

Problem J1: Squares

Gigi likes to play with squares. She has a collection of equal-sized square tiles. Gigi wants to arrange some or all of her tiles on a table to form a solid square. What is the side length of the largest possible square that Gigi can build?

For example, when Gigi has 9 tiles she can use them all to build a square whose side length is 3. But when she has only 8 tiles, the largest square that she can build has side length 2.

Write a program that asks the user for the number of tiles and then prints out the maximum side length. You may assume that the user will only type integers that are less than ten thousand. Once your program has read the user's input and printed the largest square, your program stops executing.

There are many different methods that your program might use to find the answer. You may use any method. For example, here is one method. First, check whether there are enough tiles to build a square of side length 1. If there are enough tiles, then move on to check the side lengths 2, 3, 4, etc., until your program finds a length that is too large.

Sample Session 1

Program Output: Number of tiles?
User Input: 9
Program Output: The largest square has side length 3.

Sample Session 2

Program Output: Number of tiles?
User Input: 8
Program Output: The largest square has side length 2.

Sample Session 3

Program Output: Number of tiles?
User Input: 7535
Program Output: The largest square has side length 86.



Problem J2: Terms of Office

In CS City, a mathematical place to live, the mayor is elected every 4 years, the treasurer is appointed every 2 years, the chief programmer is elected every 3 years and the dog-catcher is replaced every 5 years.

This year, Year X , the newly elected mayor announced the appointment of the new treasurer, a new dog-catcher and congratulated the chief programmer for winning the recent election. That is, all positions were changed over. This is highly unusual. You will quantify how unusual this really is.

Write a program that inputs the year X and the future year Y and lists all years between X and Y inclusive when all positions change.

Sample Session

Program Output:	Enter the current year:
User Input:	2004
Program Output:	Enter a future year:
User Input:	2100
Program Output:	All positions change in year 2004
Program Output:	All positions change in year 2064



Problem J3: Smile with similes

A simile is a combination of an adjective and noun that produces a phrase such as "Easy as pie" or "Cold as ice".

Write a program to input n adjectives ($1 \leq n \leq 5$) and m nouns ($1 \leq m \leq 5$) provided by your contest supervisor, and print out all possible similes. The first two lines of input will provide the values of n and m , in that order followed, one per line, by n adjectives and m nouns. Your program may output the similes in any order.

Sample Input:

```
3
2
Easy
Smart
Soft
pie
rock
```

Sample output:

```
Easy as pie
Easy as rock
Smart as pie
Smart as rock
Soft as pie
Soft as rock
```

Problem J4: A Simple Encryption Algorithm

One of the simplest ways of coding a message is to do a letter shift. For example if you shift the letters in the original message by 5 then A in your original message becomes F in the coded message. (B \rightarrow G, C \rightarrow H, ..., T \rightarrow Y, U \rightarrow Z, V \rightarrow A, ..., Z \rightarrow E). To decode the message, you simply need to shift back by the same number.

A slightly trickier encryption uses a keyword to determine the amount of the shift. Suppose you were using a keyword "ACT". To encode the message, you take the original message, remove everything but the alphabetic characters, and form the message into a block that has the same width as the keyword. Here is a sample message to encrypt:

BANANA & PEEL

The blocked version of the message is shown below with the keyword ACT as a header.

A	C	T
B	A	N
A	N	A
P	E	E
L		

Now, the message is encoded using a letter shift. However, this time it is not a uniform shift; it will depend upon the keyword letter at the top of the column. If the letter at the top of the column is an 'A', then the letters in that column are not shifted. If the letter is a 'B', then the letters in that column shift by 1, and so on. In the example, the letters in the third column will shift by 19 since the 'T' is the 20th letter of the alphabet.

The encoded message is:

A	C	T
B	C	G
A	P	T
P	G	X
L		

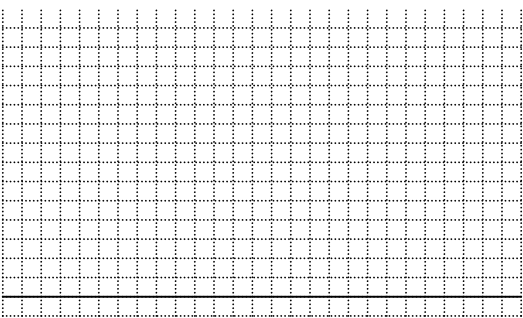
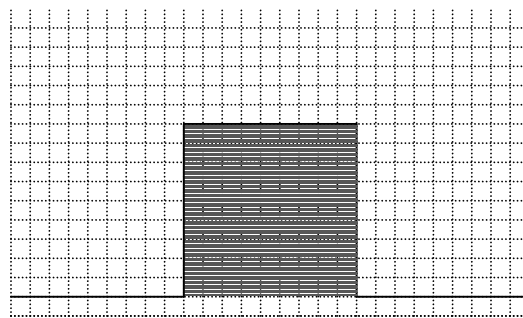
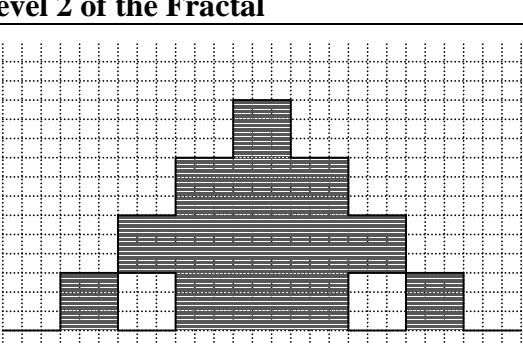
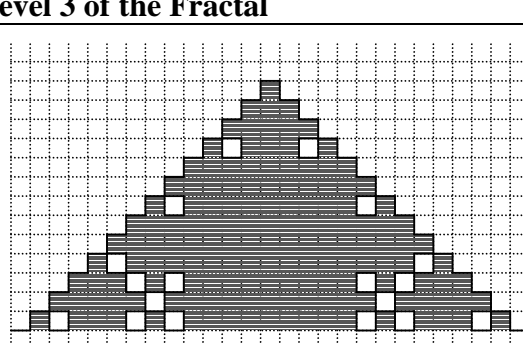
You will write a program that will accept a keyword and a string to be encoded. The keyword will never be more than 6 characters. The message will always be given in all upper case characters. The total message length will never be more than 60 characters.



Sample Input 1	Sample Output 1
ACT	BCGAPTPGXL
BANANA & PEEL	
Sample Input 2	Sample Output 2
TRICKY	BCWXONKFOTKKFZVI
I LOVE PROGRAMMING!	

Problem J5: Fractals

A fractal is a geometric shape where the pattern of the whole shape is self-replicating at each subsection of the shape. A simple “block fractal” is shown below. At each stage of the fractal growth, every straight line in the fractal is divided into three equal parts. The first and last sections stay straight; the middle section contains a square “bump” which has the same height as the width of the middle section. (You will want to consider the four orientations of a line segment within the fractal. Depending upon which line segment is currently being generated, the bump may protrude up, down, left, or right.)

Level 0 of the Fractal	Level 1 of the Fractal
	
Level 2 of the Fractal	Level 3 of the Fractal
	

Suppose this fractal is drawn on a Cartesian plane, where $(0, 0)$ is at the bottom left corner. Assume that in the example above, the bottom left point of the fractal is at $(0, 1)$ and the bottom right point of the fractal is at $(27, 1)$. For example the top of the Level 3 fractal is a line from $(13, 14)$ to $(14, 14)$.

Write a program that will keep track of the integer coordinate points of the lines in a similar “block fractal” with its bottom left corner at $(0, 1)$. The program will accept three integers as input: the level of the fractal, the width of the fractal, and an x-coordinate. You may assume that the width of the fractal will be some power of three, and that it will be large enough so that every corner of the fractal will fall on an integer intersection in the Cartesian plane. The width will never be more

than 81. The x -coordinate, x , will be in the range $0 - \text{width}$ (inclusive). Your program should output the y -coordinate value(s), y , where lines of the fractal intersect the point (x, y) .

You may draw a graphic representation of the fractal for debugging (and interest). However, test cases may ask you to define fractals that are too large to fit on a single screen.

Sample Input 1	Sample Output 1
3 27 5	4 5 6
Sample Input 2	Sample Output 2
3 27 18	1 2 3 4 7 8 9 10
Sample Input 3	Sample Output 3
2 27 19	1 4 7
Sample Input 4	Sample Output 4
4 81 38	37 38 39