National 1976

PROBLEM # 1

LARGE INTEGERS

Your program is to read a card containing two non-negative integers (not both zero), print their sum on a double-spaced line, and continue in this fashion until the deck terminator, a blank card, is encountered. On each card, one integer is contained somewhere in the first forty columns and the other is somewhere in the latter forty. Non-trailing zeroes may appear as blanks and the values of the integer data may exceed the capacity of an integer variable. The sum is to be printed without leading zeroes, right-justified at column 50, and with commas separating each group of three digits starting at the right.

For example, the card

```
                        (Col 40)
                            |            |
     05 16523892            |            6723519440
is to produce the line      |            |
                        11,740,043,332
                            |            |
                            |    (Col 50)
```

PROBLEM # 2

VIRTUAL BRICK WALLS

When building a brick wall it is essential to purchase enough material. If bricks of various sizes are used an interesting simulation of the "virtual wall" can be made to determine the number of bricks needed. For this problem you are given the size of the wall and the number of each type of bricks available. You are to determine if the desired wall can be constructed from these bricks and what its courses can look like.

Input: A sequence of data cards, format 5I5, terminated by a blank card. Each card describes a wall-building problem as follows:

Field 1 - width of wall in units (≤ 20)
Field 2 - number of courses (rows of bricks) needed (≤ 12)
Field 3 - number of type 3 bricks available
Field 4 - number of type 2 bricks available
Field 5 - number of type 1 bricks available

Bricks of type 3 are 3 units wide, bricks of type 2 are 2 units wide and bricks of type 1 are 1 unit wide. Bricks are always to be laid flat, never placed on edge or end.

Output: For each wall-building problem, on a separate page, either print only the message

CAN'T BE DONE

if the bricks supplied cannot build the wall requested; or if it can be done print a picture of the wall as described below.

Print pictures of bricks in the following sizes

Type 3 Brick

```
******************
*                *
*                *
******************
```

Type 2 Brick

```
************
*          *
*          *
************
```

Type 1 Brick

```
******
*    *
*    *
******
```

PROBLEM #2 - Page 2

So that the set of problem data consisting of

5   3   4   2   6

has as one of its correct responses the picture

```
******************************
*                **          *
*                **          *
******************************
******************************
*          **                *
*          **                *
******************************
******************************
*    **                **    *
*    **                **    *
******************************
```

Zero in the first field of a card indicate the termination of the data deck and hence the problems.

PROBLEM # 3

SECURITY CODES

To insure the correctness of frequently used codes, a sequence of check digits can be systematically embedded with each usage. For example, a particular security code algorithm could use the sum of all digits modulo 26 and the corresponding character of the alphabet attached to the end. In such a case 57359 would require the check character 'D' because 5+7+3+5+9=29 and 29 modulo 26 equals 3 which indexes the fourth letter in the alphabet or 'D' (A being 0). There are reasonably sophisticated variations of the scheme in wide use today in credit cards, security codes and passwords.

The inverse of this process is to break such a coding scheme given enough information. For this problem you are given a particular check digit coding scheme to break. This specific scheme uses a summation from left to right with either addition, subtraction or multiplication among adjacent digits. You also know that each digit is used once and only once in the algorithm. Given a set of five digit codes followed by a single alphabetic character, write a program to break this code.

The input will be 10 sets of six character fields as examples of the code on a single card in columns 1 thru 60 respectively. This will be followed by a second card with 10 sets of five digit fields in columns 1 thru 50 respectively. You are to determine the check digit code algorithm and provide the check characters. Print out the initial sequences, the check digit algorithm you have determined, as in A+B+C+D+E in the initial example, and the new sets of codes you have created using this algorithm.

National 1910

PROBLEM # 4

KNIGHT'S TOUR SIMULATION

Given an n x n chessboard, a Knight's Tour is the series of moves whereby each square is 'visited' once and only once by a piece performing legal (chess) Knight moves. In the following diagram, the Knight (K) can move to any one of the squares numbered 1 through 8.

	8		1	
7				2
		K		
6				3
	5		4	

(n = 5)

Obviously, a Knight near an edge or corner of the board has fewer potential moves available.

This problem requires the generation of one of the correct Knight's tours on a 5 X 5 board. The initial starting location for each game will be provided as a set of integers (i, j) where i is right justified in column 1 and j is right justified in column 2. A zero in column 1 and 2 indicates a termination of the problem.

Output should be a 5 X 5 integer array of numbers representing the moves from square to square. Also, the output above should be followed by the word 'open' or the word 'closed' (a closed Knight's tour is where the starting and ending positions are a Knight's move apart; otherwise the tour is open) as appropriate.

Sample input might be: 33 with one of the possible outputs:

```
        25     8     3    14    19

         2    13    18     9     4

         7    24     1    20    15

        12    17    22     5    10

        23     6    11    16    21

OPEN
```