

# The Puzzles of MrHeer

MrHeer

August 3, 2019

# Contents

1	Throwing eggs from a building	1
2	Throwing two eggs from a building	2
3	3-collinearity	3
4	Queue with three stacks	4
5	Queue with two stacks	5

# 1 Throwing eggs from a building

## Question

Suppose that you have an  $N$ -story building and plenty of eggs. Suppose also that an egg is broken if it is thrown off floor  $F$  or higher, and unhurt otherwise. First, devise a strategy to determine the value of  $F$  such that the number of broken eggs is  $\sim \lg N$  when using  $\sim \lg N$  throws, then find a way to reduce the cost to  $\sim 2 \lg F$ .

## Answer

$\sim \lg N$ : start at the middle, always cut search space in half  $\rightarrow$  binary search.  
 $\sim 2 \lg F$ : start at 1, next 2, 4, 8 (i.e.,  $2^i$ ), once the egg breaks after ( $\sim \lg F$  steps) do binary search in the smaller search space (range  $< F$  and hence number of searches  $< \sim \lg F$ )  $\rightarrow$  exponential search.

$$2^{\lceil \lg F \rceil - 1} < F \leq 2^{\lceil \lg F \rceil}$$

$$range = 2^{\lceil \lg F \rceil} - 2^{\lceil \lg F \rceil - 1} = 2^{\lceil \lg F \rceil - 1} < 2^{\lg F} = F$$

$$range < F$$

## 2 Throwing two eggs from a building

### Question

Consider the previous question, but now suppose you only have two eggs, and your cost model is the number of throws. Devise a strategy to determine  $F$  such that the number of throws is at most  $2\sqrt{N}$ , then find a way to reduce the cost to  $\sim c\sqrt{F}$ . This is analogous to a situation where search hits (egg intact) are much cheaper than misses (egg broken).

### Answer

Let us make our first attempt on  $k$ 'th floor. If it breaks, we try remaining  $(k - 1)$  floors one by one. So in worst case, we make  $k$  trials. If it doesn't break, we jump  $(k - 1)$  floors (Because we have already made one attempt and we don't want to go beyond  $k$  attempts. Therefore  $(k - 1)$  attempts are available), Next floor we try is floor  $k + (k - 1)$  Similarly, if this drop does not break, next need to jump up to floor  $k + (k - 1) + (k - 2)$ , then  $k + (k - 1) + (k - 2) + (k - 3)$  and so on. Since the last floor to be tried is  $F$ 'th floor, sum of series should be  $F$  for optimal value of  $k$ .

$$k + (k - 1) + (k - 2) + (k - 3) + \cdots + 1 \geq F$$

$$\frac{k(k + 1)}{2} \geq F$$

$$k \geq \frac{\sqrt{8F + 1} - 1}{2}$$

$$k_{min} = \lceil \frac{\sqrt{8F + 1} - 1}{2} \rceil \sim \sqrt{2F}$$

### Official solution

Solution to Part 1: To achieve  $2\sqrt{N}$ , drop eggs at floors  $\sqrt{N}$ ,  $2*\sqrt{N}$ ,  $3*\sqrt{N}$ , ...,  $\sqrt{N}\sqrt{N}$ . (For simplicity, we assume here that  $\sqrt{N}$  is an integer.) Let assume that the egg broke at level  $k\sqrt{N}$ . With the second egg you should then perform a linear search in the interval  $(k - 1)\sqrt{N}$  to  $k\sqrt{N}$ . In total you will be able to find the floor  $F$  in at most  $2\sqrt{N}$  trials.

Hint for Part 2:  $1 + 2 + 3 + \cdots + k \sim \frac{1}{2}k^2 \geq F$ .

### 3 3-collinearity

#### Question

Suppose that you have an algorithm that takes as input  $N$  distinct points in the plane and can return the number of triples that fall on the same line. Show that you can use this algorithm to solve the 3-sum problem. Strong hint: Use algebra to show that  $(a, a^3)$ ,  $(b, b^3)$ , and  $(c, c^3)$  are collinear if and only if  $a + b + c = 0$ .

#### Answer

*Proof.*  $(a, a^3)$ ,  $(b, b^3)$ , and  $(c, c^3)$  are collinear if and only if  $a + b + c = 0$ : We use a formulation of collinearity which equates gradients (assuming our points are distinct)

$$\frac{y_2 - y_1}{x_2 - x_1} = \frac{y_3 - y_1}{x_3 - x_1}$$

This becomes

$$\frac{b^3 - a^3}{b - a} = \frac{c^3 - a^3}{c - a}$$

which leaves us with

$$b^2 + ab + a^2 = c^2 + ac + a^2$$

so that

$$b^2 - c^2 = a(c - b)$$

$c \neq b$  so we have  $a = -(b + c)$

$$a + b + c = 0$$

□

## 4 Queue with three stacks

### Question

Implement a queue with three stacks so that each queue operation takes a constant (worst-case) number of stack operations.

### Answer

## 5 Queue with two stacks

### Question

Implement a queue with two stacks so that each queue operation takes a constant amortized number of stack operations. Hint: If you push elements onto a stack and then pop them all, they appear in reverse order. If you repeat this process, they're now back in order.

### Answer

---

Transfer

---

**Result:** stack1 to stack2 and reverse order

```
while stack1 is not empty do  
    | Item tmp = stack1.pop();  
    | stack2.push(tmp);  
end
```

---

---

Enqueue

---

**Input:** element item

**Result:** enqueue

```
stack1.push(item);
```

---

---

Dequeue

---

**Result:** dequeue

```
if queue is empty then  
    | throw NoSuchElementException;  
end  
if stack2 is empty then  
    | transfer();  
end  
return stack2.pop();
```

---

## Code

```
public class QueueWithTwoStacks<Item> {
    private Stack<Item> stack1;
    private Stack<Item> stack2;

    public QueueWithTwoStacks() {
        stack1 = new Stack<>();
        stack2 = new Stack<>();
    }

    private void transfer() {
        while (!stack1.isEmpty()) {
            Item tmp = stack1.pop();
            stack2.push(tmp);
        }
    }

    public void enqueue(Item item) {
        stack1.push(item);
    }

    public Item dequeue() {
        if (isEmpty()) throw new NoSuchElementException("Queue underflow");
        if (stack2.isEmpty()) transfer();
        return stack2.pop();
    }
}
```

## Analysis