



# 第十二届全国青少年信息学奥林匹克 联赛复赛试题

(NOIP2006 提高组)

竞赛时间：2006 年 11 月 18 日上午 8：30—11：30

试题名称	energy	budget	jsp	digital
目录	energy	budget	jsp	digital
输入文件名	energy.in	budget.in	jsp.in	digital.in
输出文件名	energy.out	budget.out	jsp.out	digital.out
试题类型	非交互式程序题	非交互式程序题	非交互式程序题	非交互式程序题
附加文件	无	无	无	无
时限	1 秒	1 秒	1 秒	1 秒

## 关于竞赛中不同语言使用限制的说明

### 一．关于使用 Pascal 语言与编译结果的说明

- 1．对于 Pascal 语言的程序，当使用 IDE 和 fpc 编译结果不一致时，以 fpc 的编译结果为准。
- 2．允许使用数学库(uses math 子句)，以及 ansistring。但不允许使用编译开关（最后测试时 pascal 的范围检查开关默认关闭：{\$R-,Q-,S-}），也不支持与优化相关的选项。

### 二．关于 C++语言中模板使用的限制说明

#### 1．允许使用的部分：

标准容器中的布尔集合，迭代器，串，流。

相关的头文件：<bitset> <iterator> <string> <iostream>

#### 2．禁止使用的部分：

序列：vector，list，deque

序列适配器：stack，queue，priority\_queue

关联容器：map，multimap，set，multiset

拟容器：valarray

散列容器：hash\_map，hash\_set，hash\_multimap，hash\_multiset

所有的标准库算法

相关头文件：<vector> <list> <deque> <stack> <map> <set> <algorithm>



## 1. 能量项链

(energy.pas/c/cpp)

### 【问题描述】

在 Mars 星球上，每个 Mars 人都随身佩带着一串能量项链。在项链上有  $N$  颗能量珠。能量珠是一颗有头标记与尾标记的珠子，这些标记对应着某个正整数。并且，对于相邻的两颗珠子，前一颗珠子的尾标记一定等于后一颗珠子的头标记。因为只有这样，通过吸盘（吸盘是 Mars 人吸收能量的一种器官）的作用，这两颗珠子才能聚合成一颗珠子，同时释放出可以被吸盘吸收的能量。如果前一颗能量珠的头标记为  $m$ ，尾标记为  $r$ ，后一颗能量珠的头标记为  $r$ ，尾标记为  $n$ ，则聚合后释放的能量为  $m \times r \times n$  (Mars 单位)，新产生的珠子的头标记为  $m$ ，尾标记为  $n$ 。

需要时，Mars 人就用吸盘夹住相邻的两颗珠子，通过聚合得到能量，直到项链上只剩下一颗珠子为止。显然，不同的聚合顺序得到的总能量是不同的，请你设计一个聚合顺序，使一串项链释放出的总能量最大。

例如：设  $N=4$ ，4 颗珠子的头标记与尾标记依次为  $(2, 3)$   $(3, 5)$   $(5, 10)$   $(10, 2)$ 。我们用记号  $\oplus$  表示两颗珠子的聚合操作， $(j \oplus k)$  表示第  $j, k$  两颗珠子聚合后所释放的能量。则第 4、1 两颗珠子聚合后释放的能量为：

$$(4 \oplus 1) = 10 \times 2 \times 3 = 60。$$

这一串项链可以得到最优值的一个聚合顺序所释放的总能量为

$$((4 \oplus 1) \oplus 2) \oplus 3 = 10 \times 2 \times 3 + 10 \times 3 \times 5 + 10 \times 5 \times 10 = 710。$$

### 【输入文件】

输入文件 energy.in 的第一行是一个正整数  $N$  ( $4 \leq N \leq 100$ )，表示项链上珠子的个数。第二行是  $N$  个用空格隔开的正整数，所有的数均不超过 1000。第  $i$  个数为第  $i$  颗珠子的头标记 ( $1 \leq i \leq N$ )，当  $i < N$  时，第  $i$  颗珠子的尾标记应该等于第  $i+1$  颗珠子的头标记。第  $N$  颗珠子的尾标记应该等于第 1 颗珠子的头标记。

至于珠子的顺序，你可以这样确定：将项链放到桌面上，不要出现交叉，随意指定第一颗珠子，然后按顺时针方向确定其他珠子的顺序。

### 【输出文件】

输出文件 energy.out 只有一行，是一个正整数  $E$  ( $E \leq 2.1 \times 10^9$ )，为一个最优聚合顺序所释放的总能量。

### 【输入样例】

```
4
2 3 5 10
```

### 【输出样例】

```
710
```

## 2. 金明的预算方案

(budget.pas/c/cpp)

### 【问题描述】



金明今天很开心，家里购置的新房就要领钥匙了，新房里有一间金明自己专用的很宽敞的房间。更让他高兴的是，妈妈昨天对他说：“你的房间需要购买哪些物品，怎么布置，你说了算，只要不超过  $N$  元钱就行”。今天一早，金明就开始做预算了，他把想买的物品分为两类：主件与附件，附件是从属于某个主件的，下表就是一些主件与附件的例子：

主件	附件
电脑	打印机，扫描仪
书柜	图书
书桌	台灯，文具
工作椅	无

如果要买归类为附件的物品，必须先买该附件所属的主件。每个主件可以有 0 个、1 个或 2 个附件。附件不再有从属于自己的附件。金明想买的东西很多，肯定会超过妈妈限定的  $N$  元。于是，他把每件物品规定了一个重要度，分为 5 等：用整数 1~5 表示，第 5 等最重要。他还从因特网上查到了每件物品的价格（都是 10 元的整数倍）。他希望在不超过  $N$  元（可以等于  $N$  元）的前提下，使每件物品的价格与重要度的乘积的总和最大。

设第  $j$  件物品的价格为  $v[j]$ ，重要度为  $w[j]$ ，共选中了  $k$  件物品，编号依次为  $j_1, j_2, \dots, j_k$ ，则所求的总和为：

$$v[j_1] * w[j_1] + v[j_2] * w[j_2] + \dots + v[j_k] * w[j_k]。 \quad (\text{其中 } * \text{ 为乘号})$$

请你帮助金明设计一个满足要求的购物单。

#### 【输入文件】

输入文件 `budget.in` 的第 1 行，为两个正整数，用一个空格隔开：

$N \quad m$

（其中  $N$  ( $<32000$ ) 表示总钱数， $m$  ( $<60$ ) 为希望购买物品的个数。）

从第 2 行到第  $m+1$  行，第  $j$  行给出了编号为  $j-1$  的物品的的基本数据，每行有 3 个非负整数

$v \quad p \quad q$

（其中  $v$  表示该物品的价格 ( $v < 10000$ )， $p$  表示该物品的重要度 (1~5)， $q$  表示该物品是主件还是附件。如果  $q=0$ ，表示该物品为主件，如果  $q>0$ ，表示该物品为附件， $q$  是所属主件的编号)

#### 【输出文件】

输出文件 `budget.out` 只有一个正整数，为不超过总钱数的物品的价格与重要度乘积的总和的最大值 ( $<200000$ )。

#### 【输入样例】

1000 5

800 2 0

400 5 1

300 5 1

400 3 0

500 2 0

#### 【输出样例】

2200

### 3. 作业调度方案

(jsp.pas/c/cpp)

### 【问题描述】

我们现在要利用  $m$  台机器加工  $n$  个工件，每个工件都有  $m$  道工序，每道工序都在不同的指定的机器上完成。每个工件的每道工序都有指定的加工时间。

每个工件的每个工序称为一个操作，我们用记号  $j-k$  表示一个操作，其中  $j$  为  $1$  到  $n$  中的某个数字，为工件号； $k$  为  $1$  到  $m$  中的某个数字，为工序号，例如  $2-4$  表示第  $2$  个工件第  $4$  道工序的这个操作。在本题中，我们还给定对于各操作的一个安排顺序。

例如，当  $n=3, m=2$  时，“ $1-1, 1-2, 2-1, 3-1, 3-2, 2-2$ ”就是一个给定的安排顺序，即先安排第  $1$  个工件的第  $1$  个工序，再安排第  $1$  个工件的第  $2$  个工序，然后再安排第  $2$  个工件的第  $1$  个工序，等等。

一方面，每个操作的安排都要满足以下的两个约束条件。

- (1) 对同一个工件，每道工序必须在它前面的工序完成后才能开始；
- (2) 同一时刻每一台机器至多只能加工一个工件。

另一方面，在安排后面的操作时，不能改动前面已安排的操作的工作状态。

由于同一工件都是按工序的顺序安排的，因此，只按原顺序给出工件号，仍可得到同样的安排顺序，于是，在输入数据中，我们将这个安排顺序简写为“ $1\ 1\ 2\ 3\ 3\ 2$ ”。

还要注意，“安排顺序”只要求按照给定的顺序安排每个操作。不一定是各机器上的实际操作顺序。在具体实施时，有可能排在后面的某个操作比前面的某个操作先完成。

例如，取  $n=3, m=2$ ，已知数据如下：

工件号	机器号/加工时间	
	工序 1	工序 2
1	1/3	2/2
2	1/2	2/5
3	2/2	1/4

则对于安排顺序“ $1\ 1\ 2\ 3\ 3\ 2$ ”，下图中的两个实施方案都是正确的。但所需要的总时间分别是  $10$  与  $12$ 。



当一个操作插入到某台机器的某个空档时（机器上最后的尚未安排操作的部分也可以看作一个空档），可以靠前插入，也可以靠后或居中插入。为了使问题简单一些，我们约定：在保证约束条件（1）（2）的条件下，尽量靠前插入。并且，我们还约定，如果有多个空档可以插入，就在保证约束条件（1）（2）的条件下，插入到最前面的一个空档。于是，在这些约定下，上例中的方案一是正确的，而方案二是不正确的。

显然，在这些约定下，对于给定的安排顺序，符合该安排顺序的实施方案是唯一的，请你计算出该方案完成全部任务所需的总时间。

### 【输入文件】

输入文件 `jsp.in` 的第  $1$  行为两个正整数，用一个空格隔开：



$m$   $n$

(其中  $m$  ( $<20$ ) 表示机器数,  $n$  ( $<20$ ) 表示工件数)

第 2 行:  $m \times n$  个用空格隔开的数, 为给定的安排顺序。

接下来的  $2n$  行, 每行都是用空格隔开的  $m$  个正整数, 每个数不超过 20。

其中前  $n$  行依次表示每个工件的每个工序所使用的机器号, 第 1 个数为第 1 个工序的机器号, 第 2 个数为第 2 个工序机器号, 等等。

后  $n$  行依次表示每个工件的每个工序的加工时间。

可以保证, 以上各数据都是正确的, 不必检验。

#### 【输出文件】

输出文件 jsp.out 只有一个正整数, 为最少的加工时间。

#### 【输入样例】

```
2 3
1 1 2 3 3 2
1 2
1 2
2 1
3 2
2 5
2 4
```

#### 【输出样例】

```
10
```

## 4. $2^k$ 进制数

(digital.pas/c/cpp)

#### 【问题描述】

设  $r$  是个  $2^k$  进制数, 并满足以下条件:

- (1)  $r$  至少是个 2 位的  $2^k$  进制数。
- (2) 作为  $2^k$  进制数, 除最后一位外,  $r$  的每一位严格小于它右边相邻的那一位。
- (3) 将  $r$  转换为 2 进制数  $q$  后, 则  $q$  的总位数不超过  $w$ 。

在这里, 正整数  $k$  ( $1 \leq k \leq 9$ ) 和  $w$  ( $k < w \leq 30000$ ) 是事先给定的。

问: 满足上述条件的不同的  $r$  共有多少个?

我们再从另一角度作些解释: 设  $S$  是长度为  $w$  的 01 字符串 (即字符串  $S$  由  $w$  个“0”或“1”组成),  $S$  对应于上述条件 (3) 中的  $q$ 。将  $S$  从右起划分为若干个长度为  $k$  的段, 每段对应一位  $2^k$  进制的数, 如果  $S$  至少可分成 2 段, 则  $S$  所对应的二进制数又可以转换为上述的  $2^k$  进制数  $r$ 。

例: 设  $k=3, w=7$ 。则  $r$  是个八进制数 ( $2^3=8$ )。由于  $w=7$ , 长度为 7 的 01 字符串按 3 位一段分, 可分为 3 段 (即 1, 3, 3, 左边第一段只有一个二进制位), 则满足条件的八进制数有:

2 位数: 高位为 1: 6 个 (即 12, 13, 14, 15, 16, 17), 高位为 2: 5 个, ..., 高位为 6: 1 个 (即 67)。共  $6+5+\dots+1=21$  个。



3 位数：高位只能是 1，第 2 位为 2 : 5 个（即 123，124，125，126，127），第 2 位为 3 : 4 个，…，第 2 位为 6 : 1 个（即 167）。共  $5+4+\dots+1=15$  个。

所以，满足要求的  $r$  共有 36 个。

**【输入文件】**

输入文件 digital.in 只有 1 行，为两个正整数，用一个空格隔开：

k w

**【输出文件】**

输出文件 digital.out 为 1 行，是一个正整数，为所求的计算结果，即满足条件的不同的  $r$  的个数（用十进制数表示），要求最高位不得为 0，各数字之间不得插入数字以外的其他字符（例如空格、换行符、逗号等）。

（提示：作为结果的正整数可能很大，但不会超过 200 位）

**【输入样例】**

3 7

**【输出样例】**

36