Why Computing?

[1] Mohammed VI Polytechnic University (UM6P), Morocco

October 17, 2025

Warm-up: Addition (by hand)

$$27 + 18 = \ 45$$

Warm-up: Addition (by hand)

$$27 + 18 = \ 45$$

Easy!

When the numbers blow up...

$$873{,}942{,}551 \; + \; 6{,}408{,}177{,}903 \; = ?$$

$$37{,}941 \; \times \; 28{,}307 \; = ?$$

When the numbers blow up...

$$873{,}942{,}551 \; + \; 6{,}408{,}177{,}903 \; = ?$$

$$37{,}941 \; \times \; 28{,}307 \; = ?$$

Takeaway

Still "just" arithmetic, but now a **calculator** is the right tool.

New task: Do two lists share a number?

List A: [2, 7, 5, 9]
List B: [4, 1, 7]

New task: Do two lists share a number?

List A: [2, 7, 5, 9]
List B: [4, 1, 7]

Answer: Yes (they both contain 7).

We solved it by inspection, pen & paper is fine here.

What does “big” look like? (snapshot: 50 numbers each)

Each list really has 100,000 numbers. Below is just a tiny snapshot:

List A (50 of 100,000)

102	745	23	901	66
118	432	987	54	210
77	863	44	593	12
380	259	711	33	642
508	19	274	826	941
305	718	660	421	88
137	999	246	555	802
319	703	28	474	615
91	284	736	467	50
122	803	968	39	702

· · · and **99,950 more**.

List B (50 of 100,000)

14	745	990	301	508
72	463	211	86	593
640	19	377	258	44
736	402	55	118	28
274	903	501	66	839
702	33	947	284	129
615	470	12	777	260
421	137	380	967	777
259	555	642	901	473
946	320	802	77	468

· · · and **99,950 more**.

Why we need a computer

You *might* spot a common number here by eye, but at full size (100k vs 100k) we need a precise **procedure**, not a calculator. A computer is **fast but dumb**: it does exactly what we tell it.

A tiny program that answers it (what we'll learn)

```
def share_element(A, B):
    seen = set(A)
    for x in B:
        if x in seen:
            return True
    return False


print(share_element([2,7,5,9], [4,1,7]))   # -> True
```

Over the next weeks we'll practice:

- starting on paper → writing clear steps (pseudocode)
- analyzing cost (time/memory) → picking good data structures
- translating into code → testing on small, then scaling up

Puzzle: The 20 doors

There are **20 closed** doors, numbered 1..20.
For pass $p = 1$ to 20: **toggle** every door whose number is a multiple of p.

Question: Which doors end up **open**?

Puzzle: The 20 doors

There are **20 closed** doors, numbered 1..20.
For pass $p = 1$ to 20: **toggle** every door whose number is a multiple of p.

Question: Which doors end up **open**?

Try a smaller case first (by hand), then generalize.

Try a tiny case (by hand): $N = 10$

Toggle pattern $\Rightarrow$ door d flips once per **divisor** of d.
Count divisors for 1..10 and mark odd/even:

$$\begin{array}{ll} 1: & 1 \Rightarrow 1 \text{ flip (odd)} \\ 2: & 1,2 \Rightarrow 2 \text{ flips (even)} \\ 3: & 1,3 \Rightarrow 2 \text{ flips (even)} \\ 4: & 1,2,4 \Rightarrow 3 \text{ flips (odd)} \\ 5: & 1,5 \Rightarrow 2 \text{ flips (even)} \\ 6: & 1,2,3,6 \Rightarrow 4 \text{ flips (even)} \\ 7: & 1,7 \Rightarrow 2 \text{ flips (even)} \\ 8: & 1,2,4,8 \Rightarrow 4 \text{ flips (even)} \\ 9: & 1,3,9 \Rightarrow 3 \text{ flips (odd)} \\ 10: & 1,2,5,10 \Rightarrow 4 \text{ flips (even)} \end{array}$$

For $N = 10$, the open doors are $\{1, 4, 9\}$.

Key insight: Why do some doors flip an odd number of times?

Divisors usually come in **pairs**: if $a \mid d$, then $a \cdot b = d$ with partner $b = \frac{d}{a}$.

Key insight: Why do some doors flip an odd number of times?

Divisors usually come in **pairs**: if $a \mid d$, then $a \cdot b = d$ with partner $b = \frac{d}{a}$.

- Paired divisors contribute *two* flips $\Rightarrow$ even.
- **Unpaired** divisor happens only when $a = b \Rightarrow a^2 = d$.

Conclusion

A door flips an **odd** number of times $\iff$ its number is a **perfect square**.

Back to 20 doors

Perfect squares ≤ 20 are: $1, 4, 9, 16.$

So the open doors are $\boxed{\{1, 4, 9, 16\}}$.

(You can verify quickly: each of these has an odd number of divisors.)

Scaling up: N doors

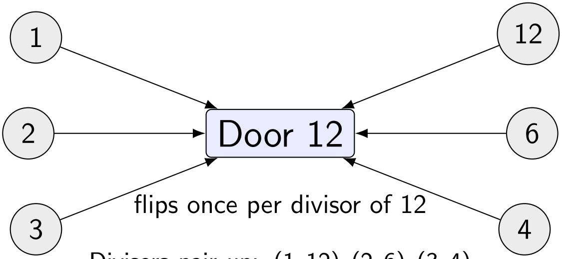
Open doors $=$ all perfect squares $\leq N$. Count $= \lfloor \sqrt{N} \rfloor$.

Examples:

- $N = 20 \Rightarrow \lfloor \sqrt{20} \rfloor = 4$ doors open: $\{1, 4, 9, 16\}$.
- $N = 100 \Rightarrow \lfloor \sqrt{100} \rfloor = 10$ doors open: $\{1^2, \ldots, 10^2\}$.
- $N = 1000 \Rightarrow \lfloor \sqrt{1000} \rfloor = 31$ doors open.

The "calculator way" would toggle door-by-door; the **structure** (divisors) gives a one-line answer.

Why flips follow divisors: example door 12



Divisors pair up: $(1,12),(2,6),(3,4)$

$\Rightarrow$ even flips $\Rightarrow$ door 12 ends closed.

Only a perfect square has an unpaired divisor (e.g., $4 = 2^2$).

20 doors. Which are open at the end?

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20

Open doors are the perfect squares: $\{1, 4, 9, 16\}$.

Python demo: simulate vs. use the math

```
def open_doors_sim(N):
    state = [False] * (N + 1)
    for p in range(1, N + 1):
        for d in range(p, N + 1, p):
            state[d] = not state[d]
    return [i for i in range(1, N + 1) if state[i]]

def open_doors_math(N):
    k = int(N ** 0.5)
    return [i * i for i in range(1, k + 1)]

print(open_doors_sim(20))   # -> [1, 4, 9, 16]
print(open_doors_math(20))  # -> [1, 4, 9, 16]
```

Takeaways

Computers = calculators on steroids

Fast but not clever. They need clear, step-by-step instructions.

Over the next weeks, we'll solve a lot of problems
and write a lot of `Python` to do it.

Thank you!

Questions?