ACM South American Regional

Collegiate Programming Contest

Warmup Session

November 18, 2000

Buenos Aires – Argentina
Campina Grande – Brazil
Porto Alegre – Brazil
São Paulo – Brazil
Santiago – Chile
Merida – Venezuela

acm International Collegiate Programming Contest 2001 IBM event sponsor
CREDITS HOME REGIONALS WORLD FINALS INFO PAST CONTACTS

PROBLEM 1: Recognizing Good ISBNs

File Names

Source File: isbn.c, etc
Input File: isbn.in
Output File: isbn.out

Statement of the Problem

Most books now published are assigned a code which uniquely identifies the book. The International Standard Book Number, or ISBN, is normally a sequence of 10 decimal digits, but in some cases, the capital letter X may also appear as the tenth digit. Hyphens are included at various places in the ISBN to make them easier to read, but have no other significance. The sample input and expected output shown below illustrate many valid, and a few invalid, forms for ISBNs.

Actually, only the first nine digits in an ISBN are used to identify a book. The tenth character serves as a check digit to verify that the preceding 9 digits are correctly formed. This check digit is selected so that the value computed as shown in the following algorithm is divisible by 11. Since the check digit may sometimes need to be as large as 10 to guarantee divisibility by 11, a special symbol was selected by the ISBN designers to represent 10, and that is the role played by X. The algorithm used to check an ISBN is relatively simple. Two sums, s_1 and s_2, are computed over the digits of the ISBN. s_1 is a partial sum of the digits in the ISBN and s_2 is the sum of the partial sums in s1. The ISBN is correct if the final value of s_2 is divisible by 11. An example will clarify the procedure. Consider the (correct) ISBN 0-13-162959-X. First look at the calculation of s_1:

```
-----------------------------------------------------------------
digits in the ISBN    0   1   3   1   6   2   9   5   9   10(X)
s1 (partial sums)     0   1   4   5  11  13  22  27  36   46
-----------------------------------------------------------------
```

The calculation of $s2$ is done by computing the total of the partial sums in the calculation of $s1$:

```
-----------------------------------------------------------------
s2 (running totals)   0   1   5  10  21  34  56  83  119  165
-----------------------------------------------------------------
```

We now verify the correctness of the ISBN by noting that 165 is, indeed, divisible by 11.

Input Format

The input data for this problem will contain one candidate ISBN per line of input, perhaps preceded and/or followed by additional spaces. No line will contain more than 80 characters, but the candidate ISBN may contain illegal characters, and more or fewer than the required 10 digits. Valid ISBNs may include hyphens at arbitrary locations. The input ends with a special ISBN: 0-00000-000-0.

Output Format

The output should include a display of the candidate ISBN (without spaces) and a statement of whether it is correct or wrong. The expected output shown below illustrates the expected format.

Sample Input

```
0-89237-010-6
  0-8306-3637-4
  0-8306-3637-5
0-00000-000-0
```

Sample Output

```
0-89237-010-6 is correct.
0-8306-3637-4 is correct.
0-8306-3637-5 is wrong.
```

PROBLEM 2: Network Message Distribution

File Names

Source File: network.c, etc
Input File: network.in
Output File: network.out

Statement of the Problem

A message network is composed by message centers and directed links between centers, i.e. link (i, j) indicates that center i can send messages to center j. Links are not symmetric; the existence of link (i, j) do not imply that center j can send messages to center i.

This particular message network is designed to operate as follows. All messages that arrive from outside the message network are broadcast to a subset S of network centers. This set S is such that there are no links in the message network connecting any two centers in S. In the next step, the centers in S will forward the received messages through links to other centers. The centers outside S will do the same. The messages must reach every center in the message network provided that they pass through at most one center outside S.

Your task is to write a program to obtain one such set S.

Input Format

A network (a directed graph) using the following format

```
np
n1
nc1  x11 x12 x13 ... x1(nc1)
nc2  x21 x22 x23 ... x2(nc2)
...
nc(n1)  x(n1)1 x(n1)2 x(n1)3 ... x(nl)(nc(nl))
n2
nc1  x11 x12 x13 ... x1(nc1)
nc2  x21 x22 x23 ... x2(nc2)
...
nc(n2)  x(n2)1 x(n2)2 x(n2)3 ... x(n2)(nc(n2))
...
n(np)
nc1  x11 x12 x13 ... x1(nc1)
nc2  x21 x22 x23 ... x2(nc2)
...
nc(n(np))  x(n(np))1 x(n(np))2 x(n(np))3 ... x(n(np))(nc(n(np)))
```

where np is the number of problems in the file; $n1, n2, ..., n(np)$ are the number of centers in problem 1, 2, ..., np, respectively; ncj is the number of centers directly reachable (through a link) from center

j, for all centers j on all problems; and xjk is the k-th center that can be directly reached from center j. All values are separated by 2 spaces.

Output Format

A set of centers (a vertex subset of the input graph) with the indication of which problem they refer to (see example).

Sample Input

```
2
5
0
2  3  4
2  1  2
1  2
2  3  4
7
0
4  1  3  5  7
2  1  6
4  1  2  3  5
1  1
3  2  4  7
2  2  4
```

Sample Output

```
Problem 1: S={1,5}
Problem 2: S={6}
```

PROBLEM 3: Editing Distance

File Names

Source File: dna.c, etc
Input File: dna.in
Output File: dna.out

Statement of the Problem

Since the acceptance of Darwin's theory on evolution, many researchers have tried to understand the evolutionary history of living organisms. This study is done normally through the construction of *phylogenetic trees*. Up to 1950, the construction of these trees was based on experiments and intuition. Gradually, mathematical formalism appeared and now, with the development of molecular biology, data about the molecular structure of organisms is being used in these constructions.

One of the sources of data is the evolutionary distance between species. A tentative way to determine this distance is to study the differences between the DNA of two species. A DNA molecule consists of a double chain of basis. There are four basis, each one denoted usually by one of the letters A, C, G or T.

The simplest events that occur during the course of molecular evolution are substitution of one base for another and the insertion or deletion of a base-pair. The *editing distance* between the DNA molecules of the two species is the minimum number of these events—deletion, insertion and substitution—that transform a DNA molecule into the other one.

Your mission is to write a program that reads a sequence of pairs of DNA molecules (given by the sequence of basis of one of the chains) and computes, for each pair, its editing distance.

Input Format

The first line of the input file contains a number n ($0 < n \leq 20$), which indicates the number of given pairs of DNA sequences. In each of the next $2n$ lines, there is a sequence of basis, that is, a sequence of A, C, G or T, of length at most 200.

Output Format

The output should consist of n lines, one for each pair of DNA sequences. Each line consists of `Pair \#i: editing distance d`, where $i \in \{1, \ldots, n\}$ and d is the editing distance of the ith pair of DNA sequences.

Sample Input

```
3
ATAAGC
AAAAACG
AATCGTATGGCCCTA
ATGGTGATGGCCAT
GCTCTGCGAATA
```

```
CGTTGAGATACT
```

Sample Output

```
Pair #1: editing distance 3
Pair #2: editing distance 5
Pair #3: editing distance 7
```