编程与赛题基础

—— 郑任毅

编程基础

- 数组的大小?

时间限制
200 ms
内存限制
65536 kB
代码长度限制
8000 B
判题程序
Standard
作者
CHEN, Yue

编程基础

- 数组的大小?
  - 大部分赛题是 64MB （ 65536 KB ）内存限制
  - int 型是 4B
  - 64MB/4B = 16M = 16e6 = 1.6e7
  - int[10000000] 的数组基本没有压力

编程基础

- 数组的大小?
  - 大部分赛题是 64MB （ 65536 KB ）内存限制
  - int 型是 4B
  - 64MB/4B = 16M = 16e6 = 1.6e7
  - int[10000000] 的数组基本没有压力
- 声明数组的习惯?

编程基础

- 数组的大小?
  - 大部分赛题是 64MB （ 65536 KB ）内存限制
  - int 型是 4B
  - 64MB/4B = 16M = 16e6 = 1.6e7
  - int[10000000] 的数组基本没有压力
- 声明数组的习惯?
  - 稍微开大一点，比如 10000 + 5
  - 大的数组开成全局数组

编程基础

- 做题的过程?

编程基础

- 做题的过程?
  - 仔细读题：模拟题要我们干什么？超过 / 不低于这类词的描述，以及一些不确定的信息能在样例中找到答案
  - 数据范围：什么级别的数据，能接受什么复杂度?
  - 制定做法并用代码实现
  - debug： 制造样例并测试

编程基础

- int 整数的范围是多少?

编程基础

- int 整数的范围是多少?
  - [-2147483648, 2147483647]
  - 简单来说就是略大于 2e9

编程基础

- int 整数的范围是多少?
  - [-2147483648, 2147483647]
  - 简单来说就是略大于 2e9
- 变量或数组的初始值?

编程基础

- int 整数的范围是多少?
  - [-2147483648, 2147483647]
  - 简单来说就是略大于 2e9
- 变量或数组的初始值?
  - 全局变量初始值为 0(或二进制位全为 0)
  - map 初始 value 为 0
  - 局部变量根据开发环境，很可能是随机值

编程基础

- 写完程序之后做什么？提交?

编程基础

- 写完程序之后做什么？提交?
  - 测试题目给出的样例
  - 自己造一些样例进行测试，尤其注意考虑边界情况，会不会有数组访问越界、除数为 0 、初始化这类的问题

编程基础

- 出现 bug 怎么 debug？

编程基础

- 出现 bug 怎么 debug？
  - 语法问题：编译不过
  - 运行时问题：死循环、数组越界、除数为 0 、递归爆栈、初始化不正确
  - 逻辑问题：算法有错、实现有错

编程基础

- 出现 bug 怎么 debug？
  - 设置断点，单步运行，追踪变量
  - 单元测试：注释其他部分，对某一部分代码设置初始值后单独运行，观察结果
  - 运行时输出（个人推荐）：在代码中增加一些输出语句输出中间变量，观察是否是预期的值

赛题类型

- PAT(B) 101-141-1-2017-03-04
- A 数组存储、遍历、数值比较　1066
- B 字符串读入、比较、数值计数 1067
- C 数组存储、元素访问、遍历　1068
- D 字符串存储、比较　1069
- E 数组排序、贪心　1070
- PAT(B) 101-140-3-2016-12-10
- A 数组存储、比较、访问　1061
- B 数值遍历、判断、 gcd　1062
- C sqrt、数组最大值　1063
- D 字符串读入、求和、排序去重 / 比较　1064
- E vis 数组的使用、排序　1065
-

赛题类型

- PAT(B) 101-139-2-2016-09-11
- A 类似数字拆位、遍历求和1056
- B 求和、数制拆分　　1057
- C 数组存储、字符读入比较、排序 1058
- D 数字读入、 vis 使用、素数判断　1059
- E 数组排序、贪心　　1060
- PAT(B) 101-133-1-2016-03-12
- A 三角函数 1051
- B 字符串读入、存储、访问1052
- C 数组存储、比较、计数　1053
- D 字符串处理、求和求均值　　1054
- E 结构体排序　　1055
-

赛题类型

- PAT(B) 101-131-3-2015-12-05
- A 求和、比较、计数　1046
- B 格式化读入、求和、求最大值 1047
- C 字符串处理　　1048
- D 计算贡献、求和　　1049
- E 排序、枚举因数、数组赋值　　1050
- PAT(B) 101-131-2-2015-09-12
- A 字符串与数的存储、访问1041
- B 字符串行读入、遍历与统计　　1042
- C 字符串读入、字符统计、输出 1043
- D 进制转换、字符串比较　1044
- E 前后缀最大值　1045
-

赛题类型

- PAT(A) 101-141-1-2017-03-04
- C 统计点的度数 1126
- D 模拟瞎搞（记录深度遍历输出） 1127
-
- PAT(A) 101-140-3-2016-12-10
- C 邻接矩阵、判断点数、重复点、是否有边 1122
- D 模拟 AVL 1123
-
- PAT(A) 101-139-2-2016-09-11
- C 并查集 1118
- D dfs 区间判定 1119

赛题类型

- PAT(Basic):
  - 1、前半：
    - 数组存储、访问、各种求和计数比较
    - 字符串存储、处理、比较
    - 数制与数位
    - 一些数学函数的使用
  - 以花式模拟为主

赛题类型

- PAT(Basic):
  - 2、后半：
    - 排序
    - 思路
    - 贪心
  - 以排序之后找思路为主

赛题类型

- PAT(Advanced):
  - 1、前半：
    - PAT(Basic) 的后两题
    - 英文版
  - 以读题之后排序找思路为主

赛题类型

- PAT(Advanced):
  - 2、后半：
    - 图论基础（图的存储与边的访问）
    - 模拟（以二叉树为主）
    - 简单算法题（并查集等）
  - 模拟以及简单算法

读入

- 整数：
  - 对于数组的依次读入
  - 格式化读入： scanf(“%d:%d:%d”,&hour,&min,&sec)

读入

- 整数：
  - 对于数组的依次读入
  - 格式化读入： scanf(“%d:%d:%d”,&hour,&min,&sec)
- 字符串：
  - %c or %s ?
  - char[] or string ?

读入

- 整数：
  - 对于数组的依次读入
  - 格式化读入： scanf(“%d:%d:%d”,&hour,&min,&sec)
- 字符串：
  - %c or %s ?
  - char[] or string ?
- scanf/printf or cin/cout ?
- 作为字符串读入还是作为整数读入?

例题 1047

- 1047 编程团体赛
- 编程团体赛的规则为：每个参赛队由若干队员组成；所有队员独立比赛；参赛队的成绩为所有队员的成绩和；成绩最高的队获胜。
- 现给定所有队员的比赛成绩，请你编写程序找出冠军队。
- 输入格式:
- 输入第一行给出一个正整数 N （<=10000），即所有参赛队员总数。随后 N 行，每行给出一位队员的成绩，格式为：“队伍编号 - 队员编号 成绩”，其中“队伍编号”为 1 到 1000 的正整数，“队员编号”为 1 到 10 的正整数，“成绩”为 0 到 100 的整数。
- 输出格式:
- 在一行中输出冠军队的编号和总成绩，其间以一个空格分隔。注意：题目保证冠军队是唯一的。

例题 1047

- 输入样例:
- 6
- 3-10 99
- 11-5 87
- 102-1 0
- 102-3 100
- 11-9 89
- 3-2 61

- 输出样例:
- 11 176

例题 1047

- 输入样例:
- 6
- 3-10 99
- 11-5 87
- 102-1 0
- 102-3 100
- 11-9 89
- 3-2 61

适合用:
scanf("%d-%d%d",&a,&b,&c);
的格式化读入

例题 1047

- 完整做法：
  - 由于队伍编号小于 1000, 我们用 1000 的数组记录每个队的总得分
  - 对每一行读入队伍编号、队员编号、成绩
  - 将成绩加入相应队伍编号的总的分中
  - 遍历数组求最大值
  -
  - 保险起见，我们还可以用 1000 的数组记录每个队是否出现过

例题 1064

- 1064 朋友数
- 如果两个整数各位数字的和是一样的，则被称为是“朋友数”，而那个公共的和就是它们的“朋友证号”。例如 123 和 51 就是朋友数，因为 1+2+3 = 5+1 = 6，而 6 就是它们的朋友证号。给定一些整数，要求你统计一下它们中有多少个不同的朋友证号。注意：我们默认一个整数自己是自己的朋友。
- 输入格式:
- 输入第一行给出正整数 N。随后一行给出 N 个正整数，数字间以空格分隔。题目保证所有数字小于 104。
- 输出格式:
- 首先第一行输出给定数字中不同的朋友证号的个数；随后一行按递增顺序输出这些朋友证号，数字间隔一个空格，且行末不得有多余空格。

例题 1064

- 输入样例:
- 8
- 123 899 51 998 27 33 36 12
- 输出样例:
- 4
- 3 6 9 26
- 适合作为字符串读入!

例题 1064

- 完整做法
  - 将数以字符串的形式读入，对字符串中每一个字符的数字值求和
  - 由于数都小于 1e4 ，所以和的范围不超过 4*9, 我们可以开一个数组记录某一个朋友证号是否出现过
  - 遍历并按要求输出（行末无空格）

例題 1064

```
for(int i = 0 ; i < 40 ; ++i ){
    if(vis[i]){
        if(cnt > 0)printf(" ");
        printf("%d",i);
        cnt++;
    }
}
printf("\n");
```

字符串的简单操作

- char[]:
  - strlen(s)
  - 下标 0 ~ strlen(s)-1
  - strcmp(s,t) → - / 0 / + →s<t / s==t / s>t
  - strcpy(s,t)

字符串的简单操作

- string:
  - s.length()
  - 下标 0 ~ s.length()-1
  - s<t / s==t / s>t
  - s=t
  - 操作简单方便但运行较慢

一些数学函数

- #include <math.h>
-
- pow(a,n)
- sqrt(a)
- cos(a)
- sin(a)
-
- 例如： 1063、 1051

数位与数制

- 拆出一个数的每一个数位?

```
while(n){
    tmp = n % 10;    // 每次循环中的 tmp 即为最后一位
    n /= 10;
}
```

数位与数制

- 拆出一个数的每一个数位?

```
while(n){
    tmp = n % 10;    // 每次循环中的 tmp 即为最后一位
    n /= 10;
}
```

- 拆出一个数的每一个二进制位?

```
while(n){
    tmp = n % 2; // 每次循环中的 tmp 即为最后二进制位
    n >>= 1;
}
```

数位与数制

- 字符串转换为数 / 通过每一位的数字生成数?

```
for( int i = 0 ; s[i] ; ++i ){
    n = n * 10 + s[i] - ‘0’;
}
```

数位与数制

- 字符串转换为数 / 通过每一位的数字生成数?

```
for( int i = 0 ; s[i] ; ++i ){
    n = n * 10 + s[i] - ‘0’;
}
```

- 十进制与其他进制之间一般可以这样类似的进行转换

例题 1044

- 1044 火星数字
- 火星人是以 13 进制计数的：
- 地球人的 0 被火星人称为 tret。
- 地球人数字 1 到 12 的火星文分别为： jan, feb, mar, apr, may, jun, jly, aug, sep, oct, nov, dec。
- 火星人将进位以后的 12 个高位数字分别称为： tam, hel, maa, huh, tou, kes, hei, elo, syy, lok, mer, jou。
- 例如地球人的数字“29”翻译成火星文就是“hel mar”；而火星文“elo nov”对应地球数字“115”。为了方便交流，请你编写程序实现地球和火星数字之间的互译。
- 输入格式：输入第一行给出一个正整数 N （<100），随后 N 行，每行给出一个 [0, 169) 区间内的数字 —— 或者是地球文，或者是火星文。
- 输出格式：对应输入的每一行，在一行中输出翻译后的另一种语言的数字。

例题 1044

- 输入样例:
- 4
- 29
- 5
- elo nov
- tam
- 输出样例:
- hel mar
- may
- 115
- 13

例题 1044

- 输入样例:
- 4
- 29
- 5
- elo nov
- tam
- 输出样例:
- hel mar
- may
- 115
- 13

- 十进制和十三进制之间的转换

例题 1044

- 完整做法：
  - 以字符串行读入，先判断是火星文还是地球文
  - 进行进制转换

例题 1044

- 完整做法：
  - 以字符串行读入，先判断是火星文还是地球文
  - 进行进制转换
- 各种坑点！
  - 如果从第二行开始进行行读入（ gets ），那么第一行的行末回车可能会导致读入有问题！
  - 判断火星文是几位数?
  - 火星文中，低位是 0 但高位有值时，不显示 0, 只有高位串！

例题 1044

```cpp
#include <stdio.h>
#include <string.h>
#include <string>
using namespace std;

char low[13][5] = {"tret", "jan", "feb", "mar", "apr", "may", "jun"
, "jly", "aug", "sep", "oct", "nov", "dec"};
char high[13][5] = {"", "tam", "hel", "maa", "huh", "tou", "kes", "
hei", "elo", "syy", "lok", "mer", "jou"};
```

```
gets(s);
int num = 0;
if(s[0] >= '0' && s[0] <= '9'){
    for(int i = 0 ; s[i] ; ++i){
        num = num * 10 + s[i] - '0';
    }
    if(num <= 12)printf("%s\n", low[num]);
    else if(num%13)printf("%s %s\n", high[num/13], low[num%13]);
    else printf("%s\n",high[num/13]);
}
else{
    string t(s);
    int pos = t.find(' ');
    if(pos == -1){
        for(int i = 0 ; i < 13 ; ++i){
            if(t == string(low[i]))num += i;
            if(t == string(high[i]))num += i * 13;
        }
    }
    else{
        string h = t.substr(0,pos);
        string l = t.substr(pos+1);
        for(int i = 0 ; i < 13 ; ++i){
            if(l == string(low[i]))num += i;
            if(h == string(high[i]))num += i * 13;
        }
    }
    printf("%d\n",num);
}
```

数组中的最值

- 求数组中的最大值：

```
int Max = -1;  // 初始化为极小值，极大值 0x3f3f3f3f
for( int i = 1 ; i <= n ; ++i ){
    if( a[i] > Max ) Max = a[i] ;
}
```

数组中的最值

- 求数组中的最大值：

```
int Max = -1;   // 初始化为极小值
for( int i = 1 ; i <= n ; ++i ){
    if( a[i] > Max ) Max = a[i] ;
}
```

- 那么数组的前两大值呢?

数组中的最值

- 求数组中的前两大值
- 同时维护最大值和次大值

```
int Max_1 = -1, Max_2 = -1;
for( int i = 1 ; i <= n ; ++i ){
    if( a[i] > Max_1 ){
        Max_2 = Max_1;
        Max_1 = a[i];
    }
    else if( a[i] > Max_2 ) Max_2 = a[i];
}
```

vis/count 数组的使用

- 记录一个数是否出现过，出现过几次
- 记录一个数是否有信息
- 通过下标从小到大访问，天然有序
- 要求：数的范围不大，可以开下那么大的数组

例题 1059

- 1059 C 语言竞赛
- C 语言竞赛是浙江大学计算机学院主持的一个欢乐的竞赛。既然竞赛主旨是为了好玩，颁奖规则也就制定得很滑稽：
- 0. 冠军将赢得一份“神秘大奖”（比如很巨大的一本学生研究论文集……）。
- 1. 排名为素数的学生将赢得最好的奖品 —— 小黄人玩偶！
- 2. 其他人将得到巧克力。
- 给定比赛的最终排名以及一系列参赛者的 ID，你要给出这些参赛者应该获得的奖品。
- 输入格式：输入第一行给出一个正整数 N（<=10000），是参赛者人数。随后 N 行给出最终排名，每行按排名顺序给出一位参赛者的 ID（4 位数字组成）。接下来给出一个正整数 K 以及 K 个需要查询的 ID。
- 输出格式：对每个要查询的 ID，在一行中输出“ID: 奖品”，其中奖品或者是“Mystery Award”（神秘大奖）、或者是“Minion”（小黄人）、或者是“Chocolate”（巧克力）。如果所查 ID 根本不在排名里，打印“Are you kidding?”（耍我呢？）。如果该 ID 已经查过了（即奖品已经领过了），打印“ID: Checked”（不能多吃多占）。

例题 1059

- 输入样例：
- 6
- 1111
- 6666
- 8888
- 1234
- 5555
- 0001
- 6
- 8888
- 0001
- 1111
- 2222
- 8888
- 2222

- 输出样例：
- 8888: Minion
- 0001: Chocolate
- 1111: Mystery Award
- 2222: Are you kidding?
- 8888: Checked
- 2222: Are you kidding?

例题 1059

由于编号范围不大，可以直接将排名保存在数组下标为该编号的元素中，查询时就能根据那个元素是否有排名来确定输出了

- 输入样例：
- 6
- 1111
- 6666
- 8888
- 1234
- 5555
- 0001
- 6
- 8888
- 0001
- 1111
- 2222
- 8888
- 2222

- 输出样例：
- 8888: Minion
- 0001: Chocolate
- 1111: Mystery Award
- 2222: Are you kidding?
- 8888: Checked
- 2222: Are you kidding?

例题 1059

- 完整做法：
  - 将编号当成整数，依次读入，并用一个数组存储每个编号对应的名次
  - 对于每个查询，查看该编号是否有名次，若有名次，进行相关处理，输出，并标记为已处理过的，若无名次，进行输出
    - 名次为 1, 直接输出
    - 判断名次是否为素数，再输出
    - 输出后，可以将名次标为 -1, 以便下次直接输出

例题 1059

- 判断素数?
  - O(√n) 判断：
    - 对于根号 n 以内的数全部判断是否为 n 的因数
    - 无因数则为素数
  - 素数筛 O(nlogn)：
    - 初始化 2 为素数
    - 对于所有遍历到的素数，将它的倍数标记为非素数
    - 遍历时跳过非素数
  - 欧拉筛 O(n)：

例題 1059

```
bool isPrime(int rk){
    for(int i = 2 ; i * i <= rk ; ++i){
        if(rk % i == 0)return false;
    }
    return true;
}
```

例題 1059

```
for(int i = 1 ; i <= n ; ++i){
    int id;
    scanf("%d",&id);
    Rank[id] = i;
}
scanf("%d",&m);
while(m--){
    int id;
    scanf("%d",&id);
    printf("%04d: ",id);
    if(Rank[id] == 0)printf("Are you kidding?\n");
    else if(Rank[id] == -1)printf("Checked\n");
    else{
        if(Rank[id] == 1)printf("Mystery Award\n");
        else if(isPrime(Rank[id]) == true)printf("Minion\n");
        else printf("Chocolate\n");
        Rank[id] = -1;
    }
}
```

最大公因数 gcd

- 两个数的最大公因数 gcd

最大公因数 gcd

- 两个数的最大公因数 gcd
- 实现：辗转相除法

最大公因数 gcd

- 两个数的最大公因数 gcd
- 实现：辗转相除法：
  - gcd(a, b) = gcd(b , a%b)

最大公因数 gcd

- 两个数的最大公因数 gcd
- 实现：辗转相除法：
  - gcd(a, b) = gcd(b, a%b)

```
int gcd( int a, int b){
    return  b ? gcd( b, a % b ) : a;
}
```

最大公因数 gcd

- 两个数的最大公因数 gcd
- 实现：辗转相除法：
  - gcd(a, b) = gcd(b, a%b)

```
int gcd( int a, int b){
    return  b ? gcd( b, a % b ) : a;
}
```

- 库函数：　__gcd(a, b)

最大公因数 gcd

- 两个数的最大公因数 gcd
- 实现：辗转相除法：
  - gcd(a, b) = gcd(b, a%b)

```
int gcd( int a, int b){
    return  b ? gcd( b, a % b ) : a;
}
```

- 库函数： __gcd(a, b)
- 例题： 1062

1070 结绳

- 思路是什么？为什么？

1070 结绳

```
#include <stdio.h>
#include <algorithm>
using namespace std;

const int maxn = 1e4 + 5;

int len[maxn];

int main(){
    int n;
    scanf("%d",&n);
    for(int i = 1 ; i <= n ; ++i)scanf("%d",&len[i]);
    sort(len + 1, len + n + 1);
    double ans = len[1];
    for(int i = 2 ; i <= n ; ++i){
        ans = ( ans + len[i] ) / 2.0;
    }
    printf("%d\n",(int)ans);
    return 0;
}
```

谢谢！