PADMA ROBOTICS PROGRAMME — TERM 3 LESSON PLANS

Term 3 · Autonomy · Weeks 1–10 · Grades 6–8

Format: 10 × 40-minute lessons · **Devices:** 1 per 4 students **Hardware:** 4 KanKaung robots, one per table, every week

> **Prerequisite:** Terms 1 and 2. **This term assumes them.** Conditionals, variables, modes and functions are not re-taught — they are used.
>
> **What this term is.** For two terms students wrote rules on a screen. Now the rules run on four wheels, four kilos, and a floor that doesn't care how clever the code is.
>
> **What this term is not.** It is not an introduction to programming. If a student cannot fill in the four-state table from memory in Week 1, they are not ready — and that is a Term 2 problem, not a Term 3 one.

The assumption rule

Every lesson names what it is building on. If a class is shaky on that thing, stop and fix it before continuing — do not push forward hoping the robot will teach it.

WEEK	BUILDS ON	THE THING THEY MUST ALREADY HAVE
1	T1 W4, T2 W5	Events, broadcast, trim as a concept
2	T2 W5	Calibration by measurement
3	T2 W1–W2	Conditions and comparisons
4	**T2 W3**	**The four-state table they wrote on paper**
5	T2 W4–W6	Variables, clamping, modes
6	T2 W10	Testing against criteria
7	T2 W2, W4	Thresholds, two-branch logic
8	T2 W6	State machines
9	T1 W9	The six-step debug protocol

WEEK	BUILDS ON	THE THING THEY MUST ALREADY HAVE
10	everything	—

Blocks this term

BLOCK	MEANING
`set motor [L] to (n)` / `set motor [R] to (n)`	Wheel speed, −100 to 100
`drive (l) (r)`	Shorthand for both wheels
`stop motors`	Both to zero
`[IR left] sees line?` / `[IR right] sees line?`	Boolean
`distance (cm)`	Ultrasonic, 2–400
`run without screen refresh`	Custom-block toggle

> **Confirm against your build in Week 1.** If the names differ, change the table on the wall and nothing else.

File naming

`P3_01_Pins`, `P3_02_Drift`, … `P3_10_Runner`.

Standard materials

Devices · 4 robots · **fresh AA cells every week** · 1 m taped corridor per table · **mat per table** (60×90 cm white board, 19 mm black tape) · stopwatch · spare servo · roll of black tape · the class's four-state tables from Term 2.

WEEK 1

Meet KanKaung

1. Lesson Info Block

Topic	The robot's pin map; first motor command; trim as remembered knowledge
Duration	40 minutes
Keywords	pin · GPIO · motor · command · speed · direction · trim
Mastery statement	*"I can make this machine move, and stop it, on purpose."*
Materials	Devices · 4 robots · fresh cells · **printed pin map** · 1 m corridor per table · `P3_01_Start` · exit tickets

2. Teacher Preparation Checklist

- [ ] **Pair and charge every robot before the lesson.** Do not spend lesson time here.
- [ ] Print the pin map: motors, IR left, IR right, ultrasonic trig/echo, and the 8 free pins.
- [ ] Tape a 1 m corridor per table.
- [ ] Test each robot: one wheel forward on command. Label the faulty ones.
- [ ] Write on the board: **this term, you are not in the loop.**

3. Step-by-Step Instructional Guide

WARM-UP — "WHO'S DRIVING?" (5 MIN)

> **Script:** "In Week 4 of Term 1, who decided where this machine went? … You did. You pressed the button. In Week 7 of Term 2, who decided? … You did, but you wrote the rule first. Today — nobody. From now on the robot decides, and you'll be standing next to it with your hands in your pockets."

> **Script:** "That's harder than it sounds. Every time it goes wrong you'll want to grab it. Don't. Fix the rule, not the robot."

GUIDED INSTRUCTION — PINS AND FIRST MOTION (10 MIN)

Step 1 (3 min). Hand out the pin map. Name the occupied pins.

> **Script:** "Four pins for the motors, four for the sensors. Eight free. In this course you use none of the free ones — those are club work. Today: motors only."

Step 2 (4 min). `set motor [L] to (50)` … `stop motors`.

> **Script:** "One command, one wheel. Now notice — it doesn't stop. It runs until you tell it. **The motor has no idea what you meant.** That's the whole term in one sentence."

Step 3 (3 min). `drive (50) (50)` and `drive (50) (-50)`.

> **Script:** "Two wheels, three behaviours: forward, turn, stop. Every movement this term is some combination of those. There is nothing else."

INDEPENDENT PRACTICE — STUDIO TASK: WAKE IT UP (20 MIN)

TASK	REQUIREMENT	TIME
A	Left wheel forward, 1 second, stop	4 min
B	`drive (50) (50)` forward, then stop	4 min
C	Spin in place with `drive (60) (-60)`	4 min
D	**Check last term's trim values** on the corridor	8 min

Save as `P3_01_Pins`.

Task D is revision: their trim values from Term 2 Week 5 should still work. Most will — the ones that don't are the first clue for next week.

DEBUGGING & DISCUSSION — "IT ONLY TURNS" (5 MIN)

SYMPTOM	CAUSE
One wheel turns	Other block missing or zero
Turns instead of straight	Both signs the same but one wheel wired backwards
Nothing at all	Not paired, or not powered
Moves, ignores commands	Two commands at once; last wins

> **Script:** "The second one is the interesting one. Both commands say forward and the robot turns. Nothing is broken — one wheel is **wired** backwards. Don't fix that in code."

Exit ticket: "*Why does the robot keep moving after one forward command?*"

4. Differentiations

Supporting struggling students

- Tasks A and B only.
- Label the robot physically: red dot left wheel, blue right, matching dots on screen.
- Pre-built project with numbers only editable.

Advanced extension challenges

- Ramp speed 0→100 over two seconds.
- Figure of eight on the floor with timed turns.
- Predict the distance for a 2-second run, then test.

Robot-table adjustments

- One driver, seated, corridor only (Week 1–2).
- Swap to the spare on any pairing failure over 60 seconds.

WEEK 2

Trim, Properly

1. Lesson Info Block

Topic	**Why trim drifts** — battery, heat, wear — and how to detect it
Duration	40 minutes
Keywords	trim · drift · battery sag · thermal · wear · re-calibration
Mastery statement	"*The number that was right in October is wrong now, and I can tell you why.*"
Materials	Devices · 4 robots · **one deliberately weak battery pack** · corridor · `P3_02_Drift` · exit tickets

> **This is not "what is trim".** They learned that in Term 2. This is why the value they wrote down is no longer true.

2. Teacher Preparation Checklist

- [ ] Have one **weak pack** ready. Test that it produces visible drift.
- [ ] Check each robot against its Term 2 trim values. Several will already have drifted.
- [ ] Print a drift log: pack type, run, drift, trim used.
- [ ] Prepare the closing demo: a robot driving straight at full charge, then curving badly at the end of the pack.

3. Step-by-Step Instructional Guide

WARM-UP — "SAME ROBOT, DIFFERENT DAY" (5 MIN)

> **Script:** "Your trim value is in your notebook from Term 2 — Week 5. Read it out. Now run the corridor with that value. Write down what happens."

Half the class will report drift. Let them be annoyed for thirty seconds.

> **Script:** "Nothing in your code changed. Your value is exactly what you measured. So what changed?"

GUIDED INSTRUCTION — THE THREE CAUSES (10 MIN)

CAUSE	WHAT HAPPENS	HOW YOU NOTICE
Battery sag	Voltage falls, servo timing shifts	Drift grows through the lesson
Heat	Servos run faster when warm	First runs differ from the fifth
Wear	A gear loses a tooth, a wheel slips	Sudden step change, not gradual

> **Script:** "Three causes, one symptom. You can't tell them apart by looking. You tell them apart by **changing one thing and re-running** — which is the six-step protocol from Term 1, applied to a machine."

The demo (3 min). Full pack: straight. Swap to the weak pack: curves within two metres.

> **Script:** "Same code. Same trim. Different battery. **Trim is a snapshot of one moment's hardware**, not a property of the robot."

INDEPENDENT PRACTICE — STUDIO TASK: DRIFT LOG (20 MIN)

TASK	REQUIREMENT	TIME
A	Run the corridor with Term 2 values. Record drift	5 min
B	Re-trim to under 3 cm	6 min
C	Run five times in a row. Does the drift change?	5 min
D	Swap to a weak pack and record the change	4 min

Save as `P3_02_Drift`**.** One sentence in the notebook: "*The most likely cause of our drift today was ___ because ___.*"

DEBUGGING & DISCUSSION — "WHICH CAUSE WAS IT?" (5 MIN)

Collect three teams' conclusions. Push on evidence:

> **Script:** "You said battery. What proved it? … You swapped the pack and the drift changed. That's evidence. 'I think it's the battery' is not."

> **Script:** "Next week you stop steering with trim and start steering with a sensor. The sensor doesn't care about your battery — but it has its own problems."

Exit ticket: "*Name two reasons a robot that drove straight yesterday curves today.*"

4. Differentiations

Supporting struggling students

- Give the measured values; they run and record only.

Advanced extension challenges

- Plot drift against run number over ten runs.
- Build a "check trim" routine that runs at the start of every session.
- Write the maintenance card.

WEEK 3

What the Sensors See

1. Lesson Info Block

Topic	IR reflectance; both sensors read; the physical truth table
Duration	40 minutes
Keywords	sensor · infrared · reflectance · boolean · threshold · surface
Mastery statement	*"The robot can tell me which side of the line it is on."*
Materials	Devices · 4 robots · 1 m straight line per table · `P3_03_Sensors` (two large indicators) · white paper and black tape scraps · exit tickets

2. Teacher Preparation Checklist

- [ ] **Know what black reads as on your units.** `1` or `0`? Write it on the board before students enter. This flips every conditional in Week 4.
- [ ] Build `P3_03_Sensors`: two large on-stage shapes, green when "sees line", red when not.
- [ ] Tape a straight 1 m line per table.
- [ ] Check sensor height: 3–8 mm above the surface.
- [ ] Have scraps of white paper, black tape, and something shiny.

3. Step-by-Step Instructional Guide

WARM-UP — "TORCH AND WALL" (5 MIN)

Shine a torch on white paper, then on black paper.

> **Script:** "The sensor sends infrared — invisible light — and measures how much bounces back. That's it. Which means it doesn't see a *line*. It sees **reflectivity**. And it will be fooled by a shiny floor or a red line."

GUIDED INSTRUCTION — READ BOTH (10 MIN)

Step 1 (3 min). Live readout of both sensors. Move a hand under one.

Step 2 (4 min). Build the truth table on the board — the same four rows they wrote in Term 2 Week 3, now with a machine attached:

LEFT	RIGHT	MEANING
line	line	Centred
—	line	Drifted right
line	—	Drifted left
—	—	Lost

> **Script:** "You wrote this in Term 2, on paper, for a machine you hadn't met. Open your notebook. Compare. Is your table right?"

This is the moment the term pays off. Let them sit with it.

Step 3 (3 min). Physically move the robot across the line by hand, motors off, watching indicators.

> **Script:** "We can read the robot's situation without it doing anything. That separation — read the world, then act on it — is the whole of Week 4."

INDEPENDENT PRACTICE — STUDIO TASK: SENSOR TRUTH TABLE (20 MIN)

TASK	REQUIREMENT	TIME
A	Two on-screen indicators, one per sensor	5 min
B	Place the robot in all four positions by hand; record	6 min
C	Compare with your Term 2 paper table; note any differences	4 min
D	Test a shiny surface and a red line; record	5 min

Save as `P3_03_Sensors`**.**

DEBUGGING & DISCUSSION — "THE SENSOR IS BROKEN" (5 MIN)

It almost never is. Check order: close enough? surface matte? LED lit? black reading correct?

> **Script:** "And the fourth check — is black `1` or `0` on *this* unit? I've written it on the board. Read it before you change your code."

Exit ticket: "*Both sensors read 'sees line'. Where is the robot?*"

4. Differentiations

Supporting struggling students

- Tasks A and B only.
- Use your hand instead of the line — faster and more visual.

Advanced extension challenges

- Sensor height experiment at 2/5/10 mm, graph reliability.
- What would a third, centre sensor change?
- Ambient light test.

WEEK 4

Four States, Four Branches

1. Lesson Info Block

Topic	Implementing the table they wrote in Term 2 — selection on a real machine
Duration	40 minutes
Keywords	selection · branch · chain · bang-bang control · correction
Mastery statement	*"The robot decides from what it senses."*
Materials	Devices · 4 robots · 1 m line per table · **the Term 2 four-state tables** · `P3_04_Follow` · exit tickets

> **The lesson this term exists for.** Everything before it and after it supports this hour.

2. Teacher Preparation Checklist

- [ ] **Bring the Term 2 tables AND flowcharts.** Hand each team its own two sheets. This is not optional.
- [ ] If a team has only one of the two, re-run Term 2 Week 3 Part C with them during the warm-up — ten minutes now, or a lost lesson later.
- [ ] Set speed to **40** for the first attempt. This single choice decides whether the lesson works.
- [ ] Keep the line short and straight this week. Curves are Week 6.
- [ ] Confirm the `if / else if / else` chain nests comfortably in your Padma build.

3. Step-by-Step Instructional Guide

WARM-UP — "YOUR TABLE, YOUR MACHINE" (5 MIN)

Hand out the Term 2 sheets — table and flowchart side by side.

> **Script:** "Two pieces of paper from nine weeks ago. The table says **what** the robot should do in each of four situations. The flowchart says **in what order it checks them**. Today you need both."

> **Script:** "Nine weeks ago you filled this in for a machine you had never seen. Today that machine is on your table and this is the program. Read your own handwriting. Is it right?"

> **Script:** "If your table is wrong, your robot will be wrong — and you'll find out in four minutes, which is the best possible thing to happen to you today."

GUIDED INSTRUCTION — FROM TABLE TO CHAIN (10 MIN)

Step 1 (4 min). Read row one aloud together, then write it:

```
if <[IR left] sees line?> and <[IR right] sees line?> then [ drive 50 50 ]
```

> **Script:** "One row, one branch. `and` because both must be true."

Step 2 (4 min). The shape of the whole thing:

```
if      L and R        → both forward
else if not L and R    → swing right
else if L and not R    → swing left
else                   → stop
```

> **Script:** "Four rows, four branches, one `else` for everything unexpected. The `else` is the robot saying **I do not know**."

Step 3 (2 min). Wrap in `forever`.

> **Script:** "The whole thing repeats, hundreds of times a second. Look, decide, correct. That's a **control loop**."

INDEPENDENT PRACTICE — STUDIO TASK: FOLLOW THE LINE (20 MIN)

TASK	REQUIREMENT	TIME
A	Compare your Term 2 table with the robot's actual behaviour	4 min
B	Build the four branches	10 min
C	Place it left of the line, then right, then on it — must recover from both	6 min

Save as `P3_04_Follow`.

The most common failure: students write `if L` and `if R` as two separate `if` blocks instead of a chain. Both fire in the same pass and the robot jerks. The chain is the lesson.

DEBUGGING & DISCUSSION — "IT TURNS THE WRONG WAY" (5 MIN)

> **Script:** "It drifts left and turns further left. Either your table is wrong or your **sensor wiring is swapped**. How do you tell?"

Answer: move it by hand. Which indicator turns red?

> **Script:** "You cannot debug a robot by reading the code. You debug it by **asking what it sees** and comparing with what you expected."

Exit ticket: "*Reads left = nothing, right = line. Which wheel speeds up?*"

4. Differentiations

Supporting struggling students

- Two branches: "centred" and "lost".
- Printed skeleton with conditions spelled in English above each branch.
- Speed 30.

Advanced extension challenges

- Proportional correction instead of two speeds.
- A fifth state: a wide line or a crossing.
- Recovery: when lost, reverse then turn back.

WEEK 5

Variables and Loop Rate

1. Lesson Info Block

Topic	Speed and correction as variables; `run without screen refresh`
Duration	40 minutes
Keywords	parameter · clamp · loop rate · frame · latency · responsiveness
Mastery statement	*"A fast robot needs a fast loop."*
Materials	Devices · 4 robots · line per table · `P3_05_Rate` · stopwatch · exit tickets

2. Teacher Preparation Checklist

- [ ] Build two versions of the follower: one with a visible `wait` and no refresh toggle; one with the loop in a custom block with **`run without screen refresh`** on.
- [ ] Time both yourself. You want real numbers.
- [ ] Confirm the toggle's location in your Padma build.

3. Step-by-Step Instructional Guide

WARM-UP — "SLOW-MOTION DRIVER" (5 MIN)

A student drives another along a line, but may only correct once every two seconds.

> **Script:** "You were always correcting something that happened two seconds ago. That's your robot right now."

GUIDED INSTRUCTION — TWO VERSIONS (10 MIN)

Step 1 (4 min). Run the slow version.

> **Script:** "Watch the wheels. They don't spin, they start and stop. The robot looks like it's thinking. It isn't — it's **waiting**."

Step 2 (4 min). Move the loop into a custom block, tick `run without screen refresh`.

> **Script:** "Same code. One checkbox. The block now runs to completion inside one frame instead of being sliced across frames."

Step 3 (2 min). Compare lap times.

> **Script:** "Same logic, same sensors, same robot. Different **loop rate**. This is the first time this year the difference is measurable in seconds."

INDEPENDENT PRACTICE — STUDIO TASK: FAST AND SMOOTH (20 MIN)

TASK	REQUIREMENT	TIME
A	Time three runs with screen refresh on	5 min
B	Move the loop into a custom block, refresh off	6 min
C	Time three runs again, same speed	5 min
D	Raise speed until it fails. Record the speed and the failure mode	4 min

Save as `P3_05_Rate`.

Task D produces the second insight: at high speed the failure is **oscillation** — past the line, overcorrect, back.

DEBUGGING & DISCUSSION — "IT'S WORSE NOW" (5 MIN)

> **Script:** "Sometimes the toggle makes it worse. A long loop with no exit can lock the frame. If your custom block contains a `forever`, the whole project freezes."

Second: the **screen stops updating**. Indicators freeze. That's normal.

Exit ticket: "*Why does a faster robot need a faster loop?*"

4. Differentiations

Supporting struggling students

- Both versions pre-built; they only time and record.

Advanced extension challenges

- Count loop iterations per second.
- Throttle deliberately and show stability improving.
- Split sensing and acting into two loops.

WEEK 6

Tuning

1. Lesson Info Block

Topic	Overshoot, oscillation, the speed/stability trade-off
Duration	40 minutes
Keywords	tuning · overshoot · oscillation · damping · stability · trade-off
Mastery statement	*"There is no best speed — there is a trade-off, and I can choose it."*
Materials	Devices · 4 robots · **curved mats** (add two 90° curves) · stopwatch · `P3_06_Tune` · exit tickets

2. Teacher Preparation Checklist

- [ ] Add curves to the mats — 90°, 15 cm radius. Keep the straight.
- [ ] Fresh batteries; tuning data on weak cells is worthless.
- [ ] Print the three-speed test sheet: speed, lap time, success out of 5.
- [ ] Mark the point where each robot leaves the line with tape during testing.

3. Step-by-Step Instructional Guide

WARM-UP — "TOO FAST TO TURN" (5 MIN)

A student runs a chair slalom at walking pace, then sprinting.

> **Script:** "Walking: every turn. Sprint: two chairs hit. Did your legs get worse? No — **your reaction time stayed the same while your speed increased.**"

GUIDED INSTRUCTION — THREE SPEEDS (10 MIN)

Step 1 (3 min). Speed 40 on the curved mat. Note lap time and failures. **Step 2 (4 min).** Speed 70. Watch it swing wide.

> **Script:** "It sees the line late, corrects hard, goes past, corrects back. Each correction is correct. Together they're a disaster."

Step 3 (3 min). `correction` as the damping dial.

> **Script:** "Oscillates? Correction too strong. Drifts off? Too weak. There is one right value **for this speed on this mat** — and it won't be right for the next speed."

INDEPENDENT PRACTICE — STUDIO TASK: TUNING TABLE · CARD 11 (20 MIN)

SPEED	CORRECTION	LAP TIME	SUCCESS /5	FAILURE MODE
40				
55				
70				
85				

Rule: find the best correction per speed, then record.

Target: fastest speed with 4/5 success.

Save as `P3_06_Tune`**.** One sentence: "*Our fastest reliable speed is ___ because ___.*"

DEBUGGING & DISCUSSION — "WHY ONLY ON THE CURVE?" (5 MIN)

> **Script:** "Straight lines are easy — barely correcting. Curves are where correction matters and where a slow loop shows up. If it only fails on curves, the problem is **latency**, not logic."

Exit ticket: "*Your robot oscillates. Which variable changes, up or down?*"

4. Differentiations

Supporting struggling students

- Two speeds instead of four.
- Pre-tuned robot; they record only.

Advanced extension challenges

- Speed scheduling: slow automatically on a turn.
- Plot speed against success rate.
- Explain damping to a non-engineer in three sentences.

WEEK 7

The Second Sense

1. Lesson Info Block

Topic	Ultrasonic distance; thresholds and hysteresis
Duration	40 minutes
Keywords	ultrasonic · distance · threshold · hysteresis · deadband · chatter
Mastery statement	*"Two thresholds, not one, or it will fight itself."*
Materials	Devices · 4 robots · cardboard target · `P3_07_Threshold` · hysteresis worksheet · exit tickets

2. Teacher Preparation Checklist

- [ ] Test `distance (cm)` on each robot at 10/30/60/100 cm. Note the bad ones.
- [ ] Know the reading with nothing in front — usually a fixed high number.
- [ ] Rehearse the chatter demo until it is uncomfortable to watch.
- [ ] Print the hysteresis worksheet: a number line with **two** marks to place.
- [ ] Tape a 1 m measuring strip per table.

3. Step-by-Step Instructional Guide

WARM-UP — "THE DOOR GREETER" (5 MIN)

> **Script:** "You are the greeter. When someone comes within two metres, say hello." A student approaches to exactly 2 m, steps back 10 cm, forward 10 cm — six hellos.

> **Script:** "The person barely moved. You greeted six times. The rule was correct every time, and unusable."

Change it: hello at 2 m, quiet until 3 m.

> **Script:** "Two marks instead of one. One hello. That's **hysteresis**."

GUIDED INSTRUCTION — ONE LINE, THEN TWO (10 MIN)

Step 1 (3 min). Single threshold. Target at exactly 25 cm → the display flickers.

> **Script:** "The distance wobbles by two centimetres. My threshold sits in the middle of the wobble. A hundred correct decisions a second, and it looks broken."

Step 2 (4 min). The number line:

```
400 ------- 40 ------- 25 ------- 2
 far     resume      stop      near
```

> **Script:** "Stop below 25. Resume above 40. Between them — **do nothing. Keep the current state.**"

Step 3 (3 min). Code with a state variable.

> **Script:** "The third line doesn't ask about distance. It asks what we already decided. **In the gap, the answer comes from memory.**"

INDEPENDENT PRACTICE — STUDIO TASK: RELIABLE OBSTACLE STOP · CARD 13 (20 MIN)

TASK	REQUIREMENT	TIME
A	Two marks on the worksheet, deadband shaded	5 min
B	Two-threshold stop, robot stops at 25 and resumes at 40	8 min
C	Hold the target in the deadband for 15 seconds. No flicker	4 min
D	Justify the gap width in one sentence	3 min

Save as `P3_07_Threshold`.

DEBUGGING & DISCUSSION — "IT STILL CHATTERS" (5 MIN)

Four causes: state variable reset each loop; both `if`s firing in one pass; deadband narrower than sensor noise; target soft or angled.

> **Script:** "Cause three is the one to teach. How much does your sensor wobble? Watch it thirty seconds and write the highest and lowest. That's **measurement**, and it's how you choose a gap instead of guessing one."

Exit ticket: "*Why does it stay stopped when the target is between the thresholds?*"

4. Differentiations

Supporting struggling students

- Two branches: near and far. No deadband.

Advanced extension challenges

- Three zones with hysteresis on both boundaries.
- Adaptive deadband measured at startup.
- Time-based guard: ignore readings shorter than 100 ms.

WEEK 8

Priority and Fail-Safe

1. Lesson Info Block

Topic	Combining behaviours; override > obstacle > line; the "I don't know" state
Duration	40 minutes
Keywords	priority · arbitration · fail safe · unknown · sanity check · latched
Mastery statement	*"The obstacle wins, and the machine can say it doesn't know."*
Materials	Devices · 4 robots · line mats · a block to place on the line · `P3_08_Safe` · exit tickets

2. Teacher Preparation Checklist

- [ ] Place a block on each line — book or box, 10 cm wide.
- [ ] Prepare `P3_08_Naive`: obstacle check *inside* the line branches. It half-works.
- [ ] Decide the priority order with the class before coding: **override > obstacle > line**.
- [ ] Know the impossible-value test: unplug a sensor and see what it reports.

3. Step-by-Step Instructional Guide

WARM-UP — "TWO RULES, ONE DRIVER" (5 MIN)

A student drives a finger along a taped line; a second holds a hand at the far end.

> **Script:** "Rule one: follow the line. Rule two: stop before the hand. Which wins if the line goes through the hand?"

> **Script:** "Because hitting something is **irreversible**. Getting off the line is not. When two rules conflict, the one whose failure is worse wins."

GUIDED INSTRUCTION — PRIORITY IS STRUCTURAL (10 MIN)

Step 1 (3 min). Show the naive version — obstacle check inside the branches. It drives past sometimes.

> **Script:** "The four line branches all set motor speeds. One of them overwrites your stop."

Step 2 (4 min).

```
if blocked        → stop, signal
else              → the four line branches
```

> **Script:** "The obstacle check goes first and everything else goes inside the `else`. Not tidier — **the line logic cannot run while blocked.** Priority is structural."

Step 3 (3 min). The unknown state.

> **Script:** "What does the sensor report when it's unplugged? Zero, forever. Zero might mean 'touching' or 'not connected'. Your code cannot tell. So it gets a third state: **I don't know.** Stop, signal, wait for a human."

INDEPENDENT PRACTICE — STUDIO TASK: GUARDIAN BEHAVIOUR · CARD 14 (20 MIN)

TASK	REQUIREMENT	TIME
A	Restructure: obstacle first, line logic inside `else`	6 min
B	Stops at 25 cm, resumes at 40 cm when the block is removed	6 min
C	UNKNOWN state on an impossible reading: stop, signal, no motion	5 min
D	Write the priority ladder from memory	3 min

Save as `P3_08_Safe`.

DEBUGGING & DISCUSSION — "IT STOPPED FOREVER" (5 MIN)

> **Script:** "It stopped at the block. You removed the block. It's still stopped. `blocked` was set to 1 and nothing sets it back. The robot is **latched**."

Exit ticket: "*What should a robot do when two sensors contradict each other?*"

4. Differentiations

Supporting struggling students

- Wrap the whole follower in `if not blocked`. One change, genuinely correct.
- UNKNOWN for the ultrasonic only.

Advanced extension challenges

- Three-way priority including a manual override button.
- Recovery: back up 10 cm before resuming.
- Count and display obstacles encountered.
- Write the safety case in five sentences a parent could read.

WEEK 9

Debugging a Physical System

1. Lesson Info Block

Topic	The four fault categories — code, wiring, power, surface
Duration	40 minutes
Keywords	fault · category · isolate · swap · reproduce · log
Mastery statement	*"Most robot faults are not code faults, and I can tell which is which — and the robot keeps its own record."*
Materials	Devices · 4 robots · **one injected fault per robot** · `P3_09_Log` **starter with the** `faults` **list pre-created** · spare servo, batteries, robot · exit tickets

2. Teacher Preparation Checklist

- [ ] Inject one fault per robot before class. Keep a private list.

    - swap one motor lead
    - disconnect one IR sensor
    - fit a weak pack
    - shiny tape across the line
    - raise one IR sensor 2 cm
    - set a trim value to +40

- [ ] Have the spares box ready.
- [ ] **Pre-create the `faults` list** in `P3_09_Log` and leave it empty. Make it global so the whole project can write to it.
- [ ] Test that `show list [faults]` displays clearly on the stage at projector distance. If not, print the list larger or use a text sprite bound to `item 1 of` , `item 2 of` .

3. Step-by-Step Instructional Guide

WARM-UP — "FOUR BUCKETS" (5 MIN)

Board: **CODE · WIRING · POWER · SURFACE.**

> **Script:** "Every failure goes in one of these. Your first job is to pick the bucket **before** you touch anything."

Sort three examples.

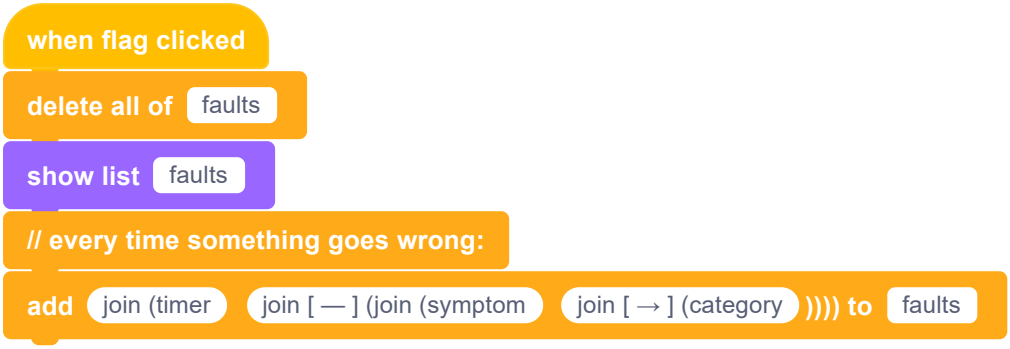
GUIDED INSTRUCTION — THE ISOLATION LADDER (10 MIN)

RUNG	QUESTION
1	Does it work on a straight line?
2	Do both sensors read correctly by hand?
3	Does the **other** robot work on this mat?
4	Does **your** robot work on another mat?
5	Do the wheels turn when commanded directly?

> **Script:** "Rung three is the important one. **Swap the robot, not the code.** If the other one works, the problem is physical. One swap, half the search space gone."

INDEPENDENT PRACTICE — STUDIO TASK: FIND THE FAULT (20 MIN)

Task 0 — build the black box (6 min, before you touch the robot). A robot that fails at minute nine leaves no evidence. So make it keep its own record.



> **Script:** "A list is a row of boxes with one name. Every time something happens you add one box. At the end you read the row and the story is there — in order, with timestamps. **You cannot count what you did not store.**"

Then the fault hunt:

1. Observe and record the symptom — **into the list, not onto paper** (4 min)
2. Name the category and add it to the list (2 min)
3. Run the isolation ladder (6 min)
4. Fix it, or make it fail safely (2 min)

At the end, `show list [faults]` on the stage and read it aloud as a team.

> **Script:** "Read your list. How many wiring faults? How many power? … Nobody counted. You just read the row. That is what a list is for — it holds the history so you don't have to remember it."

Save as `P3_09_Log`.

DEBUGGING & DISCUSSION — "WHAT DID YOU SWAP FIRST?" (5 MIN)

> **Script:** "Who swapped the servo first? Who swapped the battery? Who re-read the code? … Notice the fastest teams all did the same thing: they tested the **cheapest hypothesis first**."

> **Script:** "Battery takes five seconds. Code takes five minutes. **Always check power before you read code.**"

Point at a team's list on the stage.

> **Script:** "One more thing about that list. It has an order. Item one happened before item five. When you read it next week, the order will tell you which fault caused which — and a bag of sticky notes never does that."

Exit ticket: "*Why did we put the faults in a list instead of on paper?*"

4. Differentiations

Supporting struggling students

- A visible fault — a disconnected lead.
- Category given as a hint after two minutes.

Advanced extension challenges

- Intermittent fault that only appears when turning.
- Design a fault for another team.
- Fail-safe redesign so a disconnected sensor stops rather than drives blind.

WEEK 10

Build Sprint: The Line Runner

1. Lesson Info Block

Topic	Completing the full course; reliability
Duration	40 minutes
Keywords	course · checkpoint · reliability · coasting · stop box · done
Mastery statement	*"It finishes the course, and it finishes it more than once."*
Materials	Devices · 4 robots · **full course mats** · planning sheet · stopwatch · rubric extract · spare everything

2. Teacher Preparation Checklist

- [ ] Build the full course per table: 40 cm straight · 90° curve · 30 cm straight · **8 cm gap** · stop box.
- [ ] Build a demonstration course at the front.
- [ ] Print the definition of done as a checklist.
- [ ] Prepare `P3_10_Starter` (Week 8 build) for teams that lost work.

3. Step-by-Step Instructional Guide

WARM-UP — READ THE COURSE (5 MIN)

> **Script:** "Four features. Straight — easy. Curve — where correction matters. Gap — where you have **no reading at all** for half a second. Stop box — where you must stop without a sensor telling you to."

> **Script:** "What does the robot do when it can't see anything?"

GUIDED INSTRUCTION — THE GAP (10 MIN)

Step 1 (4 min). Manual run: it stops dead at the gap.

> **Script:** "Your `else` says 'stop'. Right for 'I'm lost', wrong for 'I'm briefly blind'. Two different situations, one branch."

Step 2 (4 min). The blind timer.

`if both off → change [blind] by 1` … `if blind < 20 → keep last speeds` … `else → stop`

> **Script:** "Now the robot has a **memory of being blind**. Short — carry on. Long — stop. That's the first time it has had a sense of duration."

Step 3 (2 min).

> **Script:** "Come out of the gap slowly. It doesn't know where it landed."

INDEPENDENT PRACTICE — SPRINT (20 MIN)

Required:

- [ ] Completes the course
- [ ] Survives the gap
- [ ] Stops inside the stop box
- [ ] **8 out of 10 attempts**
- [ ] Explains the four-state table without notes
- [ ] States trim values and why

Save as `P3_10_Runner`.

Circulate with the isolation ladder, not answers. When a team is stuck on the gap: "what did it last see?"

DEBUGGING & DISCUSSION — "RUN IT AGAIN" (5 MIN)

Five consecutive runs; class counts successes.

> **Script:** "Four out of five. Which one failed, and why? … If you don't know which one failed, you don't have a robot — you have a lucky robot."

> **Script:** "Next lesson is reliability. Ten runs. A robot that works eight times in ten is not eighty percent done — it's a robot with a fault you haven't found."

Exit ticket: "*What does your robot do in the gap, and why?*"

4. Differentiations

Supporting struggling students

- Course A: no gap, wider curves.
- `P3_10_Starter` provided.
- One success criterion: finishes without help.

Advanced extension challenges

- Recovery after the gap when landing off-axis.
- Two laps — does the failure rate rise?
- Speed scheduling on curves.
- Operator card in under 40 words.

Term 3 at a glance

WK	LESSON	BUILDS ON	BUILD
1	Meet KanKaung	T1 W4	Motor moves, trim checked
2	Trim, Properly	T2 W5	Drift explained
3	What the Sensors See	T2 W1–2	Truth table verified physically
4	Four States	**T2 W3**	**Their own table, running**
5	Variables and Loop Rate	T2 W4–6	Smooth follower

WK	LESSON	BUILDS ON	BUILD
6	Tuning	T2 W10	Fast reliable follower
7	The Second Sense	T2 W2	Obstacle stop with hysteresis
8	Priority and Fail-Safe	T2 W6	Guardian behaviour
9	Debugging a Physical System	T1 W9	Fault log
10	Build Sprint	everything	**Course completed 8/10**

Ten things every student must be able to do by Week 12

1. Read a pin map and name which pins the motors use.
2. Explain why a robot that drove straight yesterday curves today.
3. Put the robot in all four sensor states by hand and say what each means.
4. **Implement the four-state table they wrote in Term 2.**
5. Explain what `run without screen refresh` changes and when it hurts.
6. Choose a speed based on evidence, not preference.
7. Set two thresholds and explain why one chatters.
8. Name the priority ladder and defend the order.
9. Categorise a fault before touching the robot.
10. Complete the course eight times out of ten.

After Term 3

- [ ] All projects archived; one video per team of a successful course run.
- [ ] Four-state tables returned to students — they are now a personal record of a design that worked.
- [ ] Fault cards kept for next year.
- [ ] Robots serviced: servos checked, cells replaced, wheels true.
- [ ] Club rosters drawn from students who finished early.
- [ ] **Bridge to Text** recommendation written for each student who should continue.

Term 3 pack v2.0 · Padma Robotics Programme · rewritten to assume Term 2