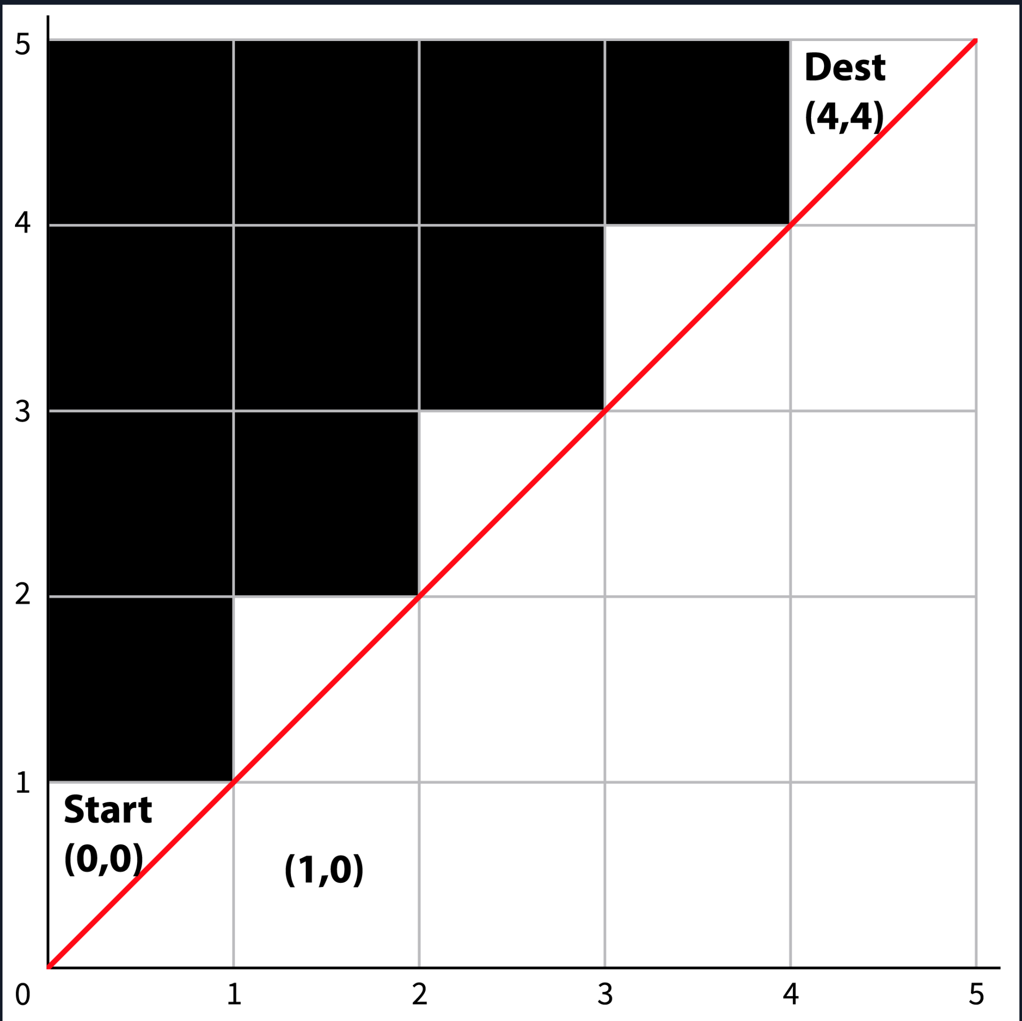
Number of Paths

You're testing a new driverless car that is located at the Southwest (bottom-left) corner of an `n×n` grid. The car is supposed to get to the opposite, Northeast (top-right), corner of the grid. Given `n`, the size of the grid's axes, write a function `numOfPathsToDest` that returns the number of the possible paths the driverless car can take.

Diagram



> **NOTE:** *"the car may move only in the white squares"*

Explanation

For convenience, let's represent every square in the grid as a pair `(i,j)`. The first coordinate in the pair denotes the east-to-west axis, and the second coordinate denotes the south-to-north axis. The initial state of the car is `(0,0)`, and the destination is `(n-1,n-1)`.

The car must abide by the following two rules: it cannot cross the diagonal border. In other words, in every step the position `(i,j)` needs to maintain `i >= j`. See the illustration above for `n = 5`. In every step, it may go one square North (up), or one square East (right), but not both. E.g. if the car is at `(3,1)`, it may go to `(3,2)` or `(4,1)`.

Assignment

Explain the correctness of your function, and analyze its time and space complexities.

Example:

```
input:  n = 4

output: 5 # since there are five possibilities:
          # “EEEENNN”, “EENENN”, “ENEENN”, “ENENEN”, “EENNEN”,
          # where the 'E' character stands for moving one step
          # East, and the 'N' character stands for moving one step
          # North (so, for instance, the path sequence “EEENNN”
          # stands for the following steps that the car took:
          # East, East, East, North, North, North)
```

Constraints:

- **[time limit] 5000ms**
- **[input] integer `n`**
  - 1 ≤ n ≤ 100
- **[output] integer**