

# **Advent of Code 2022 Day 14**

## **Selected Fun Problems of the ACM Programming Contest**

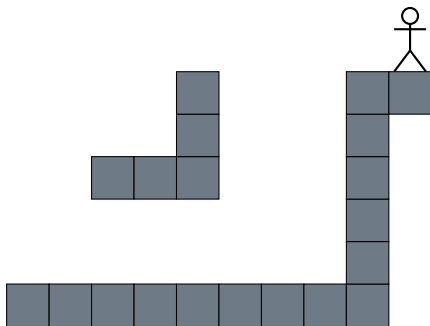
**Simon Roller**

University of Tübingen

28.06.2024

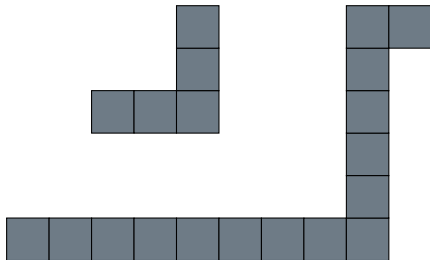
# Motivation

How much sand is needed to fill the cave and its surroundings?

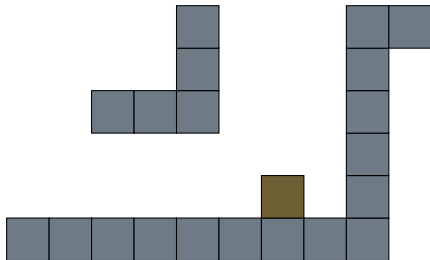


# Problem Example

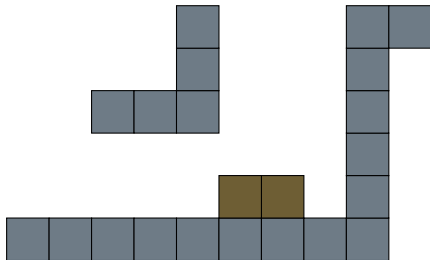
+



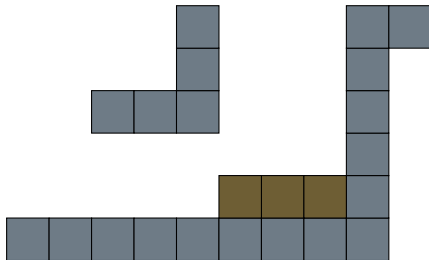
# Problem Example



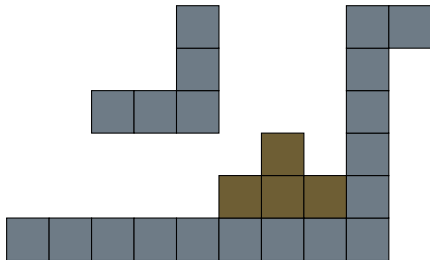
# Problem Example



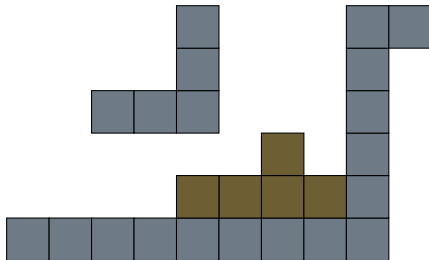
# Problem Example



# Problem Example

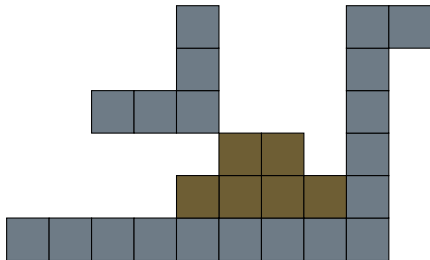


# Problem Example

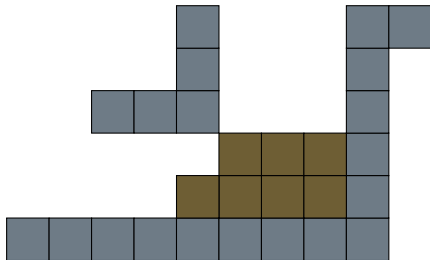




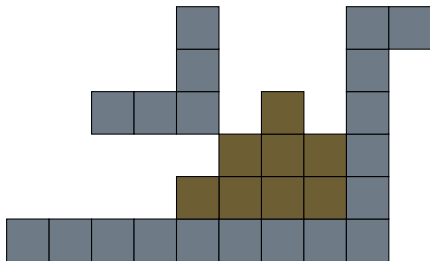
# Problem Example



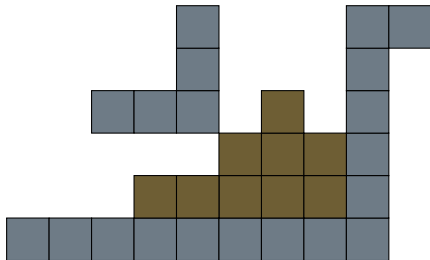
# Problem Example



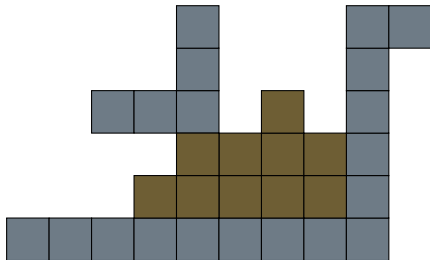
# Problem Example



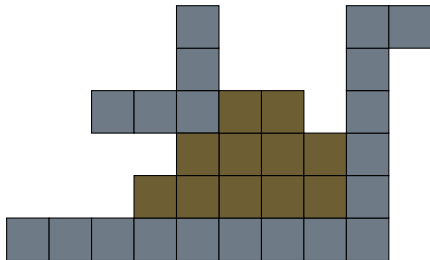
# Problem Example



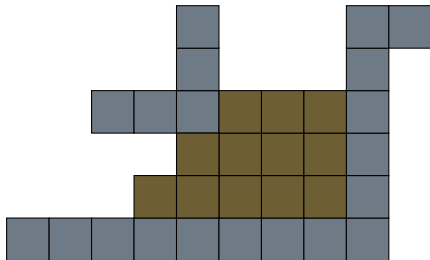
# Problem Example



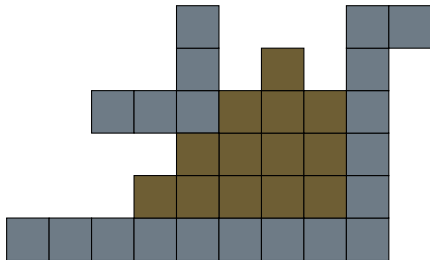
# Problem Example



# Problem Example

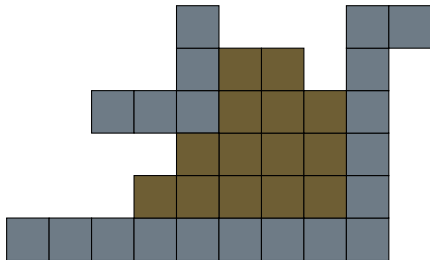


# Problem Example

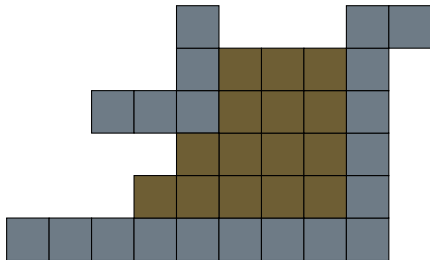




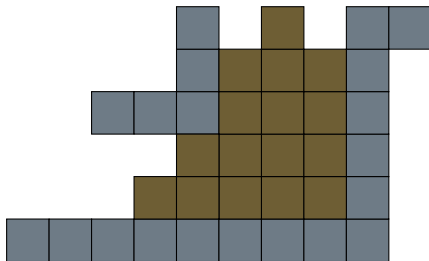
# Problem Example



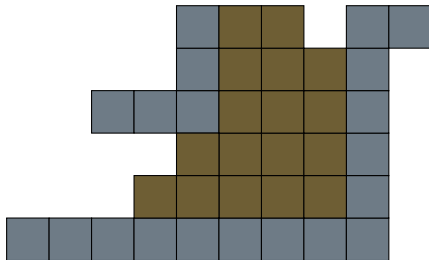
# Problem Example



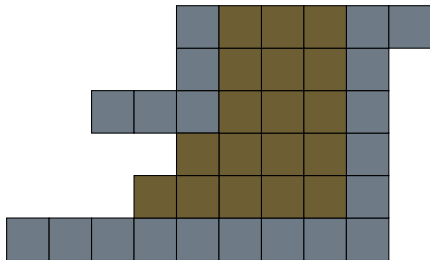
# Problem Example



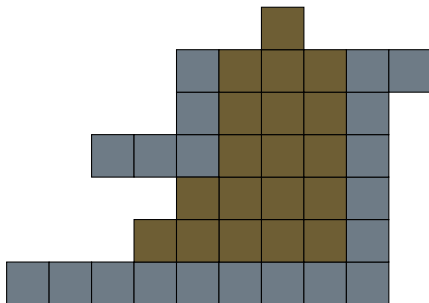
## Problem Example



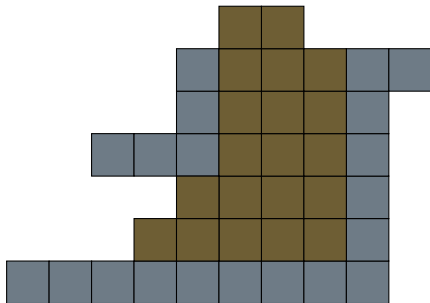
# Problem Example



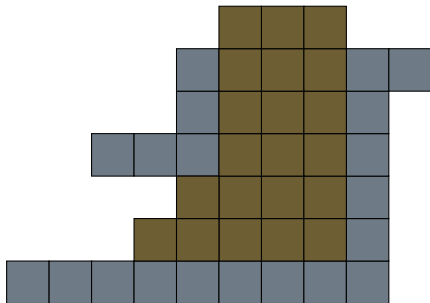
# Problem Example



# Problem Example

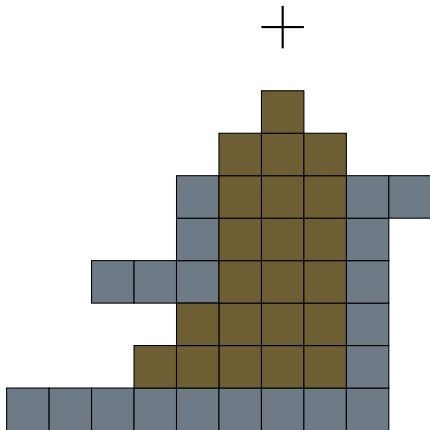


# Problem Example

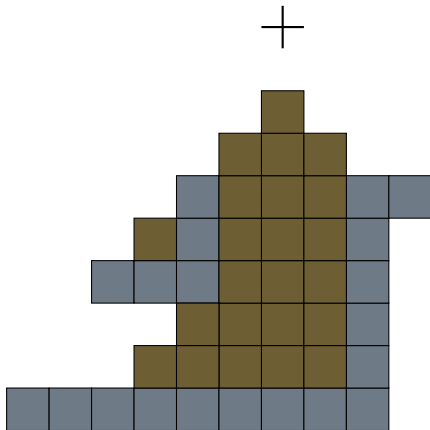




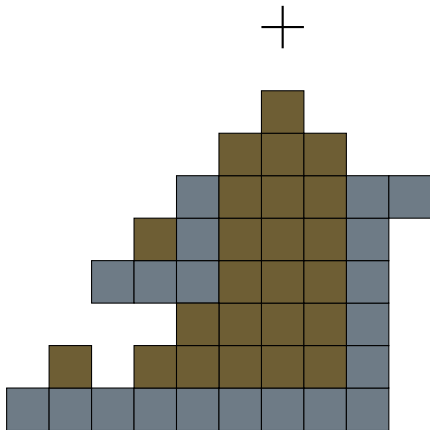
# Problem Example



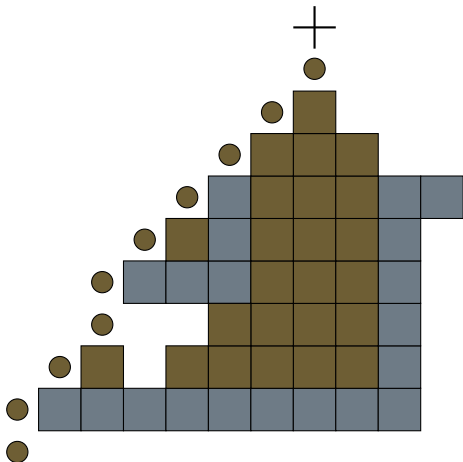
# Problem Example



# Problem Example

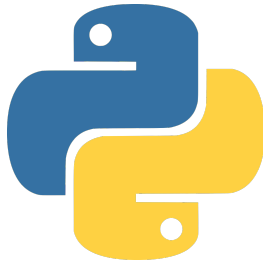


# Problem Example



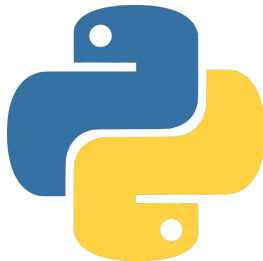
# Programming Language

- ease of use



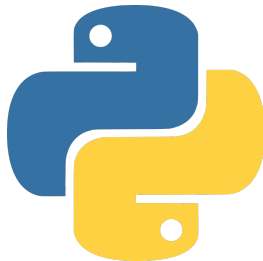
# Programming Language

- ease of use
- no runtime or memory constraints



# Programming Language

- ease of use
- no runtime or memory constraints
- me being proficient in the language



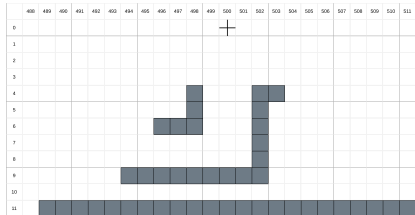
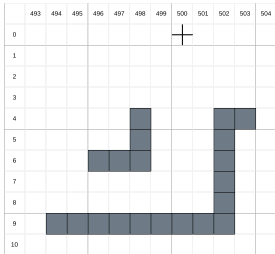
# Input Details

```
1  498,4 -> 498,6 -> 496,6
2  503,4 -> 502,4 -> 502,9 -> 494,9
```

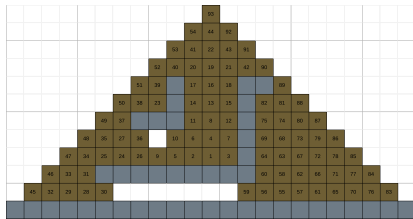
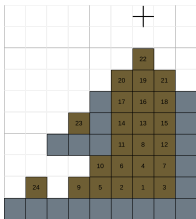


# Input Details

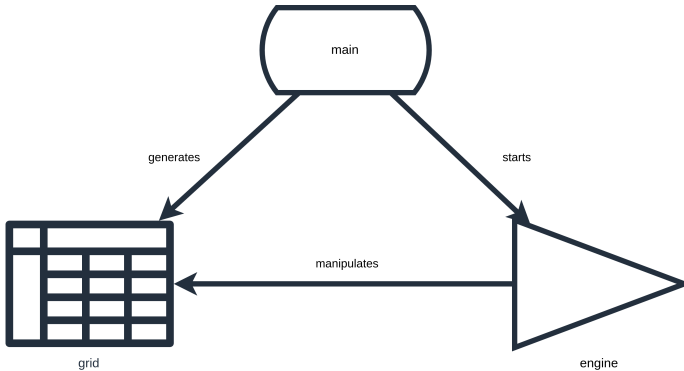
- 1 498,4 -> 498,6 -> 496,6
- 2 503,4 -> 502,4 -> 502,9 -> 494,9



# Output Details



# Solution Approach



# Code Example

```
1 num = 0
2 while True:
3     sand = (500, 0)
4     while True:
5         if self.grid.is_air_at(Coordinate((sand[0], sand[1] + 1))):
6             sand = (sand[0], sand[1] + 1)
7         elif self.grid.is_air_at(Coordinate((sand[0] - 1, sand[1] + 1))):
8             sand = (sand[0] - 1, sand[1] + 1)
9         elif self.grid.is_air_at(Coordinate((sand[0] + 1, sand[1] + 1))):
10            sand = (sand[0] + 1, sand[1] + 1)
11        else:
12            self.grid.add(Object((Material.solid_sand, Coordinate(sand))))
13            break
14        if sand[1] >= self.grid.get_last_row() + 1:
15            if not part2:
16                return num
17            self.grid.add(Object((Material.solid_sand, Coordinate(sand))))
18            break
19        num += 1
20        if sand == (500, 0):
21            return num
```

# Live Demo

# Key Takeaways & Outlook

- rock structure is created using the input coordinates

# Key Takeaways & Outlook

- rock structure is created using the input coordinates
- dynamically calculate the number of sand

# Key Takeaways & Outlook

- rock structure is created using the input coordinates
- dynamically calculate the number of sand
- adjust grid to be optimized for memory or computational performance



# Key Takeaways & Outlook

- rock structure is created using the input coordinates
- dynamically calculate the number of sand
- adjust grid to be optimized for memory or computational performance
- render falling sand