Approaches of Dynamic Programming

In this lesson, we will continue our discussion on dynamic programming and see some approaches within dynamic programming.

We'll cover the following

We saw in the last lesson how dynamic programming can be described as solving problems that depend on subproblems to find an optimal solution. There are two basic approaches to solve a problem using dynamic programming. These approaches differ from each other in the way they start solving a problem. In the **bottom-up** approach, we start from the most basic unit (sub-problem) and then build-up to the solution. Whereas, in the **top-down** approach, we start from the problem and build and use smaller components(sub-problem) as needed. Let's see each of these individually in an abstract sense.

Bottom-up approach

The basic idea of the bottom-up approach is to start by constructing the smallest units first and then use those small units to build bigger and bigger units. Let's take the example, suppose you are hired by a construction company to construct a building. There are two ways you might go about doing this task. You might want to start off by constructing the smallest unit which would be a room in this case. Then replicating multiple rooms to construct a floor, and then finally stacking multiple floors to make a building. Take a look at the visualization below.

Construct a building of 4 floors of 3 rooms each

Bottom-up approach to constructing a building

start off by constructing a room

x 3

replicate room to construct a floor

x 3

replicate room to construct a floor

x 3 x 4

replicate floor to construct a building

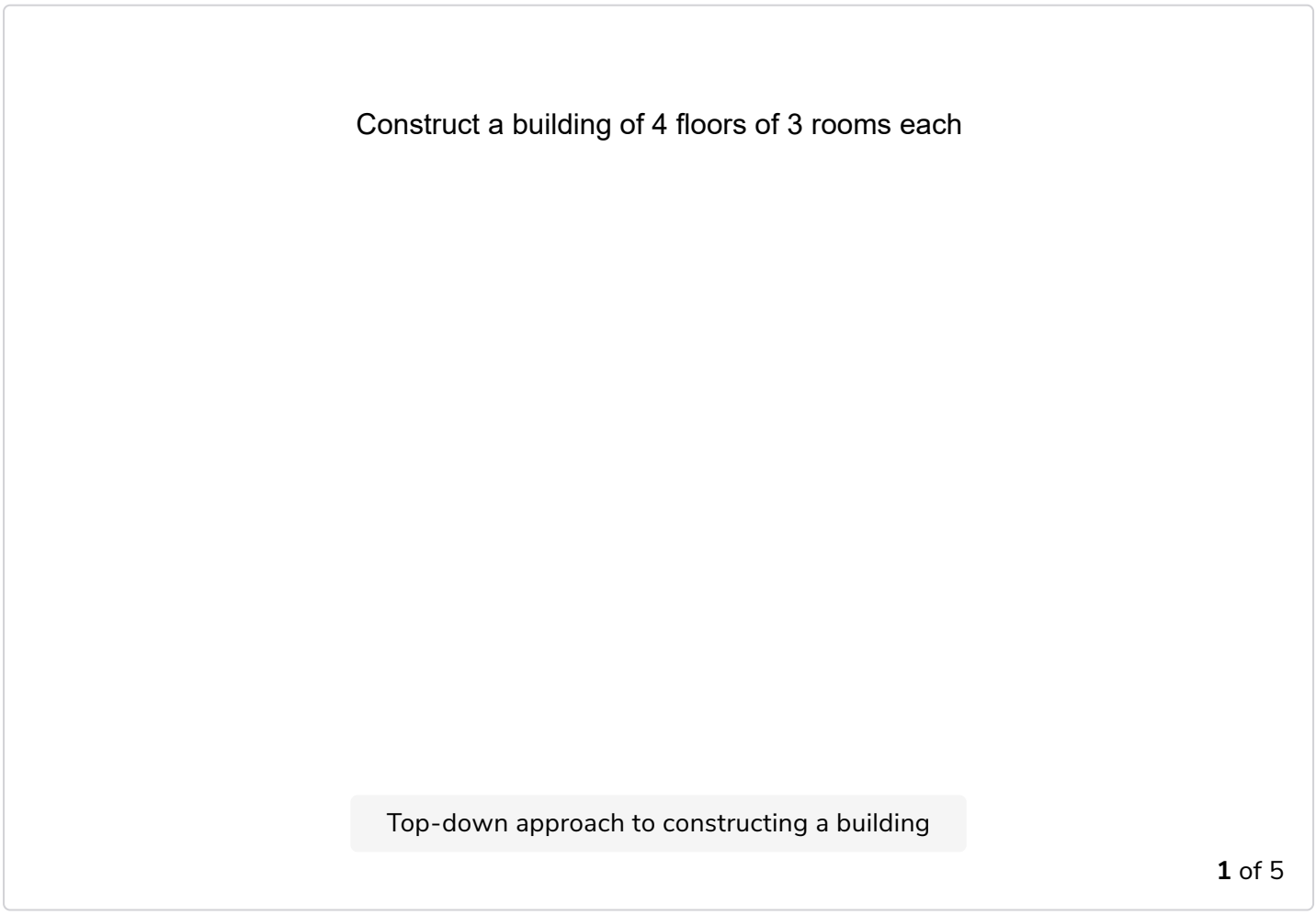
x 3

There you go, a building you constructed in a bottom-up manner

See how you started from the **bottom** with the most basic thing, a room, and constructed it **up** to your final output, a building. This is exactly how the bottom-up approach works. If you have a problem that depends on multiple subproblems, don't touch the main problem at all. Start from the most basic subproblem, find its answer, store it, and reuse it to construct answers for bigger and bigger subproblems until you find the answer to your main problem.

Top-down approach

The other way you can construct the building is to build a hollow building. Then make divisions for the floors and finally build rooms on each floor. Look at the following visualization for a better understanding.



Top-down approach to constructing a building

1 of 5

Construct a hollow structure the size of the building

Make divisions for floors

Fill each floor with rooms

There you go, you made a building with top-down approach

Notice how you started this time from the **top**, i.e., by constructing the structure of the building and then constructed it **down** with the most basic units, i.e., rooms.

This is what you have been doing so far with recursion. You start off at the top with a bigger problem, make a recursive call to subproblems and that recursion goes down until you reach your base cases. Recursion is essentially the top-down approach. In the top-down dynamic programming approach, you evaluate something as it is needed and then you can store it and reuse it when needed again.

In the next lesson, we will see what kinds of problems benefit from dynamic programming.