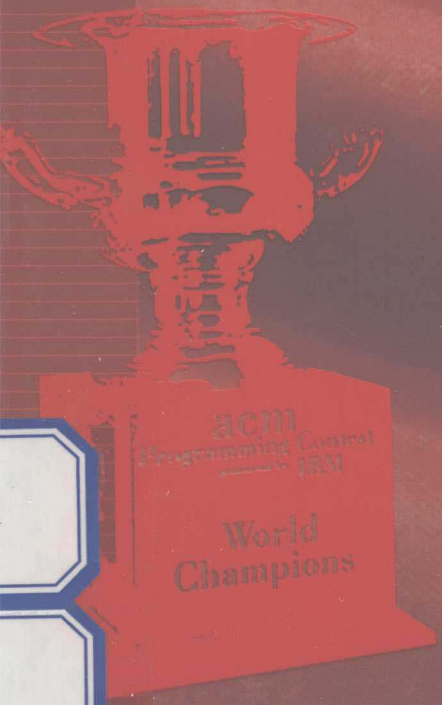
谨以此书献给上海交通大学获得ACM-ICPC世界冠军十周年

ACM国际大学生程序设计竞赛（ACM-ICPC）系列丛书

ACM国际大学生程序设计竞赛
算法与实现

俞　勇　主编



清华大学出版社

ACM国际大学生程序设计竞赛（ACM-ICPC）系列丛书

ACM国际大学生程序设计竞赛
算法与实现

俞　勇　主编

清华大学出版社
北京

内 容 简 介

ACM 国际大学生程序设计竞赛（ACM-ICPC）是国际上公认的水平最高、规模最大、影响最深的计算机专业竞赛，目前全球参与人数达 20 多万。本书作者将 16 年的教练经验与积累撰写成本系列丛书，全面、深入而系统地将 ACM-ICPC 展现给读者。本系列丛书包括《ACM 国际大学生程序设计竞赛：知识与入门》、《ACM 国际大学生程序设计竞赛：算法与实现》、《ACM 国际大学生程序设计竞赛：题目与解读》、《ACM 国际大学生程序设计竞赛：比赛与思考》等 4 册，其中《ACM 国际大学生程序设计竞赛：知识与入门》介绍了 ACM-ICPC 的知识及其分类、进阶与角色、在线评测系统；《ACM 国际大学生程序设计竞赛：算法与实现》介绍了 ACM-ICPC 算法分类、实现及索引；《ACM 国际大学生程序设计竞赛：题目与解读》为各类算法配备经典例题及题库，并提供解题思路；《ACM 国际大学生程序设计竞赛：比赛与思考》介绍了上海交通大学 ACM-ICPC 的训练及比赛，包括训练札记、赛场风云、赛季纵横、冠军之路、峥嵘岁月。

本丛书适用于参加 ACM 国际大学生程序设计竞赛的本科生和研究生，对参加青少年信息学奥林匹克竞赛的中学生也很有指导价值。同时，作为程序设计、数据结构、算法等相关课程的拓展与提升，本丛书也是难得的教学辅助读物。

图书在版编目（CIP）数据

ACM 国际大学生程序设计竞赛：算法与实现/俞勇主编. —北京：清华大学出版社，2013.1
ACM 国际大学生程序设计竞赛（ACM-ICPC）系列丛书
ISBN 978-7-302-29413-9

Ⅰ. ①A…　Ⅱ. ①俞…　Ⅲ. ①程序设计–竞赛–高等学校–教学参考资料　Ⅳ. ①TP311.1

中国版本图书馆 CIP 数据核字（2012）第 161172 号

责任编辑： 龙启铭
封面设计： 傅瑞学
责任校对： 李建庄
责任印制： 何　芊

出版发行： 清华大学出版社
网　　址：http://www.tup.com.cn, http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮编：100084
社 总 机：010-62770175　　邮购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课 件 下 载：http://www.tup.com.cn,010-62795954

印 装 者： 北京国马印刷厂
经　　销： 全国新华书店
开　　本： 185mm×230mm　　印　张：17.75　　字　数：445 千字
版　　次： 2013 年 1 月第 1 版　　印　次：2013 年 1 月第 1 次印刷
印　　数： 1～3000
定　　价： 36.00 元

产品编号：047607-01

写在最前面的话

自从上海交通大学2002年第一次、2005年第二次获得ACM国际大学生程序设计竞赛（ACM International Collegiate Programming Contest，简称ACM-ICPC或ICPC）世界冠军以来，总有记者邀请编者撰写冠军之路类的文章，也总有出版社希望编者出版ACM-ICPC竞赛类的书籍，因为没有想清楚怎么写，所以一直没动笔。直到2010年上海交通大学第三次获得ACM-ICPC世界冠军后，编者决定出版一套系列丛书，包括《ACM国际大学生程序设计竞赛：知识与入门》、《ACM国际大学生程序设计竞赛：算法与实现》、《ACM国际大学生程序设计竞赛：题目与解读》及《ACM国际大学生程序设计竞赛：比赛与思考》4册书籍，全面、深入而系统地将ACM-ICPC展现给读者，把上海交通大学十多年来对ACM-ICPC竞赛的感悟分享给读者。

编写此系列丛书的另一个重要原因是ACM-ICPC竞赛在中国大陆的迅猛发展。自从1996年ACM-ICPC引入中国大陆，前六届仅设立1个赛区，目前每年一般设立5个赛区，并已有30所高校承办过亚洲区预赛；参赛学校从不满20所，到如今已达200多所；参赛人数从不到100人，到如今超过12万人次；总决赛名额从起初的3个，到如今已超过15个。同时，中国大陆在ACM-ICPC竞赛上所取得的成绩也举世瞩目。清华大学9次获得总决赛奖牌（3金5银1铜），位居奖牌榜之首，是实力最强、表现最稳定的高校；上海交通大学8次获得总决赛奖牌（4金3银1铜），3次夺得世界冠军，算是目前国内成绩最好的高校；中山大学4次获得总决赛奖牌（2银2铜），在生源不占优势的情况下，这一成绩令人敬佩；复旦大学3次获得总决赛奖牌（1银2铜），是公认的强校；浙江大学2次获得总决赛奖牌（1金1银），1次夺得世界冠军，再次让国人欢欣鼓舞；北京大学1次获得总决赛奖牌（1铜），队员的综合实力堪称一流；最难能可贵的是，华南理工大学也获得过总决赛的奖牌（1铜），它告诉我们，ACM-ICPC不仅仅是“强校”之间的“对话”，只要坚持参与就会斩获成果。另外，至今已有37所大陆高校参加过全球总决赛，且不论成绩如何，他们在赛场上的奋斗亦值得称道。

本系列丛书的第一册《ACM 国际大学生程序设计竞赛：知识与入门》分为三个部分。知识点部分基本涵盖了竞赛中所涉及的主要知识点，包括数学基础、数据结构、图论、计算几何、论题选编、求解策略等六个大类内容。入门与进阶部分介绍了包括如何快速入门、如何提高自身以及团队水平等，主要根据上海交通大学 ACM-ICPC 队多年参赛经验总结而来。在线资源部分对一些常用的在线评测系统和网上比赛进行了介绍。

本系列丛书的第二册《ACM 国际大学生程序设计竞赛：算法与实现》涵盖了大部分 ACM-ICPC 竞赛常用的经典算法，包括数学、图论、数据结构、计算几何、论题选编五个大类，对每个算法的代码实现，都配有接口说明以及简略的算法阐述，并提供算法的完整程序，贴士部分收集了一些实用的知识点及积分表，方便读者查找使用。

本系列丛书的第三册《ACM 国际大学生程序设计竞赛：题目与解读》分为两个部分。例题精讲部分针对第二册《ACM 国际大学生程序设计竞赛：算法与实现》中的算法配备经典例题，并提供细致的解题思路，读者可以通过这一部分学习和掌握算法；海量题库部分按照算法分类罗列出大量习题，并提供相应的题解，读者可以利用这一部分的题目进行训练，更加熟练地运用各类算法。

本系列丛书的第四册《ACM 国际大学生程序设计竞赛：比赛与思考》从 120 多名队员、2400 余篇文档中精心挑选、编纂而成的文集，包括训练札记、赛场风云、赛季纵横、冠军之路、峥嵘岁月，集中展现了上海交通大学 ACM-ICPC 队 16 年的奋斗历程，记载了这些队员为了实现自己的梦想而不懈努力、勇于拼搏的故事。

这是一套全面、系统地学习 ACM-ICPC 竞赛的知识类书籍；

这是一套详尽、深入地熟悉 ACM-ICPC 竞赛的算法及题目的手册类书籍；

这是一套程序设计、数据结构、算法等相关课程的拓展与提升类书籍；

这是一部上海交通大学 ACM-ICPC 队的成长史；

这是一部激励更多学子勇敢追寻并实现自己最初梦想的励志书。

历时 2 年零 5 个月，终于完成了本系列丛书，编者与队员有一种如释重负的感觉，因为我们把出版这套丛书看得很重，这是我们 16 年的经验与积累，希望对广大读者有用。

值此 ACM-ICPC 进入中国大陆 16 周年、上海交通大学获得 ACM-ICPC 世界冠军 10 周年之际，谨以此系列丛书——

纪念我们曾经走过的路、度过的岁月；

献给所有支持、帮助过我们的人……

俞　勇

2012 年 10 月于上海

前 言

在 ACM 国际大学生程序设计竞赛（ACM International Collegiate Programming Contest，ACM-ICPC 或 ICPC）中，实现算法的能力是非常重要的。尤其是对新手来说，在了解到一个新的算法后，有时会对如何实现该算法产生困惑，也许并不能一下想到很好的实现，这时就需要参考一些已有的实现。

另外，ACM-ICPC 比赛中允许选手将一定量的（一般为 25 页）纸质资料带入比赛现场进行参考。队伍往往会将一些相对较难实现的常用算法的代码整理为 SCL（Standard Code Library，标准代码库）带入赛场，在需要的时候可以直接抄写已有代码，既节省时间，也保证了正确性。

因此我们出版这本收集了大量经典常用算法的用 C++语言实现的代码库，希望可以帮助读者学习算法以及准备比赛用的 SCL。

本书分为两个部分。第一部分为代码库，涵盖了大部分比赛常用的经典算法，包括数学、图论、数据结构、计算几何、论题选编五个大类，对每个算法的代码实现，都配有接口说明以及简略的算法阐述，便于读者理解。第二部分为贴士，收集了一些实用的知识点以及积分表，适合于带入赛场进行参考。

本书编写工作历时两年左右，参与编写工作的人员全部为上海交通大学 ACM-ICPC 队的现役队员。代码大多来自于往年上海交通大学 ACM-ICPC 队使用的 SCL 以及队员的日常训练。同时，本书的编写也得到上海交通大学 ACM-ICPC 队的退役队员大力帮助，他们参与了代码库的收集、整理、校验等工作。

参与本书写稿、审稿的人员主要有（按姓氏笔画为序）：尹天蛟，乌辰洋，任春旭，刘奇，刘彦，寿鹤鸣，李说，杨思逸，吴卓杰，张捷钧，陈明骋，陈泽佳，陈彬毅，陈爽，林承宇，金斌，郑翌，胡张广达，郭晓旭，曹雪智，康南茜，章雍哲，商静波，彭上夫，谭天，缪沛晗，瞿钧等。

在此，衷心感谢所有为此书出版做出直接或间接贡献的人！也真心祝愿此书能够在算法实现和 SCL 的准备上给读者带来帮助。

由于时间仓促，作者水平有限，疏漏、不当和不足之处在所难免，真诚地希望专家和读者朋友们不吝赐教。如果您能在阅读和使用此书过程中发现任何问题或有任何建议，恳请发邮件至 yyu@cs.stu.edu.cn，我们将不胜感激。

编　者

2012 年 10 月于上海

目 录

第一部分 算 法

第一部分

算　　法

第1章 数 学

1.1 矩 阵

1.1.1 矩阵类

【任务】

实现矩阵的基本变换。

【接口】

结构体：*Matrix*

成员变量：

int *n*,*m*	矩阵大小
int *a*[][]	矩阵内容

重载运算符：+、－、×

成员函数：

void clear()	清空矩阵

【代码】

```
const int MAXN=1010;
const int MAXM=1010;
struct Matrix{
    int n,m;
    int a[MAXN][MAXM];
    void clear(){
        n=m=0;
        memset(a,0,sizeof(a));
    }
    Matrix operator +(const Matrix &b) const{
        Matrix tmp;
        tmp.n=n; tmp.m=m;
```

```
        for (int i=0; i<n; ++i)
            for (int j=0; j<m; ++j)
                tmp.a[i][j]=a[i][j]+b.a[i][j];
        return tmp;
    }
    Matrix operator -(const Matrix &b) const{
        Matrix tmp;
        tmp.n=n; tmp.m=m;
        for (int i=0; i<n; ++i)
            for (int j=0; j<m; ++j)
                tmp.a[i][j]=a[i][j]-b.a[i][j];
        return tmp;
    }
    Matrix operator *(const Matrix &b) const{
        Matrix tmp;
        tmp.clear();
        tmp.n=n; tmp.m=b.m;
        for (int i=0; i<n; ++i)
            for (int j=0; j<b.m; ++j)
                for (int k=0; k<m; ++k)
                    tmp.a[i][j]+=a[i][k]*b.a[k][j];
        return tmp;
    }
};
```

【使用范例】

参见程序 POJ3420.CPP。

1.1.2 Gauss 消元

【任务】

给一个n元一次方程组，求它们的解集。

【说明】

将方程组做成矩阵形式，再利用三种初等矩阵变换，得到上三角矩阵，最后回代得到解集。

【接口】

int solve(double a[ ][MAXN], bool l[ ], double ans[ ], const int& n);

复杂度：$O(n^3)$

输　入：a　　方程组对应的矩阵

　　n　　未知数个数

　　l, ans　　存储解，$l[]$表示是否为自由元

输　出：解空间的维数

【代码】

```
inline int solve(double a[][MAXN], bool l[], double ans[],
const int& n) {
    int res = 0, r = 0;
    for (int i = 0; i < n; ++i)
        l[i] = false;
    for (int i = 0; i < n; ++i) {
        for (int j = r; j < n; ++j)
            if (fabs(a[j][i]) > EPS) {
                for (int k = i; k <= n; ++k)
                    swap(a[j][k], a[r][k]);
                break;
            }
        if (fabs(a[r][i]) < EPS) {
            ++res;
            continue;
        }
        for (int j = 0; j < n; ++j)
            if (j != r && fabs(a[j][i]) > EPS) {
                double tmp = a[j][i] / a[r][i];
                for (int k = i; k <= n; ++k)
                    a[j][k] -= tmp * a[r][k];
            }
        l[i] = true, ++r;
    }
    for (int i = 0; i < n; ++i)
        if (l[i])
            for (int j = 0; j < n; ++j)
                if (fabs(a[j][i]) > 0)
                    ans[i] = a[j][n] / a[j][i];
    return res;
}
```

【使用范例】

参见程序 POJ1830.CPP。

1.1.3 矩阵的逆

【任务】

给一个矩阵，求它的逆。

【说明】

将原矩阵A和一个单位矩阵E作成大矩阵(A,E)，用初等行变换将大矩阵中的A变为E，则会得到(E,A^{-1})的形式。

【接口】

void inverse(vector<double> A[ ], vector<double> C[ ], int N);

复杂度：$O(n^3)$

输　入：A　原矩阵

　　C　逆矩阵

　　N　矩阵的阶数

【代码】

```
1   inline vector<double> operator * (vector<double> a, double b) {
2       int N = a.size();
3       vector<double> res(N, 0);
4       for (int i = 0; i < N; ++i)
5           res[i] = a[i] * b;
6       return res;
7   }
8   inline vector<double> operator - (vector<double> a, vector<double> b){
9       int N = a.size();
10      vector<double> res(N, 0);
11      for (int i = 0; i < N; ++i)
12          res[i] = a[i] - b[i];
13      return res;
14  }
15  inline void inverse(vector<double> A[], vector<double> C[], int N) {
16      for (int i = 0; i < N; ++i)
17          C[i] = vector<double>(N, 0);
```

```
    for (int i = 0; i < N; ++i)
        C[i][i] = 1;
    for (int i = 0; i < N; ++i) {
        for (int j = i; j < N; ++j)
            if (fabs(A[j][i]) > 0) {
                swap(A[i], A[j]);
                swap(C[i], C[j]);
                break;
            }
        C[i] = C[i] * (1 / A[i][i]);
        A[i] = A[i] * (1 / A[i][i]);
        for (int j = 0; j < N; ++j)
            if (j != i && fabs(A[j][i] > 0)) {
                C[j] = C[j] - C[i] * A[j][i];
                A[j] = A[j] - A[i] * A[j][i];
            }
    }
```

【使用范例】

参见程序 POJ1166.CPP。

1.1.4　常系数线性齐次递推

【任务】

已知$f_x = a_0 f_{x-1} + a_1 f_{x-2} + \cdots + a_{n-1} f_{x-n}$和$f_0, f_1, \cdots, f_{n-1}$，给定$t$，求$f_t$。

【说明】

f的递推可以看成一个$n \times n$的矩阵$\boldsymbol{A}$乘以一个n维列向量$\boldsymbol{\beta}$，因为矩阵乘法满足结合律，用快速幂可以加速。其中，

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & & 0 \\ & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ a_{n-1} & a_{n-2} & a_{n-3} & & a_0 \end{bmatrix}, \quad \boldsymbol{\beta} = \begin{bmatrix} f_{x-n} \\ f_{x-n+1} \\ \vdots \\ f_{x-2} \\ f_{x-1} \end{bmatrix}$$

【接口】

int solve(int a[ ], int b[ ], int n, int t);

复杂度：$O(n^3 \log t)$