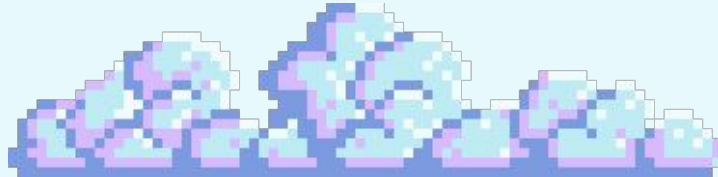




What even IS "Computational Thinking"?

START



C.T. is:

Explicit reasoning.

A problem solving method that is:

- **Clear:** Unambiguous. assumptions stated
- **Repeatable:** Same input, same outcome
- **Testable:** does the model work as expected?

If not, what step is wrong?



C.T. isn't:

A computer 'thinking'

– Rather –

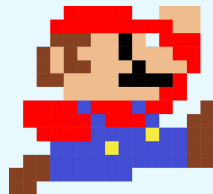
A way of stepwise, repeatable, thinking for unambiguous solutions



C.T. is:

The 4 steps:

1. **Decompose** a Big problem → small steps
2. **Recognize patterns**
3. **Simplify, purposefully.** Focus on what matters
4. **Algorithmic design.** Explicit steps + decision rules



C.T. isn't:

Truth - Limited by inputs and assumptions

No implicit sense of:

- **Time** - before/after, recent
- **Learning** - knowledge from past events
- **Self** - no values, ethics, fairness

YOU must explicitly define these!



C.T. VS. Coding:

C.T. is the Plan:

Define inputs/outputs

Decompose,

Recognize patterns,

Simplify,

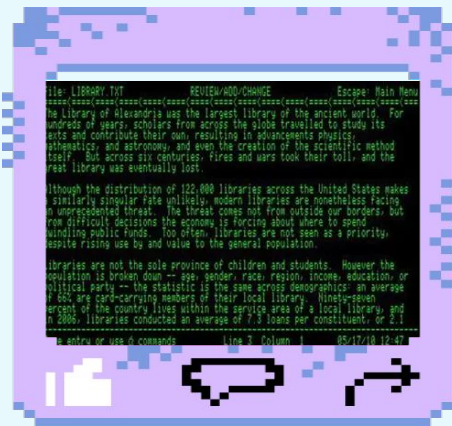
Design.

Coding is the implementation:

Expression of steps in language so a computer can follow them exactly

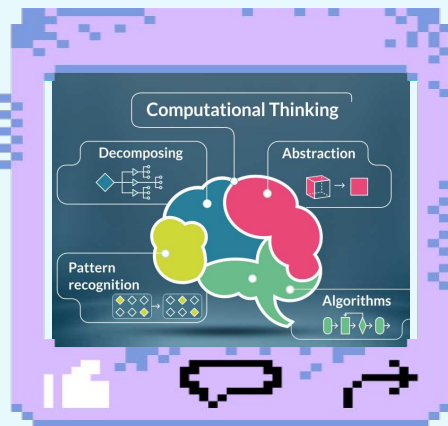


You don't need to know the backend...



Coding feels like this when unfamiliar

Green phosphorous blinking at you unforgettingly



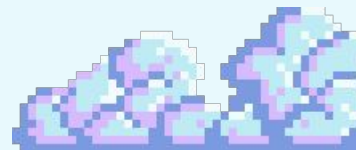
When really using this

Computational Thinking as a way of using what we know of how computers function to use the tool and get solutions



Can lead to this

Becoming a coding guru





But it helps!



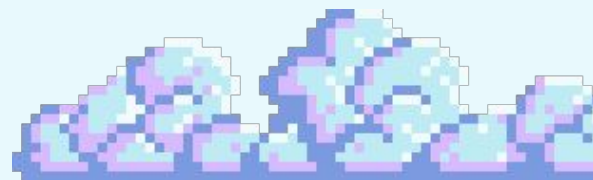
Just like...

You can't design a good protocol without knowing what size pipette to use, or how big a sample to collect

Understanding the tools (computers) teaches:

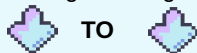
Loops (can) cost time ie: sorting 1M things \neq sorting 10 things

Memory is finite and what are ways/how can you can work with the data to make computing easier/faster



Computers can teach us

Learning how to go from "I want to solve this problem" (but don't know how to in an efficient manner)

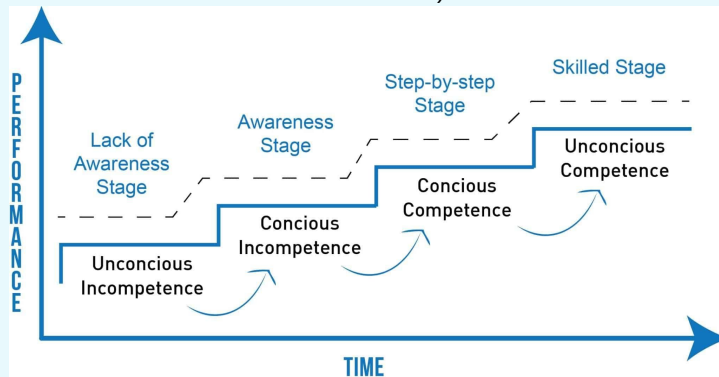


"I can design a process a machine can execute efficiently"

This is hard to learn...

But can be done!

Patience and grace with ourselves (and others)





Application



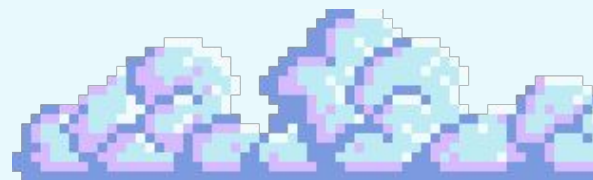
When collecting data, it's very helpful to think about what analyses you plan to run down the line

- Help you collect all the necessary data and format it correctly up front (saves time on the back end)
- “Your worst enemy in science is you from 1 year ago”



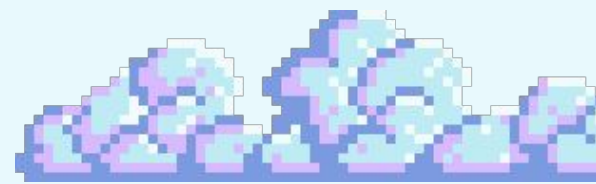
With coding sometimes less is more

- Making sure that you explain the code that isn't your strong suit



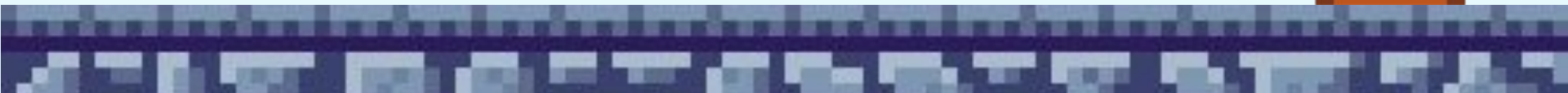


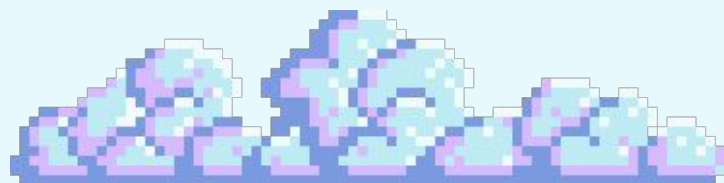
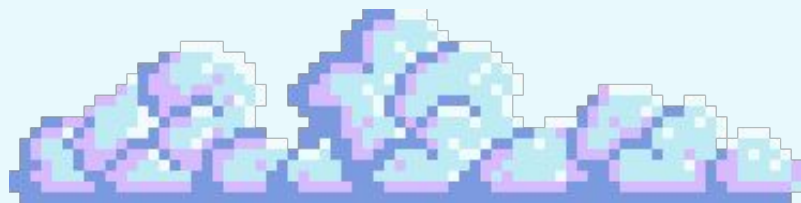
Application Activity



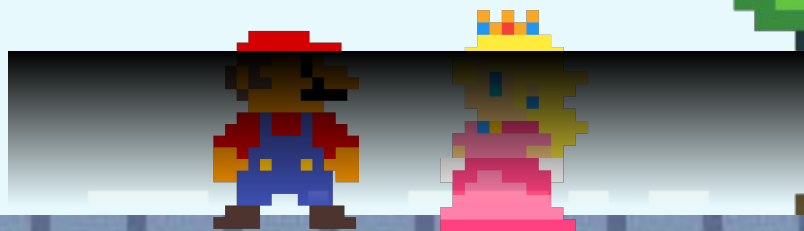
◆ Objective: With a partner, discuss a time where you could have used **computational thinking** without even knowing (0/1 complete)

◆ Objective: Discuss a time where using **computational thinking** would have helped answer your questions (0/1 complete)





GAME
OVER



What is "machine" learning

