

AE2ADS

Reading

M. T. Goodrich, R. Tamassia and M. H. Goldwasser,
Data Structures and Algorithms in Java, 6th Edition,
2014.

- Chapter 5. Recursion

Exercise 1

You are climbing a staircase. It takes n steps to reach the top.

Each time you can either climb **1** or **2** steps. In how many distinct ways can you climb to the top?

Example 1:

Input: $n = 2$

Output: 2

Explanation: There are two ways to climb to the top.

1. 1 step + 1 step

2. 2 steps

Example 2:

Input: $n = 3$

Output: 3

Explanation: There are three ways to climb to the top.

1. 1 step + 1 step + 1 step

2. 1 step + 2 steps

3. 2 steps + 1 step

Exercise 2

Implement `pow(x, n)`, which calculates x raised to the power n (i.e., x^n).

Example 1:

Input: $x = 2.00000$, $n = 10$

Output: 1024.00000

Example 2:

Input: $x = 2.10000$, $n = 3$

Output: 9.26100

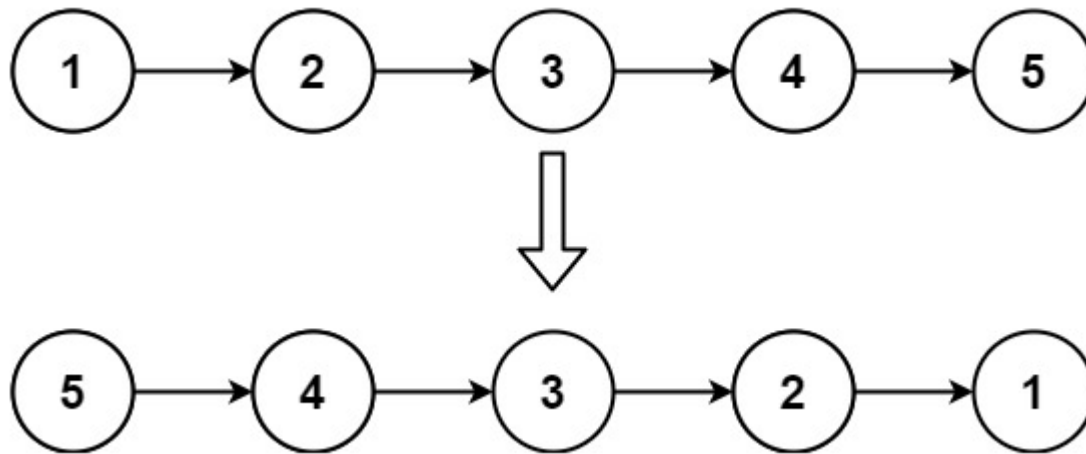
Example 3:

Input: $x = 2.00000$, $n = -2$

Output: 0.25000

Exercise 3

Given the head of a singly linked list, reverse the list, and return the reversed list.



Input: head = [1,2,3,4,5]

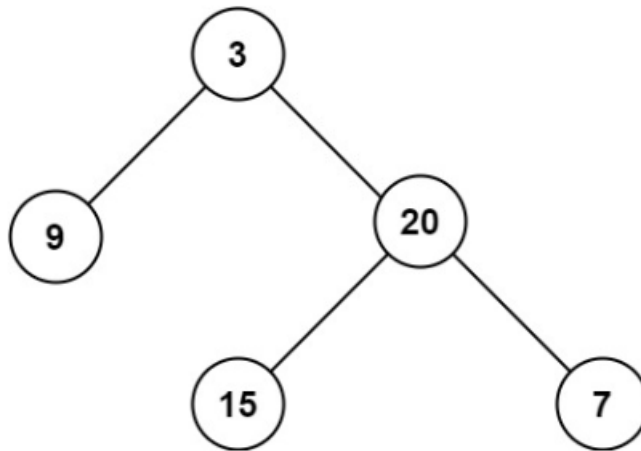
Output: [5,4,3,2,1]

Exercise 4

Given the root of a binary tree, return its maximum depth.

A binary tree's maximum depth is the number of nodes along the longest path from the root node down to the farthest leaf node.

Example 1:



Input: root = [3,9,20,null,null,15,7]

Output: 3

Exercise 5

Given an array ***nums*** of distinct integers, return all the possible permutations. You can return the answer in any order.

Example 1:

Input: `nums = [1,2,3]`

Output: `[[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1]]`

Example 2:

Input: `nums = [0,1]`

Output: `[[0,1],[1,0]]`

Example 3:

Input: `nums = [1]`

Output: `[[1]]`

Exercise 6

Write a program to solve a Sudoku puzzle by filling the empty cells.

A sudoku solution must satisfy all of the following rules:

Each of the digits 1-9 must occur exactly once in each row.

Each of the digits 1-9 must occur exactly once in each column.

Each of the digits 1-9 must occur exactly once in each of the 9 3x3 sub-boxes of the grid.

The '.' character indicates empty cells.

Exercise 6

Input: board =

```
[["5","3",".",".", "7",".", "6",".", "1","9","5",".", "8",".", "2"],
["6",".", "8",".", "1","9","5",".", "3","4","8"],
["1","9","8","3","4","2","5","6","7"],
["8","5","9","7","6","1","4","2","3"],
["4","2","6","8","5","3","7","9","1"],
["7","1","3","9","2","4","8","5","6"],
["9","6","1","5","3","7","2","8","4"],
["2","8","7","4","1","9","6","3","5"],
["3","4","5","2","8","6","1","7","9"]]
```

Output:

```
[["5","3","4","6","7","8","9","1","2"],
["6","7","2","1","9","5","3","4","8"],
["1","9","8","3","4","2","5","6","7"],
["8","5","9","7","6","1","4","2","3"],
["4","2","6","8","5","3","7","9","1"],
["7","1","3","9","2","4","8","5","6"],
["9","6","1","5","3","7","2","8","4"],
["2","8","7","4","1","9","6","3","5"],
["3","4","5","2","8","6","1","7","9"]]
```

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9