FLORIDA HIGH SCHOOLS COMPUTING COMPETITION '85

1.1 Write a program to place numbers on a stack with the options of retrieving from or adding to the stack. A Stack is a Last In--First Out (LIFO) type of storing. The user may enter one of three commands: ADD to the stack; TAKE from the stack; or QUIT. If ADD is entered, then the program accepts a value to place on the stack. If TAKE is entered, then the last value placed on the stack is removed and displayed. If QUIT is entered, then the program ends. Example:

```
 INPUT: Enter command: ADD
        Enter number: 80
        Enter command: ADD
        Enter number: 45
        Enter command: ADD
        Enter number: 98
        Enter command: TAKE
OUTPUT: 98
 INPUT: Enter command: TAKE
OUTPUT: 45
 INPUT: Enter command: QUIT
OUTPUT: (program ends)
```

1.2 A teacher writes a set of N consecutive positive integers beginning with 1 on a blackboard. The teacher then erases one of the numbers. Write a program to find the number that was erased if the average of the remaining integers is AV, where N and AV are given as input. Example:

```
 INPUT: Enter N, AV: 11, 4.7

OUTPUT: NUMBER ERASED WAS 9
```

1.3 Write a program to round the square root of a given number at a designated place value and then to sum the digits in the rounded number. The program will accept as input a whole number, N, and an integer between 3 and -4 inclusive for the exponent of the power of 10, D. Let S equal the square root of N rounded to the nearest 10^D, and then display the number S in the form S=####.#### and display the sum of all the digits of S next to the phrase "SUM=". Examples:

```
 INPUT: Enter N, D: 2, -3
OUTPUT: S=   1.4140
        SUM=10

 INPUT: Enter N, D: 625, 1
OUTPUT: S=  30.0000
        SUM=3
```

1.4 The year is 1985. As the U.S.S. RETUPMOC spacecraft whirls through space, Captain James T. Irk notices that the time dial is malfunctioning and is increasing the year every second. He then notices that the years are steadily counting faster and faster. The next thing that he realizes is that the spacecraft has crashed and the time dial reads 2345. Write a program to simulate the behavior of the time dial, displaying the year in the center of the screen. Beginning with the year 1985, increase the count by one year approximately every second for several seconds, and then begin to steadily and rapidly increase the years with less time between each succeeding display until it is counting faster than the eye can comprehend, ending with the year 2345, which remains on the screen. The displays should not flicker. The program should take less than 60 seconds to run.

1.5 In a Tennis tournament there are N number of participants. In the first round there are N/2 or (N-1)/2 games depending on whether N is even or odd respectively. If N is odd then there must be one bye. A bye allows a player to proceed to the second round without playing in the first round. For successive rounds, the number of players is equal to the number of games and byes in the previous round. Write a program that will prompt the user for the number of participants, N, between 2 and 99 inclusive, and will then calculate the number of games and byes starting at the initial round through the final round. For each round and the grand total, print the number of games and byes in the format shown below. Example:

```
 INPUT: Enter N: 27

OUTPUT: ROUND 1  13 GAMES  1 BYE
        ROUND 2   7 GAMES
        ROUND 3   3 GAMES  1 BYE
        ROUND 4   2 GAMES
        ROUND 5   1 GAMES
        TOTAL    26 GAMES  2 BYES
```

1.6 Write a program to find the smallest, the largest, and the sum of all 3-digit numbers, whose 3 digits are non-zero and distinct, between N and M inclusive, where N and M are input as positive integers with N being less than M. Example:

```
 INPUT: Enter N, M: 90, 125

OUTPUT: SMALLEST = 123
        LARGEST = 125
        SUM = 372
```

1.7 Bob Simon of Bob's Cycle Shop wants to improve his billing procedure and decides that a computer program that would calculate values for an invoice and then prints the invoice would be very helpful. Write a program for Bob! Allow him to input the customer's name, product number of part sold, and labor time in hours. Bob charges $10 per hour for labor. The program must prepare an invoice on the screen with the following information: customer name, part # and description, part cost, labor cost, invoice total including 5% sales tax on part cost, rounded to the nearest penny. All amounts must be displaye in the form ###.## as shown below. Bob has seven parts that are to be used in this program.

Part#	Description	Cost
S193	10 INCH SPROCKET	13.95
S867	30 INCH CHAIN	27.50
F234	BLITZ MAG FRAME	119.00
S445	COMPUTCYCLE COMPUTER	33.95
C492	JET BRAKE SET	29.98
J273	27 INCH WHEEL	32.00
T100	27X1 INCH TIRE TUBE	12.50

Example:

```
 INPUT: Enter name: CRAIG
        Enter part#: F234
        Enter time: 10.5

OUTPUT: CUSTOMER NAME: CRAIG
        PART #: F234
        DESCRIPTION: BLITZ MAG FRAME
        PART COST:  119.00
        LABOR COST: 105.00
        5% TAX:       5.95
        TOTAL:      229.95


 INPUT: Enter name: BARBARA
        Enter part#: S193
        Enter time: 4

OUTPUT: CUSTOMER NAME: BARBARA
        PART #: S193
        DESCRIPTION: 10 INCH SPROCKET
        PART COST:   13.95
        LABOR COST:  40.00
        5% TAX:       0.70
        TOTAL:       54.65
```

1.8 Write a program that will prepare a set of labels (on the screen of course), given the number of lines on the label as input (between 3 and 5 inclusive). Each label will have blanks on the first line, the individual's name (last name first, comma and space then first name) on the second line, and the telephone number on the third line. The following names are to be stored in the program either in DATA statements or in arrays:

```
DATA LISA SPINXS, 987-6543
DATA BOB SIMON, 923-4455
DATA BILL SIMON, 123-4567
DATA HARRY TROUTMAN, 876-2174
DATA HARRY PARKER, 222-3333
DATA *END*, 0
```

DATA or array statements will be terminated by "*END*, 0". The program will then alphabetically sort the entries on the basis of last name, then first name. After printing the first three lines of the label, the program is to skip the appropriate number of lines before printing the next label. Example:

```
     INPUT: Enter # of lines on label: 5
    OUTPUT:
            PARKER, HARRY
            222-3333



            SIMON, BILL
            123-4567



            SIMON, BOB
            123-4455



            SPINXS, LISA
            987-6543



            TROUTMAN, HARRY
            876-2174
```

1.9 Write a program that will randomly generate a 5x5 matrix of 25 letters, A through Y, centered on the top part of the screen, with every adjacent letter on a row separated by a space. Allow the user to think of a secret letter. The computer must ask the user yes(Y)-or-no(N) questions to logically determine the secret letter (using similar questions as shown in the examples). The computer will start with a score of 11 points and will deduct 1 point for each question that is asked and answered. Display the score in the upper right corner after each question is asked. If the program does not determine the letter before the computer score reaches 0, then no credit is awarded at this time (try again). Possible examples:

RUN PROGRAM:

```
    OUTPUT:         Q W E R T         SCORE=11
                    Y U I O P
                    A S D F G
                    H J K L M
                    X C V B N

     OUTPUT/INPUT: IS THE LETTER IN ROW 1? N
     OUTPUT: (The score decreases to 10 at the top right)
     OUTPUT/INPUT: IS THE LETTER IN ROW 2? Y
     OUTPUT: (The score decreases to 9)
     OUTPUT/INPUT: IS THE LETTER IN COL 1? N
     OUTPUT: (The score decreases to 8)
     OUTPUT/INPUT: IS THE LETTER IN COL 2? Y
     OUTPUT: (The score decreases to 7)
             YOUR LETTER IS U
```

RUN PROGRAM:

```
    OUTPUT:         X C V B N         SCORE=11
                    M L K J H
                    A S D F G
                    P O I U Y
                    T R E W Q

     OUTPUT/INPUT: IS THE LETTER IN ROW 1? N
     OUTPUT: (The score decreases to 10 at the top right)
     OUTPUT/INPUT: IS THE LETTER IN ROW 2? N
     OUTPUT: (The score decreases to 9)
     OUTPUT/INPUT: IS THE LETTER IN ROW 2? Y
     OUTPUT: (The score decreases to 8)
     OUTPUT/INPUT: IS THE LETTER IN COL 1? N
     OUTPUT: (The score decreases to 7)
     OUTPUT/INPUT: IS THE LETTER IN COL 2? Y
     OUTPUT: (The score decreases to 6)
             YOUR LETTER IS S
```

1.10 Write a program that will allow the user to locate the cursor on the screen by using the keys I,J,K, and L to move up, left, right, and down respectively. Upon pressing 1,2,3, or 4, a square is drawn relative to the cursor's position, if possible, as shown below, using the # symbol to designate the cursor.

```
#                        #
*********       *********        *********       *********
*       *       *       *        *       *       *       *
*   1   *       *   2   *        *   3   *       *   4   *
*       *       *       *        *       *       *       *
*********       *********        *********       *********
                                         #       #
```

If the picture would print off the screen, then display the error message "OFF THE SCREEN" instead. Examples:

```
RUN PROGRAM: Press the appropriate keys I,J,K,M to place the
             cursor in the center of the screen.
     INPUT: 1
    OUTPUT: The box below with respect to the cursor (#):

                 #
                 *********
                 *       *
                 *   1   *
                 *       *
                 *********


RUN PROGRAM: Place the cursor in the center of the screen.
     INPUT: 3
    OUTPUT:      *********
                 *       *
                 *   3   *
                 *       *
                 *********
                         #


RUN PROGRAM: Place the cursor at the absolute right side of the
             screen.
     INPUT: 4
    OUTPUT: OFF THE SCREEN


RUN PROGRAM: Place the cursor at the bottom of the screen.
     INPUT: 2
    OUTPUT: OFF THE SCREEN
```

2.1 Write a program to outline the border of the screen with a random letter; then when the space bar is pressed, the inside border of the new screen will be outlined by a random letter; afterwards, when the space bar is pressed, the inside border of the new screen will be outlined by a random letter, and so on. Continue drawing these rectangles until the whole screen is filled, then press the space bar once again and the screen will clear and start over with a new outer border. A miniature sample run would look like this:

```
RRRRRRRRRRR          RRRRRRRRRRR          RRRRRRRRRRR
R         R          RQQQQQQQQQR          RQQQQQQQQQR
R         R          RQ       QR          RQYYYYYYYQR
R         R          RQ       QR          RQYYYYYYYQR
R         R          RQQQQQQQQQR          RQQQQQQQQQR
RRRRRRRRRRR          RRRRRRRRRRR          RRRRRRRRRRR
```

2.2 Write a program to find and print the longest sequence of letters in alphabetical order in a set of N letters entered one at a time. If two or more sequences tie for the longest, then print each of the sequences on successive lines. Each letter must be separated by a space. If a letter repeats, such as the letter "D" in the series of letters A,B,C,D,D,E, then the first sequence is considered to end at the first "D" and the second sequence begins at the second "D". Example:

```
 INPUT: Enter N: 5
        Enter letter: J
        Enter letter: E
        Enter letter: S
        Enter letter: S
        Enter letter: U
OUTPUT: E S
        S U


 INPUT: Enter N: 7
        Enter letter: S
        Enter letter: I
        Enter letter: S
        Enter letter: L
        Enter letter: O
        Enter letter: R
        Enter letter: D
OUTPUT: L O R
```

2.3 Write a program to simulate a word processor: Allow the program to accept as input a paragraph (one string with no leading spaces and no more than 127 characters in length) and to print the words out on the screen providing a 5-character left margin and no more than 30 characters on a line. Do not split up words. A word is defined as a set of characters in between two spaces (except for the first and last words of the string). The first word of the paragraph is to be indented 5 spaces. The last word of a sentence is followed by a period and 2 spaces (as input), except for the last word of the paragraph which is followed by just a period. New sentences beginning at the left margin are not to be indented by the trailing spaces of the previous sentence. Example:

```
 INPUT: Enter text: DO NOT SPLIT UP WORDS.  THE FIRST WORD OF
                    THE PARAGRAPH IS TO BE INDENTED 5 SPACES.

OUTPUT:                  DO NOT SPLIT UP WORDS.
                    THE FIRST WORD OF THE
                    PARAGRAPH IS TO BE INDENTED 5
                    SPACES.
```

2.4 Write a program to accept a word as input and print the word with its consonants alphabetized and placed back into the consonant positions, and likewise have the vowels alphabetized and placed back into the vowel positions. Example:

```
 INPUT: Enter word: ELEPHANT
OUTPUT: AHELNEPT
```

2.5 Write a program that will accept at most 9 words, each having at most 10 characters, and then will check the words for common letters and print those letters that are occur in all the words. The program is to allow the user to choose one of the common letters, then the program is to arrange the words in a list so that the FIRST common letter appears in the same column. If there are no common letters then print the message: NO COMMON LETTERS. Examples:

```
 INPUT: Enter number of words: 3
        Enter word: BROTHER
        Enter word: MOTHER
        Enter word: TUTOR
OUTPUT: O R T
 INPUT: Choose letter: T
OUTPUT: BROTHER
         MOTHER
           TUTOR

 INPUT: Enter number of words: 2
        Enter word: MOM
        Enter word: DAD
OUTPUT: NO COMMON LETTERS
```

2.6 Three local cross country teams compete in a double dual race. Each team consists of seven runners, but only the first five finishers of a team contribute to that team's score. As the runners cross the finish line, gasping for breath, the judge writes the INITIAL of the runner's team name and the NUMBER indicating the runner's finishing position, e.g. 1 for 1st, 2 for 2nd and so on. To find the score for teams A and B and to decide which of the two wins, the scorekeeper temporarily eliminates all of C's positions, and then repositions the runners from A and B into places 1 through 14. The team's score consists of the sum of the places of their first five runners. The lower team score wins. If there is a tie then the team whose sixth runner crossed the finish line first is the winner.

Write a program that computes the score for each pair of three teams and determines the winner of each pair. The program must allow the user to assign all 21 runners' team INITIAL to finishing places. Team initials can be any letter in the alphabet. Example:

```
     INPUT: Place 1: A
            Place 2: B
            Place 3: A
            Place 4: B
            Place 5: A
            Place 6: B
            Place 7: C
            Place 8: C
            Place 9: C
            Place 10: C
            Place 11: B
            Place 12: A
            Place 13: C
            Place 14: B
            Place 15: C
            Place 16: B
            Place 17: A
            Place 18: A
            Place 19: C
            Place 20: B
            Place 21: A
```

```
 Example of
 repositioning
 teams A and B:

  1: A
  2: B
  3: A
  4: B
  5: A
  6: B
  7: B
  8: A
  9: B
 10: B
 11: A
 12: A
 13: B
 14: A
```

```
    OUTPUT: (in any order)

            TEAM A: 28 POINTS
            TEAM B: 28 POINTS
            TEAM B WINS!

            TEAM A: 25 POINTS
            TEAM C: 31 POINTS
            TEAM A WINS!

            TEAM B: 24 POINTS
            TEAM C: 31 POINTS
            TEAM B WINS!
```

2.7 Write a program that will make it easy to manipulate tables of numerical data with at most 4 digits and one decimal point. The program must first initialize a 3x3 array with the data shown in the example below. The array will be displayed as a table consisting of three rows with three items in each row. Complete the table so that sums of the rows are found in a fourth column and sums of the columns are found in the fourth row. After the initial table is loaded, the program must display the following menu:

```
A. EDIT OR CHANGE A VALUE
B. DISPLAY THE RESULTS
C. QUIT
```

If option (A) is chosen, the user will enter the row and column to be changed, then the new value. The program then returns to the menu upon hitting a key. If option (B) is chosen, the table will be displayed; Both the numbers and the sums are to be displayed in the form ###.## and separated from each other by two spaces. The program then returns to the menu upon hitting a key. If option (C) is chosen, the program will terminate. Example:

RUN PROGRAM:

```
    OUTPUT: A. EDIT OR CHANGE A VALUE
            B. DISPLAY THE RESULTS
            C. QUIT
     INPUT: Enter option: B
    OUTPUT:  10.11   20.22   30.33   60.66
             11.10   22.20   33.30   66.60
             10.00   20.00   30.00   60.00
             31.21   62.42   93.63  187.26
     INPUT: (press any key)
    OUTPUT: A. EDIT OR CHANGE A VALUE
            B. DISPLAY THE RESULTS
            C. QUIT
     INPUT: Enter option: A
            Enter row, col: 1, 2
            Enter number: 60.55
     INPUT: (press any key)
    OUTPUT: A. EDIT OR CHANGE A VALUE
            B. DISPLAY THE RESULTS
            C. QUIT
     INPUT: Enter option: B
    OUTPUT:  10.11   60.55   30.33  100.99
             11.10   22.20   33.30   66.60
             10.00   20.00   30.00   60.00
             31.21  102.75   93.63  227.59
     INPUT: (press any key)
    OUTPUT: A. EDIT OR CHANGE A VALUE
            B. DISPLAY THE RESULTS
            C. QUIT
     INPUT: Enter option: C
    OUTPUT: (program terminates)
```

2.8 Write a program that checks all combinations of four different whole numbers between 0 and 9 inclusive to see if the product of any two can be expressed as a 2-digit numeral using the other two numbers as digits. The program must print the combinations for which the condition holds true along with the illustration. Each set must be displayed with its elements in the following order: smaller multiplicand, larger multiplicand, digits of product. Amongst the other sets, each set must be displayed in ascending order when considering the smaller and larger multiplicands together. In the example below, set 2,5,1,0 comes before 2,7,1,4 because 25 comes before 27. Similarly, 27 comes before 28 in the set 2,8,1,6. A final total must also be displayed in the form shown below. Partial example:

```
RUN PROGRAM:

    OUTPUT: 2  5  1  0    2 X 5 = 10
            2  7  1  4    2 X 7 = 14
            2  8  1  6    2 X 8 = 16
            :  :  :  :    :       :
            :  :  :  :    :       :
            8  9  7  2    8 X 9 = 72
            TOTAL = ##
```

2.9 Write a program to accept N words, after N is input as a number less than 15, and then to accept a word with a wildcard, "*". The wildcard could come before, after, or in between the letters of the word. Your program must then display all words that were input that match the format of the wildcard expression, where the wildcard could represent 0 to 10 letters. All words are displayed in the order in which they were input. If no words are found then display the message "NO WORDS FOUND". The program is to continue to accept as input a wildcard word until a word is entered without a wildcard. Example:

```
     INPUT: Enter N: 3
            Enter word: RUN
            Enter word: RUNNING
            Enter word: RUNS
            Enter string: RUN*
    OUTPUT: RUN
            RUNNING
            RUNS
     INPUT: Enter string: *UN
    OUTPUT: RUN
     INPUT: Enter string: R*N
    OUTPUT: RUN
     INPUT: Enter string: *INGS
    OUTPUT: NO WORDS FOUND
     INPUT: Enter string: END
    OUTPUT: (program terminates)
```

2.10 A particular building has three major sections, an office section, a computer section, and a dry storage section. Each section's temperature is controlled by a central processing unit for different maximums and minimums as set by a section's thermostat. Heat is generated in the office section so that the temperature rises at a rate of .1 degree Fahrenheit (F) every 15 seconds. Heat is generated at the rate of .2(F) every 15 seconds in the computer room. The rate in dry storage is .1(F) every minute. The central air conditioning unit has the capacity to cool any one section at the rate of .1(F) every 3 seconds if no heat is being generated. If 2 sections are being cooled the air conditioner will cool at the rate of .1(F) every 6 seconds and for all 3 sections the rate is .1(F) every 9 seconds.

Write a program for the controller that will monitor the temperatures in each section every 15 seconds, alert the controller to turn air conditioning off or on in a section when necessary and will print a status report every 5 minutes or when the air conditioner is turned off or on in any section. The status report must include the sections that the air conditioning is turned off or on (denoted by 0 or 1 respectively), the temperature in each room (rounded to the nearest tenth of a degree), and the time of the status report (of the form Minutes:Seconds). Assume that heat is being generated at the same time that a section is being cooled. Initialize the program with minimum temperatures for each section and the air conditioning off. Air conditioning is changed (turned on or off) when it exceeds or precedes its maximum or minimum temperature respectively during a certain interval of time. For example if the computer room reads 64.8(F) and the air conditioning is currently off, then a message should not be displayed to turn the air conditioning off again for that section if the temperature reads 64.9(F) fifteen seconds later. The program ends when it reaches the last 5-minute interval, entered by the user.

Minimum and Maximum Acceptable Temperatures

Office	72.0F to 78.0F
Computer Rm	65.0F to 70.0F
Dry Storage	79.0F to 85.0F

Example:

```
     INPUT: Enter last 5-minutes: 30

    OUTPUT: OF CO DS  OFFICE  COMP.   DRY.   MIN:SEC
            0  0  0   72.0    65.0    79.0     0:00
            0  0  0   74.0    69.0    79.5     5:00
            0  1  0   74.6    70.2    79.7     6:30
            0  1  0   76.0    66.0    80.0    10:00
            0  0  0   76.4    64.8    80.1    11:00
            0  0  0   78.0    68.0    80.5    15:00
            1  0  0   78.1    68.2    80.5    15:15
            1  1  0   74.1    70.2    80.8    17:45
            1  1  0   72.7    69.7    81.0    20:00
            0  1  0   72.0    69.5    81.1    21:15
            0  0  0   73.5    65.0    81.5    25:00
            0  0  0   75.5    69.0    82.0    30:00
```

3.1 A standard six sided die has a 1 on the opposite side of the 6, and a 5 on the opposite side of the 2. When the 1 is on the top and the 5 is in front, the 4 is on the right side, and the 3 is on the left side. Write a program that, accepts as input, the numbers on the top and front sides of the die and prints out the numbers on all six sides of the die clearly labeled. Example:

```
 INPUT: Enter top, front: 1, 3
OUTPUT: TOP=1  FRONT=3  RIGHT=5
        BACK=4  LEFT=2  BOTTOM=6
```

3.2 Write a program to factor a quadratic equation with integral coefficients and rational roots. Input will be A,B,C of Ax^2+Bx+C, where A is greater than 0 and C is not equal to 0. Output should be of this form: Q(Rx+S)(Tx+U), where Q,R,S,T, and U are integers and R,Q are positive. For example 4x^2+2x-12 should yield 2(x+2)(2x-3) or 2(2x-3)(x+2), order does not matter. Note that R=1 is omitted and that there is a '-' sign instead of '+-'. If Q=1 then Q must not be printed. If the quadratic equation cannot be factored then display the message: CANNOT BE FACTORED.

```
 INPUT: Enter A, B, C: 30, 4, -2
OUTPUT: 2(5X-1)(3X+1)

 INPUT: Enter A, B, C: 2, 4, 6
OUTPUT: CANNOT BE FACTORED
```

3.3 Write a program to simulate a calculator. Input will be a mathematical expression using whole numbers (less than 1000) and the symbols +,-,*,/. By following the rules of algebraic order, compute the numerical value of the expression and display it in the form ###.###. Examples:

```
 INPUT: Enter expression: 3+4*2-15
OUTPUT:  -4.000

 INPUT: Enter expression: 250*4-50*5+100/3
OUTPUT: 783.333
```

3.4 Write a program to print out all the digits of N factorial (N!), once given N as a whole number less than 50. The output displayed must contain all the digits in the resulting factorial number and not an abbreviated form using exponential notation (i.e. 2.432902E+18).

```
 INPUT: Enter N: 20
OUTPUT: 2432902008176640000
```

3.5 Write a program that will print the sum and difference (clearly labeled) of any two positive decimal numbers of up to 30 digits each with the decimal point between any two of the digits. Since the first number input will be numerically larger than the second number input, the difference output will always be positive. Example:

```
    INPUT: Enter #1: 987654321.123456789
           Enter #2: 123456789.987654321

   OUTPUT: SUM = 1111111111.111111110
           DIFFERENCE = 864197531.135802468
```

3.6 Write a program to print a snake (a trail of 30 asterisks '*') centered on the screen. Upon hitting appropriate keys (like I,J,K,M or something similar), the snake's head moves in the appropriate direction while the rest of the snake slithers along the same right angle paths. The snake is to move CONTINUOUSLY in the designated direction UNTIL a new directional key is hit. The snake will be 30 asterisks long throughout the entire run; Do not leave a sketched path. The snake cannot go backwards, e.g. if it is going to the right its next direction cannot be to the left. The snake continues moving until it runs into itself or it runs off the screen or a non-directional key is pressed.

3.7 Write an EFFICIENT program to accept as input a word with at most 7 distinct letters and a positive integer K. Your program is to print out the Kth, 2*Kth, and 3*Kth elements of the alphabetized list of all the permutations of the word. For example if the word CAT is entered with K=2 then the computer would form the list ACT, ATC, CAT, CTA, TAC, TCA, and output would be: ATC, CTA, TCA. Example:

```
    INPUT: Enter word: CAT
           Enter K: 2
   OUTPUT: ATC  CTA  TCA
```

3.8 Suppose you had a playing board with N rows with N squares in each row. You are to place pennies on the squares so that each row, column, and main diagonal has at most one penny. Write a program that will accept as input a positive number N, between 3 and 14 inclusive, and will print the largest possible number of pennies that can be placed on the board. The program also prints one of many solutions to the puzzle, displaying the units digit of the N numbers horizontally, each separated by a space, and the units digit for the N numbers vertically; Place an asterisk in the position where a penny can be placed and display the coordinates of each penny to the right with the sum of the coordinates. Examples:

```
 INPUT: Enter N: 3

OUTPUT: TOTAL = 3
          1 2 3
        1 *     (1,1) SUM = 2
        2     * (2,3) SUM = 5
        3   *   (3,2) SUM = 5

 INPUT: Enter N: 4

OUTPUT: TOTAL = 4
          1 2 3 4
        1   *     (1,2) SUM = 3
        2       * (2,4) SUM = 6
        3 *       (3,1) SUM = 4
        4     *   (4,3) SUM = 7
```

3.9 Write a program to determine the minimum number of moves required to move a stack of N blocks having graduated sizes to one of two alternate locations. (There are three locations all together.) N will be input as a positive integer less than 16. When moving blocks, you can only move one block at a time and you cannot stack a larger block on a smaller block. When finished, the blocks in the new stack will be in the same order as they were in the beginning. Your program could be written in a very few program lines if you can develop an algorithm for determining the required number of moves. For example, the following 7 moves will transfer the three graduated blocks from one location (-) to one of two other locations (- -); The blocks are designated by the numerical values 3, 2, and 1, which represent the magnitude of the block:

```
1                                                         1
2       2                   1       1           2         2
3       3 1     3 1 2   3   2     3 2   1 3 2   1 3       3
- - -   - - -   - - -   - - -   - - -   - - -   - - -   - - -
```

Example:

```
 INPUT: Enter N: 3
OUTPUT: 7
```

3.10 Write a program that will find the set of numbers P,Q,R, where the following conditions hold:

1. Q is a 2-digit number and R is a 3-digit number.
2. P is a 5-digit number that is the product of Q and R.
3. Each of the digits 0-9 is found exactly once in the number sentence P=QxR.
4. Q is as small as possible but greater than S, where S is a 2-digit number input.

Examples:

```
 INPUT: Enter S: 44

OUTPUT: P = 17820  Q = 45  R = 396


 INPUT: Enter S: 52

OUTPUT: P = 16038  Q = 54  R = 297
```