，老师不能带钥匙回家。每次老师上课前，都从公共钥匙盒里找到自己上课的教室的钥匙去开门，上完课后，再将钥匙放回到钥匙盒中。

钥匙盒一共有N个挂钩，从左到右排成一排，用来挂N个教室的钥匙。一串钥匙没有固定的悬挂位置，但钥匙上有标识，所以老师们不会弄混钥匙。

每次取钥匙的时候，老师们都会找到自己所需要的钥匙将其取走，而不会移动其他钥匙。每次还钥匙的时候，还钥匙的老师会找到最左边的空的挂钩，将钥匙挂在这个挂钩上。如果有多位老师还钥匙，则他们按钥匙编号从小到大的顺序

还。如果同一时刻既有老师还钥匙又有老师取钥匙，则老师们会先将钥匙全还回去再取出。

今天开始的时候钥匙是按编号从小到大的顺序放在钥匙盒里的。有位老师要上课，给出每位老师所需要的钥匙、开始上课的时间和上课的时长，假设下课时间就是还钥匙时间，请问最终钥匙盒里面钥匙的顺序是怎样的?

[输入格式]

输入的第一行包含两个整数N, K

接下来行，每行三个整数w, s, c, 分别表示- -位老师要使用的钥匙编号、开始.上课的时间和上课的时长。可能有多位老师使用同一把钥匙，但是老师使用钥匙的时间不会重叠。

保证输入数据满足输入格式，你不用检查数据合法性。

[输出格式]

输出一行，包含N个整数，相邻整数间用一个空格分隔，依次表示每个挂钩上挂的钥匙编号。

* + 1. 问题分析及边界条件：

要处理的问题：

1. 钥匙可能存在多次被取走的情况。
2. 同一时刻，还钥匙在先，取钥匙在后。
3. 多位老师同时还钥匙，按钥匙编号从小到大的顺序还。
4. 老师们会将钥匙放到最左边的空挂钩上。
   * 1. 算法设计：
5. 声明结构体，存放在一次取还的过程中钥匙的编号，取还的时间。
6. 在所有还钥匙的时间中，找到最大的时刻，以方便时刻的遍历。
7. 以时刻作为循环的条件，可以遍历每一时刻，钥匙可能存在多次被取走的情况也在其中。
8. 循环时，先考虑还在考虑取。
9. 循环前，将所有老师要取的钥匙按编号排序，方便出现多位老师同时还钥匙的情况时的顺序问题。
10. 建立一个一维数组作为挂钩，存放钥匙的编号。
    * 1. 详细设计：
11. 声明结构体

struct teacher

{

int num;//编号

int s;//开始时间

int e;//结束时间

int se;//还钥匙时间

};

1. 函数cmp用于sort（）排序

bool cmp(teacher t1, teacher t2)

{

return t1.num < t2.num;

}

1. 程序主框架设计：

int main()

{

……//输入N，K

teacher t[K];//老师的取还行为

int ans[N];//钥匙挂钩

for(int i = 0; i < N; i++)

ans[i] = i+1;//初始化

int zt = 0;//最后还钥匙时间

for(int i = 0; i < K; i++)//初始化

{

cin >> t[i].num >> t[i].s >> t[i].e;

t[i].se = t[i].s + t[i].e;

if(zt < t[i].se)

zt = t[i].se;

}

sort(t, t+K, cmp);//按钥匙编号排序

for(int time = 0; time <= zt; time++)

{

for(int i = 0; i < K; i++)//还钥匙

if(time == t[i].se)

for(int j = 0; j < N; j++)

if(ans[j] == 0){

ans[j] = t[i].num;

break;}

for(int i = 0; i < K; i++)//取钥匙

if(time == t[i].s)

for(int j = 0; j < N; j++)

if(ans[j] == t[i].num){

ans[j] = 0;

break;}

}

for(int i = 0; i < N; i++)

cout<<ans[i]<<" ";

return 0;

}

* + 1. 样例设计与测试：

[样例1输入]

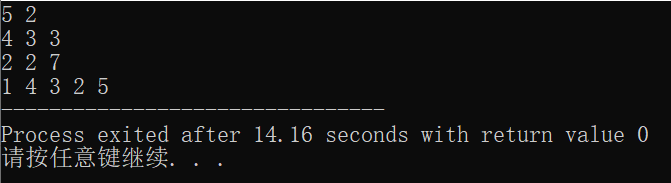
5 2

4 3 3

2 2 7

[样例1输出]

1 4 3 2 5



样例输入

5 7

1 11 4

3 3 12

1 15 12

2 7 20

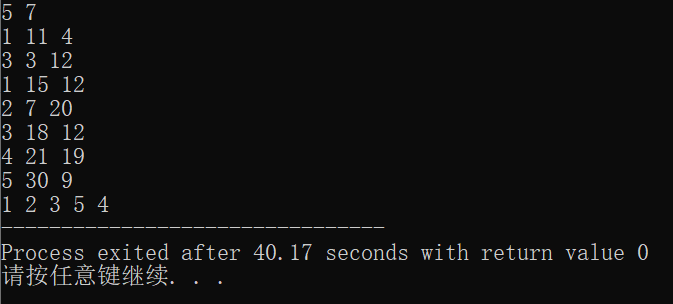
3 18 12

4 21 19

5 30 9

[样例2输出]

1 2 3 5 4



[样例3输入]

1 3

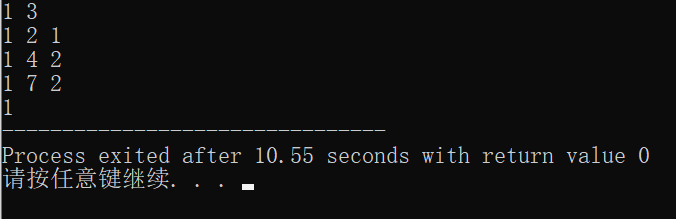
1 2 1

1 4 2

1 7 2

[样例3输出]

1



* + 1. 分析与总结：

对结构体的应用

利用sort（）函数对结构体排序

对for循环if条件的恰当应用

题3：化学方程式（ID：201912-3）

1. 问题描述：

[题目描述]

化学方程式，也称为化学反应方程式，是用化学式表示化学反应的式子。给出一组化学方程式，请你编写程序判断每个方程式是否配平（也就是方程式中等号左右两边的元素种类和对应的原子个数是否相同）。

本题给出的化学方程式由大小写字母、数字和符号（包括等号=、加号+、左圆括号和右圆括号）组成，不会出现其他字符（包括空白字符，如空格、制表符等），化学方程式的格式与化学课本中的形式基本相同（化学式中表示元素原子个数的下标用正常文本，如H2O写成H2O），用自然语言描述如下：

化学方程式由左右两个表达式组成，中间用一个等号三连接，如2H2+O2=2H2O；

表达式由若干部分组成，每部分由系数和化学式构成，部分之间用加号+连接，如2H2+O2、2H2O；

系数是整数或空串，如为空串表示系数为1；

整数由一个或多个数字构成；

化学式由若干部分组成，每部分由项和系数构成，部分之间直接连接，如H2O、CO2、Ca(OH)2、Ba3(PO4)2；

项是元素或用左右圆括号括起来的化学式，如H、Ca、(OH)、(P04)；

元素可以是一个大写字母，也可以是一个大写字母跟着一个小写字母，如H、O、Ca。

[输入格式]

从标准输入读入数据。  
输入的第一行包含一个正整数n，表示输入的化学方程式个数。  
接下来n行，每行是一个符合定义的化学方程式。

[输出格式]

输出到标准输出。  
输出共n行，每行是一个大写字母Y或N，回答输入中相应的化学方程式是否配平。

* 1. 问题分析及边界条件：

要处理的问题

1. 化学方程式的读入和元素的存储
2. 如何判断方程式守恒

边界条件：

输入的化学式中含有相同元素

* 1. 算法设计：

通过上述分析，将采用下面的方法解决相关问题:

1. 采用map储存化学元素和对应的系数

unordered\_map<string, int> ans;

1. 通过函数来完成对化学方程式中化学元素的分离提取

int computeDigit(int& first, int last)

void f(int first, int last, int e)

void expression(int first, int last, int e)

1. 通过最后判断map中元素系数是否为0判断是否守恒

auto i = find\_if(ans.begin(), ans.end(), [](const pair<string, int>& p) {

return p.second != 0;

});

cout << ((i == ans.end()) ? "Y" : "N") << "\n";

* 1. 详细设计：

程序主框架设计

int n;

string formula;

unordered\_map<string, int> ans;

//返回formula的[first,last]区间对应的数字,注意函数返回之后传递给first的实参将移动到第一个非数字字符的位置

int computeDigit(int& first, int last) {

int i = 0;

for (; first <= last and isdigit(formula[first]); ++first)

i = i \* 10 + formula[first] - '0';

return i == 0 ? 1 : i;

}

void f(int first, int last, int e) { //计算formula的[first,last]区间的原子及其对应系数，基本系数为e

if (first == last or (last - first == 1 and islower(formula[last]))) { //化学式是单个原子

ans[formula.substr(first, last - first + 1)] += e; //该原子个数递增e

return;

}

e \*= computeDigit(first, last); //该化学式内所有原子基本系数要乘上该化学式起始系数

for (int i = first, j = i + 1; i <= last; i = j, ++j) { //遍历化学式

if (isupper(formula[i])) { //是原子

if (j <= last and islower(formula[j]))

++j;

int k = j;

f(i, k - 1, e \* computeDigit(j, last)); //递归处理

} else if (formula[i] == '(') { //遇到(

for (int num = 1; num != 0; ++j) { //找到对应的)位置存储在j中

if (formula[j] == '(')

++num;

else if (formula[j] == ')')

--num;

}

int k = j;

f(i + 1, k - 1, e \* computeDigit(j, last)); //递归处理

}

}

}

void expression(int first, int last, int e) { //按+分离出所有化学式

for (int i = first, j = first; i <= last; i = j + 1) {

j = formula.find('+', i);

if (j == string::npos or j > last)

j = last + 1;

f(i, j - 1, e);

}

}

int main() {

cin >> n;

while (n--) {

cin >> formula;

ans.clear();

int k = formula.find('='); //按=将方程式分割成两部分

expression(0, k - 1, 1);

expression(k + 1, formula.size() - 1, -1);

//查找有没有原子对应个数不为0

auto i = find\_if(ans.begin(), ans.end(), [](const pair<string, int>& p) {

return p.second != 0;

});

cout << ((i == ans.end()) ? "Y" : "N") << "\n";

}

return 0;

}

* 1. 样例设计与测试

[样例输入]

11

4Au+8NaCN+2H2O+O2=4Na(Au(CN)2)+4NaOH

4Zn+10HNO3=4Zn(NO3)2+NH4NO3+3H2O

H2+O2=H2O

2H2+O2=2H2O

H2+Cl2=2NaCl

H2+Cl2=2HCl

CH4+2O2=CO2+2H2O

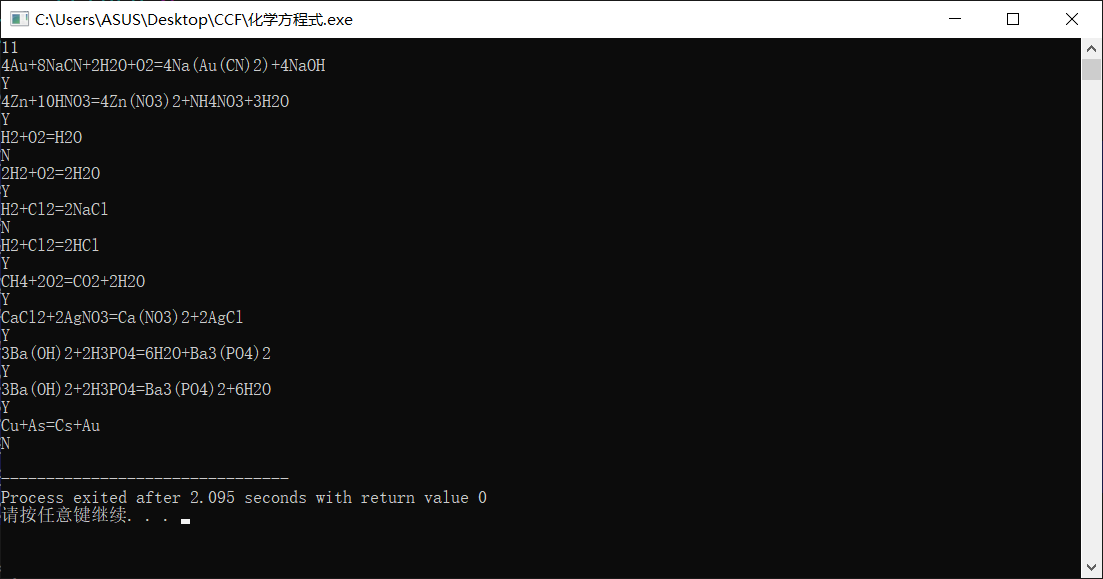
CaCl2+2AgNO3=Ca(NO3)2+2AgCl

3Ba(OH)2+2H3PO4=6H2O+Ba3(PO4)2

3Ba(OH)2+2H3PO4=Ba3(PO4)2+6H2O

Cu+As=Cs+Au

[样例输出]



* 1. 分析与总结

需要按照化学式和化学元素进行分离，不能只是按照是否是字母就完成，并且需要设计进行左加右减的处理函数。

题4：学生排队（ID：201703-2）

1. 问题描述：

[问题描述]

体育老师小明要将自己班上的学生按顺序排队。他首先让学生按学号从小到大的顺序排成一排，学号小的排在前面，然后进行多次调整。一次调整小明可能让一位同学出队，向前或者向后移动一段距离后再插入队列。

例如，下面给出了一组移动的例子，例子中学生的人数为8人。

0)初始队列中学生的学号依次为1，2，3，4, 5，6, 7，8;

1)第一次调整，命令为“3号同学向后移动2”，表示3号同学出队，向后移动2名同学的距离，再插入到队列中，新队列中学生的学号依次为1，2，4，5，3，6，7，8;

2)第二次调整，命令为“8号同学向前移动3”，表示8号同学出队，向前移动3名同学的距离，再插入到队列中，新队列中学生的学号依次为1，2，4，5，8，3，6，7;

3)第三次调整，命令为“3号同学向前移动2”，表示3号同学出队，向前移动2名同学的距离，再插入到队列中，新队列中学生的学号依次为1，2，4，3，5，8，6，7。

小明记录了所有调整的过程，请问，最终从前向后所有学生的学号依次是多少?

请特别注意，上述移动过程中所涉及的号码指的是学号，而不是在队伍中的位置。在向后移动时，移动的距离不超过对应同学后面的人数，如果向后移动的距离正好等于对应同学后面的人数则该同学会移动到队列的最后面。在向前移动时，移动的距离不超过对应同学前面的人数，如果向前移动的距离正好等于对应同学前面的人数则该同学会移动到队列的最前面。

[输入格式]

输入的第一行包含- 一个整数n，表示学生的数量，学生的学号由1到n编号。

第二行包含一个整数m，表示调整的次数。

接下来m行，每行两个整数p，q，如果q为正，表示学号为p的同学向后移动q，如果q为负，表示学号为p的同学向前移动-q。

[输出格式]

输出一行，包含n个整数，相邻两个整数之间由一个空格分隔，表示最终从前向后所有学生的学号。

二、 问题分析与边界条件：

要解决的问题：

1. 找到编号为i的学生
2. 不能单纯的与相隔t位置的同学换位，需要平移
3. 如何移动
4. 分别区分前后移动

三、 算法设计：

1. 移动可以递推，与相邻的同学调换位置，t次之后到达目标位置。
2. t--；次数递减。
3. 在t>0时，判断结束为t<=0; 在t<0时，判断结束为t>=0;

四、 详细设计：

主程序设计：

int main()

{

……//输入n，m，初始化num[n]

while(m>0)

{

int s = 0, t, r;//学号r，移动方向t，s学生现在所在位置

cin >> r >> t;

for (int i=1; i<=n; i++)

if (num[i] == r)

s = i;

if (t>0)

for (;t>0;t--)

{

int temp = num[s];

num[s] = num[s+1];

num[s+1] = temp;

s++;

}

else if (t<0)

for (;t<0;t++)

{

int temp = num[s];

num[s] = num[s-1];

num[s-1] = temp;

s--;

}

m--;

}

for (int i = 1; i <= n; i++)

cout << num[i] << " ";

return 0;

}

五、 样例设计与测试

[样例1输入]

8

3

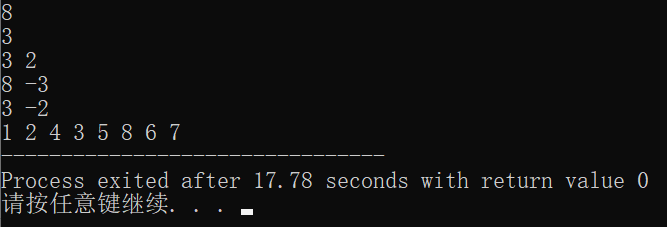
3 2

8 -3

3 -2

[样例1输出]

1 2 4 3 5 8 6 7



[样例2输入]

10

4

2 6

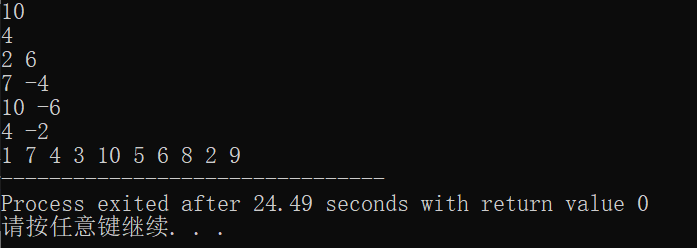
7 -4

10 -6

4 -2

[样例2输出]

1 7 4 3 10 5 6 8 2 9



六、 分析与总结

移动位置时一定要找到对应的同学，移动的方法来自线性代数。

题5：游戏（ID：201712-2）

1. 问题描述

[问题描述]

有*n*个小朋友围成一圈玩游戏，小朋友从1至*n*编号，2号小朋友坐在1号小朋友的顺时针方向，3号小朋友坐在2号小朋友的顺时针方向，……，1号小朋友坐在*n*号小朋友的顺时针方向。  
　　游戏开始，从1号小朋友开始顺时针报数，接下来每个小朋友的报数是上一个小朋友报的数加1。若一个小朋友报的数为*k*的倍数或其末位数（即数的个位）为*k*，则该小朋友被淘汰出局，不再参加以后的报数。当游戏中只剩下一个小朋友时，该小朋友获胜。  
　　例如，当n=5, k=2时：  
　　1号小朋友报数1；  
　　2号小朋友报数2淘汰；  
　　3号小朋友报数3；  
　　4号小朋友报数4淘汰；  
　　5号小朋友报数5；  
　　1号小朋友报数6淘汰；  
　　3号小朋友报数7；  
　　5号小朋友报数8淘汰；  
　　3号小朋友获胜。  
  
　　给定*n*和*k*，请问最后获胜的小朋友编号为多少？[提示]

因为所有小球的初始位置都为偶数，而且线段的长度为偶数，可以证明，不会有三个小球同时相撞，小球到达线段端点以及小球之间的碰撞时刻均为整数。

同时也可以证明两个小球发生碰撞的位置一定是整数(但不一定是偶数)。

[输入格式]

　输入一行，包括两个整数*n*和*k*，意义如题目所述。

[输出格式]

输出一行，包含一个整数，表示获胜的小朋友编号。

二、 问题分析与边界条件：

要解决的问题：

1. 报数和循环同时进行
2. 判断小朋友是否出局

三、 算法设计：

1. 使用一维数组并且初始化为他们的序号
2. 让报数的变量在不符合条件时加一
3. 如果小朋友出局，则元素值变为0
   1. 详细设计：

程序主框架设计：

int main() {

int n,k;

cin >> n >> k;

int a[n];

for(int i=0; i<n; i++) {

a[i]=i+1;

}

int count=n;

int num=1;

for(int i=0; count>1; i++) {

if(a[i%n]!=0) { //i号小朋友未出局

if(num%k==0||num%10==k) {

count--; //总人数-1

a[i%n]=0; //标记出局

}

num++; //报数+1

}

}

for(int i=0; i<n; i++) {

if(a[i]!=0) {

cout << a[i] << endl;

}

}

return 0;

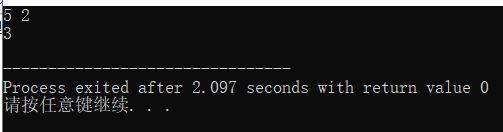
}

* 1. 样例设计与测试

样例输入

5 2

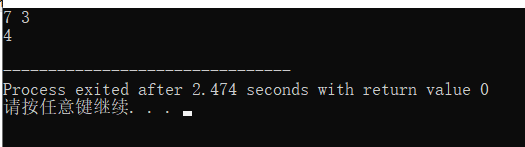
样例输出



样例输入

7 3

样例输出



* 1. 分析与总结

使用数组模拟进行求解