CSE 231 Fall 2008

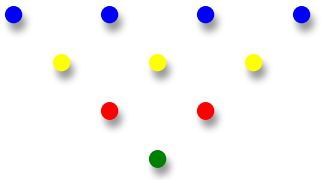
Programming Project 3

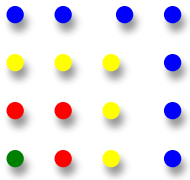
This assignment is worth 20 points (2.0% of the course grade) and must be **completed and turned in before 11:59 on Monday, September 15, 2008.**

Assignment Overview
The aim of this project is mostly an opportunity to work on your problem-solving skills, but it also emphasizes the use of Boolean logic and branching statements. You are going to determine the two *triangular* numbers which, when summed, give a *square* number.

Background
The ancient Greeks were fascinated with numbers, especially integers. They particularly were interested in the relationship between numbers and their physical realization. That is, numbers were very concrete to the Greeks, as they often talked and represented them in physical terms.

Consider the relationship between numbers and two-dimensional figures, called *polygonal numbers*. The Greeks would arrange numbers in successive layers, called *gnomons*, to generate a new number related to a particular polygon. Lets look at some examples:





The figure on the left is an example of "triangular" numbers, the figure on the right an example of "square" numbers. Each figure adds a gnomon, another layer, to the figure based on the shape of the figure.

For each figure, the first gnomon is indicated by the ball, the second by the balls, the third by the balls and the fourth gnomon by the balls.

The actual triangular and square numbers are the sum of the balls in the respective figure. That is, you take the sum of all the balls in each gnomon up to the last layer you added, and that is the number. For example, the left figure represents the triangular number 10, and the right figure represents the square number 16. Below is a table of the triangular and square numbers.

Take a look at those numbers, do you see any patterns?

Rank (first, second, etc)	Triangular	Square
1	1	1
2	3	4
3	6	9
4	10	16
5	15	25
6	21	36
7	28	49
8	36	64
9	45	81
10	55	100

The relationship for the square numbers should be pretty obvious. It is literally the n^2 operation. That is, to find the n^{th} square number, simply multiply n * n. So the 4^{th} square number is 4^2, or 16. We get the term "square" from this geometric representation. For the triangular numbers it is a little less obvious, but the formula for the n^{th} triangular number turns out to be $(n^2 + n)/2$. Thus, the 5^{th} triangular number is $(5^2 + 5)/2$ which is (25+5)/2, or 15. Surprisingly, some numbers are both triangular **and** square numbers. Which of the above is both?

A particularly interesting relationship is that any square number can be written as the sum of two triangular numbers. For example, 9 = 6 + 3. We are going to use that relationship in this project.

Note that these are all *integer* numbers. The Greeks were interested in integers as they could work with them as if they represented real, physical objects (which to them, they did).

Finally, there are many kinds of polygonal numbers. The next are pentagonal numbers (each gnomon adds a pentagon), hexagonal numbers, and so on. If you are curious, see http://en.wikipedia.org/wiki/Polygonal_number for more on the matter.

Program Specifications

So the problem is this. It is kind of a puzzle that, once solved, should end up as a rather small program.

1. Ask the user for a square number (a number which has an integer square root).
2. If the number provided is not a square, report an error and end the program.
3. If the number is a square, then find the two triangular numbers that sum to the square number (remember: there is a proof that every square number is the sum of two triangular numbers).
4. Print out the two triangular numbers and the original square number.

What are the two triangular numbers which, when added together, give a square number? That's the puzzle.

Once you figure out the puzzle, the program is quite short.

Deliverables

proj03.py -- your source code solution (remember to include your section, the date, project number and comments).

1. Please be sure to use the specified file name, i.e. "proj03.py"
2. Save a copy of your file in your CS account disk space (H drive on CS computers).
3. Electronically submit a copy of the file.

Assignment Notes

There are two ways to figure out the relationship between which triangular numbers sum to square numbers.

Look for the pattern

Look at the table of triangular and square numbers in the table. Can you find a relationship between the square numbers and the triangular numbers that sum to them? One way to help discover the relationship is to mark each square number and the triangular numbers that sum to it (highlight them, circle them, color them, etc.).

Draw a picture

The Greeks thought of these numbers as figures. Draw a square number figure. How can you divide the square figure into two triangular figures? Come up with a guess (hypothesis) and test it on other square number figures.

Other things to think about

1. How can you figure out if a number is truly a square number? As we said, the square root should be an integer. The square root function is part of the math module. You need to `import math` and then use `math.sqrt()`. Think about how you would determine whether the result of `math.sqrt()` is truly an integer. The `math.sqrt()` *always* returns a float.
2. How do you determine which square a square number is? That is, is 64 the 4^{th} square number (no), the 7^{th} square number (no) or the 8^{th} square number (yes).
3. If you know which square number a square number is, then if you found the pattern above you should be able to generate the triangular numbers required.

Other things to try

If you want a bigger challenge, here is a nice set of problems. Fermat gave the following proposition, called the *polygonal number theorem*. Every integer of "reasonable size" could be represented as a sum of:

1. at most three triangular numbers

2. at most four square numbers
3. at most five pentagonal numbers

and so forth. Lagrange proved it for the square case, Gauss the triangular case, but Cauchy proved the whole theory.

So, if the user provides you a large integer and an integer representing a polygonal number (3 for triangular, 4 for square etc), you should be able to provide the sum of those polygonal numbers that equal the original integer.