Expand All Collapse All

Trinomial coefficients (brute force)

Why use the primitive type `long` instead of `int`?

The numbers can get too large to fit in an `int`. For example, $T(22, 0) = 3{,}241{,}135{,}527$ cannot be represented as an `int` since the largest `int` is $2^{31} - 1 = 2{,}147{,}483{,}647$.

Why does it take "forever" to compute T(30, 0)?

That's to be expected. Rewatch the *Exponential waste* video segment. You'll fix this performance bug in the next exercise.

Can I use arrays, memoization, or dynamic programming to speed things up?

No. Please implement the recursive function by applying the recurrence relation directly. You will get a chance to use *dynamic programming* in the next problem.

Trinomial coefficients (dynamic programming)

Can I use negative indices with Java arrays?

No. Instead, consider declaring a private helper method that translates from indices in the desired range (e.g., between $-n$ and n) to indices in an allowable range (e.g., between 0 and $2n + 1$). Alternatively, use the fact that $T(n, k) = T(n, -k)$ for all n and k and avoid storing any coefficients when k is negative.

Can I use memoization instead of dynamic programming?

No. Use (bottom-up) dynamic programming.

Reve's puzzle

How do I transfer the remaining *n* – *k* discs using only three poles?

Use the classic algorithm (from lecture) for the 3-pole towers of Hanoi problem. You will need to modify the code from lecture because you must move the *largest* $n - k$ discs, not the *smallest* $n - k$ discs.

What will the structure of my program look like?

We recommend defining *two* recursive functions: one for the 3-pole version of the problem and one for the 4-pole version. A good starting point is [Hanoi.java]().

For debugging, can you provide solutions for some larger values of *n*?

Here are solutions for n = [6](), [7](), [8](), [9](), [10](), [15](), and [20]().

The solution to the towers of Hanoi problem that uses the fewest moves is unique. Is the same true for Reve's puzzle?

No. Since poles B and C are indistinguishable, interchanging B and C throughout any optimal solution yields another optimal solution. The autograder will accept any optimal solution.

Does the Frame–Stewart algorithm work for 5 (or more) poles?

Excellent question. The [Frame–Stewart conjecture]() is that, with a suitable choice of k, the Frame–Stewart algorithm solves the problem using the fewest moves. Unfortunately, the conjecture remains open for 5 (or more) poles.

Recursive squares

Which pen colors should I use?

Use `StdDraw.LIGHT_GRAY` to fill the squares; use `StdDraw.BLACK` to draw the outline of the squares.

Which methods should I use to draw the filled square and outline square?

Use `StdDraw.filledSquare(x, y, halfLength)` and `StdDraw.square(x, y, halfLength)`. Recall that (`x`, `y`) is the center of the square and `halfLength` is one-half the side length of the square.

What happens when I draw two shapes that overlap?

The second shape drawn will be visible; any overlapping parts of the first shape will be hidden.