**Java program:** Prob11.java

**Input File:** Prob11.in.txt

**Output:** Your output needs to be directed to stdout (i.e., using System.out.println())

**Introduction**

If you’ve ever looked at a word search puzzle and thought “hey, I wonder if there’s some hidden message in there somewhere?” then this problem is for you. Your task is to take some text and create a counter clockwise spiral out of it.

**Program Input**

The file Prob11.in.txt will contain some number of lines of text. Your program should read in the text and print it out in a counter clockwise spiral. Here are the parameters for creating the spiral:

* The spiral should be contained within a square, and the square should be as small as possible.
* The first character of the text should go in the exact center of the square.
* The second character of text should go to the left of the first character.
* The spiral of text should go counter clockwise from there.
* Insert spaces as necessary to separate words from different lines (watch out for input lines with leading and/or trailing spaces).
* When you reach the end of the text, fill the remainder of the spiral with spaces.

**Example Input:**

You should

turn this

text into

a spiral.

Note:take a look at the input file for this – there are spaces in the file that you can’t see here.

**Program Output**

Your program should print the text in spiral form as described above. Again, take a look at the example file provided. The extra spaces are not readily apparent here.

**Example Output:**

txet

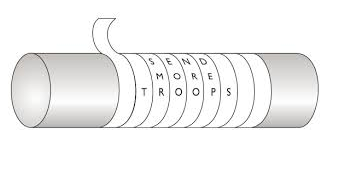
idluos

noYhi.

ttushl

ourn ta

aspir

**Java program:** Prob12.java

**Input File:** Prob12.in.txt

**Output:** Your output needs to be directed to stdout (i.e., using System.out.println())

**Introduction**

The Caesar and Syctale ciphers are very old ciphers used by the ancient Greeks to encrypt secret messages. In this problem you will be decrypting a message that has been encoded using both methods.

**The Caesar Cipher**

The Caesar cipher is a form of a substitution cipher because it substitutes letters for other letters. A Caesar cipher just shifts letters however many positions to the left or the right. For example:

LOCKHEED --> right shift 3 --> ORFNKHHG

Letters are shifted as if we were using an alphabet ring. That is, shifting Z to the right produces an A, and so forth. So, the string “Zebra” shifted 1 position to the right would be “Afcsb”. Note that capitalization is retained.

**The Scytale Cipher**

The Scytale cipher is a form of transposition in that it rearranges letters into different positions. In a Scytale cipher, you would wrap a very long piece of paper around a stick and write your message going across. The message would be unreadable when unwrapped. You had to use the same stick to decrypt the message. This is similar to picking a two dimensional array and filling it with the message. For example, the word “LOCKHEED” using a 4x3 array would produce this:

|  |  |  |
| --- | --- | --- |
| L | O | C |
| K | H | E |
| E | D | X |
| X | X | X |

A capital X is used to pad the array if the message doesn’t fill the array exactly. To “unravel” the paper, read the array from top to bottom, left to right:

LOCKHEED --> LKEXOHDXCEXX

**Putting Them Together**

For this problem, you must decrypt a message that was encrypted using the Caesar cipher followed by a Scytale. Let’s look at the “LOCKHEED” example:

LOCKHEED --> Caesar +3 --> ORFNKHHG -->Scytale 4x3 --> ONHXRKGXFHXX

There are a couple of things that are important to note:

* If the input string was too short to fill the array exactly, the encrypted Scytale string will have more characters than the original decrypted message. Be careful when printing your output because of this. You can assume that the original message does not end in a capital X.
* You will not know the array size that was used to perform the Scytale cipher, but it is guaranteed that the number of characters in the encoded input file will fit exactly into the array used.
* You will not know the shift value that was used to perform the Caesar cipher, but the original text is guaranteed to have the word “Dear” in it. Shifting the letters in the word “Dear” n times produces no other word in the English language, so it is a sufficient key to know that the decoding has been successful.

**Program Input**

The file Prob12.in.txt will contain some number of lines of encoded text. The file in its entirety should be decoded all at once – each line is not separately encoded. Rather, the lines together form one long string for encoding and decoding purposes.

**Example Input:**

JouyWOjkg

rkgjIacuyzrgxguk

oszoXxeejyreZ

yXJZokzrh

ngX

**Program Output**

Your program should output the decoded message. Your line lengths should match the input file, with the additional padded characters (if any) removed from the end of the file.

**Example Output:**

DearDiary

TodayisCodeQuest

Iwilldomybest

Thatisall