1 Alphabetically Speaking (Regionals 1986)

DATA21.txt (DATA22.txt for the second try) contains 4 lines of text. Write a program that accepts each line and reorders the letters within the sentence into alphabetical order according to the following rules:
(You may assume that your sample data is a sentence not over 40 characters long.)

Only the letters from A-Z are affected.

The sorted output reflects the format of the original sentence with all non-alphabetic characters in their original positions.

The output consists of the original sentence displayed on the screen with the output directly below it.

Sample Input

```
THE PRICE OF BREAD IS $1.25 PER POUND
THE LICENCE PLATE READS G76-ZA
THE EXPRESSION 4+3=7 READS: FOUR PLUS THREE IS SEVEN
3 OF 4 CHILDREN DROP OUT OF SCHOOL
```

Sample Output

```
THE PRICE OF BREAD IS $1.25 PER POUND
ABC DDEEE EF HIINO OP $1.25 PPR RRSTU

THE LICENCE PLATE READS G76-ZA3
AAA CCDEEEE EGHIL LNPRS T76-TZ3

THE EXPRESSION 4+3=7 READS: FOUR PLUS THREE IS SEVEN
ADE EEEEEEEFHH 4+3=7 IILNN: OOPP RRRR SSSSS ST TUUVX

3 OF 4 CHILDREN DROP OUT OF SCHOOL
3 CC 4 DDEFFHHI LLNO OOO OO PRRSTU
```

2 Diamond Design (Regionals 1986)

The data for this program will be a single word of not more than 9 letters. The output for the program will be a diamond-shaped array of letters centered on the screen. The word radiates up, down, left and right to spell the word. In addition while moving up, down, left or right, you can make one right-angled turn and still spell out the entire word. Notice that the last letter of the word, in each case, surrounds the outer edge of the array.

DATA21.txt (DATA22.txt for the second try) contains 5 words on 5 separate lines, as shown in the sample input. Use input controls to display each diamond on the screen in turn.

Sample Input

SCHOOL
DIAMOND
WEDNESDAY
LOTUS
X

Sample Output

```
     L              D                 Y               S         X
    LOL            DND               YAY             SUS
   LOOOL          DNOND             YADAY           SUTUS
  LOOHOOL        DNOMOND           YADSDAY         SUTOTUS
 LOOHCHOOL     DNOMAMOND         YADSESDAY       SUTOLOTUS
LOOHCSCHOOL   DNOMAIAMOND       YADSENESDAY       SUTOTUS
 LOOHCHOOL   DNOMAIDIAMOND     YADSENDNESDAY       SUTUS
  LOOHOOL     DNOMAIAMOND     YADSENDEDNESDAY       SUS
   LOOOL       DNOMAMOND     YADSENDEWEDNESDAY       S
    LOL         DNOMOND       YADSENDEDNESDAY
     L           DNOND         YADSENDNESDAY
                  DND           YADSENESDAY
                   D             YADSESDAY
                                  YADSDAY
                                   YADAY
                                    YAY
                                     Y
```

3 Bridge Hand Display (Regionals 1986)

In this program, 52 ordered integers represent the 52 cards of a deck of playing cards. The following table shows the value of each card.

	A	2	3	4	5	6	7	8	9	T	J	Q	K
C	1	2	3	4	5	6	7	8	9	10	11	12	13
D	14	15	16	17	18	19	20	21	22	23	24	25	26
H	27	28	29	30	31	32	33	34	35	36	37	38	39
S	40	41	42	43	44	45	46	47	48	49	50	51	52

The suits are Clubs, Diamonds, Hearts and Spades, denoted in the first column of the above table by C, D, H, and S. Individual cards names are displayed across the top of the table (A = Ace, J = Jack, Q = Queen, K = King, T = ten). For example, card 40 is the ace of spades and card 23 is the ten of diamonds.

In the game of Bridge, all 52 cards are dealt one at a time into four groups. or hands, denoted as WEST, NORTH, EAST and SOUTH. Each player's hand consists of 13 cards. You must display each player's hand sorted into suits. Each suit must be arranged from highest to lowest value. The ace of a suit has the highest value, followed by king, queen, jack, ten, 9, 8, ... ,3, 2. The suits are displayed in order: Spades, Hearts, Diamonds and Clubs, labeled as S:, H:, D:, and C:, as shown below. After each label, the cards are displayed using single characters from the above table, and are separated by a single space. If any suit in a hand is a void (meaning that there are no cards in that suit) the suit label should be followed by a space and three dashes.

SAMPLE DATA FILE (each comma represents a new line):

42, 30, 11, 48, 17, 19, 14, 31, 45, 4, 5, 6, 46, 2, 36, 3, 18, 16, 24, 40, 47, 29, 51, 32, 20, 28, 35, 8, 37, 10, 23, 12, 50, 21, 49, 15, 52, 9, 34, 33, 25, 39, 44, 22, 27, 38, 43, 1, 7, 26, 41, 13,

SAMPLE OUTPUT (on a cleared screen)

WEST:
S: K J 8 7 6 3
H: A J
D: Q 7 5 4
C: 7

EAST:
S: Q T 5 4 2
H: T 9 8
D: A J T
C: J

NORTH:
S: ---
H: K Q 4 3 2
D: K 8 6 3
C: T 9 4 2

SOUTH:
S: A 9
H: 7 6 5
D: 9 2
C: A K Q 8 6 3

4 Word Search (Regionals 1986)

In this program, you must find words in a word search array. DATA31.txt (DATA32.txt for the second try) will consist of N, a positive integer between 4 and 8 inclusive, N strings containing N letters each to make up the word search array, and **four** words to search for.

For each word found, find the starting and ending coordinates of its position in the grid. the upper left corner of the grid is (1,1), the lower left corner is (N,1), the upper right corner is (1,N), and the lower right corner is (N,N).

Words start anywhere in the grid, then are spelled out up, down, left, right or along any diagonal of the grid.

SAMPLE DATA FILE:

```
4
PIND
EFUN
ENRE
KILL
PEEK
DUNK
ILL
RUN
```

P	I	N	D
E	F	U	N
E	N	R	E
K	I	L	L

SAMPLE OUTPUT (on a cleared screen):

```
          START       END
  PEEK    (1,1)       (4,1)
  DUNK    (1,4)       (4,1)
  ILL     (4,2)       (4,4)
  RUN     (3,3)       (1,3)
```

5 Decoding (Finals 1986)

You have intercepted a coded message. Your agent has been able to determine one of the actual words in the message.

You have been able to determine that the secret code works as follows: Only the upper case alphabet and the blank character are used. There is a secret code number, N that is changed every day. It is an integer from 1 and 27, including 1 and 27. To encode the first letter of the message, imagine two alphabets, the second shifted forwards N letters. For example, if N were 7, the shifted alphabet would appear as follows:

To

```
Alphabet         " ABCDEFGHIJKLMNOPQRSTUVWQYZ"
Shifted Alphabet "TUVWXYZ ABCDEFGHIJKLMNOPQRS"
```

encode the second letter, the shifted alphabet is shifted N more letters. If N = 7 and the second letter of the message was an 'A' it would then be encoded as the letter 'N'. The shifted alphabet should be shifted another N letters forwards before encoding each successive letter.

Data41.txt (Data42.txt for the second try) contains 4 coded message of 80 characters or less in length that has been encoded using a secret code number, each followed on a separate line one of the words in the original message, as revealed to you by one of your secret agents, for a total of 10 lines. You must figure out the secret code number and the original message, as in the sample data below.

SAMPLE INPUT:

JYSXCJNTLFKQXAVF IZ ORSQXJL
WALL
ZCMGBYRMUDYTPIHSM SIGINAKIFSUL
GREAT
XD K KDVBWXMUYBEDAJTXUCPHELHSHSLCJDEUBFOCH
KINGS
NJ BELJVKBBRASKHKYMBOSYAJIHPMFRHMEU
COULD

SAMPLE OUTPUT

The secret code number is 25
HUMPTY DUMPTY SAT ON A WALL
The secret code number is 9
HUMPTY DUMPTY HAD A GREAT FALL
The secret code number is 4
ALL THE KINGS HORSES AND ALL THE KINGS MEN
The secret code number is 16
COULD NOT PUT HUMPTY TOGETHER AGAIN

6 Shuffling the Deck (Finals 1986)

The cards in a pack of 52 cards are in a precise order, starting with the top card, the Ace of Spades, then the 2, 3, 4, 5, 6, 7, 8, 9, T, J, Q, and finally the King of Spades. The other suits follow in order: Hearts, Diamonds and Clubs.

They are shuffled in a special way. The first data statement tells you the number of cards, N, that are removed from the exact centre of the deck (N must be an even number) and placed on top of the deck. Another N cards are then removed from the exact centre of the deck and placed on the bottom of the deck.

The second data item tells you how many times this operation is performed. The third data tells you how many cards to deal to each player, one at a time, in order, around the table, in the usual way, after the shuffling is completed.

The players' names are Al, Bob, Carla and Don. Al gets the first card, Bob the second, etc.

Print out the cards in each person's hand, sorted from lowest value to highest value. The highest cards in the deck are, in order, the Ace of Spades, Hearts, Diamonds and then Clubs. The next highest cards are the Kings of Spades, Hearts, Diamonds and Clubs, and so on. The lowest card in the deck is the 2 of Clubs.

Sample Input

```
12
4
6
```

Sample Output (on a cleared screen):

```
Al:    3 D  4 H  6 C  T C  K S  A S
Bob:   2 S  4 D  5 H  7 C  J C  A H
Carla: 2 H  3 S  5 D  6 H  8 C  Q C
Don:   3 H  4 S  6 D  7 H  9 C  K C
```

7 Spelling Checker (Finals 1986)

In this problem, the DATA31.txt (DATA32.txt for the second try) contains 8 pairs of words. The first word in each pair is a word spelled correctly. The second word in each pair is the word to be tested for spelling accuracy. It may or may not be spelled correctly.

The output for each pair will be a number, either -1, 0, or 1.

If the two words match perfectly, the output will be a 1.
If there is no match, the output will be a 0

If there is a probable match, the output will be a -1. A probable match means that there is an error, but the two words are close enough, as defined by one, and only one, of the following conditions:

- One of the letters does not match, but there is the same number of letters.
- One letter is missing, but all the matching letters are in order.
- There is one extra letter, but the rest of the letters are in order.
- Two, and only two, letters are swapped, but the rest of the letters are in order.

Print the two words together with its appropriate number in three columns as shown in the Sample Output below.

Sample Input

```
Contest
Contest
Program
Progran
Carpet
capet
pitcher
pitscher
framework
fromewark
mother
moethre
carport
acrpor
```

Sample Output (on a cleared screen):

```
Correct        Guess         Result
contest        contest         1
program        progran        -1
carpet         capet          -1
pitcher        pitscher       -1
framework      fromewark      -1
mother         moethre         0
carport        acrpor          0
```

8 Evaluator (Finals 1986)

In this program you must read a single character string, containing an expression to be evaluated. The following describes the restrictions and characteristics of this data string.

1) The string will contain only integers, spaces, the operations +, -, *, and / and parentheses (brackets)
2) No two operation symbols will be used without either an integer number or an expression in parentheses between them.
3) All operations will be binary, that is, operations will always be performed with two values on either side of the operation. the minus sign will not be used as the "negative" of a value or expression.
4) One or more spaces may occur anywhere in the string and have no effect on the order of operations or the value of the expression.
5) Sets of parentheses (brackets) may be used and may occur within other sets of parentheses. There will never be more than five nested levels of parentheses in the expression. the operations in the innermost parentheses must be performed first.
6) The standard order of operations must be used to evaluate the expression. The output, on a cleared screen, will be the original data string, followed by the correct value of the expression on the next line. A non integer result must be displayed rounded to two decimal places.

DATA41.txt (DATA42.txt for the second try) contains four lines each containing an expression as stated above. Print each expression with its evaluation as shown in the sample output.

Sample Input

```
5
12 * 9
(4+ 12) / (9 - 13) + 3
2 * (31*5 - 4)+ (3*2) -( 4 * (7-8)) +2
```

Sample Output (on a cleared screen):

```
5 = 5
12 * 9 = 108
(4+ 12) / (9 - 13) + 3 = -1
2 * (31*5 - 4)+ (3*2) -( 4 * (7-8)) +2 = 314
```

9 Equations (Regionals 1987)

DATA11.txt (DATA1.2.txt for the second try) contain four linear equations in x and y on four lines. They are written in standard math notation, not the way they would be written for use in a computer program. Parentheses (brackets) and blank spaces are not allowed. Only integers or terms with x and y are allowed. You must rewrite the equations, collecting all the like terms and reducing the coefficients to their lowest terms.

The output should show both the original equations and the final equations written in standard mathematical notation using the format: Ax+By+C=0

Sample Input

```
3x+4y=10
x+2y-19=y-5x+29-3y+2x-3+4y
2x+7=5-4y+2x
38x-14y-6x-35-7-x+4y
```

Sample Output

```
3x+4y=10
3x+4y-10=0

x+2y-19=y-5x+29-3y+2x-3+4y
4x-45=0

2x+7=5-4y+2x
2y+1=0

38x-14y-6x-35-7-x+4y
11x-6y-14=0
```

10 Integers (Regionals 1987)

DATA21.txt (DATA22.txt for the second try) contains 4 integers. N, less than 100. These numbers represent the number of days rent owed to King Chiplit by his subjects who must pay the rent as follows:

$1.00 on the first day,
$2.00 on the second day,
$4.00 on the third day,
$8.00 on the fourth day,

And so on, doubling the amount each day up to and including the Nth day. You must calculate exactly the total number of dollars paid over the N days, including all its digits. The output should be displayed on a cleared screen. The number of days should be followed by the total rent.

Sample Input

```
6
18
57
99
```

Sample Output

```
6 days rent is $63
18 days rent is $262143
57 days rent is $144115188075855871
99 days rent is $633825300114114700748351602687
```

11 Straight Line (Regionals 1987)

Imagine x and y axes on a screen completely filled with asterisks (*). The x-axis is horizontal and the y-axis is vertical, as usual. The asterisk in the lower left corner of the screen represents the point (0,0). If there are 25 rows of 40 columns on the screen the upper right asterisk on the screen would be the point (39,24).

First clear the screen then place a plus sign (+) in the position (0,0).

DATA31.txt (DATA32.txt for the second try) contains four lines containing 4 integers between 1 and 40 inclusive, representing two ordered pairs, the starting and end points of a straight line. The slope of this line is greater than or equal to zero. Plot these points on the screen using asterisks (*). Now draw the best straight line possible between the two points using asterisks and the following rules:

Situation 1: If the slope of the line is less than 1, There must be one asterisk in each text column between the two points.

If the line does not pass exactly through the centre of an asterisk, the nearest asterisk should be plotted, either the one directly above it or the one directly below it. The distance between the asterisks and the line is to be measured vertically. If the two asterisks are the same distance from the line, the lower one should be plotted.

Situation 2: If the slope of the line is more than 1, There must be one asterisk in each text row between the two points.

If the line does not pass exactly through the centre of an asterisk, the nearest asterisk should be plotted, either the one directly to the left of it or the one directly to the right of it. the distance between the asterisks and the line is to be measured horizontally. If the two asterisks are the same distance from the line, the one on the left should be plotted.

Situation 3: If the slope of the line is exactly 1, then the line passes directly through the centres of a line of asterisks, one per row and column between the two points.

Note: the equation for the straight line between the two points (x1, y1) and (x2, y2) is:

$$y - y1 = m(x - x1)$$

where the slope, m, is found from the two point by:

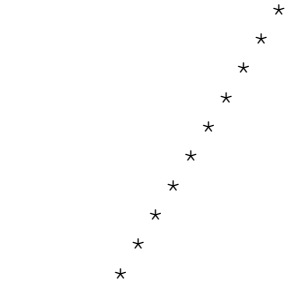
$$\frac{y2 - y1}{x2 - x1}$$

Sample Input

```
1 1 10 5
3 3 12 12
3 10 20 10
10 1 12 10
```

Sample Output

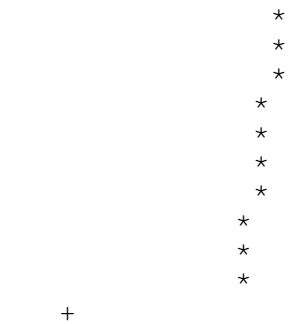
```
          **
        **
      **
    **
  **
+
press any key to continue..
```



```
+
press any key to continue..
```

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

```
+
press any key to continue..
```



```
+
press any key to continue..
```

12 Crossword (Regionals 1987)

DATA41.txt (DATA42.txt for the second try) consists of 10 words. Arrange these words to make a crossword puzzle. Words can be arranged either horizontally, reading from left to right, or vertically, reading from top to bottom. There must be a blank space at the beginning and end of each word. There must be spaces between adjacent words as in a regular crossword puzzle.

The output should be displayed on a cleared screen. The word list should appear on the left of the screen, with the "across" words listed first, in alphabetical order, followed by the "down words, also in alphabetical order.

Sample Input

```
canada
computer
contest
thought
program
algorithm
plotter
careful
thinker
print
```

Sample Output

```
across                careful
------                o
algorithm          canada
careful               t   p
canada          plotter   r
plotter         r     s   o
thinker         i     thought
thought         n         r
                thinker   algorithm
down                      m
----
contest
print
program
```

13 Poker (Finals 1987)

Four people are playing poker. The players' names are Al, Bob, Carol and Diane. Each gets five cards.
The values of the cards are 2, 3, 4, 5, 6, 7, 8, 9, T, J, Q, K and A. The four suits are Spades (S), Hearts (H), Diamonds (D) and Clubs (C). Aces (A) are considered to be the high cards in this game.
Hands are evaluated in poker according to the arrangements and combinations of cards in the hand.

Poker Rules:

PAIR
A pair of cards with the same value, such as a pair of 5's.

TWO PAIR
The next best hand is two pairs, such as a pair of 4's and a pair of 10's

THREE OF A KIND
The next best hand is three of a kind, such as three 4's

STRAIGHT
The next best hand is called a straight. It consists of five cards in consecutive numerical order, though not all of the same suit.

FLUSH
The next best hand is called a flush. It consists of five cards all of the same suit, as in 5 diamonds or 5 spades.

FULL HOUSE
The next highest hand is called a full house. It consists of three of a kind plus two of a kind. A sample hand would be three 8's and two 3's.

FOUR OF A KIND
The next highest hand is four of a kind, as in four 3's

STRAIGHT FLUSH
The next highest hand is called a straight flush. It consists of five cards in consecutive numerical order, all the same suit. A sample would be the 2, 3, 4, 5, 6 of clubs.

ROYAL FLUSH
The best hand in poker is called a Royal Flush. It is a straight flush that includes the Ace. A sample Royal Flush would be the A, K, Q, J, 10 of spades.

NOTHING
None of the above

DATA11.txt (DATA12.txt for the second try) contains four lines of data. Each line contains one player's name followed by five cards, as shown in the sample input. For each person, identify his or her hand.

Sample Input

```
AL 9S TH JC QC KD
BOB AH AS 3C 5D JH
DIANE 5H 7H KH 3H 9H
CAROL 5S 2S 2H 2D 5C
```

Sample Output

```
AL         STRAIGHT
BOB        PAIR
DIANE      FLUSH
CAROL      FULL HOUSE
```

14 Life (Finals 1987)

Life is a game that simulates life and death in an imaginary bacteria colony. The bacteria are represented by asterisks (*) that live in the character locations on the computer screen. Each bacteria can have as many as 8 neighbouring bacteria, one in each of the adjoining character locations.
The game is divided into "generations". In each generation, bacteria live, died or are born depending on their number of neighbours and certain preset rules. The version of LIFE that is supposed to be running is determined by a three digit number following the word "LIFE" in the last data statement.

Rules for L I F E x y z

- Bacteria with fewer than x neighbours die from loneliness.
- Bacteria with more than y neighbours die from overpopulation.
- A bacteria is born in every empty square with exactly z neighbours.

At the start of each move, evaluate each of the squares on the screen. Decide whether a bacteria should live or die. Determine in which squares births take place. Only when all the decisions have been made should you actually remove bacteria or add new ones.

DATA21.txt (DATA22.txt for the second try) contains 8 lines of 8 characters, either a star (*) or a dot (.). The dots represent empty spaces and the stars represent one bacterium each. Place this arrangement in the middle of a 40x40 grid on the screen to start, replacing the dots with blank spaces. This is the first generation. The 9th line of the data file contains the word LIFE followed by three digits representing x,y,z.
Display the 2nd, 3rd, 4th and 5th generation in succession under input control

Sample Input:

```
........
........
...**...
..****..
..****..
...**...
........
........
LIFE243
```

Sample Output

```
2  ****
   *  *
   *  *
   ****

3   **
   ****
  **  **
  **  **
   ****
    **

4  ****
  **  **
  *    *
  *    *
  **  **
   ****

5   **
  ******
  **  **
 **    **
 **    **
  **  **
  ******
    **
```

15 Parity (Finals 1987)

Data is sent by modem in binary. If seven digits or "bits" are used for each character, numbers from 0 to 127 can be sent. In binary notation, these are the numbers from 0000000 to 1111111. The letter "B" is normally sent as the number 66 (1000010 in binary). "A" is 65, and "C" is 67. A blank space is sent as the number 32 (0100000 in binary).
One way to detect transmission errors is to add a bit to each character as it is sent, called the parity bit. This bit becomes the eight bit or "high" bit of each number. It can either be a 1 or a 0.

Data is sent using either "even" or "odd" parity. If you are using even parity, the total number of 1's including the parity bit, must be even. The computer must look at the number to be sent, and decide whether the parity bit is to be "set" (made equal to 1) or not (made equal to 0).

If you are using "odd" parity, the parity bit is set, or not, to make the total number of bits set in the number odd.
Examples:

	decimal	binary	even parity	odd parity
A	65	1000001	01000001	11000001
B	66	1000010	01000010	11000010
C	67	1000011	11000011	01000011
blank	32	0100000	10100000	00100000

DATA31.txt (DATA32.txt for the second try) contains 4 lines containing a word in capital letters and the word “EVEN” or “ODD” as in the sample.
For each word on a blank screen with its binary equivalent using the indicated parity as shown in the sample. Proceed from one word to the next under input control.

Sample Input

```
CAT EVEN
DOG ODD
HORSE ODD
CHICKEN EVEN
```

Sample Output

```
EVEN PARITY

C     11000011
A     01000001
T     11010100

Press any key to continue..
```

```
ODD PARITY

D     11000100
O     01001111
G     11000111

Press any key to continue..
```

```
ODD PARITY

H     11001000
O     01001111
R     01010010
S     11010011
E     01000101

Press any key to continue..
```

```
EVEN PARITY

C     11000011
H     01001000
I     11001001
C     11000011
K     01001011
E     11000101
N     01001110

Press any key to continue..
```

16 Sorting (Finals 1987)

Sorting usually follows the standard alphabet, with digits less than any letter. Here the letters and digits are jumbled in a special way. Your task it so sort a set of “words” according to the jumbled “alphabet”.

DATA41.txt (DATA42.txt for the second try) contains a jumbled order of letters, digits and the space character. Following this are 12 “words” and phrases, each on a separate line, to be sorted according to the new jumbled “alphabet”

Print out the sorted list together with the place number as shown in the sample. You will loose 10 marks for each word not associated with the correct place number.

Sample Input

```
qwertyp4s0vb1 7n3uc2il9m6k5dxj8azfhgo
able
amble
awful
a4n 9w9
plonk
732
73295
blob
thread
ablative
quirk
quark
```

Sample Output

```
 1. quirk
 2. quark
 3. thread
 4. plonk
 5. blob
 6. 732
 7. 73295
 8. awful
 9. a4n 9w9
10.able
11.ablative
12.amble
```

17 Boxes, Little Boxes (Regionals 1988)

DATA11.txt (DATA12.txt for the second try) contains an 4 groups of 4 integers, each group on a separate line, as shown in the sample input. The first two values are the row-column location of the top left corner of a rectangle. the second two values are the row-column position of the bottom right corner. Screen rows are numbered 0 through 20. Columns are numbered 0 through 40. First, on a cleared screen, display axes 0 to 40 and 0 to 20 (0123456...) as illustrated below. (the printed example is incomplete to conserve space). For each group of four values draw a rectangle on the screen. The corners of the rectangle are to be labeled with letters of the alphabet. the first rectangle is labeled with A's, the second with B's and so on. The horizontal sides are drawn with hyphens. The vertical sides are drawn with either exclamation points (!) or piping symbols (|).
For example: For the numbers 3 4 8 9:

```
 012345678901234567...
0
1
2
3    A----A
4    |    |
5    |    |
6    |    |
7    |    |
8    A----A
9
```

Each succeeding rectangle must be drawn over existing rectangles (if, of course, the coordinates require it). Any lines covered be the content of the new rectangle must be erased. If a line of the new rectangle crosses a line of an existing rectangle the crossings are marked with plus (+) signs. If a lettered corner and a plus sigh conflict, the plus sign takes precedence. If there is a conflict between letters, the newest letter takes precedence.

Sample Input

```
4 3 9 8
4 5 12 9
10 6 17 7
7 1 7 6
```

Sample Output

```
 012345678901234567...
0
1       D
2       |
3    A--+-A
4    |B-+-+--B
5    || |    |
6    || D  C-+----C
7    ||    C-+----C
8    A+      |
9     B------B
0
```

18 Pretty Printer (Regionals 1988)

Write a pretty printer program for structured BASIC programs saved in ASCII format. A pretty printer program will read a program and indent the code according to the following rules:

- all lines between LOOP and ENDLOOP are indented 2 spaces;
- all lines between IF and ELSEIF and ELSE and ENDIF are indented 2 spaces
- all lines between FOR and NEXT
- all lines between PROC and ENDPROC
- all lines between OPEN and CLOSE
- multiple spaces outside of quotation marks are reduced to a single space;
- equal, plus, and minus signs, asterisks and slashes are preceded and succeeded with a single space;
- commas outside of quotation marks are succeeded with a single space;
- indents are accumulative: If a LOOP contains an IF then the contents of the IF indent a further 2 spaces for a total of four spaces.

DATA21.txt (DATA22.txt for the second try) contains a BASIC program of up to 40 lines. Format the program according to the rules stated and display the program on the screen.

Sample Input

```
FOR I=1 TO 10
 LOOP
 PRINT              I
          OPEN#2,"TESTER",APPEND
   IF I=2 OR I=4
   PRINT,I,"HI THERE"
  ELSE
         PRINT,"NO WAY THIS DAY     -HOZEE-."
         ENDIF
 CLOSE#2
             ENDLOOP
               NEXT I
```

Sample Output

```
FOR I = 1 TO 10
   LOOP
     PRINT I
     OPEN#2, "TESTER", APPEND
       IF I = 2 OR I = 4
         PRINT, I, "HI THERE"
       ELSE
         PRINT, "NO WAY THIS DAY     -HOZEE-."
       ENDIF
     CLOSE#2
   ENDLOOP
 NEXT I
```

19 Histogram (Regionals 1988)

Write a program that will read data from the file DATA31.txt (DATA32.txt for the second try). Print a vertical histogram (bar graph) representing the occurrences of each letter of the alphabet (A-Z and a-z). Capitals and lower case are equivalent.. Characters other than those in the alphabet are to be ignored.

Output your result on a 26x26 frame as shown in the sample output. (No letter will occur more frequently than 20 times)

Sample Input

```
The quick, brown, fox jumped over the lazy dog. This, is an
example of how to test your histogram program.
```

Sample Output

```
*       * *             *
*       * *   *         *          *
*   * * * * * * *     * * * * *   * * * *   * * *
* * * * * * * * * * * * * * * * * * * * * * * * *
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
```

20 Visible Insertion Sort (Regionals 1988)

The algorithm for an insertion sort works like this: Suppose the list to be sorted is 1, 3, 4, 5, 6, 2, 11. Comparisons from the left end of the list indicated that the items are in order, except for 2. The 2 is copied down below the list:

```
1  3  4  5  6  2  11
```

Then it is compared to each item to its left and moved under it until the correct position if found (to the left of the 3):

```
            1  3  4  5  6     11
                           2

            1  3  4  5  6     11
                        2

            1  3  4  5  6     11
                     2

            1  3  4  5  6     11
                  2

            1  3  4  5  6     11
               2
```

The 3 4 5 6 are all moved down the list one position:

```
            1     3  4  5  6  11
               2
```

The 2 is copied into the space previously occupied by 3.

```
            1  2  3  4  5  6  11
```

This procedure is repeated through the entire list.

DATA41.txt (DATA42.txt for the second try) contains two lines of data: The first line contains N<20, the number of integers to follow. The next line contains the N integers separated by a space. The integers are between –9 and 99.

Write a program that will display and sort the numbers as shown above, by performing an animated insertion sort on that data showing the movements of each out-of-order number to a position directly below its original spot in the list, and then moving left to its correct position. Show the shifting of all necessary values to the front, and the movement of the out-of-order number up into its correct position into the list. (See the example printed above.) Enough delay must be introduced into the movement that the appearance of animation results.

Sample Input

10
9 2 5 1 6 11 –3 8 2 10

21 Postfix Notation (Finals 1988)

In 1951, the Polish mathematician Jan Lukasiewicz devised a parenthesis-free notation which has become known as Polish Notation or Reverse Polish Notation (RPN) or Postfix Notation. In postfix notation, the operators follow the operands:

```
Algebra:   2 + 3              Postfix:    2 3 +
```

Eliminating parentheses is possible because each operator requires a fixed number of operands.
Thus if + is ordinary addition, we expect two operands (e.g., `2 + 3` or `2 3 +`).

```
Algebra    4(x - 2*y)         Postfix:    x 2 y * - + 4 *
           35 - 12/b                      35 12 b / - +
           (35 + 4 * a) + 3               35 4 a * + 3 +
```

The minus sign causes problems since it could represent either subtraction (2 operands) or negation (one operand). The problem is solved by agreeing that the minus sign always represents negation and thus
subtraction is always `a + (-b)` which in postfix is `a b - +`

Write a program that will convert an expression in DATA from postfix notation to algebraic notation with parentheses. the program will involve only +, -, *, / and ^. Note that in the DATA statement, the space is used as a separator. Extraneous parentheses in your final solution will not count against you.

Sample Input

```
a b c + *
5 6 - + 7 ^ 13 14 + - *
a 2 ^ b 2 ^ + 13 * 2 3 ^ /
45 - x 2 ^ *
```

Sample Output

```
a * (b + c)
(5 + (-6))^7 * (13 + (-14))
(a ^ 2 + b ^ 2) * 13 / 2 ^ 3
-45 * x^2
```

22 Marquee (Finals 1988)

Write a program that uses the bit pattern in the character generator ROM to define the structure for characters in a moving marquee on the screen. Each 8x8 bit character is defined by 8 numbers, one per row of bits. Each number represents a bit pattern and is the sum of the values of the bits which are ON in that row of the character.

For example, The letter “A” is defined by the eight numbers: 120 204 204 252 204 204 204 0. The letter is then created by first converting each number to binary and then turning on or printing those bits: For the A-shape use the letter A as shown. For the B-shape similarly use the letter B, and so on.

```
120   =  %01111000          AAAA
204   =  %11001100        AA  AA
204   =  %11001100        AA  AA
252   =  %11111100        AAAAAA
204   =  %11001100        AA  AA
204   =  %11001100        AA  AA
204   =  %11001100        AA  AA
```

Your program will (a) read a message and (b) a file name from DATA21.txt (DATA22.txt for the second try). The file contains the bit patterns for characters A to Z as shown in the sample input, followed by one line containing a message to be printed. Your program must then print the message in banner letters as in the samples above. the message should start at the left side of the screen and print towards the right side putting two spaces between each letter of the message, and four spaces between each word. When the right edge of the screen is reached, the message should scroll off the the LEFT so that new letters are added to the right side. When the end of the message is reached the message should begin again without disturbing the sideways scroll. the program should be in an infinite loop at this time.

Sample Input

```
A 120 204 204 252 204 204 204 0
B 248 204 204 248 204 204 248 0
C 120 204 192 192 192 204 120 0
D 240 216 204 204 204 216 240 0
E 252 192 192 252 192 192 252 0
F 252 192 192 252 192 192 192 0
G 120 204 192 220 204 204 120 0
H 204 204 204 252 204 204 204 0
I 48 48 48 48 48 48 48 0
J 12 12 12 12 12 204 120 0
K 204 216 240 224 240 216 204 0
L 192 192 192 192 192 192 252 0
M 132 204 252 252 252 204 204 0
N 140 204 236 252 220 204 196 0
O 120 204 204 204 204 204 120 0
P 248 204 204 248 192 192 192 0
Q 120 204 204 204 220 216 116 0
R 248 204 204 248 204 204 204 0
S 248 204 192 120 12 204 248 0
T 252 48 48 48 48 48 48 0
U 204 204 204 204 204 204 120 0
V 204 204 204 204 204 120 48 0
W 204 204 204 252 252 204 72 0
X 204 204 204 48 204 204 204 0
Y 204 204 204 120 48 48 48 0
Z 252 12 24 48 96 192 252 0
GOOD LUCK
```

Sample Output:

```
 GGGG      OOOO      OOOO     DDDD         LL        UU  UU     CCCC     KK  KK
GG  GG    OO  OO    OO  OO    DD DD       LL        UU  UU    CC  CC    KK KK
GG        OO  OO    OO  OO    DD  DD      LL        UU  UU    CC        KKKK
GG GGG    OO  OO    OO  OO    DD  DD      LL        UU  UU    CC        KKK
GG  GG    OO  OO    OO  OO    DD  DD      LL        UU  UU    CC        KKKK
GG  GG    OO  OO    OO  OO    DD DD       LL        UU  UU    CC  CC    KK KK
 GGGG      OOOO      OOOO     DDDD        LLLLLL     UUUU      CCCC     KK  KK
```

23 Cross Reference (Finals 1988)

Write a program that will read a BASIC program saved as an ASCII file and create a variable cross reference table. a cross reference table lists all variables in alphabetic order along with a sorted list of the line numbers in which the variable occurs. Note that in BASIC variables may be upper or lower case, must start with a letter, may contain digits and may end with a "$" or "%".

DATA31.txt (DATA32.txt for the second try) contains the BASIC program, each new line starting with a number. Following the program is a list of reserve words. (e.g. FOR, NEXT, TO, STEP, LOOP, ENDLOOP, etc. one per line).

The output is to be presented on a cleared screen as shown in the example.

Sample Input

(Note that to save space, the input is presented in two columns. In the input file, the data of the second column is presented following the data of the first column, with no intervening blank line.)

```
1000 LET X% = 5
1010 Y = 2
1015 ALPHA$ = "Halley's Comet"
1020 LOOP
1030   FOR I = 1 TO 10
1040     Z = Y*I + X%
1050     PRINT I,
1060     PRINT X%,Y,Z,I
1070   NEXT I
1080   INPUT "ENTER A NEW NUMBER",J
1090   IF J>10 THEN QUIT
1100   X% = J
1110   Y = J*J
1120 ENDLOOP
1130 PRINT ALPHA$
1140 END
```

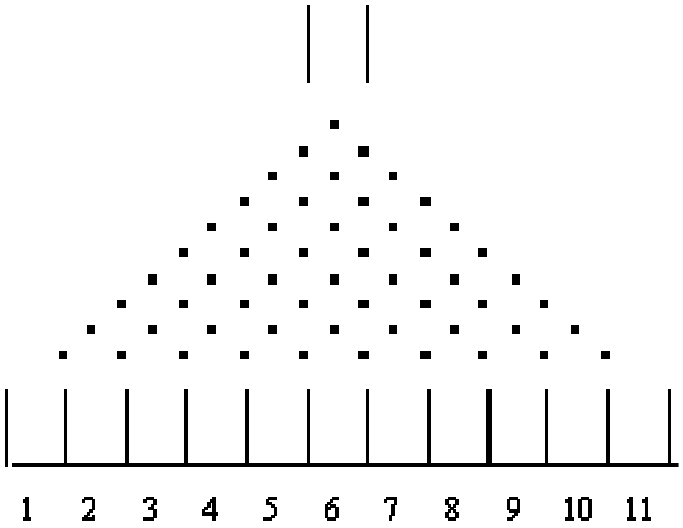
```
LET
INPUT
PRINT
LOOP
ENDLOOP
FOR
TO
NEXT
IF
THEN
QUIT
ELSE
ELSEIF
ENDIF
OPEN
CLOSE
END
```

Sample Output

```
ALPHA$          1015,1130
I               1030,1040,1060,1070
J               1080,1100,1110,
X%              1000,1040,1060,1100
Y               1010,1040,1060,1110
Z               1040,1060
```

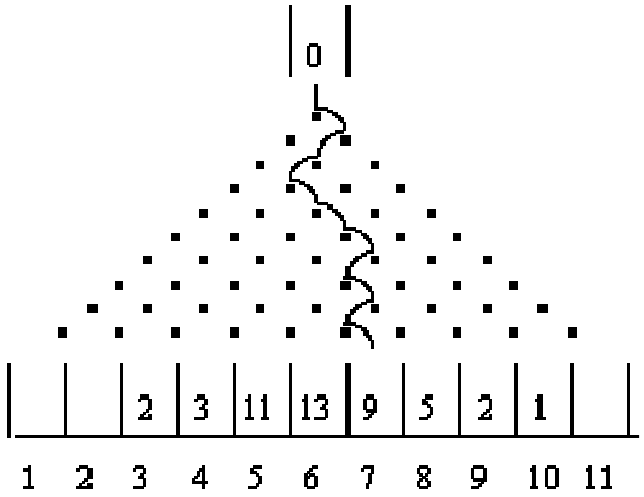
24 Pinball (Finals 1988)

Write a program that demonstrates the normal distribution by the random action of a ball rolling down a pinball machine.



As you can see from the diagram above, the pinball machine consists of 10 rows of pins which fan out to form a triangle. A ball (the small letter o or its equivalent on your system) is animated to fall through the chute at the top of the machine. as it reaches (or strikes) each pin, the probability is 50% that the ball will roll to the right or to the left and then down into the next row of pins. a sample path has been drawn in below.

Write a program which will demonstrate through animation the rolling of balls through the pinball machine. The program speed must be such that the illusion of movement is achieved. The total number of balls that fall into each pocket must be printed in the pocket. The program should roll a total of 100 balls.



25 ZAP computer (Regionals 1989)

The ZAP computer has 99 storage locations. Each storage location is capable of holding one complete instruction or one data value. For example: Read the value A where a is equal to 1000. Read is the instruction or command and it is stored in one location of storage. Since this particular computer has 99 locations of storage, it is capable of holding only 99 instructions or 99 data items. This is a most unlikely situation, however, since we need both instructions and data in the computer at the same time. Each of the locations in the ZAP computer is addressable. The address range is from 01 (the first usable location) to 99 (the last usable location). The ZAP computer uses only integer arithmetic. Reading data or an instruction into a storage location will cause the previous contents of that location to be destroyed(destructive read-in). Writing from a location is non-destructive, allowing the same data to be written several times. Data may be stored in the system, moved from location to location and manipulated arithmetically. The ZAP computer reserves one storage location as an arithmetic "register", location 99. This does not prevent the programmer from addressing location 99 as they would any other in storage.
Caution is required at this point. Since location 99 is used to store the result of a calculation, it is advisable to consider carefully whether you wish to use this location for storing other data.

THE LANGUAGE
Each instruction must be written in a prescribed format and placed in sequential locations in storage. All machine language instructions or commands are numeric and are 6 or 8 digits long. (Only the HALT instruction takes 4 digits.) An instruction consists of:

Instruction Address: This instruction address is a 2 digit code that indicates where the instruction will be placed in storage. This will be the first two digits. Program execution begins at storage location 01, therefore, the first instruction must be placed in location 01 of storage. The instructions are then processed sequentially. This means that all instructions must be stored in ascending sequence and in successive storage locations.

Operation Code: The operation code is the command that directs the computer to do what the programmer wishes. It consists of 2 digits and will be the 3rd and 4th digits.

X address and Y address: The ZAP computer is a 2 address computer, allowing data to be moved from one location to another. These 2 addresses are referred to as the X address and the Y address and identify the location where something is stored. The X or Y address generally refers to data, but may be the address of an instruction. the X address will be the 5th and 6th digits and the Y address will be the 7th and 8th digits.

Data: The value 99 signifies the end of the program and that what will follow will be data. the value 99 occupies the 1st and 2nd digits. all data must be integers. Only one data value will be on each line. -999 will be used to signify the end of data.

Below is a list of the ZAP operation codes:

Operation Code	Instruction
01	**Read** data and store in X address
02	**Write** data from X address
03	**Move** data from X address to Y address
09	**Add** data in X address to data in Y address and store the sum in 99
10	**Subtract** data in X from data in Y and store the difference in 99
11	**Multiply** data in X by data in Y and store the product in 99
12	**Divide** data in X by data in Y and store the quotient in 99
13	**Halt** execution of program

The problem:

Write a program that will read in a ZAP program from the data file DATA11.txt (DATA12.txt for the second try) and execute it. You may assume that the ZAP program is error free. The output from the program should be printed on a clear screen.
Note: Any answer produced by an integer arithmetic computer is converted to the largest integer value less than or equal to the actual answer.

Sample Input

Note that the explanation on the right is not part of the data file)

```
010130          read in A (8--> location 30)
020131          read in B (3--> location 31)
030230          output A
040231          output B
05113030        calculate A*A  (A*A = 64)
06099931        add B              ( A*A+B=67)
07039932        save the sum into C (67--> location 32)
08103130        calculate A-B (A-B = 5)
09123299        divide C by the result (67/5 = 13)
100299          output result
1113            halt operation
99              end of program and beginning of data
8               the value of A
3               the value of B
-999            signifies end of data file.
```

Sample Output

```
8
3
13
```

26 Tic - Tac - Toe (Regionals 1989)

Given the placement of X's and O's in a game of Tic-tac-toe, determine and output the "best" next move.
Obtain your input for the program from the data file called DATA21.txt (DATA22.txt for the second try). The file will consist of 4 game boards. Each game board is a nine character string on one line, representing the state of each position of the game board.

1	2	3
4	5	6
7	8	9

The first character describes the upper left corner position; the fifth character describes the middle position; and so on. Each position on the game board contains an X, or an O, or is empty. An empty position on the game board is represented by an asterisk in the data string. We guarantee that each board will represent a valid game, and that it is always X's turn to move. Unlike many styles of Tic-tac-toe, we do not guarantee that X made the first move.

To determine the location of X's "best" next move, use the following prioritized list:

X should make any move which will result in a win. A win occurs when there are three markers of the same type in the same row, column or diagonal. Make any move which will block the other player from winning. That is, if there is a spot where O on his or her next turn could move in order to win, put an X in that spot to block. Fill in the lowest numbered square available, using the numbering scheme shown above. If there is move than one valid "best" next move, (i.e. X could block in two different ways or X could win in more than one way), then any such move is a valid move.

Sample Input

```
O**OXX**O
XOO*O*X*X
O*X**O***
OXO*X*XO*
```

Sample Output

```
GAME #1 BEST NEXT MOVE 7
GAME #2 BEST NEXT MOVE 4
GAME #3 BEST NEXT MOVE 7
GAME #3 BEST NEXT MOVE 6
```

27 Big Numbers (Regionals 1989)

The data file, DATA31.txt (DATA32.txt for the second try), contains four lines of data. The first is a three digit whole number, the second is a five digit whole number, the third is a thirteen digit whole number, and the fourth is a fifteen digit whole number. (012 is a valid three digit whole number.)
If the four inputs are represented by `A, B, C, D` respectively, your program should output the results of the following:

```
C + B – A
A * (B – D)
((A * B) – D) * (C – B)
(D + C + B + A) – (D * C * B * A) – (A – B – C – D)
```

Each output must appear on a separate line. There must not be any space between the digits of each result. Negative numbers must be preceded by a negative sign. Leading zeros must be suppressed.

Sample Input

```
012
54321
1234567890123
543210987654321
```

Sample Output

```
54333
651852
1234567944432
-6518531851200000
-67063081260750650456059513 8
-437152056219104837799469238338986
```

28 Binary Tree (Regionals 1989)

The expression (X = Y) * (A - B) could be represented in a binary tree as illustrated below:

```
        *
      +   -
     X Y A B
```

The root node of the tree contains the operator (*) that is evaluated last. Each subtree contains an operator and is in itself a complete expression. The subtree to the left of the root node represents the expression `(X+Y)`; and the subtree to the right of the root node contains the expression `(A-B)`. Four set of expressions are given in the data file DATA41.txt (DATA42.txt for the second try). Each expression can contain parentheses (to any level of nesting), +, -, *, /, letters and/or integer constants. There are no spaces in the expressions. Each expression is no more than 20 characters long. there are sufficient parentheses in each expression to indicate explicitly the order of operations. (i.e., the number of pairs of parentheses is one less than the number of arithmetic operators.)

On a clear screen, print out an expression with exactly one blank before and after a bracket, and one blank before and after an arithmetic operator. Below it print the binary tree of the expression, with one blank after each character. Then, when any key is pressed, print the next expression on a clear screen, and so on until all the expressions from the data file have been displayed in binary tree form.

Sample Input

```
X+Y
(X+Y)/(-65*M)
Y*(A/(B-C))
((A+(B*C))-X)/(((33+X)*(7+Y))+(Z-8))
```

Sample Output

```
X+Y
Binary Tree
 +
X Y
Press any key to continue..

(X+Y)/(-65*M)
Binary Tree
   /
 +     *
X Y -65 M
Press any key to continue..

Y*(A/(B-C))
Binary Tree
   *
 Y    /
    A    -
       B   C
Press any key to continue..

((A+(B*C))-X)/(((33+X)*(7+Y))+(Z-8))
Binary Tree
        /
      -         +
 +     X    *     -
A  *      +   +  Z 8
  B C   33 X 7 Y
Press any key to continue..
```

29 Zing Zong (Finals 1989)

The game of Zing Zong is played by two people, designated as player a and player b, on a square table. The table is 80 cm. x 80cm. and is divided into two courts by a black line on the surface of the table. Each player has a metal paddle not unlike a ping ping paddle. The players are situated at opposite ends of the table. A ball is served downward towards the table in a random direction (towards either player A's or player B's court) from a machine high above the centre of the table. An electric field keeps the ball from ever touching the black centre line.

The idea of the game is to keep the ball from touching the table.
A player gains a point if: the opponent lets the ball tough his court, or the opponent hits the ball when it is out of bounds (ie: anywhere not on his court)
A player loses a point if: He lets the ball touch the playing surface of his court, or he propels the ball into the out of bounds area and it is not touched by his opponent.
The game is over when one player reaches a score of 6.
A player keeps the ball off his court by hitting it with the paddle.
A player may hit the ball as often as he wishes before propelling it into his opponent's court.
The paddles, the playing surface and the surrounding out of bounds areas are electrified so that a computer connected to sensors can score the game. The game is scored in real time but a record is kept of all data used in the scoring. the data collected by the computer is based on a Cartesian co-ordinate grid arranged with the centre of the playing surface as the origin.
Quadrants 1 and 2 correspond to B's side and quadrants 3 and 4 correspond to A's side.
The data is recorded as a string of codes separated by a space according to the following rules:
If the ball touches the playing surface or the surface of the out-of-bound area, the point of contact is recorded. for example, if 23 -6 is recorded, it means that the ball touched the court at (23,-6).
If the ball is touched by a paddle, the specific paddle and the hit co-ordinates are recorded. The hit co-ordinates are the co-ordinates of the point directly below where the paddle touched the ball. For example, if A -8 -12 is recorded, it means that player A's paddle touched the ball above the playing surface at the point (-8, -12).
So a typical record leading up to a point for B might be as follows:

A 15 -12 B-26 34 A -16 -28 23 -4

Note that the ball hit the playing surface at (23, -4), a point on player A's side. Here is another sample:

B -10 38 A -39 -2 B -38 6 A 36 -35 B 29 23 B 6 32 B -4 18 -23 -43

The point was won by A because B propelled the ball into the out of bounds area and player A did not touch it.

DATA11.txt (DATA12.txt for the second try) contains one or more lines of data as shown in the Sample Input, representing 4 games each. Each game 's data is terminated by the letter "E".
Write a program that will process the recorded data of the 4 ZING-ZONG games and will tabulate the score point by point. If one player reaches 6 points, the game is over and a winner is declared. Your output should be similar to the output illustrated below.

Note that the sample only contains two of 4 games.

Sample Input

```
A -10 -10 B 12 12 A -40 -15 20 -6 B 29 32 A 36 -3 B -37 4 A -36 -
5 -42 6 B 13 3 A 13 -3 -16 37 -10 13 A 15 -6 B 35 42 A 6 -40 A -
10 -15 B 3 39 B -32 39 A -1 -1 B -1 2 A 38 -39 -1 42 B 10 10 A 10
-6 B -16 -1 E -27 -3 A 16 -4 B -25 34 A 39 -3 -37 -1 31 -2 B 17
32 A -23 -6 B 13 23 B 29 13 A -42 -45 A 3 -2 B 2 5 A 6 -1 -29 1 B
-38 28 -32 -37 19 -2 E
```

Sample Output

The output on a cleared screen would be similar to the following: (FIRST SCREEN)

```
POINT     ======SCORE======
          PLAYER A PLAYER B
          ======== ========
1            0        1
2            0        2
3            1        2
4            2        2
5            3        2
6            3        3
7            4        3

PRESS ANY KEY TO CONTINUE..
```

And your next screen of output would be similar to : (SECOND SCREEN)

```
POINT     ======SCORE======
          PLAYER A PLAYER B
          ======== ========
1            0        1
2            0        2
3            0        3
4            0        4
5            1        4
6            1        5
7            1        6
                   WINNER

PRESS ANY KEY TO CONTINUE..
```

30 Factoring (Finals 1989)

Given a quadratic polynomial, $Ax^2 + Bx + C$, factor it completely over the integers. The polynomials in this problem are guaranteed to factor. The coefficient **A** will be non-zero, and at least one of **B** and **C** will be non-zero. The data file DATA21.txt (DATA22.txt for the second try) will contain 4 sets of data. Each set will contain three integers, A, B, and C, corresponding to the coefficients of the polynomial $Ax^2 + Bx + C$. Each set of data in the data file is on one line and the three numbers are separated by spaces.

Your output should begin on a clear screen. Each line of output should contain the simplified factored form of each polynomial given in the data file. Embedded spaces in the output are allowed. The following prioritized set of factoring rules must be obeyed in determining your output. If there is a conflict, the lower numbered rule takes precedence.

1. Brackets must appear around binomial factors, but must not appear around monomial factors. 2x(x-4) is correct but (2x)(x-4) is incorrect.
2. Coefficients of 1 must not be printed. (x+2)(2x-1) is correct but (1x+2)(2x+1) and 19x+2)(2x+1) are both incorrect.
3. Zero must not be printed. The factor x is correct but the factor x+0 is incorrect.
4. Two operators must not appear together. (x-2) is correct, but (x+-2) and (x-+2) are both incorrect.
5. All leading coefficients of the x term with a binomial must be positive. -(x+2)(x-3) is correct but (-x-2)(x-3), -1(x+2)(x-3) and (x+2)(-x+3) are incorrect.
6. Factors must be in standard form with descending powers of x. (x+2)(x-3) is correct but (2+x)(x-3) and (x+2)(-3+x) are both incorrect.
7. ^2 must not be used. (x+1)(x+1) may not be printed as (x+1)^2
8. The order of the binomial factors is irrelevant. (x+2)(x-3) and (x-3)(x+2) are equally correct.
9. Monomial factors must come before binomial factors. x(x+3) is correct but (x+3)x is incorrect.
10. Constants must appear before variable within a single term. 2x(3x+1) is correct but x2(3x+1) and 2x(x3+1) are incorrect.

Sample Input

1 5 6
1 -1 -6
2 -3 0
6 -5 -6

Sample Output

(x+2)(x+3)
(x+2)(x-3)
x(2x-3)
(2x-3)(3x+2)

31 Round Robin (Finals 1989)

You are responsible for setting up the round robin schedule for the city tennis league. Write a program which will set up the schedule of matches with court assignment for each round of the tournament. the number of players (P<12) and the number of available courts (2 or 3) are stored on a single line in the data file DATA31.txt (DATA32.txt for the second try) separated by spaces. The file contains 4 sets of data, each on a separate line as in the Sample Input below.

Your output should be a list of (P-1) rounds so that every player plays every other player exactly once in the tournament. If P is odd, then one person will draw a bye in each round. There may be fewer courts than the number of matches in any given round, but the numbers are sufficiently small that each schedule can easily fit on one screen. it should also be noted that C will be at most 3.

Sample Input (note there are only 2 sets of data of 4 in this sample)

```
5 2
8 3
```

Sample Output

```
NUMBER OF PLAYERS: 5
NUMBER OF COURTS: 2
COURT #       1       2
ROUND 1 :   2 - 5   3 - 4   1 - BYE
ROUND 2 :   3 - 1   4 - 5   2 - BYE
ROUND 3 :   4 - 2   5 - 1   3 - BYE
ROUND 4 :   5 - 3   1 - 2   4 - BYE
ROUND 5 :   1 - 4   2 - 3   5 - BYE

Press any key to continue..

NUMBER OF PLAYERS: 8
NUMBER OF COURTS: 3
COURT #       1       2       3
ROUND 1 :   1 - 8   2 - 7   3 - 6
            4 - 5
ROUND 2 :   2 - 8   3 - 1   4 - 7
            5 - 6
ROUND 3 :   3 - 8   4 - 2   5 - 1
            6 - 7
ROUND 4 :   4 - 8   5 - 3   6 - 2
            7 - 1
ROUND 5 :   5 - 8   6 - 4   7 - 3
            1 - 2
ROUND 6 :   6 - 8   7 - 5   1 - 4
            2 - 3
ROUND 7 :   7 - 8   1 - 6   2 - 5
            3 - 4

Press any key to continue..
```

32 Long Multiplication (Finals 1989)

Given any two natural (non-zero counting) numbers, find the product showing all non-zero partial products as if the problem were done the old-fashioned "conventional" way. Four pairs of numbers are given in the data file DATA41.txt (DATA42.txt for the second try). Each pair is on a separate line and the two numbers are separated by a space.

For each set of numbers, A and B, read from the file, print `A*B`. the product will have less than 20 digits. The columns must all align properly. Partial products that have a value of zero must not be printed. The first line of dashes (see sample output below) must be exactly as long as the largest of the two numbers above it. the last line of dashes (see the sample output below) must be exactly the same length as the final product. Each output produced by the data file should be on a clear screen. the prompt: PRESS ANY KEY TO CONTINUE should appear on the screen after the product has been printed and the program should indeed wit before moving on to the next screen.

The following example should be examined carefully to help clarify the meanings of the definitions given above.

```
   1025
    103
   ----
   3075
 102500
 ------
 105575
```

The two lines between the dashed lines are referred to as partial products. The first is obtained by multiplying 1025 by 3. the second is obtained by multiplying 1025 by 1. Note the two trailing zeroes. the rightmost one is a place holder, and the next one (to the left) is the result of 1025 times zero. It would be incorrect, for purposes of this problem, to have a partial product row containing all zeroes. Trailing zeroes in the second partial product must be include as illustrated.

Sample Input

```
5281 205300
```

Sample Output

Only one sample of 4 has been included here

```
      5281
    205300
    ------
   1584300
  26405000
1056200000
----------
1084189300

PRESS ANY KEY TO CONTINUE
```

33 Buzz (Regionals 1990)

In the standard version of the math game Buzz, players begin counting and whenever a player reaches a number that has seven as a digit or is divisible by seven, he says "Buzz" instead of the number. Numbers such as seventy require the player to say "Buzz Buzz": Once because the numer contains the digit "7" and once because the number is divisible by "7". However, if one of the two properties holds more than once (for example, the number 771 contains the digit 7 twice) the player only says "Buzz" once for each property that holds.

In the ECOO version of Buzz, the following properties would cause a player to buzz:

1. if the number contains the digit 9
2. if 8 is a factor of the number
3. if the sum of the digits is a multiple of 4
4. if the digital root of the number is equal to 7
5. if the number contains d, where d is its digital root

The digital root of a number is found by adding all the digits together. If this sum is less than ten, then that number is the digital root. Otherwise, one considers the sum of the digits of the sum, and the process continues until a single digit results.
For examle, the digital root of 988 is 7, because 9+8+8=25, and 2+5=7

From the file DATA11.txt (DATA12.txt for the second try), read a series of 4 positive integers. Each number will be less than 20 000.

For each number read from the file, output the number itself if none of the properties above hold, or the word "BUZZ" followed by the digit 1-5 indicating the property that caused the buzz. the correct number of times. Note that the most times that you would "BUZZ" would be 5 times. (If each of the above properties hold)

Sample Input:

```
9479
72
1287
39
```

Sample Output:

```
9479 BUZZ 1
72   BUZZ 2
1287
39   BUZZ 1 BUZZ 3 BUZZ 5
```

34 Word Processing (Regionals 1990)

Input: Four paragraphs to analyze. Each line will contain no more than 16 characters, which must be read in as a single string
The punctuation will be periods, commas, semicolons, colons, exclamation points, question marks. Sentences are terminated by a period, exclamation mark, or question mark. The other characters will be A-Z and 0-9.

Output: For each paragraph, find the following:

1. The number of words in it. By "word" we mean the intuitive meaning of the word: a contiguous series of alphanumerics surrounded by a space, punctuation, the beginning of a line, or the end of a line. Note that spaces and/or punctuation are not to be considered as part of the word.
2. The longest word(s) in the paragraph. If there are more than one word, print them all.
3. All words containing non-distinct letters. that is, a word that contains at least one letter which is repeated in the word.
4. The paragraph reformatted with 20 characters per line. Use a sharp to indicate spaces. Output 2 spaces a the end of a sentence. Indent paragraphs by 5-spaces. All other lines begin in the first column. Do not split words between lines. Add blanks at the end of the line to fill it up to 20 characters.

Sample Input:

```
3
TWINKLE TWINKLE LITTLE STAR.
HOW I WONDER
WHAT YOU ARE.
2
UP ABOVE THE WORLD SO HIGH,
LIKE A DIAMOND IN THE SKY!
4
WHEN THE BLAZING SUN IS GONE,
WHEN HE NOTHING SHINES UPON,
THEN YOU SHOW YOUR LITTLE LIGHT,
TWINKLE, TWINKLE, ALL THE NIGHT.
4
THEN THE TRAVELLER IN THE DARK,
THANKS YOU FOR YOUR TINY SPARK!
HE COULD NOT SEE WHICH WAY TO
GO,
IF YOU DID NOT TWINKLE SO.
```

Sample Output

```
10
TWINKLE
LITTLE
    TWINKLE TWINKLE LITTLE
STAR.  HOW I WONDER WHAT YOU
ARE.

Press any key to continue..

12
DIAMOND
DIAMOND
     UP ABOVE THE WORLD SO
HIGH, LIKE A DIAMOND IN THE
SKY!

Press any key to continue..

22
BLAZING NOTHING TWINKLE
NOTHING SHINES LITTLE ALL
     WHEN THE BLAZING SUN IS
GONE, WHEN HE NOTHING SHINES
UPON, THEN YOU SHOW YOUR
LITTLE LIGHT, TWINKLE, TWINKLE,
ALL THE NIGHT.

Press any key to continue..

26
TRAVELLER
TRAVELLER SEE WHICH DID
     THEN THE TRAVELLER IN THE
DARK, THANKS YOU FOR YOUR TINY
SPARK!  HE COULD NOT SEE WHICH
WAY TO GO, IF YOU DID NOT
TWINKLE SO.

Press any key to continue..
```

35 ECOO* Bingo (Regionals 1990)

Our version of BINGO is played using the following 5 by 5 card:

E	C	O	O	*
77	37	56	87	2
40	16	75	64	34
14	21	**	19	93
47	92	58	31	52
25	23	7	80	12

A game is played by calling out ten numbers. If the number appears on the card, then it is "covered". Initially, all numbers on the card are uncovered, except for the centre square. After the 10 numbers have been called, the game ends and one of the following prizes are awarded:

$5	The four corners
$10	A horizontal row is covered
$15	a vertical column is covered
$20	A diagonal is covered
$50	Both diagonals are covered
$100	The top row and the middle column are covered

If more than one prize could be won for a card, than the prize with the highest value is chosen. If none of the winning conditions are met, then a $1 prize is given (just for trying)

Input: From a file called DATA31.txt (DATA32.txt for the second try) read 24 numbers from one line which are the numbers on the card entered column by column, starting on the top left (don't forget the free space in centre). 10 more numbers on the next line of the file are the numbers which were called out. Continue the process four times for four games.

Output: For each game print the value of the prize won.

Sample Input:

```
77,40,14,47,25,37,16,21,92,23,56,75,58,7,87,64,19,31,80,2,34,93,52,12
40,77,16,1,31,75,12,80,25,19
33,40,19,54,75,63,97,77,67,3,94,27,72,17,78,38,34,51,73,35,4,2,79,14
51,38,94,14,33,97,77,75,67,35
69,53,53,82,97,53,51,99,74,69,55,73,27,77,86,7,10,50,8,13,30,72,88,20
7,55,73,53,51,27,1,77,20,30
8,17,23,24,25,31,37,40,45,54,61,62,65,70,74,75,79,86,88,90,91,94,95,98
65,8,86,17,61,25,90,91,37,98
```

Sample Output:

```
$20
$50
$15
$5
```

36 Family Ties (Regionals 1990)

Given the female members of a family, answer some questions about the resulting family tree. DATA41.txt (DATA42.txt for the second try) will contain N, the number of relationships that will follow. Each of these lines is in one of the following formats:

```
DAUGHTER mom daughter
```

Or

```
SISTER sis1 sis2
```

The first format states that mom is the mother of daughter. The second format states that sis1 and sis2 are sisters (i.e. they both have the same mother). Each person will have a unique name; for example, "MOTHER WILMA PEBBLES" and "SISTER BETTY WILMA" both refer to the same person WILMA (we will use all capital letters for names). We guarantee that all of the necessary information will be available in the tree. We also guarantee that the data in the tree will be valid, for example, it will not say that JILL has two different mothers, or that JILL's mother and sister are the same person. All names will be single words containing only letters.

After all of the input has been read, print the answers to the following questions:

1. How many people are in the tree?
2. Who is AMY's mother?
3. Who is BETH's mother?
4. Who are CAROL's daughters (List all of them, in any order)
5. Who are DIANNE's descendants? (List all of them, in any order)

A person's descendants includes her daughters, granddaughters, great-granddauthers, and so on.

Sample Input:

```
6
DAUGHTER CAROL AMY
DAUGHTER CAROL BETH
SISTER GLENDA FAITH
SISTER DIANE ESTHER
SISTER BETH DIANE
DAUGHTER DIANE FAITH
```

Sample Output

```
7
CAROL
CAROL
AMY BETH DIANE ESTHER
GLENDA FAITH
```

37 Bilingual Calendar Generator (Regionals 1991)

Given an input of a year and month (in English or French), This program will generate and display a calendar for that month on the screen.
DATA11.txt (DATA12.txt for the second try) will contain 4 lines, each containing a 4-digit year followed by a month in either English or French. Display the monthly calendar as shown in the Sample output, one at a time under input control.

See Table A below for the English and French names for the months. Your calendar will be only as big as required. (i.e. no blank weeks!) You may use any ASCII or draw characters that your computer is capable of generating in order to draw the calendar on the screen.

Table A: Months of the Year:

janvier/January	mai/May	septembre/September
fevrier/February	juin/June	octobre/October
mars/March	juillet/July	novembre/November
avril/April	aout/August	decembre/December

Note the following formula for finding the day of the week, where d = day of the month, m the month number: Mar=1, Apr=2, etc, Jan=11, Feb=12. and y the year number, where Jan and Feb are considered the months of the previous year. E.g. for 5 Jan 1995: d=5, m=11, y=1994.
Note also that "/" stands for integer division and that % is the modulus operator.
w=0= Sunday. w=1=Monday, etc.

```
w = (d + 2*m + 3*(m+3)/5 + y + y/4 - y/100 + y/400 - 2) % 7
```

Sample Input
(only 2 of 4 lines shown)

```
1945 mai
2010 August
```

Sample Output

```
-----------------------------------
|  1945        mai/May       1945  |
-----------------------------------
|    |    |  1 |  2 |  3 |  4 |  5 |
-----------------------------------
|  6 |  7 |  8 |  9 | 10 | 11 | 12 |
-----------------------------------
| 13 | 14 | 15 | 16 | 17 | 18 | 19 |
-----------------------------------
| 20 | 21 | 22 | 23 | 24 | 25 | 26 |
-----------------------------------
| 27 | 28 | 29 | 30 | 31 |    |    |
-----------------------------------

press any key to continue..
```

```
-----------------------------------
|  2010     aout/August      2010  |
-----------------------------------
|  1 |  2 |  3 |  4 |  5 |  6 |  7 |
-----------------------------------
|  8 |  9 | 10 | 11 | 12 | 13 | 14 |
-----------------------------------
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
-----------------------------------
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
-----------------------------------
| 29 | 30 | 31 |    |    |    |    |
-----------------------------------

press any key to continue..
```

38 Bar Graph Solution (Regionals 1991)

Create a program that will display a proper bar graph on the screen. From the text file DATA21.txt (DATA22.txt for the second try), you will read data corresponding to scientific readings. The readings will be in the range 0 - 50. The values will appear one per line in the file. You should continue to read the data until the eof -9999 is read.

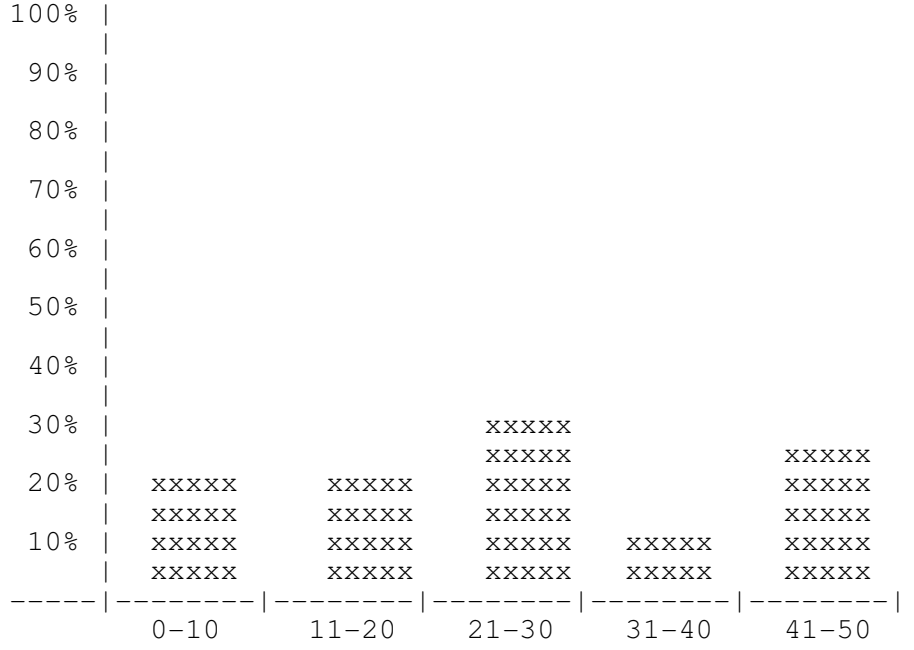
At that time, you will read one more field of information. This field will indicate how many bars will appear in your graph: n where 2<n<10. Now, your program should be able to tally the number of data in each interval and display each interval as a percentage. Sample input and output are illustrated below. The output must appear on a cleared screen and use the entire screen for the display. You may use any ASCII or draw characters that your computer is capable of generating in order to draw the graph on the screen.

Sample Input

```
30
7
31
42
17
41
27
18
25
36
0
11
38
48
46
45
47
24
25
14
2
15
```

```
25
9
3
43
42
21
42
22
20
10
5
13
30
28
4
24
-9999
5
```

Sample Output:



39 Decode this Cow, Man (Regionals 1991)

To encode a message in the "E" cypher, you take each letter of the message and advance it five places (five because 'E' is the fifth letter of the alphabet.)
Thus the message "HAVE A COW" becomes "MFAJ F HTB".
Note that the encoding rolls over (i.e. 'Z' advanced on becomes 'A')

It is possible to make this cypher much harder to decode by choosing a word key, and writing it over the mesage. For example, using the key "CAT" to encode :HAVE A COW" you would write:

```
CATC A TCA
HAVE A COW
```

Now, you encode 'H' in the 'C' code (i.e. advance it three letters so it becomes 'K', 'A' in the 'A' code (i.e. advance it one letter so it becomes 'B', 'V' in the 'T' code (i.e. advance it twenty letters so it bcomes 'P', and so forth. Thus one gets the final message:

```
CATC A TCA
HAVE A COW
KBPH B WRX
```

Thus "HAVE A COW" becomes "KBPH B WRX".

Write a program that accepts four sets of data from the file DATA31.txt (DATA32.txt for the second try). Each set is made up of the following: a (one word) keyword on one line; a message that can be multiple lines long; and the word STOP on a line by itself. The program must output in each case the code word, the original message and the encoded message, and proceed to the next set under input control.

Your program will not encode any puncuation or spaces in the original message. It may assume that all input will be in upper case.

Sample Input: (one set of input of 4)

```
SCHOOL
THIS IS A TEST MESSAGE. IT SHOULD BE
ENCODED PROPERLY.
STOP
```

Sample Output:

```
Keyword: SCHOOL

Message:
THIS IS A TEST MESSAGE. IT SHOULD BE
ENCODED PROPERLY.

Encoded Message:
MKQH XE T WMHI YXVAPVQ. BW AWDGEG JT
TZVRLTS BKRXTGXR.

Press any key to continue..
```

40 I Always Get in the Long Line (Regionals 1991)

A supermarket in Numeric City has three tellers, labeled "1", "2", and "3". When customers go to check out, they quite naturally head for the shortest line. And, in this very numeric city with an incredible desire for order, when faced with two lines of the same length, the customers always head for the teller with the lowest number! Customers in Numeric City are also very impatient about standing in line, and if standing in line, they notice a shorter line, they will move to the shorter line. Despite this impatience, cultural courtesy being what it is, it is always the last person in the longer line that will move to the shorter one.

You will read information from the file DATA41.txt (DATA42.txt for the second try) about customers at the supermarket. On the first line will be the name of the customer, on the second line will be the time that the customer is ready to begin checking out and enters a line and on the third line will be the time that the customer will spend at the teller.

Your program will simulate the activities at the supermarket starting at time 00:00. Read a customer from the file and place the customer in the correct line at the correct time. As each minute passes, you should check to see if it is appropriate for a customer to "jump line" and, if so, move the customer.

End of file will occur when you read "EOF" for a customer name. When this happens, display the following information on a cleared screen:

```
Total Customers Serviced:                    #
Total Waiting Time by All Customers:         # minutes
The following people jumped line:            XXXXXXXXXXXXXXX
                                             XXXXXXXXXXXXXXX
                                             XXXXXXXXXXXXXXX
The last customer left the supermarket at:   ##:##
```

Sample Input:

```
A
00:00
10
B
00:01
10
C
00:02
5
D
00:03
5
E
00:08
5
F
00:09
5
EOF
```

Sample Output:

```
Total Customers Serviced:                    6
Total Waiting Time by All Customers:         8 minutes
The following people jumped line:            D
The last customer left the supermarket at:   00:16
```

41 Yield (Regionals 1992)

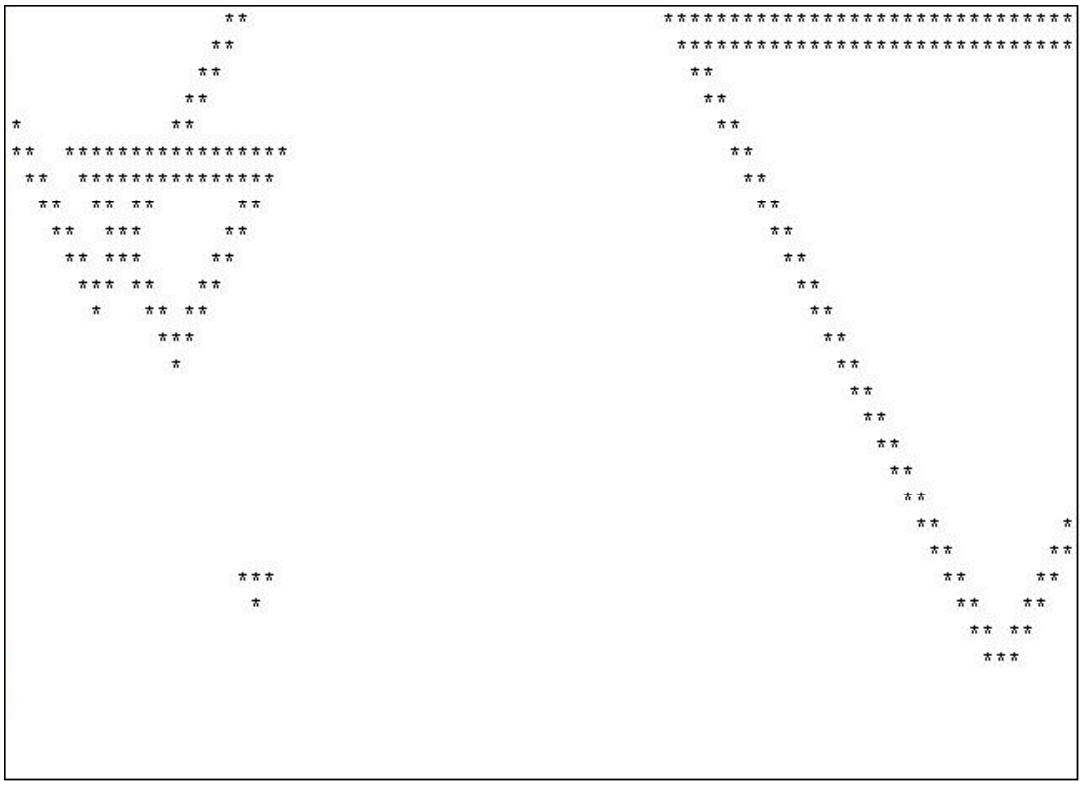
Your program will draw an inverted triangle on the screen with a border composed of two (2) asterisks. the triangle will be drawn on a 80 (wide) by 40 (high) screen at a specified position. If any part of the triangle exceeds the dimensions of the screen, that portion will not be displayed (i.e. draw only that portion of the triangle that will fit on the screen).
In order to draw the triangle, you will need to read a file called DATA11.txt from the disk. (DATA12.txt for the second try) The file will contain four sets of data, each set is composed of a vertex number (1,2 or 3) specifying one corner of the triangle; a length for the base of the triangle (which will always be an odd number); and the row and column numbers for the position of the specified corner. DO NOT CLEAR THE SCREEN BEFORE DRAWING EACH TRIANGLE.
The vertices of the triangle are numbered as follows:

```
1 ********* 2
   *       *
    *     *
     *   *
      * *
       *
       3
```

Sample Input:

```
2
17
21
6
1
51
50
1
3
31
7
12
2
3
20
22
```

Sample Output:



42 Stairs (Regionals 1992)

Write a program that will produce as output a downward stair step pattern, given two items of data: The vertical distance v (2<v<9) and the horizontal distance h (2<h<9). DATA21.txt contains 4 lines containing the two items of data as shown in the sample below. DATA22.txt is the input file for the second try.
The output should start in the top left corner of a 80 (horizontal) x 25 (vertical) screen with the first step and work down and to the right. Only complete steps should be printed out. Note that one step consists of the horizontal part followed by the vertical part.

Use the # symbol for horizontal lengths, the X for the vertical lengths and the @ at any point where the horizontal and vertical meet. Each display must follow the next under input control.

Sample Input: (Only one set of data is displayed)

```
3 5
```

Sample Output:

```
####@
    X
    @###@
        X
        @###@
            X
            @###@
                X
                @###@
                    X
                    @###@
                        X
                        @###@
                            X
                            @###@
                                X
                                @###@
                                    X
                                    @###@
                                        X
                                        @###@
                                            X
                                            X
```

Note that since 79/4=19 and 24/2=12 there are exactly 12 steps

43 Equations (Regionals 1992)

Write a program to reduce mathematical equations containing two variables (x and y) to their standard mathematical notation Ax + By + C = 0.

Your program will get its input from a file called DATA31.txt (DATA32.txt for the second try). Each line input will contain a linear mathematical equation with two terms (x and y) not in its simplest form, and there are a total of four such equations. Note that there may be extraneous blank spaces in the equations.

The output will display the equation in its simplest form, without extraneous spaces. A coeffient of 1 or 0 is not permitted. Note also that adding a negative integer (as in: x+-2) should be replaced by subtracting a positive integer (as in: x-2)

Sample Input:

```
x+2y-19= y-5x+ 29-3y
2x-4y+20=2x+10
34x=4y-12x+34-9x-12y
4x+8y-44=3x+8y-44
```

Sample Output:

3x+2y-24=0
2y-5=0
55x+8y-34=0
x=0

44 Amazing (Regionals 1992)

Write a program to create a 20x20 maze. We will use the convention that (1,1) is the top left corner and (20,1) the top right corner of the maze.

DATA41.txt (DATA42.txt for the second try) is the input file. The first 10 lines of the file contains an ordered pair each, representing the locations of 10 mines displayed in the 10x10 maze, a set of two integers separated by a space. The next 4 line contains a series of 19 letters H and 19 letters V in random order, representing paths through the maze from top left to bottom right taken by a 4 different users

Your output should be the maze as shown in the example, representing the mines using the letter “M”. Show the path of each user in succession, separated by input control. If a path crosses a mine, the path should stop and the mine replaced by the letter “X”. Identify each path by its appropriate number as shown in the example.

Sample Input:

```
1 3
1 1
8 14
11 2
12 3
5 13
19 7
18 16
7 2
12 18
VHHVHVHVVVHVVHHVVHHHHVVHHVVVVHHVHVHHVH
VHHHVVVHVHVHVVHHVVHVHHVHHHHVHVHHVVVVVH
HVVHVVHHVVVVVVVHVVVHVHHVHHHHHHHVHVHHVH
HVHHVHHHHVHVHHVVVVVHVHHHVVVHVHVHVVHHVV
```

Sample Output:

```
1.................. 2.................. 33................. 44.................
111...M...M........ 2222..M...M........ .3...M...M........ .444..M...M........
M.11.......M....... M..2.......M....... M33........M....... M..44444...M........
...11.............. ...2.............. ..3.............. .......44...........
....1.............. ...22............. ..333............. ........444.........
....1.............. ....22............ ....3............. ..........4.........
....11............M. .....22...........M. ....3............M. ..........4.......M.
.....1............. ......2............ ....3............. ..........4.........
.....111........... ......222......... ....3............. ..........4.........
.......1........... ........2.......... ....3............. ..........44........
.......11111....... ........22......... ....3............. ............4444.....
...........1....... .........222........ ....33............ ..............4.....
.....M.....111..... .....M.....22222.... .....X............ .....M........4.....
.......M.....1..... .......M.......22... .......M.......... .......M......44....
.............1..... ................222. ................. ................44...
.............1...M.. ..................M2. ................M.. ................4X..
............111.... ..................2. ................. .................
..........M...11... ..........M......2. ..........M........ ..........M........
................111. ..................2. ................. .................
..................11 ..................22 ................. .................
```

45 Master Mind (Regionals 1993)

This is a game that one player will play against another using the computer to keep track of the progress of the game. The object of the game is to figure out a hidden code made up of four different coloured markers at positions A, B, C, and D. The colours may be any of RED, ORANGE, YELLOW, GREEN, BLUE or VIOLET, but only one of each colour is selected for any game. It is assumed that both players know this rule and therefore the program need not check for four distinct colours.

One player enters the code, and the second player guesses what it is. the program should prompt the first player for the colour of each marker. Then the second player should have an opportunity to guess the code. After each guess, the computer will indicate how many colours were correct and how many were in the correct location.

If the second player guesses a colour correctly, but in the wrong position, the program shows a "0". If the colour is correct, and in the correct position, the program shows a "1". the ones are always shown first followed by any zeroes.

The program continues prompting until the code is broken.

Sample Input:

(keyboard entry)

```
ybro
rgvb
yrgo
orvb
rbgv
ybor
ybro
```

Sample Input:

```
Enter the colour of the markers, player 1
Use R, O, Y, G, B, V
What is the colour of markers A,B,C,D? ybro

press any key to continue..
<clear the screen>
```

```
Enter the colour of the markers, player 2
Use R, O, Y, G, B, V
Guess the colour of markers A,B,C,D? rgvb

press any key to continue..
<clear the screen>
```

```
Round    A   B   C   D   RESULT
  1      r   g   v   b   0 0

Enter the colour of the markers, player 2
Use R, O, Y, G, B, V
Guess the colour of markers A,B,C,D? yrgo

press any key to continue..
<clear the screen>
```

```
Round    A   B   C   D   RESULT
  1      r   g   v   b   0 0
  2      y   r   g   o   1 1 0

Enter the colour of the markers, player 2
Use R, O, Y, G, B, V
Guess the colour of markers A,B,C,D? orvb

press any key to continue..
<clear the screen>
```

```
Round    A   B   C   D   RESULT
  1      r   g   v   b   0 0
  2      y   r   g   o   1 1 0
  3      o   r   v   b   0 0 0

Enter the colour of the markers, player 2
Use R, O, Y, G, B, V
Guess the colour of of markers A,B,C,D? rbgv

press any key to continue..
<clear the screen>
```

```
Round    A   B   C   D   RESULT
  1      r   g   v   b   0 0
  2      y   r   g   o   1 1 0
  3      o   r   v   b   0 0 0
  4      r   b   g   v   1 0

Enter the colour of the markers, player 2
Use R, O, Y, G, B, V
Guess the colour of markers A,B,C,D? ybor

press any key to continue..
<clear the screen>
```

```
Round    A   B   C   D   RESULT
  1      r   g   v   b   0 0
  2      y   r   g   o   1 1 0
  3      o   r   v   b   0 0 0
  4      r   b   g   v   1 0
  5      y   b   o   r   1 1 0 0

Enter the colour of the markers, player 2
Use R, O, Y, G, B, V
Guess the colour of markers A,B,C,D? ybro

press any key to continue..
<clear the screen>
```

```
Round    A   B   C   D   RESULT
  1      r   g   v   b   0 0
  2      y   r   g   o   1 1 0
  3      o   r   v   b   0 0 0
  4      r   b   g   v   1 0
  5      y   b   o   r   1 1 0 0
  6      y   b   r   o   1 1 1 1

You found the code!
```

46 Plotter (Regionals 1993)

In this program you will write a program that will plot a linear relation, given the values of the rise, run, and y-intercept. The "*" should be used to plot the points on the line and the axis should be set up as shown below. The program will only plot values of the relation where the x element is an integral value between -39 and+39, and the y is an integral value between -10 and +10. the rise, run, and y-intercept should all be entered as integers as well.

Input: the program should prompt the user for the rise, the run, and the y-intercept as integers. (No error checking needed - integers will be used in the test data.)

Output: The output should include a set of axes with the origin at the centre of the screen. the x-axis should be drawn with the "-" sign and the y-axis with the "|". Each axis should be labeled X or Y and -10, -5, 5, and 10 should be indicated on each axis. the points should only be plotted if they fall in the range indicated above. If there are no points within these values, only the axes should be drawn.

Sample Input:
(keyboard entry)
-2
3
-2

Sample Output:

```
What is the value of rise? -2
What is the value of run? 3
What is the y-intercept? -2
<clear the screen>

                      *                Y | 10
                                         |
                         *               |
                                         |
                             *           |
                                         | 5
                                *        |
                                         |
                                    *    |
                                -10  -5  |    5     10                          X
-------------------------------------*---|-----------------------------------------
                                     |   |
                                     *   |
                                     |   |
                                     |    *
                                     |-5
                                     |      *
                                     |
                                     |          *
                                     |
                                     |-10         *
```

47 Special Number (Regionals 1993)

The number 6174 is indeed a very special number. In this problem, you will illustrate just why this number is so special.
Take any four digit number. Arrange the digits of that number so that they are in descending order to create the biggest number that can be formed by these digits. Similarly, arrange the digits in ascending order to create the smallest possible number that can be formed.
Now it gets interesting. Subtract the smallest number from the largest and take a look at the answer. If the answer is not 6174, use your answer as the four digit number to repeat the steps. Within eight attempts, your answer will be 6174!
The user will simply enter a positive four digit integer as the starting number. If the number is unsuitable, print the message, "invalid Input" and prompt the user for the number again.
Example: Enter a four digit number: 2019

Output: On a cleared screen show the detailed calculations until the answer 6174 is generated. Use the following as an example of the required formatting. Left justify all output. Scrolling of the screen is allowed if required, to accomodate all of the tries.

Sample Input
(Keyboard entry)

2019

Sample Ouput:

```
Enter a four digit number: 2019

              Try #1    9210
                       -0129
                       -----
                        9081  Not Yet

              Try #2    9810
                       -0189
                       -----
                        9621  Not Yet

              Try #3    9621
                       -1269
                       -----
                        8352  Not Yet

              Try #4    8532
                       -2358
                       -----
                        6174  It Works!
```

48 Word Wrap (Regionals 1993)

Write a program to reformat up to 50 lines of text into lines having a specified line length.
The input text lines will have a maximum length of 80 characters and will not have any leading or trailing spaces. No word in the input text will be longer than the specified line length. The maximum specified line length will be 60 characters.

Input lines that are too long may be split only at spaces, in which case any extra spaces are ignored. Other than that, spacing should be maintained as it is in the input text. Short lines must be joined together by adding one space between words in order to make lines the maximum possible length.

Input: The data file will be called DATA41.txt (DATA42.txt for the second try) and it will begin with a single number on a line by itself. This number is the specified line length. The data file will then have up to 50 lines of text followed by the word "end" on a line by itself. The number on the first line and the word "end" on the last line are not part of the text to be formatted.

Output: The program should clear the screen and then print a line of numbers showing the specified line length. Starting on the next line it should write the text out to fit in the specified line length. (Remember that spacing should be preserved if they are not split onto different lines.)

Sample Input:

```
25
The rain in Spain stays mainly on the plain, which all in all
is a very good thing.  There are three spaces after this
question mark?   Now we test the word wrap program properly by
putting three spaces after this colon:   and seeing if the program
added extra spaces.
end
```

Sample Output:

```
1234567890123456789012345
The rain in Spain stays
mainly on the plain,
which all in all is a
very good thing.  There
are three spaces after
this question mark?   Now
we test the word wrap
program properly by
putting three spaces
after this colon:   and
seeing if the program
added extra spaces.
```

49 Dr. MacDile changes the world in 10 steps (Regionals 1994)

Dr. MacDile is tired of the ordering of the decimal number system where "0" is the lowest digit (and place holder) and "9" the highest digit. The good doctor proposes a new ordering where the order will be given by a 10 digit string which includes the digits 0 to 9 in a new order. For example, "7658213094" means that the 7 is now the lowest digit (and place holder) and 4 is now the highest digit. The rules of place holding still apply in that the right-most digit is the 1's digit, the digit second from the right is the 10's digit and so on.

You will be given an ordering and up to 20 non-negative integers of a maximum length of 10 digits expressed using the new system of ordering. "-1" will be your key that the list of numbers to be sorted is done. You must sort this list in order from lowest to highest and print the sum of the column of numbers in MacDile format. A number may be repeated in the list. A number may include any number of leading zeroes (although they may not necessarily be the character "0"). One number will appear on each data line in the data file. The data file will be called **DATA11.txt** for the first try and **DATA12.txt** for the second.

Sample Input:

```
4689723105
1235
44685
896
546858
1345 845
75844
75848
-1
```

Sample Output:

```
44685
845
896
1235
1345
75844
75848
546858
```

50 Making Change (Regionals 1994)

There are many ways you can receive change for a given amount of money. Write a program that will display the number of different ways you can make change for a given amount of money. the money system consists of pennies, nickels, dimes, quarters, and loonies.

5 pennies = 1 nickel	5 nickels = 1 quarter
2 nickels = 1 dime	4 quarters = 1 loonie

Read in the data from the file called "DATA21.txt " (DATA22.txt for the second try) which contains multiple monetary values in the format shown in the Sample data below. All money is expressed as a dollar value between 0 and 2 dollars in the format $0.00 to $2:00. Print your output as shown in the sample run aligned to the left side of the screen. The following example run uses the value $0.23. The -1 in the file will serve as a flag to end the program execution. Implement a HIT ANY KEY TO CONTINUE to pause the program at the end of each full screen during the execution of the program.

Sample Input:

```
$0.23
-1
```

Sample Ouput:

#	Pennies	Nickels	Dimes	Quarters	Loonies
1	23	0	0	0	0
2	18	1	0	0	0
3	13	2	0	0	0
4	8	3	0	0	0
5	3	4	0	0	0
6	13	0	1	0	0
7	8	1	1	0	0
8	3	2	1	0	0
9	3	0	2	0	0

51 We're way off Base on this One (Regionals 1994)

You have lost your handy pocket calculator which will subtract two numbers in any base from 2 to 36 inclusive. The program will read in three numbers at a time. Numbers in bases higher than 10 will use letters for digits ie A=10, B=11, .., Z=35. Any letters will be capitalized. the first number will be a base. The next two numbers will be the two numbers to be subtracted with the second number being subtracted from the first. These numbers will be up to 40 digits in length. The flag at the end of the data file will be three zeroes. Each number will be on a separate line. You are to check for numbers in the data file that are inappropriate for the base indicated an if necessary print out the word INCORRECT DATA. (ie if the letter A was in a base 8 number you would print INCORRECT DATA).

Output the correct response for each set of numbers following the format shown in the sample data below. Data may contain leading zeroes. Note that any leading zeroes should be removed before printing the output. the data file will be called DATA31.txt (DATA32.txt for the second try).

Sample Input:

```
2
1001
11
8
00412
427
16
D4A
8F4
6
455
382
0
0
0
```

Sample Output:

```
Output #1: 110
Output #2: -15
Output #3: 456
Output #4: INCORRECT DATA
```

52 A Teacher's Best Friend (Regionals 1994)

Ms. Brodie, the history teacher at ECOO Secondary School has just completed marking her final exams and calculating her final marks. She has a list of marks, all positive integers from 0 to 100 inclusive. She asks you to take these marks and produce a histogram according to the following criteria:

The marks are grouped by letter grade accordingly:

A: 80 to 100 inclusive
B: 70 to 79 inclusive
C: 60 to 69 inclusive
D: 50 to 59 inclusive
F: 0 to 49 inclusive

Then a histogram is produced showing the number of students in each grade range with the grades sorted by the frequency at each grade range from highest to lowest. In case of a tie, the higher grade is shown first - i.i. if there are an equal number of B's and C's, then show the B's first. There will be at least one and no more than 10 marks in each grade category. The bar graph for the histogram is constructed using the letter of each respective grade. There are to be two spaces between each grade. The historgram is to be labeled up to 10, even if that is not needed for the data.

The data file will be called DATA41.txt (DATA42.txt for the second try).

Sample Input:

```
45
47
85
75
98
44
56
68
89
99
-1
```

Sample Output:

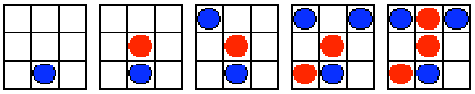
```
10 |
   |
9  |
   |
8  |
   |
7  |
   |
6  |
   |
5  |
   |
4  |AAA
   |A A
3  |A A  FFF
   |A A  F F
2  |A A  F F
   |A A  F F
1  |A A  F F  BBB  CCC  DDD
   |AAA  FFF  BBB  CCC  DDD
   ------------------------------------------------
```

53 Tic-tac-toe (Boardwide 1995)

Consider the game of tic-tac-toe on the computer. The squares are numbered as on the numeric pad of a standard keyboard. (see the figure to the right. Two players make their moves one after the other, until one wins or until there is a draw. Valid input are the digits 1...9 in some order.

7	8	9
4	5	6
1	2	3

If for example the input were: 2 5 7 9 8
then the resulting output would be:

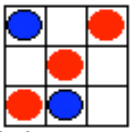


Write a program that creates a gray square in graphics mode, approximately in the middle of a black screen. The program will read its data from a text file called **DATA11.txt (DATA12.txt** for the second try). Each of five lines in the text file contains exactly 9 integers, and each line represents one game played. All input is numeric, but invalid numbers (e.g. 16) are ignored, and the player loses the turn. If a player attempts to play on a spot that is already occupied, the turn is lost also.

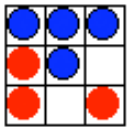
The players' names are "Blue" and "Red". Blue always plays first. When Blue plays, a blue circle will be placed in the appropriate spot in the square. When Red plays, a red circle will be placed in the square. The first player that has 3 circles in one line (horizontal, vertical or diagonal) will win, and if there are unplayed numbers in the row, they will be ignored. There should be a pause at the end of each game, as shown in the sample below:

SAMPLE INPUT:

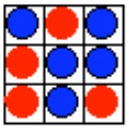
```
(three of five samples)
2 5 7 9 16 1 8 3 4
5 4 7 3 9 1 8 2 6
5 3 2 8 9 1 6 4 7
```



```
Red wins, press any key to continue..
```



```
Blue wins, press any key to continue..
```



```
A draw, press any key to continue..
```

54 Path Finder (Boardwide 1995)

Imagine that you are given instructions on grid to turn left or right and to walk a certain number of steps. You are then given another clue as to where to turn and how far to walk, and so on. **DATA21.txt (DATA22.txt** for the second try) is a text file that contains a set of positive and negative integers. These numbers represent a direction: if positive, turn right 90°, if negative, turn left 90°. The absolute value of the number reflects the distance one must travel. "-23" for example means, turn left 90° and move forward 23 steps. "10" means, turn right 90° and move forward 10 steps. Create a program that will draw this path on the 80x25 text screen. There are 4 directions on this screen: Right, Up, Left, Down. Assume that the original direction is Right, so that the first number will start a path Up, if negative, and Down if positive. Use the following rules for drawing:

1. Use the tokens below appropriately.
186 = ‖ 205 = ═ 201 = ╔ 187 = ╗ 200 = ╚ and 188 = ╝

2. A turn is an extra character on the screen, so +5 represent 6 characters:
╔═════

3. If a path crosses the end of the screen, the path should scroll:
A path that moves past the top should reappear at the bottom

4. The first character is an **"X"** and the last character a **"O"**

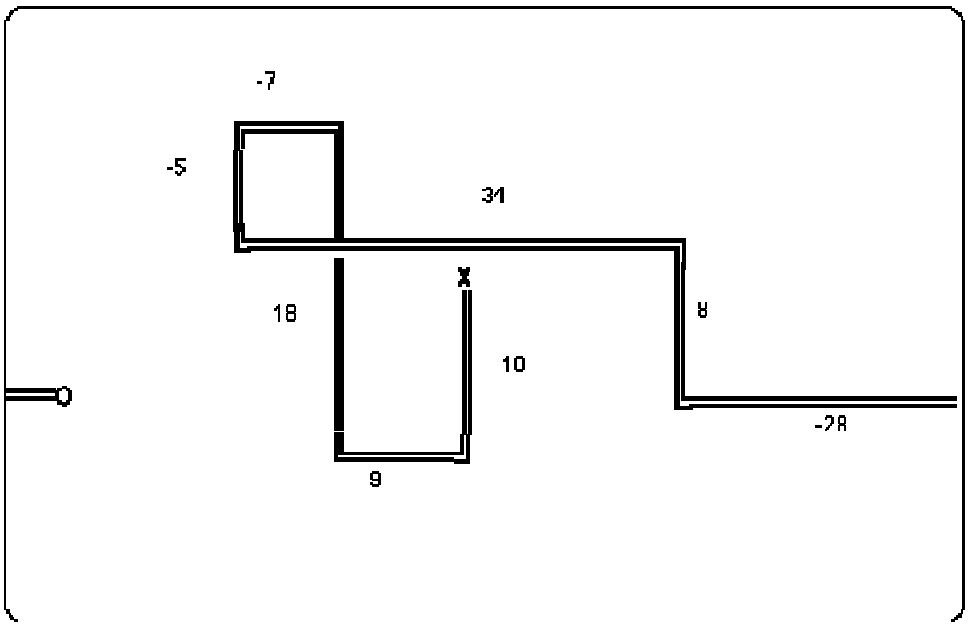
5. You may stop the path when a 0 input is read.

6. The numbers in the sample output are for clarification only: Do not output numbers

SAMPLE INPUT:

```
10
9
18
-7
-5
-34
8
-28
0
```

SAMPLE OUPUT:



55 Decoder (Boardwide 1995)

Imagine a coding system for secret messages that converts all characters and digits by moving them along the string "ABCDEFGHIOJKLMNOPQRSTUVWXYZ.1234567890" a specified number (x) of spaces. If x were 7 for example, then the letter A would become H, the letter Z would become 6, the digit 9 would become F, and so on.

There are 37 possible ways in which a message could be encoded, depending on the value of x. DATA31 is a text file that contains 5 encoded messages. Each message is on a separate line and contains no more than 250 characters. Write a program that reads each message and that will decode and print the message. Your strategy is to find the value of x that would generate a message with the least number of digits in it.

Sample Input:

```
A0Z1MGBA0MXMF591MF41D1MIXEMXMBAADMIAA.ZGFF1DMI4AM4X.MEAM9X0KMZ458.D10M
F4XFM41M4X.M0AFM9GZ4MA2M15F41DM2AA.MADMZ8AF4503MFAM35H1MF419M

7HOKZGOCBOZOH6IFG2ZMO3J3B7B5O0ZH3O7BOH63O4Z00OC4OH63OM3ZFOH63OK3ZH63FO
KZGOK702OZB2OFCI56OCIHG723OZB2O7HOKZGO1FI300MO2ZF9OH63OFZ7BO4300OZB2OH
63OK7B2O.03KOH700OH63OKZ00GOC4OH63O1CHHZ53OG6CC9O

J85OQ1BBQI1JQHEKD4QJ85Q69H5Q2KIOQM9J8QJ89IQJ89D7Q1D4QJ81JQ0KIJQJ85DQ1B
BQ1JQED35QIEC5J89D7Q71L5QJ8H55QJ1FIQ1JQJ85QM9D4EMF1D5QJ85DQJ85Q61J85HQ
M5DJQEKJQJEQI55QM81JQM1IQJ85QC1JJ5HQ1D4QM85DQ85Q7EJQEKJQE6Q4EEHIQM81JQ
I8EKB4Q85QI55Q2KJQ1Q7H51JQ297QM89J5Q251HQ

FNNCZDUDMHMFZSNZXNTZR0HCZSGDZVGHSDZAD0QZ

YMJ5XFRJ5YT53TZ5XFNI5YMJ5RFS5
```

Sample Output:

```
Offset = 23
ONCE.UPON.A.TIME.THERE.WAS.A.POOR.WOODCUTTER.WHO.HAD.SO.MANY.CHILDREN.
THAT.HE.HAD.NOT.MUCH.OF.EITHER.FOOD.OR.CLOTHING.TO.GIVE.THEM.
Offset = 25
IT.WAS.ON.A.THURSDAY.EVENING.LATE.IN.THE.FALL.OF.THE.YEAR.THE.WEATHER.
WAS.WILD.AND.ROUGH.OUTSIDE.AND.IT.WAS.CRUELLY.DARK.THE.RAIN.FELL.AND.T
HE.WIND.BLEW.TILL.THE.WALLS.OF.THE.COTTAGE.SHOOK.
Offset = 27
THEY.ALL.SAT.ROUND.THE.FIRE.BUSY.WITH.THIS.THING.AND.THAT.JUST.THEN.AL
L.AT.ONCE.SOMETHING.GAVE.THREE.TAPS.AT.THE.WINDOWPANE.THEN.THE.FATHER.
WENT.OUT.TO.SEE.WHAT.WAS.THE.MATTER.AND.WHEN.HE.GOT.OUT.OF.DOORS.WHAT.
SHOULD.HE.SEE.BUT.A.GREAT.BIG.WHITE.BEAR.
Offset = 36
GOOD.EVENING.TO.YOU.SAID.THE.WHITE.BEAR.
Offset = 5
THE.SAME.TO.YOU.SAID.THE.MAN.
```

56 Formula (Boardwide 1995)

DATA41.txt (**DATA41.txt** for the second try) is a text file that contains 5 valid mathematical expressions, one per line, that use the five operations +,-,*,/,^. (Respectively: addition, subtraction, multiplication, division and exponentation). It also uses the round brackets and floating point non-negative constants and or integers. You may expect 56.78 and 346 for example, but not **-34** or **4.5E2**. All operations are explicit, so **2*(5+3)** must be used instead of: **2(5+3)**. There are no embedded blanks in the expressions.

Write a program that evaluates each expression, as shown in the sample below to 3 decimal digits accuracy.

SAMPLE INPUT:

```
45+4*5/2-1
(16-8)*3+64-4.5
(3.5+(5-0.5)/3)^3+1
2^3^4-200
(12.6+4)/3+1.6
```

SAMPLE OUTPUT:

```
45+4*5/2-1 = 54.000
(16-8)*3+64-4.5 = 83.500
(3.5+(5-0.5)/3)^3+1 = 126.000
2^3^4-200 = 3896
(12.6+4)/3+1.6 = 7.133
```

57 Doodle (Regional 1995)

You must write a program that will 'draw' five 'trails' of asterisks on the computer screen. Each series of data will represent 1 trail and there will be five series of data in the file. All series are to be drawn on the same screen. (not cleared after each series).

NOTE: the numbers on the numeric pad represent directions as represented in the chart below

up 1 left 1 **7**	up 1 **8**	up 1 right 1 **9**
left 1 **4**	toggle **5**	right 1 **6**
down 1 left 1 **1**	down 1 **2**	down 1 right 1 **3**

The first two data items found in the DATA11.txt file (DATA12.txt for the second try) represent the starting position for that series. Only valid data will be given for the screen co-ordinates. The remaining data items are the digits that represent the movement of the cursor or the condition of the pen (pen up or pen down). This data is not guaranteed to be valid! Each series of data will end with the character Z.

The starting mode for each series of data is pen down which means that when the cursor moves drawing occurs. The number 5 has NO movement function. It will be used to toggle the pen up and down. when the pen up or down is invoked the contents under the cursor will remain when the cursor moves away; In other words the pen change happens on the next position visited by the cursor and no character or movement is created when the toggle takes place.

The first character of each series will be the letter S and will replace the first asterisk. the letter S will always be created with the pen down. When the pen is down a trail of asterisks (*) will be left behind the cursor as the data moves it about the screen. If the cursor moves over any existing trail of the current or of a previous series of contents under the cursor will remain as it was when the cursor moves away, whether the pen was up or down at the time.

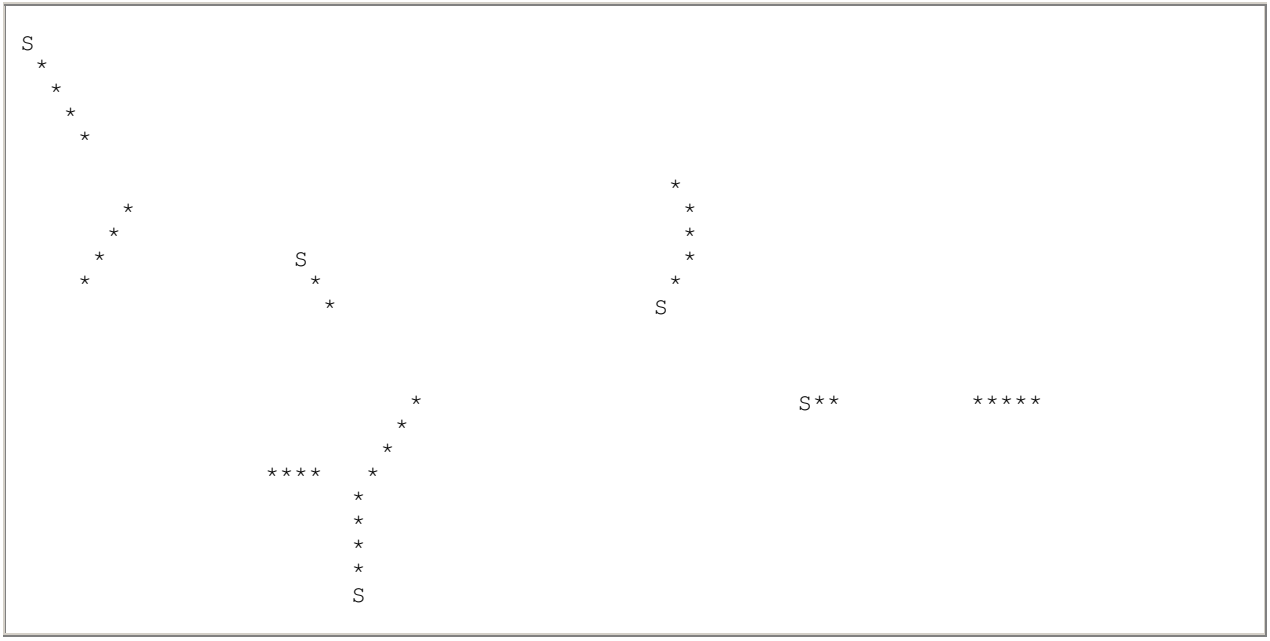
At the end of drawing all five series the cursor should move to the (79,24) position. The output should remain on the screen until the user presses a key to end the program. Any data commands that are invalid or would force the cursor off the screen are to be ignored.

The screen's xy co-ordinates are based on a 80 x 25 screen with the 0,0 point in the upper left corner and the 79,24 point in the lower right corner.

Sample Input:

```
45,12,9,9,8,8,7,5,6,6,3,2,2,Z
1,1,3,3,3,3,5,3,3,3,5,1,1,1,Z
56,16,6,6,5,6,6,6,6,6,6,6,6,6,5,6,6,6,6,Z
25,24,8,8,8,8,9,9,9,9,5,4,4,4,4,4,4,4,2,2,2,5,4,4,4,Z
20,10,3,3,Z
```

Sample Output:



58 Electronic Dictionary (Regional 1995)

When looking up words in an electronic dictionary, it is helpful to have a "Shorthand" to save typing. In this case, we wish to match word patterns to a database of words in a dictionary. The symbol "?" is used to match any single letter, and the symbol "*" is a "wild card" used to match one or more letters. for example, consider a database with the following words in it:

woes, woken, woman, womanliness, women, wombat,wombats, won,woven, born

Search String	Word(s) matched from the above list
woman	woman
wom?n	woman, women
wo??n	woken, woman, women, woven
wo*n	woken, woman, women, woven
*n	woken, woman, women, won, woven
w*s	woes, womnliness, wobats
womba?	wombat
womba??	wombats
w?m?t	(NONE)

The sample data will begin with no more than twenty words to serve as the database; each word will be no more than fifteen letters in length. Each data item will occur on a separate line. these words wil end with a flar of "<END>". there will follow five search strings. You will have the user press "Enter" after each search string, and clear the screen before each dispalyl you need not display the matching words in any particular order. There will be no more than one "*" in any one search. All data and output will be in uppercase letters with no punctuation, except for "?" and "*".

Sample Input:

```
AARDVARKS
AARDWOLVES
ABACK
ABYSSS
ABACUSS
ABACI
ABACUSES
BABIES
BABBLE
CABBIE
<END>
A???????S
A*S
AB*
AB??E
*LY
```

Sample Ouput:

```
A???????S

AARDVARKS
Hit any key to continue

A*S

AARDVARKS
ARDWOLVES
ABYSS
ABACUSS
ABACUSES
Hit any key to continue

?AB*

BABIES
BABBLE
CABBIE
Hit any key to continue

AB??E

BABBLE
CABBIE
Hit any key to continue

*LY

(NONE)
```

59 K^{th} digit beginning at N (Regional 1995)

Write a program that when given two non-negative numbers N and K, will find the K^{th} digit (not number) of the arithmetic (counting) sequence beginning at N. For example, (7,9) would yield 2: Beginning the count at 7, the ninth digit of 12345678910111213… is the digit 2 of the number 12.

The counting of digits begins with the first digit in N. The pair (1000,5) would yield the left most digit 1 of 1001.

Your program will input five pairs of numbers on five lines and print five digits on a cleared screen. the data will be such that N+K<20 000.

Sample Input:

7 9
1 15
20 20
1000 5
300 1000

Sample Output:

2
2
9
1
6

60 Passing the Time Away (Regional 1995)

In our society, both digital and analog clocks are popular. Write a program that when given two readings from an analog clock, will give the time elapsed between the two readings in digital format.

To facilitate the analog readings, the position of the hands will be given in degrees with 0° (or 360°) denoting the twelve, 90° representing the three, and so on. Each reading will have three degree values for the hour, minute and second hand respectively. The second hand value will correspond exactly to a degree reading such that the value will be from 0 to 59 seconds. The reading for the hour and minute hands may vary slightly, but will still be obvious.

Example 1: 65, 31, 18 would be 2:05:03.
Example 2: 269, 349, 180 is 8:58:30

The input file DATA41.txt (DATA42.txt for the second try) will contain five sets of two readings each. The second time will always be later than the first time, even if the second hour value is smaller than the first hour value (e.g. 10 o'clock to 3 o'clock).

Output, on a cleared screen, will be the time elapsed between the two readings, in digital format, followed by a blank line. No elapsed time will be greater than 11:59:59.

Sample Input:

```
269 349 180 65 31 18
268 253 60 360 360 0
65 61 90 8 91 84
195 183 120 30 26 174
275 60 84 276 67 102
```

Sample Output:

```
 5:06:33
 3:17:50
 10:04:59
 6:34:09
 0:01:03
```

61 Roman Numerals (Boardwide 1996)

Roman Numerals are composed of capital letters of the Roman Alphabet. The rules that govern the composition of these numbers have changed over the years, but for the sake of this problem, please follow the following rules:

`I=1, V=5, X=10, L=50, C=100, D=500, M=1000`. No other letters should be used: The input file will not contain any numbers larger than 3999

Addition principle: Any number is composed of a set of these letters, with the largest valued letter to the left of the lesser valued letter: e.g. `LXXVIII = 78`. If necessary, letters may be repeated, but not `V, L, or D`.

Subtraction principle: In order to avoid four letters of a kind in a row, `IV` is used instead of `IIII` for 4 and `IX` instead of `VIIII` for 9. Similarly `XL=40, XC=90, CD=400, CM =900`. The subtraction principle does not apply in any other case. For example, `IL` is not 49 or `MVMI` is not 1996. (note: `XXXIX=49` and `MCMXCVI=1996`)

Write a program that will read a text file called DATA11.txt (DATA12.txt for the second try), which contains one Roman numeral per line. Each line, including the last one, ends in char(13) (return or enter). Your program should print, on a cleared screen, the equivalent Arabic notation, or say that the number is invalid, as in the following example:

Input:

```
XXIX
XXXVIII
CCXCI
MCMDIII
MCMXCVI
```

Output:

```
XXIX     is 29
XXXVIII  is 38
CCXCI    is 291
MCMDIII  is invalid
MCMXCVI   is 1996
```

62 Path to the Centre (Boardwide 1996)

Imagine the screen defined so that the upper left corner is (0,0) and the lower right corner (100,100). Up to 50 points are located on the screen, such that at least two points are located in each of the four quadrants (i.e) there are at least two points (x,y) where x<50,y<50; at least two points (x,y) where x<50, y>50; at least two points (x,y) where x>50,y<50; and at least two points (x,y) where x>50, y>50.

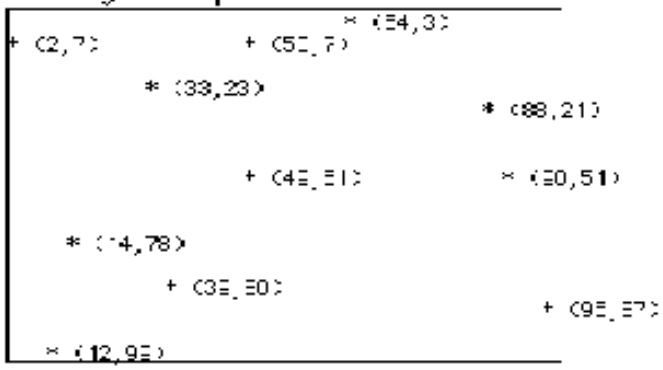
Write a program that will connect all these points with a path, that does not intersect itself. Furthermore, the path must start or end with the point that is closest to (50,50) Input data are read from a textfile called DATA2A.IN and are integers between 0 and 100 (including 0 and 100), each on separate lines, representing in groups of two, the ordered pairs (x,y) for each point. Continue reading until EOF is reached. (Like all other integers, the last integer is followed by the end-of-line character

Output should be a broken line, that does not intersect, as in the example below.

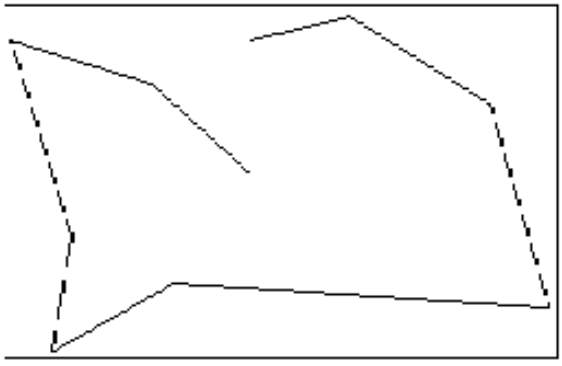
INPUT

```
50
7
49
51
39
80
78
33
23
88
21
99
95
87
2
7
89
51
64
3
```

meaning of the input



OUTPUT



63 Cipher (Boardwide 1996)

A secret message consisting of the 26 capital letters of the alphabet has been coded with the help of a keyword, that only the sender and receiver knows. The letters of the keyword are changed to their corresponding numbers: A=0, B=1, C=2, . . . , Z=25.

These numbers become the offsets to the letters of the message, once all spaces and punctuation marks have been eliminated.

For example:
The message: "Please! send me one hundred bucks" with the key: "Your son"
will become: `PLEASESENDMEONEHUNDREBUCKS`
with the key: `24/14/20/17/18/14/13`

The Message, translated into numbers:
`15/11/04/00/18/04/18/04/13/03/12/04/14/13/04/07/20/13/03/17/04/03/01/20/02/10/18`

Which is then modified by adding the values of the keyword:
`24/14/20/17/18/14/13/24/14/20/17/18/14/13/24/14/20/17/18/14/13/24/14/20/17/18/14`
repeating the keyword as often as is necessary.

Adding the numbers together will then yield the coded message:
`13/25/24/17/10/18/05/02/01/23/03/22/02/00/02/21/14/04/21/05/17/01/15/14/19/02/06`
Turning this back into Capital letters:
`NZYRKSFCBXDWCACVOEVRFRBPOTCG`

Write a program that lets you be the receiver of a coded message on a text file called DATA31.txt (DATA32.txt for the second try). The first line of the message contains the key word, which may contain embedded spaces as in the example. The next several lines (containing no more than 79 characters) contains the message you must decode. Continue reading until EOF is reached. (Like all other lines, the last line is followed by the enter character) Your output should appear on a cleared screen as shown in the following example.

Sample Input:

```
Mystery
FFWBRKCDLWMWNMDJVPMUCICTBWRUALVXVWSXNDTGVDGJDHJGGORMKIJQASFWW
MGPCGTRURQVLYMCCEKMVLFDURDBRBCPUAMLYNKCJMIOR
GQAGKRZDMOLIIJUIWFSJYUASLMDNXCEHYJCOJAVOFLMUGKHFPBGUMYIC
IGDEXRIQWGNHZPQALECKMMPWEEKCPNSZIWGXCGKIMCZYFHXYCDUWUWVPHCJ
```

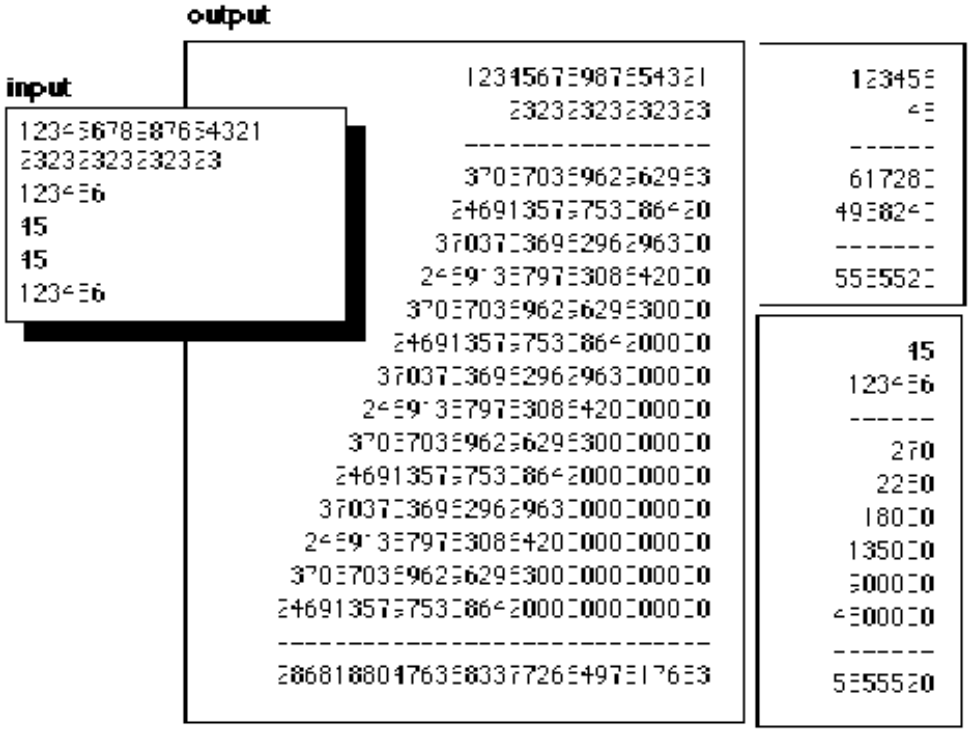
Sample Output:

```
MYSTERY
THEINTERNETSWORLDWIDEWEBISAWONDERFULPLACEFULLOFPICTURESSOUNDS
VIDEOANDTEXTFILESMUCHOFITLINKEDWITHHPYERTEXT
USINGABROWSERLIKEMOSAICASIMPLEMOUSECLICKONAWORDORPICTURE
WILLTAKEYOUDIRECTLYTOARELATEDPAGEFILEOREVENANOTHERWEBSERVER
```

64 Multiplication (Boardwide 1996)

Write a program that lets you multiply two positive integers of up to 19 digits each. The input file is a text file, DATA41.txt (DATA42.txt for the second try) that contains pairs of integers. Continue reading until EOF is reached. (Like all other integers, the last integer is followed by the end-of-line character)

These integers are to be multiplied and displayed, each problem on a separate cleared screen as shown in the example. The multiplication should be right justified as shown, and leading zeroes are not printed. The first line of separation should be as long as the larger of the two multiplicands; the second line of separation should be as long as the product. When any key is pressed, the next problem should be displayed.



65 Bogus Cancelling (Regional 1996)

The fraction 64/16 has the unusual property that its reduced value of 4 may be obtained by applying "bogus" cancelling of the 6 in the numerator (64/16 = 4/1 = 4). We will say that a fraction is "bogus" if both its numerator and denominator have two digits and it the value of the first integer divided by the second integer remains unchanged after applying "bogus" cancelling. Fractions having a value of 1 are not considered bogus, nor are fractions that have both numerator and denominator ending in zero (i.e. 50.20 is not bogus).

The text file DATA11.txt (DATA12.txt for the second try) consists of 5 sets of positive integers, both less than 100 separated by one space (numerator space denominator).

Sample Input:

64 16
27 38
50 20
12 12
19 95

Sample Output:

64/16 is a bogus fraction
27/38 is NOT a bogus fraction
50/20 is NOT a bogus fraction
12/12 is NOT a bogus fraction
19/95 is a bogus fraction

66 Gender Correctness (Regional 1996)

The text file DATA21.txt (DATA22.txt for the second try) contains 5 lines of text, each consisting of one or more short sentences taking up no more than one single line. Your job is to rewrite the lines using gender-correct terminology. To do so, you must replace all occurrences of the letters "man" with the letters "person" (case of letters must be preserved). This is to be done only when the word "man" is at the start or the end of a word. Thus the words manhole and foreman would be change to personhole and foreperson in the gender-correct sentence. In addition you are to replace any occurrences of "his" or "her" with "his/her".

If no changes are necessary to the line contents then output: "No change necessary".

Sample Input:

```
The manager looked up to the foreman
His arm broke but he managed to grab her salamander
No changes are necessary here
Tom managed to stand his ground Man to Woman
```

Sample Output:

```
Original: The manager looked up to the foreman
Gender-correct:  The personager looked up to the foreperson

Original: His arm broke but he managed to grab her salamander
Gender-correct:   His/her arm broke but he personaged to grab
his/her salamander

Original: No changes are necessary here
No change necessary

Original: Tom managed to stand his ground
Gender-correct:  Tom personaged to stand his/her ground

Original: Man to Woman
Gender-correct:  Person to Woperson
```

67 Poker Hands (Regional 1996)

A Poker hand consists of 5 playing cards. The text file DATA31.txt (DATA32.txt for the second try) contains 5 sets of numbers between 1 and 52. These numbers need to be converted to the equivalent cards using the following scheme:

The four suits are (C)lubs, (D)iamonds, (H)arts and (S)pades. The rank of the cards ranges from low to high as (A)ce, 2, 3, 4, 5, 6, 7, 8, 9, (T)en, (J)ack, (Q)ueen, and (K)ing being the highest card. The integers 1 to 13 convert to the Clubs from A,2,...,K. the integers 14 to 26 convert to Diamonds, the integers from 27 to 39 convert to Hearts and the integers 40 to 52 convert to Spades. For example, the number 48 would convert to the ninth of Spades, denoted as `S9`. As you convert these numbers to cards, you must allow for the possibility of a number outside the range 1 to 52 in the text file. If such a number occurs, immediately output the message "Bad hand, no bets on this one" and continue with the next set. Once your program has converted and verified the hand, your program will output it in a vertical list as shown in the Sample Output. Your program must now determine the best hand that can be made using the five cards. The order of hands is shown below, best to worst:

1. **A straight flush** consists of all the cards being of the same suit ie. 5 spades and the value of the cards are consecutive in rank (For example, C7 C8 C9 CT CJ)
2. **A straight** consists of all cards being consecutive in rank, but not necessarily of the same suit (For example, C7 D8 C9 HT SJ)
3. **A flush** consists of all 5 cards being of the same suit (ie 5 hearts)
4. **3-of-a-kind** means that exactly 3 of the 5 cards are of the same rank (ie 3 Queens)
5. **High card** means that none of the above hands exist and you simply output the highest card

Print a blank line between each set of data and stop the output after a full screen with a pause to continue for each additional screen of output.

Sample Input:

```
13 21 40 51 7
40 51 47 43 42
2 34 15 41 23
8 22 49 50 12
7 53 16 34 47
```

Sample Output:

```
CK D8 SA SQ C7
High card is a K
SA SQ S8 S4 S3
This is a Flush
C2 H8 D2 S2 DT
This is a 3-of-a-kind
C8 D9 ST SJ CQ
This is a Straight
C7
Bad Hand, no bets on this one
```

68 Primary Binary Images (Regional 1996)

A prime number is a number divisible only by 1 and itself. Write a program that reads a positive integer N between 1 and 32767 inclusive, and determines whether N's binary representation has a prime number of 1-bits. The program should read N, and

If the integer N has a prime number P of 1-bits, print: `N: YES → P`
Otherwise print `N: No`

For example,
63 in binary format is 111111.
Therefore it has 6 ones and 6 is not a prime. `63: NO`
55 in binary format is 110111.
Therefore it has 5 ones and 5 IS a prime. `55: YES → 5`

The text file DATA41.txt (DATA42.txt for the second try) contains 5 numbers between 1 and 32767, each on a separate line.

Sample Input:

```
63
55
32764
2049
7
```

Sample Output:

```
63:  NO
55:  YES--> 5
32764:  YES--> 13
2049:  YES--> 2
7:  YES--> 3
```

69 Eissfeldt (Final 1994)

Problem: In the first volume of the magazine The Analyst (1874), Ferdinand Eissfeldt discusses various ways of writing numbers. He mentions binary, octal, and hexadecimal, but discards them since "they have not yet come into use." He goes on to write "there is no necessity for taking any base at all. and the numbers may be made to progress in their own natural succession: or, to express it in other words, every number, even one, may be made to serve as a base for a certain time." this problem affords you the opportunity to implement Eissfeldt's method.

Eissfeldt's Method: Zero is represented by the letter X, and the first positive twenty numbers are represented by the first twenty letters of the alphabet, excluding the letter J. THUS , A = 1. B + 2, C = 3, ... , K = 10, ... , U = 20. To represent numbers after 20, this system uses the letter F, and begins as follows: FX = 21, FA = 22, FB = 23, ... , FF = 27. Because the second character now equals the first character, we increment the first character by one and continue: GX = 28, GA = 29, ... , GG = 35. When we reach the final two-digit number, UU = 240, we proceed by using FX as prefix. there are 21 numbers starting with FX: FXX, FXA, FXB, ... , FXFX. These are followed by 22 numbers starting with FA, then 23 numbers starting with FB and so on. The sample data below gives other examples.

To recap, the general rule is, as Eissfeldt wrote, "the second character is to be increased until it equals the first character, and then the first character is to be increased by one."

Input/Output. Read from the data file "DATA1" ten values. The first five values are base 10 positive integers. Convert each to its Eissfeldt notation. the last five values are numbers in Eissfeldt notation; convert each to its base 10 value. Each Eissfeldt number (the last five inputs) is a string consisting of uppercase letters A through Z only. The input data values will be separated by a comma and will contain no spaces. the decimal value of all numbers will be less than 20.000. The output format should be formatted as given in the Sample Output below.

SAMPLE INPUT:

```
16,100,333,1492,4321,T,HE,US,LET,SIX
```

SAMPLE OUTPUT:

```
Output #1: Q                    Output #1: 19
Output #1: NI                   Output #1: 41
Output #1: FDH                  Output #1: 228
Output #1: IIG                  Output #1: 2575
Output #1: NAHG                 Output #1: 16290
```

70 Working on the Chain, Gang (Final 1994)

A word chain is a sequence of words that differ by exactly one letter. For example the set of words {MOM TIP TAP MAP MOP} will produce the following different chains of length (the number of words in the chain) four or more:

MOM MOP MAP TAP
MOP MAP TAP TIP
MOM MOP MAP TAP TIP

NOTE - Two chains that are simple the reverse of each other (i.e. MOP MAP TAP TIP and TIP TAP MAP MOP) are considered the same and only need to be noted once in either order.

Your task is to read in several list of words and print out all possible word chains of length four or ore. Each list will contain 4 or 6 words. Each word in the list will have the same length (3 to 6 letters) and each list will be separated by the flag "-1". Each list of words and the associated word chains should be on a separate cleared screen. You may assume that a list of words and the associated word chains will fit on a standard screen. The data file will end with the flag "-99". All letters will be capitalized in the data file. Indicate if there are no chains of length 4 or or in a list. The data file will be called DATA21.txt (DATA22.txt for the 2nd try).

SAMPLE INPUT

```
BOAR
DOOR
MOOT
MOOR
MOAT
BOAT
-1
HELLO
COURT
COUNT
JELLO
MOUNT
-99
```

SAMPLE OUTPUT (on a cleared screen)

```
The set {BOAR DOOR MOOT MOOR MOAT BOAT}
contains the following word chains of length four or
more:

DOOR MOOR MOOT MOAT
MOOR MOOT MOAT BOAT
MOOT MOAT BOAT BOAR
DOOR MOOR MOOT MOAT BOAT
MOOR MOOT MOAT BOAT BOAR
DOOR MOOR MOOT MOAT BOAT BOAR

<Hit any key to continue>
(the screen will be cleared at this point)

The set {HELLO COURT COUNT JELLO MOUNT}
contains no word chains of length four or more.
```

71 Wrecked Angles (Final 1994)

Problem: Given a pair or rectangles, A and B, whose sides are parallel to the axes, compute some interesting facts about them. the interesting facts are:

1. the area of rectangle A
2. the perimeter of rectangle B
3. the number of vertices of A that are inside B
4. the number of points where the two rectangles intersect
5. the area of the region where the two rectangles intersect

Input: Read from the data file "DATA31.txt" (DATA32.txt for the second try) two sets of 4 positive integers with the format as shown in the Sample Input below. Rectangle a followed by rectangle B. Each rectangle is specified by four positive integers corresponding to the locations of the north, south, west, and east sides. For example, "7,4,2,9" represents a rectangle with corners at (2,7), (2,4), (9,7) and (9,4). We guarantee that each rectangle will be valid (that is, north>south, and east>west), and that no edges of the two rectangles will coincide (for example, the north edge of A will differ from both the north and south edges of B).

Output: Print the facts mentioned above with the format shown in the sample output below. Each integer is separated by a comma (no extra spaces) Each rectangle's information is on a separate line. only one set of data will be included in the test data file.

SAMPLE INPUT:
10,5,1,7
20,8,4,10

SAMPLE OUTPUT:
30
36
1
2
6

72 Common Sets (Final 1994)

Problem: Given the elements of two NON-EMPTY sets, classify the relationship between the two sets. the possible relation(s) are as follows:

Equal The two sets contain exactly the same elements.

Disjoint the two sets have no elements in common.

Subset All the elements of the first set are in the second set, but the second set contains at least one element not in the first set. (This relationship is usually called a Proper Subset.)

Superset All the elements of the second set are in the first set, but the first set contains at least one element not in the second set. (This is usually called a Proper Superset.)

Overlap The two sets each contain some elements in common and some elements not in common to both sets.

Input: Five pairs of sets will be read from a file of the DATA4. Each part of the pair will be separated by a comma and no spaces. The elements of each set are characters from among the letters A-Z and the digits 0-9. The set is specified as a single character string and will not contain any duplicate items.

Output: For each pair of sets, print their relationship.

SAMPLE INPUT:

1234,4321
1234,41
AB,ABCD89
ABCD,1234
12AB,A3B4

SAMPLE OUTPUT:

EQUAL
SUPERSET
SUBSET
DISJOINT
OVERLAP

73 Division (Boardwide 1997)

Given an integer B = $a_1a_2...a_n$, where $a_1,a_2,...,a_n$ are single digits, the following simple method may be used to check divisibility by 3 or 9:

a. Add up the digits of B to form another number C = $a_1+a_2+...+a_n$.
b. If C has 1 digit, then: B is divisible by 3 (or 9) if C is divisible by 3 (or 9).
c. If C has more than 1 digit, repeat a. with the digits of C.

Eg. if B = 12345, here are the steps taken to check divisibility for 3:

```
12345 mod 3
= 15 mod 3
= 6 mod 3
= 0
12345 is divisible by 3
```

DATA11.txt (DATA12.txt for the second try) contains an integer, N, on the first line, followed by N sets of two data separated by a single space: an integer X followed by either a 3 or a 9.
Write a program that uses the above algorithm to check X for divisibility by 3 or 9, whichever is indicated. Write out the steps as above.

SAMPLE INPUT:

```
2
10 9
12345 3
```

SAMPLE OUTPUT:

```
10 mod 9
= 1 mod 9
= 1
10 is not divisible by 9

12345 mod 3
=15 mod 3
=6 mod 3
=0
12345 is divisible by 3
```

74 In, Out, and All About (Boardwide 1997)

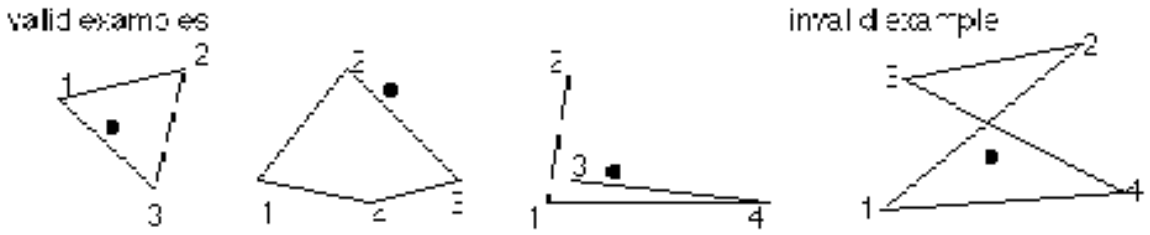
Consider an n-sided convex polygon. Given a point, it is an important problem in geometry to find out if the point is inside or outside the polygon. A point that is on the line is considered inside the shape (see case #3 of the sample)

Here, we will only be concerned with triangles and four sided figures.

DATA21.txt (DATA22.txt for the second try) contains one integer, N on the first line followed by N lines of the form:
T x1 y1 x2 y2 x3 y3 x y or: R x1 y1 x2 y2 x3 y3 x4 y4 x y

T means a triangle with vertices: (x1, y1), (x2, y2), (x3, y3)
R means a rectangle with vertices: (x1, y1), (x2, y2), (x3, y3), (x4, y4)
(x, y) is the point to be tested
all x's and y's are integers)

The vertices are given in clockwise order, and the shapes will make sense:




```
SAMPLE INPUT:
4
R -2 4 3 10 9 6 5 -4 4 4
T 2 -2 5 8 11 -1 10 7
R 2 3 5 9 8 6 10 0 6 8
R 2 -2 3 10 3 -1 10 -1 4 0

SAMPLE OUTPUT:
IN
OUT
IN
OUT
```

(in order to get any marks at all, all answers must be correct)

75 Matchit and Patchit (Boardwide 1997)

Here we will consider the problem of wild-card matching.
There are two wild card characters: * (star) and ? (question mark)

* matches any set of characters of any length (including the empty string)
? matches any single character.
all other character matches themselves.

Given a long text, we could use these to look for certain words. For example, if you wish to search for the word **HALLO**, one would simply use: **HALLO** as a search pattern. However with wildcard characters we could do more interesting searches:

ite* would find words like: **item, iterate, ite**, but not **Item**
?it would find **Hit, hit, bit**, but not **it**
i*t would find words like **intelligent, int, it** but not **bit** or **its**
??? would find any 3-digit word, such as **bit, one, Ron**
a?* would find any word starting with an **a** such as **an, article**, but not **a** itself

Note that the wildcard character **?** may occur several times inside the search pattern, however the wildcard character ***** will only occur once

Write a program that, given a long text of one or more lines (no more than 100 words), will search for some input patterns. Define words as non-blank characters surrounded by blanks.

DATA31.txt (DATA32.txt for the second try) contains one or more lines of text, followed by a blank line; an integer N, followed by N search patterns.

For each search pattern, print out all the words it matches followed by a blank line. If the search yields no result, simple print the blank line.

SAMPLE INPUT:

```
Chris and Christopher went for a walk
They walked on and on for hours
She has walking boots and he only
running shoes

5
Ch*r
walk*
??
Tom
*??n?
```

SAMPLE OUTPUT:

```
Christopher

walk
walked
walking

on
on
he

(a blank line)

went
walking
running
```

76 Mark'em Up (Boardwide 1997)

Given a list of markup tags such as the ones used in HTML (Hyper Text Markup Language) and a file with a bunch of words including these tags, output the appropriate formatted text according to the rules. The output screen is assumed to have 80 characters in width with each position labeled 1, 2, ..., 80.

All of the markup tags that you need to consider are (all in lower case)

<title>	**<tab>**
</title>	**<para>**
<heading>	**<return>**
</heading>	

exactly as they appear here. All other markup tags, such as **<wrong>**, should be ignored and should NOT appear in the output.

GENERAL RULES FOR ALL TAGS AND FORMATTING:

Any space following immediately after a tag must be ignored.
Keep outputting word by word ignoring the return characters in the original text.
If a word does not fit on a line then start the complete word on the following line.

The following are additional rules for each tag:

1. **<title>** text goes here **</title>**
   Any text between the two tags <title> and </title>, we call the title. It must be centered and All the letters must be changed to uppercase letters if they are not. Add in two blank lines below the title. All other tags within these two must be ignored.
2. **<heading>** text goes here **</heading>**
   Any text between the two tags <heading> and </heading>, we call the heading.
   It must be left aligned and on the following line there must be a '-' character for each character and space of the heading.
   Add in one blank line below the '-' characters.
   All other tags within these two must be ignored.
3. **<tab>**
   Any text following this tag should start at the next position that is a multiple of 5. (position 5, 10, 15, etc. of the possible positions 1, 2,...80)
4. **<para>**
   Any text following this tag should start on a new line at position 6.
5. **<return>**
   Inserts a newline character: Any text following this tag should start on a new line.

DATA41.txt (DATA42.txt for the second try) contains several lines of text with tags as described. Read all lines until EOF is reached.

SAMPLE INPUT

Okay, here's the text. <title> This is the Title!</title> and <heading>
this is the heading </heading> and now for
some text <return> that is formatted in a variety of ways that involves using
tabs and returns among other things. <para>Here's a new paragraph and just to
make life interesting we
include here some tags like a <tab>tab and a <return>return. <para>
Now
not all of the tags should show. Incorrect tags, <wrong>, <incorrect>
and <dummy> should not be
included in the output. Finally, notice
the tags inside the next heading. <heading> Another
</head> <head> <tab> <return>heading</heading>
<tab>Well, the tags should be ignored!

SAMPLE OUPUT

```
....|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
Okay, here's the text.
                              THIS IS THE TITLE!



and
this is the heading
-------------------

and now for some text
that is formatted in a variety of ways that involves using tabs and returns
among other things.
     Here's a new paragraph and just to make life interesting we include here
some tags like a   tab and a
return.
     Now not all of the tags should show. Incorrect tags, , and  should not be
included in the output. Finally, notice the tags inside the next heading.
Another heading
---------------

    Well, the tags should be ignored!
```

77 Short Mazes (Regional 1997)

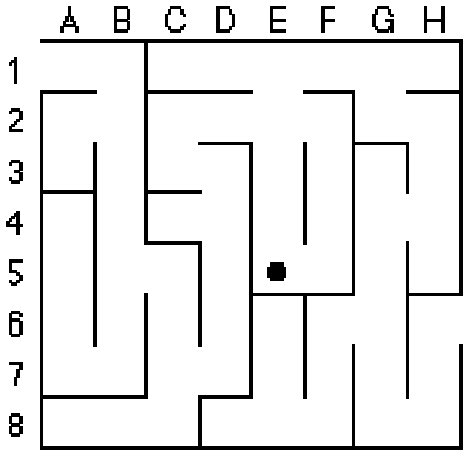
We have a two dimensional maze of a size no larger than 12x12. There will be no more than 3 external entrances with only one treasure location. Each maze will be described by a data file called DATA11.txt or DATA12.txt (for second attempt) with the following format. Line 1 is the size of the maze. Both the height and the width are always the same size. Line 2 is the co-ordinates of the treasure cell. The remaining lines of the data file describe each cell. The characteristic of each side of the cell is represented by either a 1 or a 0, where a 1 represents a closed side and a 0 represents an open side. Each side characteristic is separated by a space. The order of side descriptions are left, top, right, bottom for each cell. The descriptions start with the top left cell and work down the first column to the bottom and then go to the top of the next column and down that column to the bottom. This procedure continues until all the columns cells have been described.

Line 1: The number of outer entrances to the maze.
(One fifth of problem value)
Line 2: The shortest route to the treasure cell. (Expressed as number of cells passed through including the treasure cell).
(Two fifths of problem value)
Line 3: The longest direct route to the treasure cell. (Expressed as number of cells passed through including the treasure cell).
(Two fifths of problem value)

NOTE: In the event that there is only one route to the treasure cell the longest and the shortest routes will be the same.

DATA FILE

```
8
E5
0101 0110 1101 0101 0100 0101 0100 0111
1100 0010 1100 0101 0000 0110 1001 0110
1011 1010 1001 0110 1010 1010 1110 1010
1110 1010 1101 0010 1010 1010 1000 0010
1010 1000 0110 1010 1001 0011 1010 1011
1010 1010 1010 1010 1110 1100 0010 1100
1001 0011 1000 0011 1010 1010 1010 1010
1101 0101 0011 1101 0001 0011 1001 0011
```



PROGRAM OUTPUT:

```
Number of Entrances:  2
Shortest Route to Treasure Cell:  19 Cells
Longest Route to Treasure Cell:  21 Cells
```

78 Triangular Area (Regional 1997)

Problem:
Given the vertices of a triangle, find its area.

Input:
A file called DATA21.txt or DATA22.txt (for second attempt) that contains the vertices of 5 triangles, each ordered pair on a separate line, the coordinates separated by a single space.

Output:
The area of each triangle.

All answers must match our answer to within 1 unit (plus or minus) therefore language or machine precision limitations should not be a factor.

Formulae:
The area of a triangle can be computed by Heron's Rule when the lengths of all three sides a,b,and c, are known. The area equals

$$\sqrt{s(s-a)(s-b)(s-c)}$$

where s, the semi-perimeter is defined as `(a + b + c) / 2`

The length of the line segment a, b, or c between a point at (x_1, y_1) and a point (x_2, y_2) is

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Sample Input Data (2 of 5 triangles only!):

```
20 20
35 20
20 30
-12 3
31 20
12 -17
```

Sample Output:

```
75
634
```

79 Seeing Stars (Regional 1997)

Problem:
Use computer graphics to draw an N-pointed star on the screen.

Input:
The data file DATA31.txt or DATA32.txt (for second attempt) will contain five integers, greater than or equal to 5, that will represent the number of points that the star has.

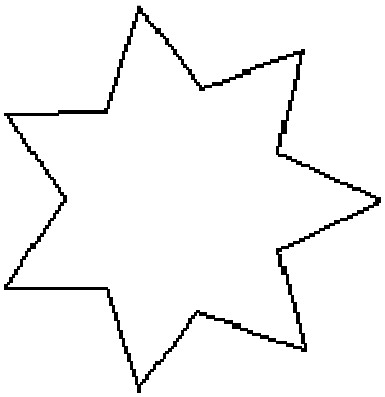
Output:
The star, in the form shown below printed in the middle of the screen. The star should occupy at least 1/2 of the screen's vertical height. The points of the star should be equidistantly aligned on an invisible circle. The troughs (low points) should also be aligned equidistantly on a (smaller) invisible circle.

Sample Input:

7
6
13
10
21

Sample Output for first sample input (7) item only.

Count the points on the star and visually inspect as to whether the points are equidistant.



Press any key to continue

80 Word Wrap (Regional 1997)

Magazines often display articles in columns. Your task in this problem is to write a program that will input text and re-format it into columns.

There will be three columns in your output, each with a maximum of 15 characters per line. Each line except the last, must also be formatted in full justification, meaning that both the left and right margins of each column are straight with the exception of when there is just one word in a line. Extra spaces must be inserted in the most balanced fashion possible, for example if there were three words in a column line with 11 characters, you would have to position four spaces to bring the total to 15 characters. In this case it could be done by putting two spaces between words. It would be wrong to put one space and then three. It the situation were changes to 12 characters, and tree words, then the spacing would be two spaces between the first pair of words and one space between the other. It the number of spaces between words cannot be equal, the higher number of spaces should appear left to right. You may assume that the maximum length of any one word is 15 characters. NOTE: To make spaces easy to read in your output, insert the character * for each space. you may assume that the character * will NOT appear in the input text.

The columns must be newspaper style, that is, the reader reads down the first column, then the second column and finally the third. The program should try to balance the length of each column with extra lines s shown below in the sample inputs.

Input
There are 5 data sets, each data set consists of one line that contains a number (< 6) and that number of lines of text, each no more than 70 characters in length. There is no punctuation in the input text. The file names are DATA41.txt and DATA42.txt for the first and second run.

Output
Each data output set should be preceded by a row of x's as shown below in the sample output. There should be two spaces between each set of x's. The columns must be formatted as described above and each space represented by an asterisk (*). Put all the output from the 5 data sets on the screen at one time, the data sets will be such that the correct output will never be more than 20 lines, including the rows of x's

Sample Input (this is one data set, the test data file will have five)

```
1
this is a sample dataline to show how the columns can be formed
from this longer line of text
```

Sample Output

```
xxxxxxxxxxxxxxx  xxxxxxxxxxxxxxx  xxxxxxxxxxxxxxx
this****is****a  how*the*columns  longer**line*of
sample*****data  can**be**formed  text
line***to**show  from*******this
```

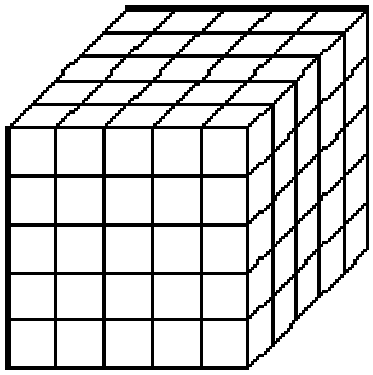
81 3D Word Hunt (Final 1997)

Find how many times do each of the words/phrases given in the input file appear in the 3 dimensional word search puzzle. The input data file called DATA11.txt or DATA12.txt (for the second attempt) has a first line that indicates the size of the 3D word search cube with one number indicating the number of letters along one side. (maximum 15) All the sides of the cube will be of the same length. Each of the following lines will be made up of letters OR SPACES to fill the cube starting in the top front line going from left to right. Each successive line will move down the front of the cube in subsequent lines. when the front square (face of the cube) is filled the subsequent squares behind the front will be filled in the same manner.

After the cube is filled, five words or phrases that will be no longer than the size of the 3D Search block (max 15 characters) will be listed one per line (5 lines of words/phrases only). Only words or phrases that are complete (all letters and spaces from beginning to end) are to be counted. You can start at any letter but you must proceed along straight lines from letter to letter. The letters of each word must proceed in normal progression i.e. not in reverse order. The following directions are permitted and are considered normal progression: top-down, left-right, front-back, diagonally top-down, diagonally front-back, and diagonally left-right.

Sample Input

```
5
ABCDE   LOVES
FELTJ   FGHII
ELMNO   RLMNO
PTRST   RQRLT
UALXY   ULETN
PRSEV   EVERY
FPHVJ   FJOIN
VEMEO   YLMNO
HQRRI   EQRST
UPOYN   UESXE
PHEAE   APPLE
HGHPO   BELT
EEPPO   EVERY
EQRLI   FELT
UPVEY   JOIN
```



In the sample, the 5x5x5 cube is filled with 25 words composed of 5 letters or spaces. the last 5 data, starting from APPLE, represent the query

Sample Output:

```
APPLE   occurs 4 times
BELT    occurs 1 times
EVERY   occurs 3 times
FELT    occurs 1 times
JOIN    occurs 4 times
```

82 Money Conversion (Final 1997)

A monetary conversion for a remote country. The basic monetary unit in this country is the *schlik*. One Schlik has a value of 2 Canadian cents and the only coins in this country are 1,5,13,23,37, and 47 Schliks. Your program is to answer the following questions. The input from the data file DATA21.txt or DATA22.txt (for the second attempt) will be two positive integers A and B each less than 100.

The output will do each of the following

1. Find the value of A schliks in Canadian dollars. (The dollar sign and the leading zero, if necessary, should be included)

2. Find the value of B schliks in Canadian dollars (The dollar sign and the leading zero, if necessary, should be included)

3. If a person had A schliks in coins, what coins do they have. Assume that they have the most 47 schlik pieces possible, then the most 37 Schlik pieces, etc. Express your answer as the 6-tuple: (A,B,C,D,E,F), representing the number of 47, 37, 23, 13, 5 and 1 Schlik coins respectively. The coins must be listed in the order stated with a comma but no spaces between each number. (See output example below)

4. If a person had B schliks in coins, what coins do they have. Assume that they have the most 47 Schlik pieces possible, then the most 37 pieces, etc. Express your answer as the 6-tuple: (A,B,C,D,E,F), representing the number of 47, 37, 23, 13, 5 and 1 Schlik coins respectively. The coins must be listed in the order stated with a comma but no spaces between each number. (See output example below)

5. How many ways are there to make change for A schliks.

Sample Input

```
12
41
```

Sample Output

```
$0.24
$0.82
0,0,0,0,2,2
0,1,0,0,0,4
3
```

83 Wheel of Fortune (Final 1997)

Two players, A and B, participate in this game of Wheel of Fortune. Player A always goes first. The object of the game is to reveal a secret phrase by guessing whether a specific letter appears in the phrase. The rules are as follows:

- If the player selects a consonant and that consonant is in the phrase, the player earns the amount of money the letter is worth, multiplied by the number of times the letter appears in the phrase. For example, if an S is worth $300 and there are two S's in the phrase, the player would earn $600.
- If the player selects a vowel (a,e,i,o,u), a flat fee of $200 is subtracted from the player's current earnings. The fee of $200 is subtracted regardless of whether or not the vowel appears in the phrase, or how many times.
- If the letter selected is not in the phrase, the player loses his turn. Conversely, if the letter is in the phrase, it is "revealed" to both players, the player continues his turn and selects another letter.
- The game ends when the letter selected is an asterisk. The player with the current turn is the winner.

DATA31.txt or DATA32.txt (for the second attempt) contains only one set of data. The set contains the phrase (input as a single string) followed by a series of guesses. Each guess is made up of an upper case letter followed by its value. Vowels will have a value of 200. The series ends with an asterisk; it will have a value of 0.

We guarantee that all data will be valid, and that a player selecting a vowel will always have enough money to pay for it. The phrase will consist of upper case letters and blanks; it will be less than 50 characters long and will have neither leading or trailing blanks.

Display the following as shown below:
- The winning player.
- The amount accumulated by the winning player.
- How many letters of the phrase were correctly selected.
- How many words of the phrase were fully revealed. That is all letters of the word that were correctly selected.
- The phrase at the end of the game. Letters that have not yet been revealed should appear as a question mark.

Sample Input Data

```
COMMEMORATIVE COIN
T 200
H 400
M 300
O 200
S 500
C 350
* O
```

Sample Output Data

```
A
900
9
0
COMM?MO??T???CO??
```

84 In the Dog House (Final 1997)

Philip has a summer job working at the Kay-9 Kennels. He has been working afternoons exercising the dogs, but this week he has to work the overnight shift.

By midnight he knows that all of the dogs have been asleep for hours and he is getting quite bored.

He starts making his rounds at exactly midnight.

As he visits each of the pens, Phil unlocks the gate. All the gates start off locked. After the first trip all gates are open. On the second round he locks every second gate. (i.e. 1,3,5, ..) and continues making subsequent rounds locking or unlocking the pens as he finds them. On the third pass he changes every third lock. (1,4,7, ..) On the fourth pass he changes every fourth lock. This process continues for W (the value for W is given in the data file) trips or until the time runs out. Every trip starts at the first pen.

Input:
The data file DATA41.txt or DATA42.txt (for second attempt) will have on subsequent lines:
The number of gates (Q)
The number of seconds to lock or unlock a pen. (Y)
The number of seconds to travel between pens (Z)
The maximum number of trips (W)
The time the burglar alarm will go off. (HHMMSS) (X)

Output:
The answers to the following questions are to be determined:

1. If it takes him Y seconds to open or close a lock and Z seconds to travel between pens, at what time would Philip have completed the W trips? (Hours, minutes and seconds)
2. How many pens would be locked if W trips where completed
3. How many pens would be unlocked when the burglar alarm goes off at exactly X hrs, awakening all the dogs? (assume that if the time to turn the lock is not complete that the lock's state has not changed by a partial time)
4. What trip would Philip be on at the time the alarm went off?
5. What door would Philip be working on or on his way to at the time the alarm went off? (after a door is locked or unlocked, Philip is considered to be working on the next door)

Sample Input Data

```
50
2
3
10
000110
```

Sample Output Data

```
00:12:30
26
14
1
15
```

85 Knight Moves (Boardwide 1998)

Consider a chess board, where each position on the board is represented by an ordered pair. A knight's move is a move that goes from a black position to any of the nearest white positions, that is not touching it, or from a white position to the nearest black position, not adjacent to it. Mathematically it moves from (x,y) to (x+a,y+b), where |a|+|b|=3.

Write a program that finds the fewest number of knight moves from any one square to any other.

In particular, the input text file **DATA11.txt** (for the first try) and **DATA12.txt** (for the second try) contains 5 lines, each containing 4 integers: a b c d, separated by one or more spaces, representing the starting position and the finishing position of the knight.

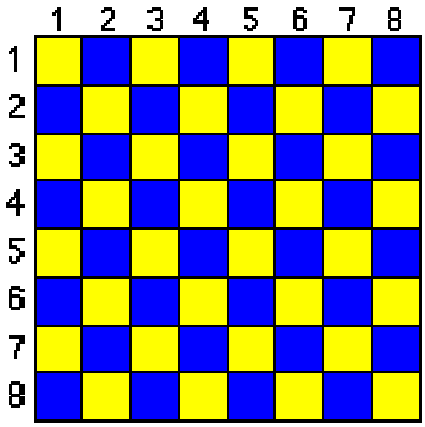
The output should be as in the following example:

Sample Input:

```
3 7 8 2
1 1 2 2
7 6 1 7
3 3 3 3
7 2 5 1
```

Sample Output (on a cleared screen):

```
There are 4 knight moves from (3,7) to (8,2)
There are 4 knight moves from (1,1) to (2,2)
There are 3 knight moves from (7,6) to (1,7)
There are 0 knight moves from (3,3) to (3,3)
There are 1 knight moves from (7,2) to (5,1)
```



86 Knight Moves (Boardwide 1998)

A maze can be constructed by using a set of 1's and 0's
Consider the example at the bottom of this page. The maze is composed of a 10 by 10 grid of cells, where each cell has some of its edges missing.

Let's look at the top row of cells.
Ignoring the tops and bottoms of each cells, we have: 

Here the dotted lines represent the missing edges of cells.
This row may be represented by: 11001000011
Columns may be constructed in a similar way.
The first column in this example, read from top to bottom is: 10000100101

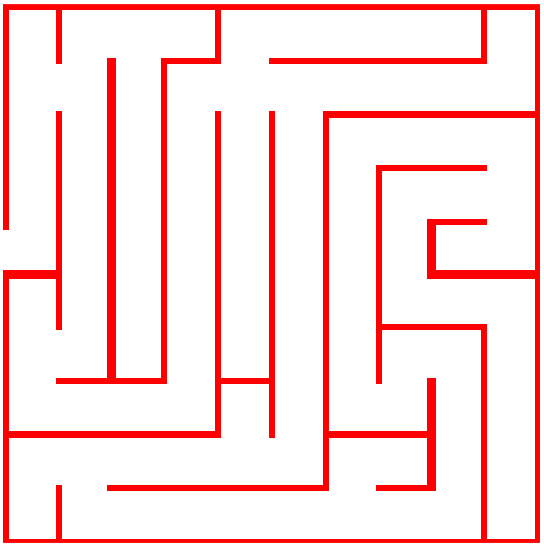
DATA21.txt (for the first try) and **DATA22.txt** (for the second try), contains information on 3 mazes: First a number representing the size of the maze (always a square), followed by all row data (from top to bottom) , followed by all column data (from left to right). The size of each maze is no larger than 20.

Output each maze on a cleared screen, and wait for any character to be pressed before the next maze.

Sample Input (one maze in this example)

```
10
11001000011
10110000001
11111110001
11111111001
01111111101
11111111001
10111111011
10001110111
10000010111
11000000011
10000100101
10000001101
10000001111
11000000111
10000001011
11000000011
11000001001
11110010111
11111110001
```

Sample Output



```
press any key to continue
```

87 The Pope's Cipher (Boardwide 1998)

In the sixteenth century, **Matteo Argenti** was the Pope's secretary.
In order to protect the Pope's correspondence from prying eyes, he developed the following code.
He gave each letter a numeric code, but to make it more difficult to decipher, he inserted the following hurdles as well:

- Since the letter **E** occurred so frequently, it should have 2 codes: 03 and 88.
- Some letters should be represented by one digit, some by two digits
- To confuse things even further, certain meaningless numbers (nulls) should be inserted at random, even between the two digits that represents a certain letter, so that for example 373 would still represent **L**.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
6	10	11	13	03 88	15	01	18	08	30	31	33	35	05	38	50	51	00

S	T	U	V	W	X	Y	Z
53	2	55	58	80	81	83	85

Nulls
4 7 9

Write a program that will read an encoded message in the Pope's cipher, and that will print out the decoded message. The input file DATA31.txt (for the first try) and DATA32.txt (for the second try) will contain 5 messages, each no longer than 255 digits. Place a blank line between each message.

Sample Input data (only two of 5 messages are displayed)

```
5303054131888733505415508113133983030548839583627276113108905013805848503355533890588085732803 8
```

Output:

```
SENDHELPQUICKLYENEMYATTACKING

ONEPLUSONEISTWO
```

88 Equation Writer (Boardwide 1998)

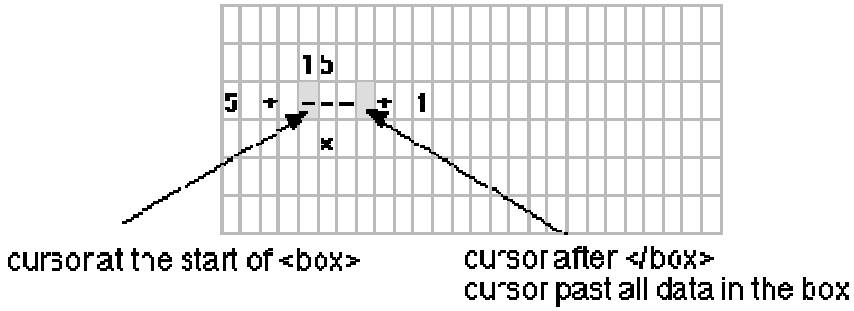
Equation Writer Unlike text, mathematical equations are frequently written in two dimensions. To assist in the representation of mathematical expressions, the following tags are made available, all in lower case letters:

<up>	Moves the cursor up one line
<down>	Moves the cursor down one line
<back>	Moves the cursor down one line, and to the beginning of the screen or the left edge of the current box
<box>	Starts a "box", which is a subset of the screen, with a newly defined left edge. Boxes may be embedded (see the second example)
</box>	Ends the last defined "box" and moves the cursor on the same vertical line as when it entered the box, but in the vertical position so that all data from the box is to the left of it.

Note: All characters, even spaces are significant, and should be printed.

e.g: **5 + <box><up>15<back>---<back> x</box> + 1**

will result in:



Write a program that reads from the input file DATA41.txt (for the first try) and DATA42.txt (for the second try) five lines of data, printing out the expressions one at the time. Each next expression should appear on a cleared screen, after any key is pressed. Each input line contains fewer than 255 characters.

The Sample input (see next page) only contains two of 5 data sets.
blank lines are added here for easier viewing
periods are added to represent spaces

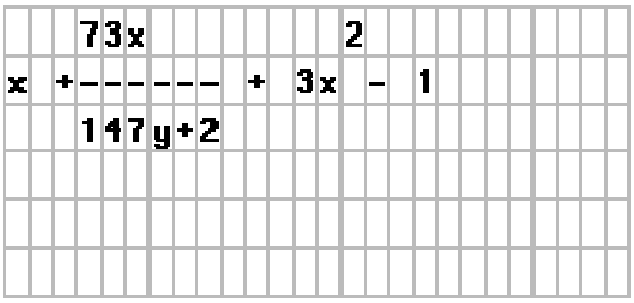
The actual input files contains 5 data lines; there are no blank lines; spaces are represented by spaces.

Sample Input

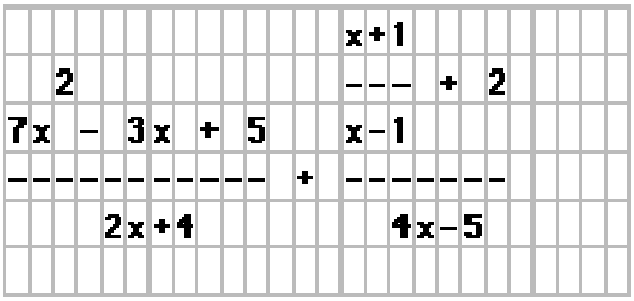
```
<down>x.+.<up><box>73x<back>------<back>147y+2</box><dow
n>.+.3x<up>2<down>-.1

<down><down><box>7x<up>2<down>-.3x.+.5<back>----------<b
ack>....2x+4</box><down>.+.<up><up><up><box><box>x+1<back
>---<back>x-1</box><down>.+.2<down><back>-------<back>..4
x-5</box>
```

Sample Output

on a cleared screen:



on a cleared screen:



89 Secret Code (Regional 1998)

A secret agent uses a system to code and decode his messages. He first creates a 6 x 6 array of letters and digits as follows:
First he uses the 26 letters of the alphabet in order.
Then he uses a keyword to insert digits in appropriate places in the alphabet.
If for example the keyword were “Top Secret”, he uses the digits 1234567890, and inserts these digits in the alphabet, before the corresponding letters.

0AB6CD58EFGHIJKLMN2O3PQ7R4S19TUVWXYZ

Notice that:

- The digit 5 (corresponding to E) is inserted before the E, and later the 8 (corresponding to the second E) is also placed before the E
- Since the keyword contains only 9 letters, only 9 digits are used.
  The remaining digits are placed, in order, before the entire set.
  In this case only the digit 0 is placed before the entire set.
- If the keyword contains blanks, or other characters, they are ignored.
  Also, if the keyword contains more than 10 letters, the extra letters are ignored also.

  The string of 36 characters are next arranged in a square (the template):

	1	2	3	4	5	6
1	0	A	B	6	C	D
2	5	8	E	F	G	H
3	I	J	K	L	M	N
4	2	0	3	P	Q	7
5	R	4	S	1	9	T
6	U	V	W	X	Y	Z

Each letter and digit is now uniquely defined by an ordered pair, (a,b), where a is the row, b is the column. He then translates his message in three steps:
Assume his message says: "**What is the weather like, down there in Mexico?**"
The message first becomes:
`WHATISTHEWEATHERLIKEDOWNTHEREINMEXICO`
Next all letters are translated to their corresponding pairs of digits:
`63/26/12/56/31/53/56/26/23/63/23/12/56/26/23/51/34/31/33/23/16/42/63/36/56/26/23/51/23/31/36/35/23/64/31/15/42`
the digit 1 is placed before and after the new string giving:
`16/32/61/25/63/15/35/62/62/36/32/31/25/62/62/35/13/43/13/32/31/64/26/33/65/62/62/35/12/33/13/63/52/36/43/11/54/21`
Then using the same square, this string is translated back to its final encoded form:
`DJUGWCMVVNJIGVVMB3BJIXHKYVVMAKBW4N3015`

What is convenient about this code, is that the encoding and decoding methods are identical: When you submit the encoded message to the same procedure, you get the original message back, ...with an extra letter before and after the message.

(This method is a modification of a code described in Andrew Pennycook's CODES and CIPHERS, David McKay Company, Inc, New York.)

Write a program that reads 5 sets of data from the file DATA11.txt (DATA12.txt for the second try). Each set consists of two lines, no more than 80 characters each. The first line represents the keyword, and the second line is the text to be encoded. The output, on a cleared screen should contain 5 sets of two lines, separated by a blank line: The keyword coded template and the encoded phrase.

Sample Input

```
Top Secret
What is the weather like, down there in Mexico?
Contest
BXNCQ7ICPV2FHO7057W0FBW1SEFW9Q63PA5EDPVNOE5EDPZ62DWBU
No
We don't have bananas
The rain in Spain falls mainly in the Plane
My Fair Lady
The sound of Music
CYYT89RGOBDK5AT4VAFBORDAA4A
```

SampleOutput

```
0AB6CD58EFGHIJKLMN2O3PQ7R4S19TUVWXYZ
DJUGWCMVVNJIGVVMB3BJIXHKYVVMAKBW4N3015

890AB1CD5EFGHIJKLM3N2OPQR6S47TUVWXYZ
8TODAYSDATEISAPRILTHIRTHYINTHEYEARNINETEENNINETYEIGHT8

34567890ABCDEFGHIJKLM1N2OPQRSTUVWXYZ
8G4Y6SWLJA4LHPHPIO

5ABCD3EFG2H68IJKLM79NOPQ4R0S1TUVWXYZ
BZREGHG4ET4

ABC8D3E0FG2HIJKLM7N59OPQR4S1T6UVWXYZ
A76TROMBONESLEDTHEBIGPARADEA
```

90 Othello (Regional 1998)

Othello is a game played by two players ("BLACK" and "WHITE" represented by "B" and "W") on an 8 x 8 checker board. Players alternate turns by placing one of their men on the board so that the man “flanks” one or more on his opponent’s men. There can be no intervening blanks. A flanking move may occur in several directions from your placement. Any men that are flanked are "flipped" over and become your colour.

You will read from a data file DATA21.txt (DATA22.txt on the second attempt) eight 8-character strings which will be a partially completed Othello game. Note that a "-" will indicate an empty square. You will use this same original board for each of the possible moves (don’t update the board).

The next five lines will each contain three data items, the row, column and colour ("B" or "W") for a possible move.

Output for each move, whether the move is valid or not. If it is valid then output the number of men flipped for that move.

Sample Input

```
--------
WBB-BW--
--WBWBW-
-B-BW---
--BWBW--
---B----
--B-----
--------
6 7 B
7 3 W
2 4 W
2 7 B
3 2 W
```

Sample Output

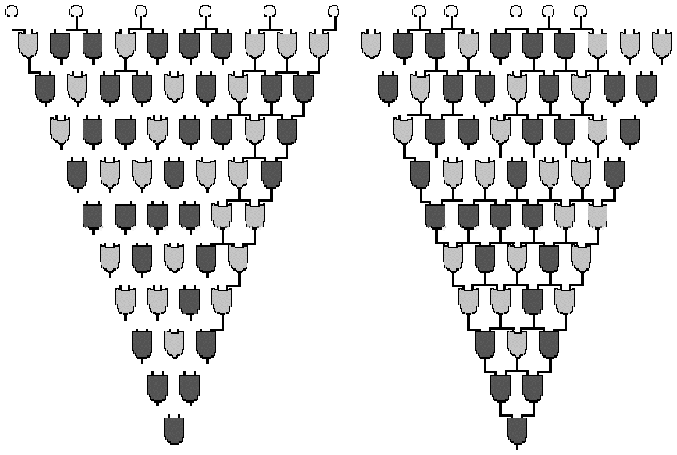
```
LEGAL MOVE WITH 2 FLIPPED
ILLEGAL MOVE
LEGAL MOVE WITH 5 FLIPPED
LEGAL MOVE WITH 1 FLIPPED
ILLEGAL MOVE
```

91 Boolean Tree (Regional 1998)

Imagine a tree composed of OR gates and AND gates in the form of an inverted triangle. The idea is that you send a set of signals down the tree. Two signals are needed for an AND gate to produce a new signal to the next level of the tree (down); only one signal is needed for an OR gate to produce a new signal to the next level.

The challenge is to produce a signal at the bottom of the tree.

The output signal generally becomes input to two new logic gates. Notice that no signal reaches the bottom of the tree on the left of the diagram, although there are 6 signals at the top. The right hand example shows the same tree with only 5 inputs, and yet the signal reaches the bottom.



The input file (DATA31.txt on the first try and DATA32.txt on the second) lets you read a boolean tree of between 2 to 12 levels inclusive: The first line contains an integer representing the number of levels to the tree (say n). This is followed by n strings of letters; These strings contain combinations of "O" and "A". the letter "O" represents an OR gate and the letter "A" an AND gate.

Write a program that will find the LEAST number of input signals needed to get one output signal, and then list all the combinations of these signals as a string of 1's and 0's, 1 standing for the presence of a signal and 0 where there is no signal.

Sample Input

```
8
OAOOAOAA
OOOAOOA
OAOAOO
OAAAO
OOAO
OOO
OA
A
6
OOOOOA
AOAOA
OOOO
AAA
AO
A
2
AO
A
```

Sample Output

```
The lowest number of signals is: 2
000011000
000100100
000101000
001000100
001001000
The lowest number of signals is: 1
0001000
The lowest number of signals is: 2
110
```

92 Special Number (Regional 1998)

The number 6174 is indeed a very special number. In this problem, you will illustrate just why this number is so special.

Take any four digit number. Arrange the digits of that number so that they are in descending order to create the biggest number that can be formed by these digits. Similarly, arrange the digits in ascending order to create the smallest possible number that can be formed.

Now it gets interesting: Subtract the smallest number from the largest and take a look at the answer. If the answer is not 6174, use your answer as the next 4 digit number to repeat the exercise. Within eight attempts, you answer will be 6174!

Input
The user will read from the file DATA41.txt (DATA42.txt if it is the second attempt) five four-digit integers as the starting numbers. If the number is unsuitable print the message, "Invalid Input". See for sample input the number 2019.

Output
Clear the screen before processing each number, then show the detailed calculations until the answer 6174 is generated. Use the following output as an example of the required formatting for the first sample data number of five that would be in the file. Prompt for a keypress for continuation after each number's solution is displayed. All numeric output should be justified to column 50. Scrolling of the screen is allowed if required to accommodate all of the tries for each data item.

Sample Input (one number in this example)

```
 2019
```

Sample Output

```
Try #1  9210
       -0129
       -----
        9081    Not Yet

Try #2  9810
       -0189
       -----
        9621    Not Yet

Try #3  9621
       -1269
       -----
        8352    Not Yet

Try #4  8532
       -2358
       -----
        6174    It works !
```

93 The Trouble with Tribles (Final 1998)

Tribles were first encountered in the 23th century by captain Kirk and his crew. They go through a unique life cycle:
On the 2nd day of any given week they are born from a male and female trible.
On the 1st day of the 2nd week they become male
On the 1st day of the 4th week they become female
On the 1st day of the 6th week they become seniors.
Depending on from where in space they come from, seniors may live from 1 to 5 weeks as seniors, before they die, and they die on the 1st day of the following week

Your task is to simulate the population growth of tribles:
Imagine a 15 by 15 grid.
If there is an empty cell on the grid, a new trible will be born there, provided there is at least one male and one female in any of the 8 adjacent cells
When a trible dies, of course, the cell becomes empty.
The bottom left cell is associated with the ordered pair (1,1)
The bottom right cell (15,1), top left: (1,15) and top right: (15,15)
The other cells are similarly associated with the ordered pairs, i.e.(x,y).

The input file (DATA11.txt for the first try, DATA12.txt for the second) contains five sets of 8 integers: Information on two tribles at the end of a week, the expected lifetime of each individual in the population, and the number of weeks the population is allowed to grow.

For example the data: 12, 6, 3, 13, 6, 2, 7, 5 means, that at the start of a week:
A trible, aged 3 weeks resides in cell (12,6)
A trible, aged 2 weeks resides in cell (13,6)
Tribles die after 7 weeks
Determine the population at the end of 5 weeks.

On your output screen draw a diagram of the population:
A newborn trible is represented by a yellow circle;
a male trible by a blue circle;
a female trible by a red one;
and a senior by a green circle.

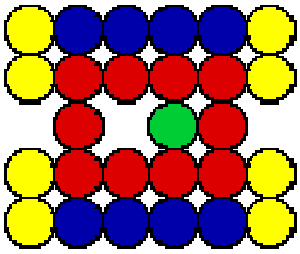
Step by step:
During the first week, one trible becomes female, the other remains male, and they have 4 children. And so, at the end of:

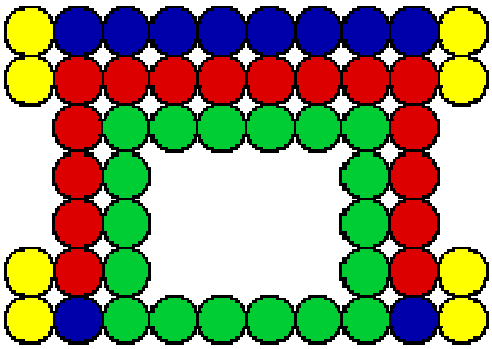
week 1	week 2	week 3	week 4	week 5
			CCCC	CHHHHC
CC	CHHC	HHHH	HFFH	CFFFFC
FH	CFFC	HSFH	HSSH	F SF
CC	CHHC	HHHH	HFFH	CFFFFC
			CCCC	CHHHHC
ages:	ages:	ages:	ages:	ages:
			1111	122221
11	1221	2332	3443	145541
43	1641	2662	3763	4 74
11	1221	2332	3443	145641
			1111	122221

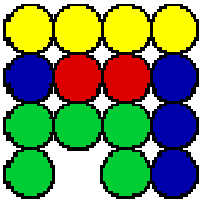
Sample Input (three of the five data sets)

```
12 6 3 13 6 2 7 5
5 3 1 6 3 2 8 12
1 1 1 2 1 3 9 7
```

Sample Output
Your output should only show the last step.







94 Tetrahedrons (Final 1998)

A tetrahedron is a 3-dimensional shape having 4 vertices: It is a pyramid on a triangular base. Your task is to write a program that will check whether a given point is inside a tetrahedron or not.
Note, that if the point is on the surface, the point is considered inside.

Input (DATA21.txt for the first try and DATA22.txt for the second) consists of 5 sets of 5 ordered triples: The first 4 are the vertices of the tetrahedron and the 5th is the point to check. Note that no entry is larger than a 2 digit integer.

Output the volume of the tetrahedron and whether the fifth point is inside the tetrahedron or not. Note that the volume of the tetrahedron with one vertex the origin (0,0) and the other three points: $A(a_1,a_2,a_3)$, $B(b_1,b_2,b_3)$ and $C(c_1,c_2,c_3)$ is:

$$\left| \frac{(a_1b_2c_3 + a_2b_3c_1 + a_3b_1c_2) - (a_3b_2c_1 + a_1b_3c_2 + a_2b_1c_3)}{6} \right|$$

As well, round off the volume to two digits behind the decimal point.

Sample Input (4 sets only in this sample)

```
0 0 0 0 1 1 2 2 2 1 1 0 0 1 1
5 1 6 8 9 0 12 -3 6 4 67 3 4 4 3
5 5 5 5 5 11 5 6 5 6 5 5 5 6 6
5 4 1 8 -3 7 14 -18 0 -8 2 14 5 -2 5
```

Sample Ouput

The point (0,1,1) is inside the tetrahedron of volume 0.33
The point (4,4,3) is outside the tetrahedron of volume 424.00
The point (5,6,6) is ouside the tetrahedron of volume 1.00
The point (5,-2,5) is inside the tetrahedron of volume 326.67

95 Wacky Markup (Final 1998)

This Markup language contains only the following tags:
(which may be in upper or lower case)

 forces an end of line character (paragraph)
<LOWER> and </LOWER> left and right container tags: forces all text to be in lower case, for example: "<lower>HeLLo DoLLy</lower>" becomes: "hello dolly"
<UPPER> and </UPPER> left and right container tags: forces all text to be in upper case, for example "<upper>HeLLo DoLLy</upper>" becomes: "HELLO DOLLY"
<PROPER> and </PROPER>left and right container tags: forces all words of text in lower case, with the first letter in capitals, for example,
"<proper> HeLLo DoLLy </proper>" becomes: "Hello Dolly"

Furthermore:
Words are separated by spaces or tags
Two or more spaces in a row are to be replaced by one space.
In the input, all container tags are properly embedded, for example:
<lower>...<proper>...</proper>...<upper>...</upper>...</lower>
Each input line is no more than 80 characters in length.

Write a program that will read from an input file (DATA31.txt for the first try and DATA32.txt for the second) five lines of no more than 80 characters. Print out the data after properly interpereting the tags. Each output must be separated from the others by a blank line.

Sample input

```
<proper>One</proper> day,<BR><upper> Henny Penny</upper> was picking up corn.
<UPPER>an acorn<LOWER> fell OUT of the TREE</LOWER> and struck her</UPPER> HeaD
<PROPER>goodness GRACIOUS<LOWER> ME!</lower></PROPER><UPPER>said HENNY</UPPER>
Where are you going<proper><br>henny penny?<br></proper>asked Cocky Locky
<lower>THE SKY IS FALLING<UPPER>and I must<br>tell</upper>the King</lower>
```

Output

```
One day,
HENNY PENNY was picking up corn.

AN ACORN fell out of the tree AND STRUCK HER HeaD

Goodness Gracious me! SAID HENNY

Where are you going
Henny Penny?
asked Cocky Locky

the sky is falling AND I MUST
TELL the king
```