全国高等院校计算机基础教育研究会

“计算机系统能力培养教学研究与改革课题”立项项目

ACM大学生程序设计竞赛

在线题库精选题解

——算法分析与设计习题解答

赵端阳　吴艳　石洗凡◎主编

题目精选 | 在浙江大学ZOJ、杭州电子科技大学HOJ中选择经典题目

算法经典 | 掌握常用的动态规划、贪心、图论、几何和数学算法

分析简洁 | 语言通俗、思路清晰、分析透彻、举一反三

北京邮电大学出版社
www.buptpress.com

ACM 大学生程序设计竞赛在线题库精选题解

——算法分析与设计习题解答

主　编　赵端阳　吴　艳　石洗凡

北京邮电大学出版社
www. buptpress. com

内 容 简 介

随着各大专院校参加 ACM/ICPC 热情的高涨，迫切需要有关介绍 ACM 国际大学生程序设计竞赛题解的书籍。本书是在浙江大学、北京大学和杭州电子科技大学的在线题库中精选了部分题目进行分析和解答，比较详细地分析和深入浅出地讲解了解题的方法和用到的算法。精选的题目算法特征明显、具有代表性，题目类型包括基础编程与技巧、模拟算法、字符串处理、大整数运算、数据结构、搜索算法、动态规划算法、贪心算法、回溯算法、图论算法、几何和数学题。

本书可以作为高等院校有关专业的本科和大专学生参加国际大学生程序设计竞赛的辅导教材，或者作为高等院校数据结构、C/C++程序设计或算法设计与分析等相关课程的教学参考书。

本书的全部源代码，都可以在北京邮电大学出版社网站下载。

图书在版编目(CIP)数据

ACM 大学生程序设计竞赛在线题库精选题解 / 赵端阳，吴艳，石洗凡主编．-- 北京 ：北京邮电大学出版社，2016.2

算法分析与设计实验教材

ISBN 978-7-5635-4673-2

Ⅰ．①A… Ⅱ．①赵… ②吴… ③石… Ⅲ．①程序设计－竞赛－高等学校－题解 Ⅳ．①TP311.1－44

中国版本图书馆 CIP 数据核字（2016）第 022315 号

书　　名：ACM 大学生程序设计竞赛在线题库精选题解
主　　编：赵端阳　吴　艳　石洗凡
责任编辑：王丹丹　刘　佳
出版发行：北京邮电大学出版社
社　　址：北京市海淀区西土城路 10 号（邮编：100876）
发 行 部：电话：010-62282185　传真：010-62283578
E-mail：publish@bupt.edu.cn
经　　销：各地新华书店
印　　刷：北京通州皇家印刷厂
开　　本：787 mm×1 092 mm　1/16
印　　张：20
字　　数：522 千字
印　　数：1—2 000 册
版　　次：2016 年 2 月第 1 版　2016 年 2 月第 1 次印刷

ISBN 978-7-5635-4673-2　　定　价：40.00 元

前　言

ACM 国际大学生程序设计竞赛（ACM/ICPC：ACM International Collegiate Programming Contest）是由国际计算机界历史悠久、颇具权威性的组织 ACM 学会（Association for Computing Machinery，美国计算机协会）主办，是世界上公认的规模最大、水平最高的国际大学生程序设计竞赛，其目的旨在使大学生运用计算机来充分展示自己分析问题和解决问题的能力。1970 年在美国 Texas A&M 大学举办了首次区域竞赛，从而拉开了国际大学生程序设计竞赛的序幕。该项竞赛从 1970 年举办至今已历 39 届，因历届竞赛都荟萃了世界各大洲的精英，云集了计算机界的“希望之星”，而受到国际各知名大学的重视，并受到全世界各著名计算机公司的高度关注，成为世界各国大学生最具影响力的国际级计算机类的赛事。

此项赛事的主办目的不单是培养参赛选手的创造力，团队合作精神以及他们在软件程序开发过程中的创新意识，同时也是检测选手们在压力下进行开发活动的能力。因此，ACM 国际大学生程序设计竞赛是参赛选手展示计算机才华的广阔舞台，是著名大学计算机教育成果的直接体现，是信息企业与世界顶尖计算机人才对话的最好机会。

ACM 国际大学生程序设计竞赛 1996 年才进入中国内地，上海交通大学作为我国内地高校最早的参赛队之一，曾 7 次进军总决赛，并于 2002 年将 ACM 金杯首次带到亚洲，打破了几十年来欧美国家对这一赛事绝对的统治地位，更震惊了世界。

2005 年 4 月 6 日，在上海举办的“第 29 届 ACM 国际大学生程序设计竞赛”总决赛中，上海交通大学团队成功解答 8 道题，以一题领先的优势力克来自世界六大洲 29 个国家和地区的 78 支参赛队，捧获“世界上最聪明的人”的冠军奖杯。时隔 3 年，ACM 全球总决赛冠军奖杯再次回到了上海交通大学。

国际 ACM 比赛是世界上规模最大、历史最长、影响最深的全球性计算机专业竞赛，它要求每一名队员不仅具有扎实的数学功底、非凡的算法设计能力、娴熟的编程技巧，而且必须具备很好的协作精神、稳定的心理素质、快速的临场应变能力。

为了帮助各个大专院校的大学生们了解国际大学生程序设计竞赛，了解其程序设计的方法，提高参与校级、省级和亚洲赛区国际大学生程序设计竞赛的兴趣，特编写这本题解。

全书共分 12 章：

第 1 章，基础编程与技巧题。主要是比较容易的题目，也称简单题，尤其适合刚刚开始熟悉 ACM 大学生程序设计竞赛的同学。通过这些题目的练习，主要是熟悉数据的输入、输出格式，基本编程方法，在线提交系统的使用，常见错误及其对策。

第 2 章，模拟算法题。根据题目的要求逐步实现目标，虽然没有经典的算法可以使用，正确理解题目是关键的要素。例如题目 Die and Chessboard 和 Counterfeit Dollar 等。

第 3 章，字符串处理题。主要是字符串的处理，这也是考察参赛者细心的一类题目，

需要对各种情况进行仔细的分析，例如题目 String Matching 等。

第 4 章，大整数运算题，例如 Martian Addition 等。

第 5 章，基本数据结构题，例如 Trees on the Level 等。

第 6 章，搜索算法题。通常有深度优先搜索和广度优先搜索算法，例如 The Same Game 等。

第 7 章，动态规划算法题。这些题目使用动态规划算法，会获得较快的运行时间和效率，例如题目 The Staircases 和 Fast Food 等。

第 8 章，贪心算法题。这些题目使用贪心算法，例如题目 Gone Fishing 等。

第 9 章，回溯算法题。这些题目使用回溯算法，例如题目 Stockbroker Grapevine 和 Ouroboros Snake 等。

第 10 章，图论题。主要包含拓朴排序、Floyd 算法和二分图等，例如 Street Directions 等。

第 11 章，几何题。主要是几何计算题目，例如题目 Laser Lines 等。

第 12 章，数学题。主要是数学计算题目，例如题目 Self Numbers 等。

本部分通过大量的实例介绍了竞赛中常用的算法，并对如何灵活地应用这些算法进行了比较详细的分析和深入浅出的讲解。

本书所用的语言是 C/C++，并在 MinGW Developer Studio 中调试通过。在运行时间和内存占用的性能方面，C++并不比 C 语言优越多少，只是 C++有丰富的模板库函数，输入/输出语句简单，编写的代码看起来格式更工整而已。在有大量输入/输出数据的题目中，书中会特别提醒读者需要使用 C 语言的输入/输出。

显然，“Accepted”是我们的目标，算法是我们不断探索的道路，好的算法让我们容易达到目标。本书中的算法，虽然作者经过反复斟酌，不断优化和简化，但仍然不敢认为是最优的。因为对算法的探索，正如金庸笔下的大侠们苦练武功一样，是没有止境的。

本书中虽然描述的是 ACM 国际大学生程序设计竞赛中最基本的算法，但它也已经超出了一般本科教科书所讲授的范围。清华大学计算机科学与技术系博士生导师、国际信息学奥林匹克中国队总教练吴文虎教授认为算法的确是艺术，“艺术与科学是相通的，都会给人以美的享受。”① 希望读者能从本书中体会到这一点。

本书可以作为高等院校有关专业的本科和大专学生参加国际大学生程序设计竞赛的辅导教材，或者作为高等院校数据结构、C/C++程序设计或算法设计与分析等相关课程的教学参考书，旨在培养和提高学生参加 ACM 国际大学生程序设计竞赛的兴趣。

本书获得全国高等院校计算机基础教育研究会计算机系统能力培养教学研究与改革课题立项的大力支持，并得到北京邮电大学出版社王丹丹老师的大力支持，在此表示非常感谢。

由于作者水平所限，书中难免有不足之处，恳请广大读者批评指正。作者电子邮箱地址：727946579@qq.com。

作者

2015 年 9 月于杭州

① 吴文虎主编，孙贺编著，程序设计中的组合数学，清华大学出版社，2005 年 5 月

目　录

第一章　基础编程与技巧题

在本章的题目大部分都比较简单，作为刚刚开始参加竞赛的练习题。

1. ZJU1164-Software CRC[①②]

【题目大意】

你在一家拥有很多个人计算机的公司上班。你的老板，Penny Pincher 博士，想要把这些个人计算机联网，但是他又不想花钱买网卡。你无意中告诉了老板每台计算机出厂时就带有一个异步串行端口，老板当然把脑筋动到这个不用花钱的解决方案上。于是指派你完成编写通信软件的任务，以实现计算机之间联网。

你阅读了许多关于通信的书籍，知道在传送及接收信息时容易产生错误。解决问题的典型的方法是，在信息的末尾附加错误校验信息。该信息允许接收程序校验传送的信息是否有错误产生(在大多数情况下)。于是，你跑到图书馆借了一本厚厚的关于通信的书，利用周末(当然没有加班费)研究错误校验方法。

最后，你决定 CRC(Cyclic Redundancy Check)是最适合的错误校验方案，并向 Penny Pincher 博士详细描述了该错误校验方案。

将待传递的信息看作是一串很长的二进制数。信息的第一个字节(Byte)是这个二进制数的最高有效字节，第二个字节是第二个最高有效字节，依此类推。把这个二进制数称为“m”。传送信息时，会在“m”之后加上 2 个字节的 CRC 校验码，整个二进制数称为“m2”。

选择 CRC 校验码的方法：当“m2”除以某个 16 位的值“g”时，余数为 0。这样就使接收程序比较容易判断信息是否产生了传送错误。所以对接收到的任何信息除以“g”，如果余数为 0 即代表此信息正确。

注意，大部分书中都建议“g”的值为奇数，你决定用 34943 作为“g”的值。

输入

必须设计一个算法，对所有传送的信息计算 CRC 的值。为了测试算法的正确性，编写一个程序读取数据(每行都是字符，不包括行结束符)，对每一行信息，计算相应的 CRC 值，并输出该 CRC 值(十六进制)，占一行。对每行输入，不会超过 1024 个 ASCII 字符。当一行的第一列是＃时，代表输入结束。

输出

对每组测试数据输出一行，是以十六进制表示的 CRC 值。注意：CRC 的取值范围应该是 0 到 34942(十进制)之间。

---

① http://acm.zju.edu.cn/onlinejudge/showProblem.do?problemCode=1164

② http://acm.uva.es/problemset/v1/128.html

样例输入

```
this is a test

A
#
```

样例输出

```
77 FD
00 00
0C 86
```

题目来源：

Zhejiang University Local Contest 2002, *Warmup*

【算法分析】

本题是用软件的方法计算 CRC 的值，在实际使用中都是用硬件实现的，以提高传送速度。从整数的 4 个字节中，取出某个字节，可以用运算的方法，而采用数据结构共用体 union 会更方便一些。

(1) 计算一行所有的字符，除以 Generator 的余数

如果直接计算，要采用大数运算的方法。取余数就是模运算，而模运算可以逐位计算的。运算过程中，将前一次余数与当前字符并起来再求模。

(2) 计算 CRC

按题目给出的要求计算，将第一步计算的结果移到高位(记为变量 L)，后面两个字节就是存放 CRC 的。如果前面计算的结果是 0，则 CRC=0；否则，CRC=Generator−L。

现在就是要取出 CRC 并输出。采用数据结构共用体 union，如图 1-1 所示，就能直接取出所需要的字节。显然，b_2 是 CRC 的高位字节，b_1 是 CRC 的低位字节。例如：L=0x01020304；则 b_1=0x04；b_2=0x03；b_3=0x02；b_4=0x01，相应的测试代码，请参考 unionTest.cpp。

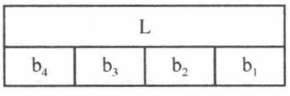

图 1-1 CRC 的数据结构

【程序代码】

```
#include <stdio.h>
#define Generator 34943

int main()
{
    unsigned char c,flag;
    int count;                          //一行中的字符数
    //CRC 的数据结构
    union
    {
        unsigned int L;
        struct
        {
            unsigned char b1;
            unsigned char b2;
            unsigned char b3;
            unsigned char b4;
        } crc;
    } data;
```

```
    while(1)
    {
        count = 0;
        //计算一行所有的字符,除以 Generator 的余数
        data.L = 0;
        while(scanf("%c",&c) && c != '\n')
        {
            count++;
            flag = c;
            //采用每个字符取模的方法
            data.L = ((data.L<<8) + c) % Generator;
        }
        //将发送信息的余数,移到整数最高和次高字节
        data.L = (data.L<<16) % Generator;
        //计算 CRC
        if(data.L != 0) data.L = Generator - data.L;
        //当一行中的第 1 个字符是'#'时,才是结束标记
        if(count == 1 && flag == '#') break;
        printf("%02X %02X\n",(int)data.crc.b2,(int)data.crc.b1);
    }
    return 0;
}
```

算法实现原代码:zju1164.c。

2. ZJU1168-Function Run Fun[①]

【题目大意】

我们都喜欢递归,不是吗?

考察一个有三个参数的递归函数 w(a,b,c):

如果 a≤0 或 b≤0 或 c≤0,则递归函数 w(a,b,c)返回 1。

如果 a>20 或 b>20 或 c>20,则递归函数 w(a,b,c)返回 w(20,20,20)。

如果 a<b 并且 b<c,则递归函数 w(a,b,c)返回 w(a,b,c−1)+w(a,b−1,c−1)−w(a,b−1,c);否则将返回 w(a−1,b,c)+w(a−1,b−1,c)+w(a−1,b,c−1)−w(a−1,b−1,c−1)。

这是一个很简单的函数。问题是,如果直接计算,当 a,b,c 取某个值时(例如 a=15,b=15,c=15),由于大量的递归,程序运行需要几小时。

输入

输入有多组测试数据,每行三个整数,结束标志是−1 −1 −1。使用上面的技术,请有效地计算函数 w(a,b,c)并输出结果。

输出

对输入的每三个数据,输出函数 w(a,b,c)的值。

样例输入

```
1 1 1
2 2 2
```

① http://acm.zju.edu.cn/onlinejudge/showProblem.do?problemCode=1168

```
10 4 6
50 50 50
-1 7 18
-1 -1 -1
```

样例输出

```
w(1,1,1) = 2
w(2,2,2) = 4
w(10,4,6) = 523
w(50,50,50) = 1048576
w(-1,7,18) = 1
```

题目来源：

Pacific Northwest 1999

【算法分析】

这是一个递归的题目。正如题目中所说，如果直接计算的话，由于大量的递归运算，运行时间会很长。由于三个整数的范围很小，就提供了采用数组保存中间结果的方法，也就是使用记忆式搜索的方法，通常称为打表计算。

【程序代码】

```
#include<stdio.h>
//存储递归的中间结果
int f[21][21][21];

//递归函数
int w(int a,int b,int c)
{
    if (a<=0 || b<=0 || c<=0) return 1;
    if (a>20 || b>20 || c>20) return w(20,20,20);
    if (f[a][b][c]>0) return f[a][b][c];
    if (a<b && b<c) return w(a,b,c-1)+w(a,b-1,c-1)-w(a,b-1,c);
    return w(a-1,b,c)+w(a-1,b-1,c)+w(a-1,b,c-1)-w(a-1,b-1,c-1);
}

int main()
{
    int a,b,c;
    //将三个参数的所有可能性全部计算出来(打表)
    int i,j,k;
    for ( i=0; i<=20; i++ )
        for ( j=0; j<=20; j++ )
            for ( k=0; k<=20; k++ )
                f[i][j][k]=w(i,j,k);

    while (scanf("%d%d%d",&a,&b,&c)!=EOF)
    {
        if (a==-1 && b==-1 && c==-1) return 0;
        printf("w(%d,%d,%d) = ",a,b,c);
        printf("%d\n",w(a,b,c));
    }
}
```

算法实现原代码：zju1168.c。

3. ZJU1195-Blowing Fuses[①②③④]

【题目大意】

也许你熟悉以下情况。你已经插入了很多的电器设备，如烤面包机、冰箱、微波炉、电脑、音响等，而且它们都在运行。但是，当你打开电视机的时候，保险丝却烧掉了，因为所有电器设备的总功率超过了保险丝的容量。当然，这是一项伟大的安全措施，它避免了由于电线过热而经常引发的房屋烧毁现象。但走进地下室（或其他一些不方便的地方）去更换保险丝或把开关拨回去，同样也是令人烦恼的。

人们多么希望有这么一个程序，它能在打开一个电器设备之前，检查所有正在运行中的设备的混合功率是否超过了保险丝的容量（保险丝烧掉），或者是否能安全地打开电器设备。

输入

输入有多个测试例。对每个测试例描绘，是一组电器设备，并给出了这些电器设备的开/关状态序列。

每个测试例的第一行是三个整数 n，m 和 c，其中 n 是设备的数目（$n \leqslant 20$），m 是对这些设备进行操作的数量，c 是保险丝的容量（单位是安培）。接下来 n 行，每行是一个正整数 ci，是第 i 个设备的消耗功率（单位是安培）。

接下来 m 行，每行一个整数，在 1 和 n（含）之间，描述了对这些设备进行打开/关闭的操作序列。对每一个数字，表示相关设备的切换状态，如果该设备正在运行，那么将开关关闭；如果该设备是关闭的，那么将开关打开。在开始时，所有设备都是关闭的。

如果一个测试案例的 $n=m=c=0$ 时，则输入结束，该测试例不需要处理。

输出

对每一个测试例，首先输出测试例的编号，然后输出保险丝在一系列的操作之后是否会被烧毁。在某个时刻，当打开设备 i，将第 i 个设备的消耗功率 ci 加到总消耗功率中，如果超过了保险丝的容量 c，保险丝将被烧毁。

如果保险丝没有被烧毁，那么输出在操作期间开启设备的最大消耗功率。

在每个测试案例之后，输出一个空行。

样例输入

```
2 2 10
5
7
1
2
3 6 10
2
5
7
2
1
2
3
1
3
0 0 0
```

① http://acm.zju.edu.cn/onlinejudge/showProblem.do?problemCode=1195

② http://acm.pku.edu.cn/JudgeOnline/problem?id=1484

③ http://acm.uva.es/p/v6/661.html

④ http://www.informatik.uni-ulm.de/acm/Regionals/1998/

样例输出

```
Sequence 1
Fuse was blown.

Sequence 2
Fuse was not blown.
Maximal power consumption was 9 amperes.
```

题目来源：

Southwestern Europe 1998

【算法分析】

有 n 个电器设备，电路上有保险丝(容量为 c)，对这些电器设备进行 m 次开关操作。程序要求检查所有正在运行中的设备的混合功率是否超过了保险丝的容量，如果不超过的话，计算在操作期间开启设备的最大消耗功率。

(1) 样例分析

对样例数据 1，保险丝的容量是 $c=10$；设备 1 的功率是 5，设备 2 的功率是 7。

第 1 次将设备 1 打开，sum=5，max=5；

第 2 次将设备 2 打开，sum=12，max=12。

因为 max>c，所以保险丝烧掉。

对样例数据 2，保险丝的容量是 $c=10$；设备 1 的功率是 2，设备 2 的功率是 5，设备 3 的功率是 7。有 6 次设备的开关操作：

第 1 次将设备 2 打开，sum=5，max=5；

第 2 次将设备 1 打开，sum=7，max=7；

第 3 次将设备 2 关掉，sum=2，max=7；

第 4 次将设备 3 打开，sum=9，max=9；

第 5 次将设备 1 关掉，sum=7，max=9；

第 6 次将设备 3 关掉，sum=0，max=9。

因为 max=9<c，所以保险丝没有烧掉。在操作期间开启设备的最大消耗功率是 9。

(2) 数据结构

定义设备的数目为 n($n\leqslant 20$)，对这些设备进行操作的数量为 m，保险丝的容量为 c。每个设备的功率定义为数组：

```
int power[21];
```

每个设备的开关状态定义为数组：

```
int toggle[21];
```

(3) 模拟每一次开关操作

定义在操作期间开启设备的最大消耗功率为 max，当前操作时的总功率为 sum。

对每一次设备的开关操作：

① 如果设备 j 是开的，则关掉。

设备关掉的时候，保存状态的数组元素 toggle[j]状态为 0，同时从总消耗功率 sum 中减去该设备的功率 power[j]。

② 如果设备 j 是关的，则打开。

设备打开的时候，保存状态的数组元素 toggle[j]状态为 1，同时从总消耗功率 sum 中加

上该设备的功率 power[j],同时更新最大功率 max。

【程序代码】

```
#include <stdio.h>
#include <memory.h>

#define maxN 21

int power[maxN];                    //设备的功率
int toggle[maxN];                   //设备的开关状态
int n,m;                            //设备的数量,对设备开关操作的次数
int c;                              //保险丝的容量

int main()
{
    int i,j;
    int number = 1;
    while(scanf("%d%d%d",&n,&m,&c) && (n||m||c))
    {
        for(i = 1; i<=n; i++)
            scanf("%d",&power[i]);
        memset(toggle,0,sizeof(toggle));
        //在操作期间开启设备的最大消耗功率
        int max = 0;
        //当前操作时的总功率
        int sum = 0;
        //模拟每一次开关操作
        for(i = 1; i<=m; i++)
        {
            scanf("%d",&j);
            //如果设备 j 是开的,则关掉
            if(toggle[j])
            {
                toggle[j] = 0;
                sum -= power[j];
            }
            //如果设备 j 是关的,则打开
            else
            {
                toggle[j] = 1;
                sum += power[j];
                //更新最大功率
                if(max<sum) max = sum;
            }
        }
        printf("Sequence %d\n",number++);
        if(max>c) printf("Fuse was blown.\n");
        else
        {
            printf("Fuse was not blown.\n");
            printf("Maximal power consumption was %d amperes.\n",max);
        }
        printf("\n");
```

```
    }
    return 0;
}
```

算法实现原代码：zju1195.c。

4. ZJU1200-Mining[①]

【题目大意】

一座矿业基地需要建造一些机器人，采集至少 10 000 个单位的资源。每个机器人从基地出发到开始挖矿要 S 分钟，工作 W 分钟，然后花费 S 分钟携带 C 单位的资源回基地。

为加快生产流程，基地将建造 K 个机器人。建造一个机器人需要 M 分钟，机器人建造完毕就立即投入工作，然后生产线上立即建造下一个机器人，直到所有的机器人都建造完毕。

由于矿藏设备的限制，在工作的地方只有一个机器人在挖矿。也就是说，只有工作的机器人完成采集工作并且开始返回基地时，另一个机器人才可以挖矿。

编程任务：模拟这个过程，并计算至少采集 10 000 单位的资源需要花费多少分钟。

输入

输入有多行，每行一个测试例。每行有 5 个整数，命名为 S，W，C，K 和 M。

输出

对每个测试例，输出一个整数 t，是至少采集 10 000 单位资源所花费的时间(分钟数)。

样例输入

10 20 10 1 5

样例输出

40005

题目来源：

Zhejiang University Local Contest 2002, Preliminary

【算法分析】

本题是要收集 10 000 个单位的资源。每个机器人从基地出发到矿区要 S 分钟，工作 W 分钟，带回 C 个单位的资源，回到基地也是 S 分钟。题目中不止一个机器人，为了加快进度，要建造 K 个机器人，建造一个机器人要花 M 分钟，建造之后马上就可以工作了，但只能一个个地建造，且在收集资源时，一次只能让一个机器人来工作。

(1) 样例分析

样例中 $S=10$，$W=20$，$C=10$，$K=1$ 和 $M=5$。只有一个机器人，每次带回 10 个资源，所以要来回：10 000/10=1 000 次。

来回一次的时间：$2\times S+W=2\times 10+20=40$。

建造机器人的时间是 5 分钟。总时间为：$1\,000\times 40+5=40\,005$。

(2) 计算需要机器人搬运的次数 cnt

显然，cnt=9 999/C+1；

当然剩下零头的不够 C 个资源时，还得工作 1 次。

① http://acm.zju.edu.cn/onlinejudge/showProblem.do?problemCode=1200