1. Encoder/Decoder for text strings

You are required to implement a program to encode and decode text using a string-based coding method as described below. Your program should have a simple menu-based structure. Its main menu is

1) Enter an encoding "key" string.
2) Encode a section of text using the current key.
3) Decode a section of text using the current key.
4) Exit the program.

Choice?

When Option 1 is selected, the program should read in a one-line string from the keyboard. This should be converted to upper case, all characters other than A to Z, comma, full stop and space should be deleted, and the string adapted to be used as a code key as follows.

The key must contain all the characters A to Z, comma, full stop and space exactly once. If a character occurs more than once, the second, third etc. occurrences should be deleted. If a character does not occur in the string, it should be added to the end in the order given above - so a blank input string would simply give the code string
'ABCDEFGHIJKLMNOPQRSTUVWXYZ,. '

For example, the input string 'I wandered lonely as a cloud' gives the code key
'I WANDERLOYSCUBFGHJKMPQTVXZ,.'

When Option 2 is chosen, the program should read in a text message from the keyboard; this may not be more than 250 characters long and your program should check for this limit. Note that Pascal will not let you enter more than 127 characters per line. Characters other than A to Z, comma, full stop and space should be removed, and the input should finish when a # character is encountered in the text. Line breaks, except when immediately preceded by a \ character (see below), should be converted into spaces. No text or spaces before the # should be deleted, but all text after it on the same line must be and the input buffer must be cleared to the end of the line.

Your program should then encode the text as follows. Start with a marker, p, pointing to the first character in the key. For each character in the message, p is moved right by the value of the character and the key symbol at p gives the next character in the encoded message.

The value of characters is given in order: A=1, Z=26, space=29. If p moves beyond the end if the key, 29 is subtracted from it (in other words, it rolls round to the start). This means that one error in copying the encoded message will result in two character errors in the decoded message, but this is acceptable.

For example, with the above code key, the message 'I WANDER' would be encoded as follows. p starts with a value of 1, pointing to the I in the key. The value of the first character

in the message (also I), 9, is added to give 10; the 10th key character is O. Space has a value of 29 - and therefore, when rolled round, p is again 10 and the first two output characters are OO. W has a value of 23, so the 4th character in the code is selected next: A. And so on, to give the encoded form 'OOANJQ,G'.

The encoded message is output in this way. It is printed as lines of maximum length 50 characters, and if it must be split after the 50th character on a line, a \ should be printed to indicate that the line end should be ignored. At the end of the text, a # should be printed. You should also output the original message, after transformation to upper case etc, in the same format, to allow the user to see what the message will look like when decoded.

Option 3 requires you to reverse this process, using the same input and output methods with the exception of printing the encoded text again. The same coding key should be used.

When option 4 is selected your program should terminate. After the other menu options have been executed, you might like to ask the user to press enter before returning to the main menu.

Sample Output for Question 1

Choose a menu option:

1) Enter an encoding "key" string.
2) Encode a section of text using the current key.
3) Decode a section of text using the current key.
4) Exit the program.

Choice? 1
Enter a code key:
I wandered lonely as a cloud

Press ENTER to continue...

Choose a menu option:

1) Enter an encoding "key" string.
2) Encode a section of text using the current key.
3) Decode a section of text using the current key.
4) Exit the program.

Choice? 2
Encode a message using the current key.
Enter Message (250 characters max) with # to finish.
I wandered lonely as a cloud
that floats on high o'er dale and hill
when all at once I saw a crowd,
a flock of wand'ring daffodils.#

```
Message is
I WANDERED LONELY AS A CLOUD THAT FLOATS ON HIGH O\
ER DALE AND HILL WHEN ALL AT ONCE I SAW A CROWD, A\
 FLOCK OF WANDRING DAFFODILS.#
Coded message is
OOANJQ,GPXXLTLUXPPQCCUUG.BEYY OY  RKDEZGGWGGVNSKKD\
Y..ANGPPQRSSK.STTHX FFG.SSCAAJAESSMMYSDDEEO,URSOOY\
YG.BH..BMMBFINQWGTT,.DSZ YQCS#

Press ENTER to continue...

Choose a menu option:

1) Enter an encoding "key" string.
2) Encode a section of text using the current key.
3) Decode a section of text using the current key.
4) Exit the program.

Choice? 3
Decode a message using the current key.
Enter Message (250 characters max) with # to finish.
OOANJQ,GPXXLTLUXPPQCCUUG.BEYY OY  RKDEZGGWGGVNSKKD\
Y..ANGPPQRSSK.STTHX FFG.SSCAAJAESSMMYSDDEEO,URSOOY\
YG.BH..BMMBFINQWGTT,.DSZ YQCS#
Decoded message is
I WANDERED LONELY AS A CLOUD THAT FLOATS ON HIGH O\
ER DALE AND HILL WHEN ALL AT ONCE I SAW A CROWD, A\
 FLOCK OF WANDRING DAFFODILS.#

Press ENTER to continue...

Choose a menu option:

1) Enter an encoding "key" string.
2) Encode a section of text using the current key.
3) Decode a section of text using the current key.
4) Exit the program.

Choice? 3
Decode a message using the current key.
Enter Message (250 characters max) with # to finish.
ooanjP,gpxxltluxppqcc#
Decoded message is
I WANAHRED LONELY AS #

Press ENTER to continue...
```

2. Magic Squares

In this question you will be asked how, in outline, you would implement a program to find magic squares. A magic square is an n by n grid containing the integers 1 to n2, which fulfils this condition: each of the sums of the n numbers on any row or column must add up to 0.5n(n2 + 1). Based on this definition, an example magic square for n = 4 is

16	15	1	2
6	4	10	14
9	8	12	5
3	7	11	13

Questions

- 2.1 Are magic squares (as defined above) unique or not? Why?
- 2.2 Devise a data structure to represent the grid for up to n = 10.
- 2.3 How would your program find magic squares?
- 2.4 Give a rough estimate of how long your program is likely to take for different
- values of n, explaining what assumptions you make.
- 2.5 Are there any ways in which the algorithm you have chosen reduces the size of the problem? Suggest further ways of speeding up the search while still finding the correct solution.

3. Squares

Given a 5x5 grid, the numbers 1 to 25 are to be placed without repetition in the grid. The placement rule is that given the number k occupying co-ordinates (x, y), 1 <= k < 25, the number k+1 can be placed in any unoccupied square on the board with the following co-ordinates (z, w):

1. (z, w) = (x +/- 3, y)
2. (z, w) = (x, y +/- 3)
3. (z, w) = (x +/- 2, y +/- 2)

For example, given the starting position of 1 at (1, 2), the number 2 can be placed in any of the squares marked with a *.

```
+-------------------+
|   |   |   |   |   |
+-------------------+
| 1 |   |   | * |   |
+-------------------+
|   |   |   |   |   |
+-------------------+
|   |   | * |   |   |
+-------------------+
| * |   |   |   |   |
+-------------------+
```

Your program should repeatedly offer 3 options: A - Problem A; B - Problem B; X - exit program.

Problem A

You are given a starting position for the number 1. Print to the screen the list of all valid squares in which the number 2 can be placed according to the above rule. The order of this list is unimportant. Append a copy of this to the file 'BIO95R2.OUT'. A sample copy of this output is given below.

Input

Starting square in format x y : 1 2

Output

Valid positions starting from (1, 2):
(1, 5)
(4, 2)
(3, 4)
blank line

Problem B

You are again given a starting position in which the number 1 is placed. Evaluate and display the number of valid combinations which result in a complete grid. This means a grid in which all pieces 2 to 25 have been placed from the given starting position following the above rule. You should also give one example complete board. A copy of your output should be appended to the file 'BIO95R2.OUT'.

Input

Starting square in format x y : 1 2

Output

Total number of valid combinations starting from (1, 2): 548
Example grid:
16 24 10 17 25
1 13 21 2 12
9 18 5 8 19
15 23 11 14 22
4 7 20 3 6