Let's Explore the Magic of Programming!

Welcome to MathCodeLab Level 1!

Together, we'll discover how to speak the language of computers and create amazing things with code!

First let's get comfortable!

Don't be shy – everyone's excited to meet you!

Let's go around and share:

Your name

Your favorite game, app, or cartoon character

What is Computer Science?

What is Programming?

Nice, Good Job!

Let’s get into some Class Rules

Our Awesome Class Ground Rules

Mute Your Mic

Click the mic button to mute when you're not talking. This helps everyone hear clearly during our coding discussions!

Raise Hand or React

Use the "Raise Hand" button or react with emojis when you want to share ideas or ask questions about your code.

Be Kind in Chat & Talk

Use friendly words in chat and when speaking. Remember, all coders make mistakes and learn from each other!

More Class Ground Rules

Keep Camera On

Seeing your face helps us feel like a real coding team! It's okay if you sometimes need it off.

Stay Focused

Stay in Teams only — no switching to games, YouTube, or texting during our coding adventures!

Use Chat Wisely

Chat is for learning, asking questions, and helping each other solve coding problems — not for spamming.

Following these rules helps everyone have the best experience possible!

Sandwich Algorithm Challenge

Can you describe steps to make a sandwich?

Sandwich Algorithm

- ✓ Take two slices of Bread
- ✓ Put Peanut Butter on One Slice
- ✓ Put Jelly on the other Slice
- ✓ Put them together

Sandwich Algorithm

Clear Requirements and Instructions

It is super essential to ask for clear requirements.
and
Give computers super precise and exact instructions, just like dish recipe!

Sandwich Algorithm

Step 1

Pick up two slices of bread.

Step 2

Spread peanut butter on one slice.

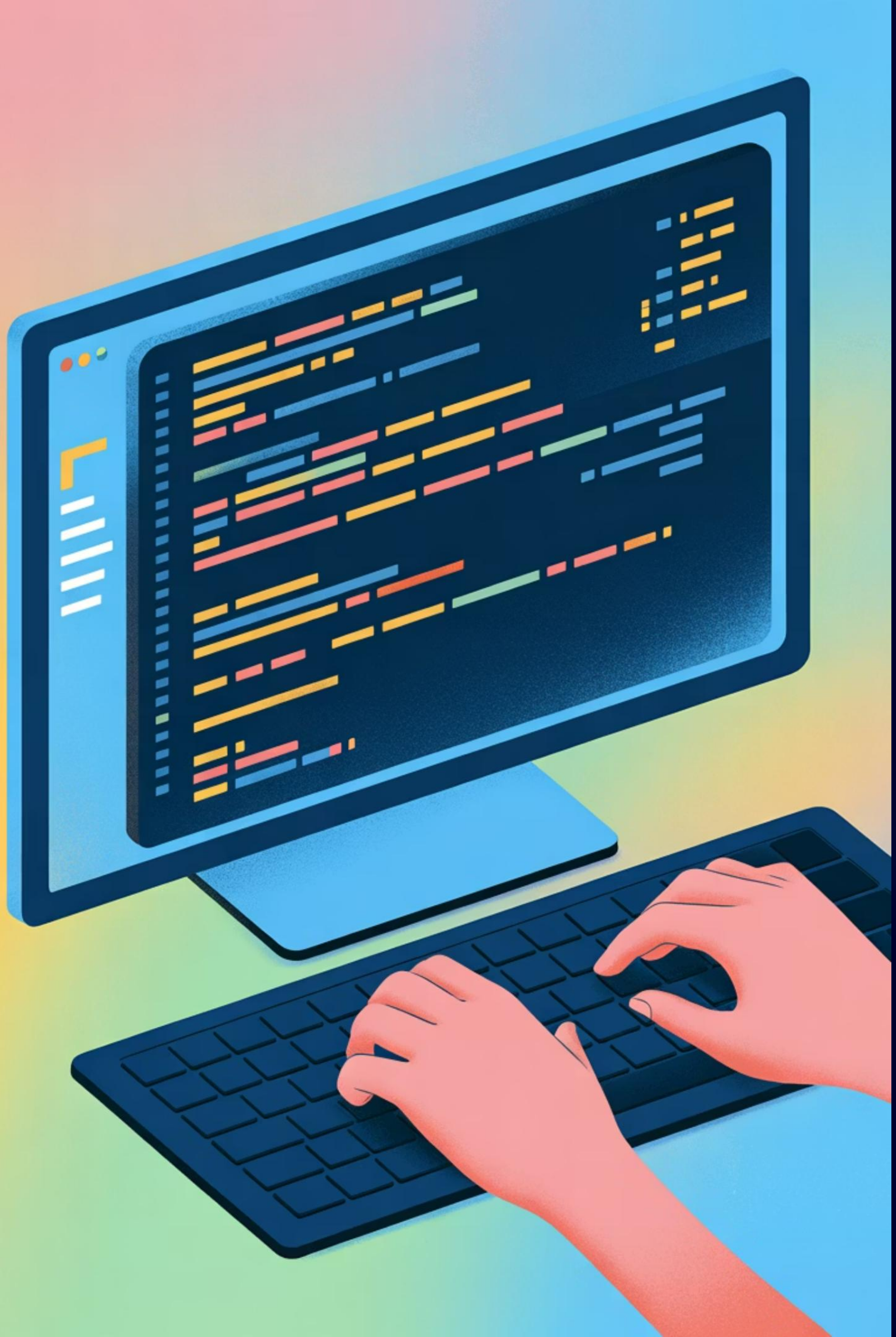
Step 3

Spread jelly on the other slice.

Step 4

Put the two slices together with spread side togather and cut in half (optional).

Computers follow algorithms just like we follow recipes!



Let's Start Our Programming Adventure!

What is Programming?

1 Programming is the **process of giving instructions to a computer** to perform specific tasks.

2 These **instructions are organized into programs** - sets of precise directions that tell computers exactly what to do and how to do it.

3 Think of programming as having a **conversation with your computer** but using a language it understands.

4 You communicate what you want it to accomplish, and **it follows your directions precisely**.

Why Learn Programming?

Solve Problems Automatically

Automate repetitive tasks and create solutions that work faster than humans ever could.

Build Digital Creations

Create websites, mobile apps, games, and other software that millions could use.

Control Hardware

Program robots, smart home devices, and other physical technology to respond to commands.

Analyze Data

Process large amounts of information to discover patterns and make predictions.

Programming empowers you to bring your ideas to life through code, opening doors to countless opportunities in our increasingly digital world.

Why Puzzles Help You Learn to Code

1

Encourage step-by-step logical reasoning.

2

Practice elimination and deduction.

3

Mirror algorithmic thinking without computers.

4

Great for building problem-solving muscles before coding.

Both puzzles and programming require systematic approaches to finding solutions through a series of logical steps.

Core Programming Concepts

Variables & Data Types

Variables are like labeled containers that store information. Data types define what kind of information they hold (numbers, text, etc.).

```
name = "Alex"  # Text
(string)age = 25      # Number
(integer)
```

Input/Output & Conditionals

Programs communicate with users through input/output. Conditionals let programs make decisions based on conditions.

```
if temperature > 80:
print("It's hot today!")
```

Loops & Functions

Loops repeat actions multiple times. Functions group code into reusable blocks that perform specific tasks.

```
def greet(name): return "Hello,
" + name
```

Thinking Like a Programmer

Break Down Problems

Divide complex challenges into smaller, manageable pieces. Solve each piece individually, then combine the solutions.

Write Clear Code

Focus on simplicity and readability. Well-organized, straightforward code is easier to understand, maintain, and debug.

Test Frequently

Run your code often to catch errors early. Make small changes and test them before moving forward.

Embrace Debugging

Errors are normal! View them as puzzles to solve rather than frustrations. Every bug fixed is a lesson learned.

Puzzle of the Day - 1

You have three light switches outside a room. Only one turns on the light inside. You can flip any switches any number of times but can enter the room only once to check.

Question: How do you determine which switch controls the light?

Think Like a Programmer

This puzzle requires you to think creatively about the problem constraints and available information - just like programming challenges do!

Solution to the Puzzle - 1

1 Step 1: Manipulate Switches

Turn on **Switch 1** and leave it on for about 5 minutes. After 5 minutes, turn **Switch 1 OFF** and immediately turn **Switch 2 ON**.

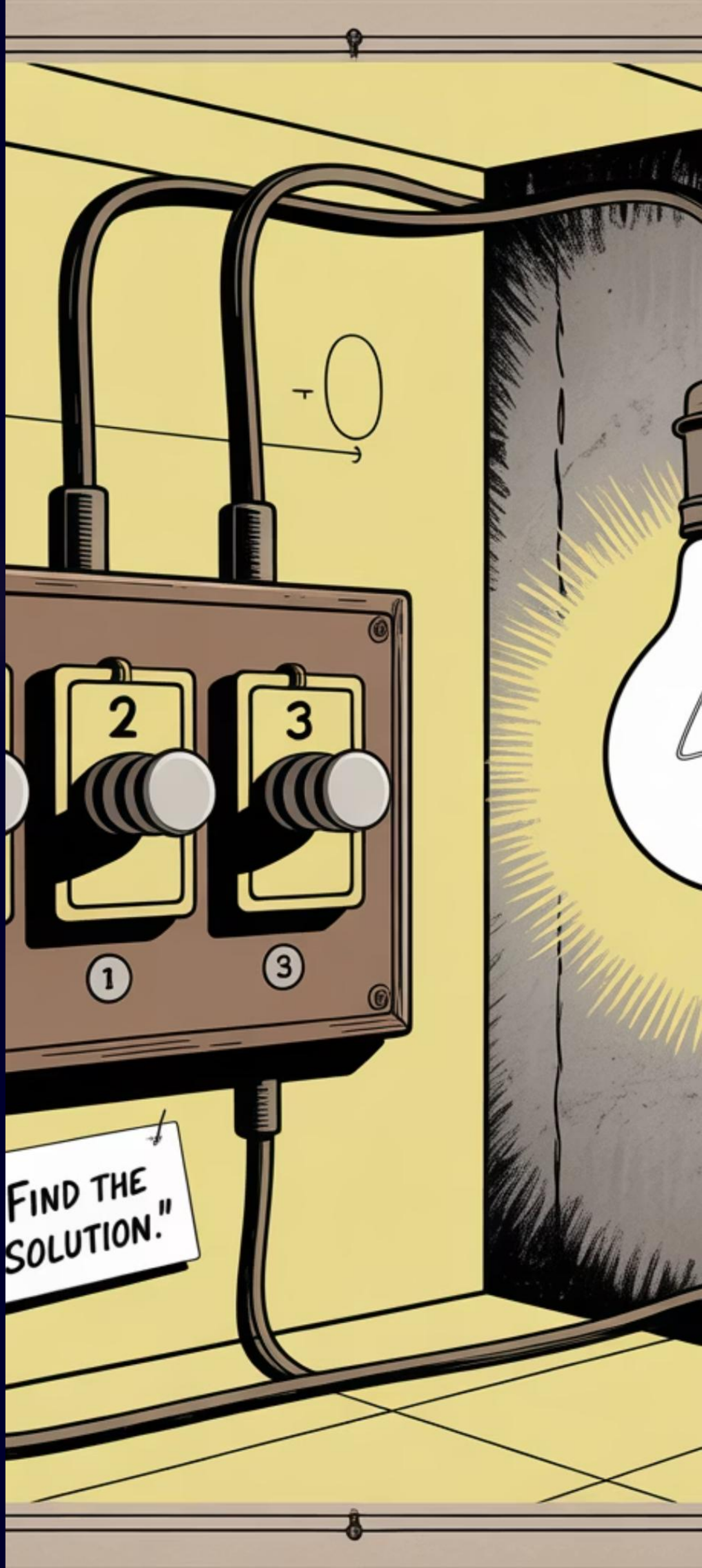
2 Step 2: Enter the Room

Immediately after adjusting the switches, enter the room to observe the state of the light bulb.

3 Step 3: Observe & Deduce

- If the light is **ON**: Switch 2 controls the light (it's the one you just turned on).
- If the light is **OFF** but the bulb is **WARM**: Switch 1 controls the light (it was on long enough to heat up).
- If the light is **OFF** and the bulb is **COLD**: Switch 3 controls the light (it was never turned on).

This clever use of light and heat allows you to determine the correct switch with just one entry into the room!



Puzzle of the Day - 2

You are standing in front of **two gates**. One gate leads to **freedom**, and the other leads to being trapped forever.
Each gate is guarded by a person:

- **One guard always tells the truth.**
- **The other always lies.**

You do **not** know which gate leads where, and you do **not** know which guard is which.

Rules:

1. You may ask **only one question**.
2. You may ask it to **only one guard**.
3. The question must be a **yes-or-no** question.

Question:
Ask **one question** to **one guard** that will **guarantee** you choose the gate to **freedom**, no matter who you ask.

Idea #1:
What question would I ask?
What would the truth-teller say?
What would the liar say?

Idea #2:
Another possible question?
Truth-teller's answer?
Liar’s answer?

Solution to the Puzzle - 2

✅ **Solution:**

> Ask either guard the following yes-or-no question: "If I were to ask the other guard which gate leads to freedom, would they say this one?" (while pointing to one of the gates)
>
> Then:
> If the guard says "Yes", choose the other gate.
> If the guard says "No", choose the gate you pointed to.

💡 **Why this works:**

If you're talking to the truth-teller, they will truthfully tell you what the liar would say — which would be a lie. So you do the opposite of their answer.

If you're talking to the liar, they will lie about what the truth-teller would say — again, giving you the opposite of the truth. So again, you do the opposite of their answer.

In both cases, reversing the answer gives you the correct gate to freedom. 🔑🚪

Coding!

www.mathcodelab.com/CodePlay