University of Scranton
ACM Student Chapter / Computing Sciences Department
10th Annual High School Programming Contest (1999)

Problem 1: What's My Score?

Ann, Bob, Carol, and Dave played nine holes of golf together. On the first hole, Ann teed off first, followed by Bob, then Carol, and, finally, Dave. At each subsequent hole, the order in which they teed off was in accord with the usual golfing convention, which is as follows: The golfer with the lowest score on the previous hole tees off first, followed by the golfer with the second lowest score, etc., etc. Golfers having equal scores on the previous hole tee off in the same relative order as they did on the previous hole.

A scorecard has four columns (one for each golfer) and nine rows (one for each of the nine holes). The score achieved by a given golfer on a given hole is to be recorded in the row corresponding to that hole in the column corresponding to that golfer.

Bob was responsible for keeping score. He was well aware that the scores achieved by the golfers on a given hole were to be recorded on the scorecard in the row corresponding to that hole. Unfortunately, he did not possess a proper understanding of the purpose of the four columns on the scorecard! He thought that the first column was for recording the score of the golfer who teed off first on the given hole, the second column was for recording the score of the golfer who teed off second on the given hole, etc., etc. Thus, in each row of the scorecard, Bob recorded the correct scores, but (usually, at least) in the wrong columns.

You are to develop a program that takes as input the scorecard filled out by Bob and that produces as output the "correct" scorecard.

As illustrated in the sample execution below, your program should prompt the user to enter Bob's original scorecard, one row at a time. The numbers entered by the user shall be separated from each other by at least one space. After consuming all the input, your program should display the correct scorecard, with one column showing hole numbers and four subsequent columns showing the scores of Ann, Bob, Carol, and Dave, in that order. Each golfer's total score (the sum of the scores on the nine holes) should be displayed at the bottom of her/his column.

Sample execution:

```
==============================================================
Enter hole 1 from original scorecard:  4  6  5  5
Enter hole 2 from original scorecard:  5  4  6  4
Enter hole 3 from original scorecard:  4  3  6  5
Enter hole 4 from original scorecard:  4  4  3  5
Enter hole 5 from original scorecard:  5  3  4  4
Enter hole 6 from original scorecard:  5  5  5  5
Enter hole 7 from original scorecard:  4  4  4  4
Enter hole 8 from original scorecard:  4  4  4  4
Enter hole 9 from original scorecard:  2  4  3  4

Corrected scorecard:
1         4   6   5   5
2         5   4   4   6
3         6   3   4   5
4         5   4   4   3
5         4   3   4   5
6         5   5   5   5
7         4   4   4   4
8         4   4   4   4
9         3   2   4   4
------------------------------------
Totals 40  35  38  41
```

University of Scranton
ACM Student Chapter / Computing Sciences Department
10th Annual High School Programming Contest (1999)

Problem 2: Spiral of Squares

Imagine a grid whose cells have been numbered, starting at zero, in the outwardly-spiraling pattern illustrated below. Imagine that the spiral extends out to infinity.

```
56 57 58 59 60 61 62 63 64
55 30 31 32 33 34 35 36 65
54 29 12 13 14 15 16 37 66
53 28 11  2  3  4 17 38 67
52 27 10  1  0  5 18 39 68
51 26  9  8  7  6 19 40 69
50 25 24 23 22 21 20 41 70
49 48 47 46 45 44 43 42 71
80 79 78 77 76 75 74 73 72
```

You are to develop a program that, given as input two nonnegative integers j and k, with neither exceeding 1000, outputs the distance on the grid between the cells numbered j and k.

By *distance between two cells* we mean the fewest number of steps by which it is possible to get from one to the other, assuming that in each step we can move one position in a horizontal or vertical (but **not** diagonal) direction.

End of input is signaled when both inputs are zero.

Sample execution:

```
=====================================
Enter two inputs:  15 42
Distance:  7

Enter two inputs:  132 132
Distance:  0

Enter two inputs:  37 25
Distance:  10

Enter two inputs:  0 0
```

University of Scranton
ACM Student Chapter / Computing Sciences Department
10th Annual High School Programming Contest (1999)

Problem 3: Hill Removal

A sequence of three or more integers whose members are in strictly increasing or strictly decreasing order is called a *hill*. (For example, $2, 5, 8, 9$ and $5, 3, -1$ are hills. However, $4, 7, 7, 12$ is not, because its 3rd member is no greater than its 2nd.)

A sequence of integers that is not itself a hill may contain one or more contiguous subsequences that *are* hills. For example, the sequence $5, 3, 5, 9, 7, 4, 1, 2$ contains as subsequences the hills $3, 5, 9$ and $9, 7, 4$ and $7, 4, 1$ and $9, 7, 4, 1$.

Develop a program that, given as input a sequence of integers, rearranges them so that the resulting sequence contains no contiguous subsequences that are hills. For each input sequence, the user shall enter its length followed by the sequence itself (as illustrated below). End of input is signaled by a sequence length of zero. The user shall not enter a sequence of length exceeding twenty.

Sample execution:

```
============================================
Enter sequence length:  9
Enter element 1:  5
Enter element 2:  8
Enter element 3:  5
Enter element 4:  3
Enter element 5:  7
Enter element 6:  9
Enter element 7:  12
Enter element 8:  6
Enter element 9:  8

5 8 3 7 5 9 6 12 8

Enter sequence length:  6
Enter element 1:  1
Enter element 2:  2
Enter element 3:  3
Enter element 4:  4
Enter element 5:  5
Enter element 6:  6

1 3 2 5 4 6

Enter sequence length:  0
```

Problem 4: Number Representation in the Factorial Base

For a positive integer m, we define $m! = 1 \cdot 2 \cdot 3 \cdot \ \cdots \ \cdot m$. That is, $m!$ is the product of the positive integers 1 through m. (This is the well known *factorial* function.)

Interestingly, for every positive integer k there exists a unique sequence of nonnegative integers d_1, d_2, ..., d_n, (where $d_j \leq j$ for all j) such that

$$k = d_1 \cdot 1! \ + \ d_2 \cdot 2! \ + \ d_3 \cdot 3! \ + \ \cdots \ + \ d_n \cdot n!$$

In the factorial base number system, the number k would be represented by the sequence of "digits" $d_n d_{n-1} \cdots d_2 d_1$. For example, the number that we express as 523 in the standard decimal number system would be expressed by 41301 in the factorial base, corresponding to the fact that

$$523 \ = \ 1 \cdot 1! \ + \ 0 \cdot 2! \ + \ 3 \cdot 3! \ + \ 1 \cdot 4! \ + \ 4 \cdot 5! \ = \ 1 + 0 + 18 + 24 + 480$$

Develop a program that takes as input a positive integer and that produces as ouput its representation in the factorial base number system, as illustrated below. (*Note:* The "digits" in the output may be separated by spaces.) End of input is signaled by the user entering zero.

You may assume that the user shall enter no input value whose factorial base representation requires more than seven digits.

Hint: By repeatedly applying the distributive property of multiplication over addition, you will find that the number $k = d_1 \cdot 1! \ + \ d_2 \cdot 2! \ + \ \cdots \ + \ d_n \cdot n!$ can also be written as follows:

$$k = d_1 + 2(d_2 + 3(d_3 + 4(d_4 + \cdots + (n-1)(d_{n-1} + n \cdot d_n) \cdots)$$

Thus, for instance, to determine the digit d_1, it suffices to take the remainder of the division $k/2$. The remaining digits can be determined by carrying out more divisions, using the same idea. *End of Hint.*

Sample execution:

```
==============================
Enter input:  1811
230121

Enter input:  0
```

Problem 5: No Hassle Auto Insurance

The No Hassle Auto Insurance company believes that everyone deserves a second chance, and therefore it does not raise the rate of a driver after he is involved in his first accident or receives his first ticket. However, the rate *will* increase for a driver involved in a second accident or receiving a second ticket. A driver whose number of accidents plus tickets exceeds four is uninsurable.

The basic formula for the rate charged by the No Hassle company is

$$r = 0.02 \cdot c \cdot a! \cdot t! \cdot y$$

where c stands for the value of the car (see below), a stands for number of accidents, t stands for number of tickets received, and y stands for the *age multiplier* (see below).

Recall that $0! = 1$ and, for $m > 0$, $m! = 1 \cdot 2 \cdot 3 \cdots m$ (i.e., the product of the integers 1 through m). This is the well known factorial function.

To calculate c (the value of the car), the purchase price and age of the car are required. Specifically, the value of a car of age zero is its purchase price. The value of a car of age greater than zero is 90 percent of its previous year's value.

To calculate y, we need the driver's age. Specifically, for a driver between the ages of 16 and 24, $y = 3$. For ages 25 through 34, $y = 1.5$. For ages 35 through 44, $y = 1$. For ages, 45 through 64, $y = 1.5$. For ages 65 and above, $y = 3$.

Develop a program that takes as input the purchase price and age of a car, the age of its driver, the number of accidents in which the driver has been involved, and the number of tickets the driver has received. (All these will be nonnegative integers.) As output, the program should display the insurance rate for the driver, accurate to at least two places after the decimal point, or else a message indicating that she is uninsurable.

End of input is signaled by the user entering zero for the purchase price of the car.

Sample execution:

```
=============================================================
Enter purchase price of car:  20000
Enter age of car:  3
Enter age of driver:  26
Enter number of accidents:  0
Enter number of tickets:  2
Rate:  874.80

Enter purchase price of car:  12000
Enter age of car:  1
Enter age of driver:  19
Enter number of accidents:  1
Enter number of tickets:  2
Rate:  1296.00

Enter purchase price of car:  34000
Enter age of car:  6
Enter age of driver:  80
Enter number of accidents:  1
Enter number of tickets:  4
Driver is uninsurable

Enter purchase price of car:  0
```

Problem 6: Lax Security

The Lax Hazardous Waste Company must maintain a certain level of security at its plant because of the nature of the materials handled there. The main door to the plant has a keypad similar to the one shown below. To open the door, an employee must first enter her five-digit personal identification number (PIN). If the PIN is valid, the door opens. Otherwise, it does not.

```
+---+ +---+ +---+
| 1 | | 2 | | 3 |
+---+ +---+ +---+
+---+ +---+ +---+
| 4 | | 5 | | 6 |
+---+ +---+ +---+
+---+ +---+ +---+
| 7 | | 8 | | 9 |
+---+ +---+ +---+
      +---+
      | 0 |
      +---+
```

However, Bocefus Lax, the company owner, is a nice lady who doesn't like to burden her employees unnecessarily. The digits on the keypad are close together and employees often enter their PIN incorrectly by pressing an adjacent button instead of the intended one. Therefore, she wants to modify the access system so that it allows an employee to open the door even if the PIN they enter is slightly incorrect. Specifically, Bocefus is willing to assume that the person using the keypad is one of her employees if the number the person enters has no more than one digit that is off-by-one from some valid PIN. Here, off-by-one means that, with respect to the keypad, the incorrect digit is either immediately above, below, to the right of, or to the left of the correct digit. (For example, the digits that are off-by-one from 4 are 1, 5, and 7.)

Develop a program that takes as input a list of valid PINs followed by a list of five-digit numbers. For each number in the latter list, the program is to report whether it is accepted or rejected. For each accepted number, the program reports which valid PIN it matches (exactly or approximately).

Note that a number entered on the keypad could match more than one valid PIN. For example, the PIN 22222 is off-by-one from 21222, 52222, 22223, and many others. If a number entered on the keypad matches a valid PIN exactly, that PIN should be identified. If the number matches no valid PIN exactly, but approximately matches some valid PIN, report the first such valid PIN in the list.

End of input is signaled by the user entering all zeros.

Sample execution:

```
================================================
Enter number of valid PINs:  4
Enter valid PIN number 1:  10360
Enter valid PIN number 2:  36243
Enter valid PIN number 3:  39491
Enter valid PIN number 4:  26543

Enter a number:  39491
Matches 39491

Enter a number:  36491
Matches 39491

Enter a number:  14543
No matches

Enter a number:  26243
Matches 36243

Enter a number:  00000
```