## Mogussort

- 1. Mogussort
- 2. **L**ore
- 3. Installation
- 4. How it works
- 5. Performance
- 6. Sus
- 7. Language





#### 1. Mogussort

World's ( SUSSIEST sorting algorithm

# 2. Lore

Your normal and boring sorting algorithms are just too fast? You want to introduce some sussy code to your programs?

Then you have come to the right place.

Inspired by the fantastic Csussus version and certified extra sussy by the original author!

#### 3. Installation

You can install Mogussort from PyPi with pip install Mogussort.



SUSSY

구FS 1/5

#### 4. How it works

There are 2 (two, 2 ) lists, crew\_in and crew\_out.

- crew\_in contains all crewmates at the start of the game.

crew\_out is empty.

The algorithm will vote out one random crewmate and put it in crew\_out That crewmate is then removed from crew\_in. Then, it checks if crew\_out is correctly sorted (ascending).

If it is correnct, it will continue to vote and populate crew\_out.

If the crewmates are not in the right order an Impostor (sus, baka) has been spotted. This means that the game is lost, and crew\_out will be cleared, and crew\_in will be reset to the original form.

The algorithm will stop once all crewmates are in the right order.

구FS 2 / 5

### 5. Performance

The mogussort solves all of your sorting problems with an incredible, luck based runtime. You might ask "how fast is it exactly?".



Well, having c crew mates the probability p of sorting voting correctly in 1 attempts to sort emergency meeting is:

$$p = \frac{1}{c} \cdot \frac{1}{c - 1} \cdot \frac{1}{c - 2} \cdot \dots \cdot \frac{1}{1} \tag{1}$$

$$=\frac{1}{c!}\tag{2}$$

That means the probability to fail in 1 emergency meeting is:

$$1 - p = 1 - \frac{1}{c!} \tag{3}$$

If we make n emergency meetings, the chance P of 1 success is:

$$P = 1 - (1 - p)^n \tag{4}$$

$$=1-(1-\frac{1}{c!})^n\tag{5}$$

So, how many meetings do we need until we have a chance of  ${\cal P}$  to have them sorted correctly once?

$$P = 1 - (1 - \frac{1}{c!})^n \tag{6}$$

$$(1 - \frac{1}{c!})^n = 1 - P \tag{7}$$

$$n = \log_{1 - \frac{1}{c!}} (1 - P) \tag{8}$$

$$n = \frac{\lg(1-P)}{\lg(1-\frac{1}{c!})} \tag{9}$$

So if we manage m meetings per second, mogussort will finish with a probability P within time  $t_P$ :

$$t_P = \frac{n}{m} \tag{10}$$

$$=\frac{\lg(1-P)}{\lg(1-\frac{1}{c!})}\cdot\frac{1}{m}\tag{11}$$

구FS 3 / 5

Let's assume we manage  $m=60\mathrm{k}\frac{\mathrm{meeting}}{\mathrm{s}}$  , which is a value I roughly get on my computer (Intel Core i7-8550U). Then mogussort will finish with a probability P of 0,50 and 0,95 within:

Crew Mates c	Time $t_{0,50}$	Time $t_{0,50}$	Time $t_{0,95}$	Time $t_{0,95}$
2	16,67µs		72, 03µs	
3	63,36µs		273, 9μs	
4	271, 4μs		1, 173ms	
5	1,381ms		5,966ms	
6	8,312ms		35, 92ms	
7	58, 22ms		251,6ms	
8	465, 8ms		2,013s	
9	4, 192s		18, 12s	
10	41,92s		181,2s	(3, 0min)
11	461, 1s	(7, 7min)	1,993ks	(33min)
12	5, 534ks	(1,5h)	23,92ks	(6, 6h)
13	71,94ks	(20h)	310, 9ks	(3, 6d)
14	1,007Ms	(12d)	4,353Ms	(50d)
15	15, 11Ms	(170d)	65, 29Ms	(2, 1a)
16	242, 0Ms	(7,7a)	1,046Gs	(33a)
17	4, 162Gs	(132a)	17, 99Gs	(570a)
18	104, 1Gs	(3.297a)	449, 7Gs	(14.251a)

This even means you might be able to play a game of Among Us while you wait for the algorithm to finish (now isn't that fantastic?).

구FS 4 / 5

#### 6. Sus

#### 7. Language

Made in py sus to improve performance in comparison to the original Csussus version.



pysus

구FS 5 / 5