PRESENTATION 10 · LEARNING WITH PYTHON

Avoid the traps

See the warning band, back away, and find the way back to FINISH.

TheBotCup 2026 · AiBotCup Simulator



RECAP · COLOUR SENSORS

Every colour on the floor means something

Metals

collect them (P6–P8)

Green

deposit there (P9)

Pink

FINISH or a target

Red

a trap: stay away! (today)

Today the colour sensors are used for **safety**, not for collecting.

PART 1

The mission: Session 10

START to FINISH, traps in the way

SESSION 10 · AVOID THE TRAPS

Reach FINISH without touching a trap

What happens	Points
Pass trap 1 without touching	+50
Pass trap 2 without touching	+50
Touch a trap	−20
Reach FINISH	+100

Time limit **4:00**. Best score: **200**.

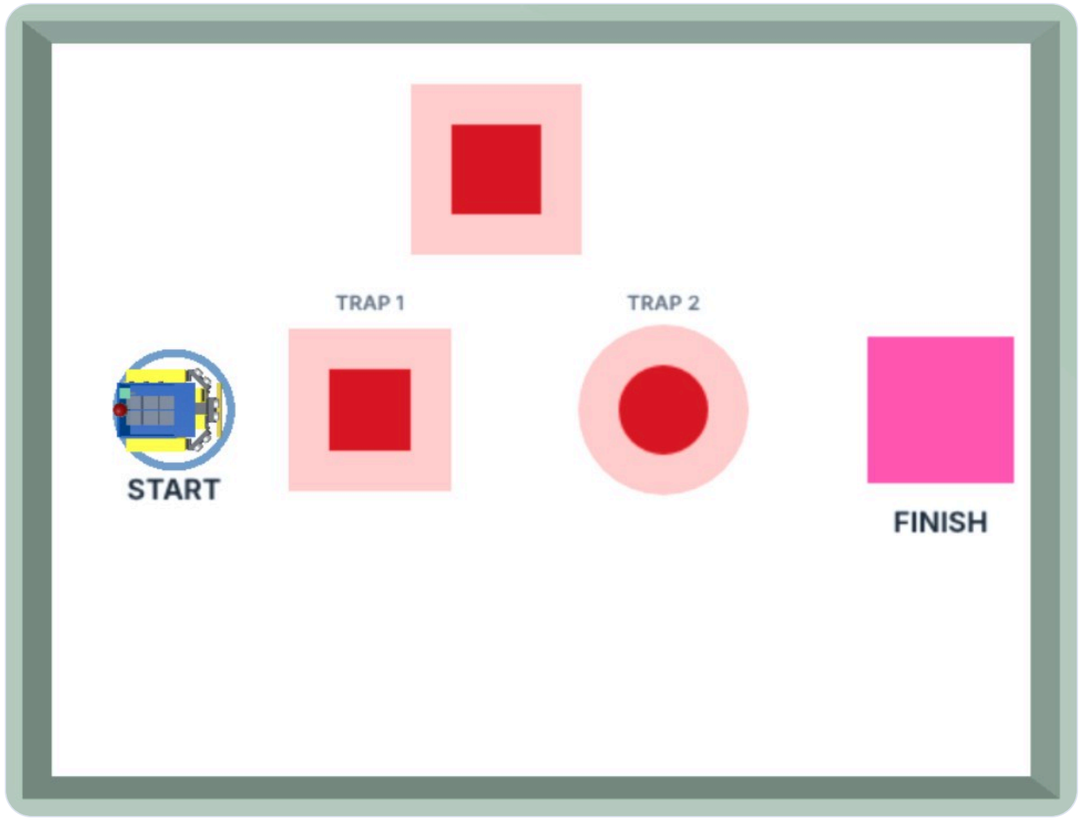
10

Session 10 · Avoid the traps

Drive from START to the pink FINISH pad. Two unsafe areas (traps) are on the straight route and a third is nearby: go around them.

Each of the two traps passed without touching a trap: +50. Touching a trap: −20. Reach FINISH: +100. Time limit 4:00.

Not done yet



THE FIELD

Straight ahead is blocked twice

START on the left, facing **FINISH** on the right.

Trap 1 (square) and **trap 2** (round) sit right on that line.

A **third trap** waits above: going round the top is risky.



WHAT IS A TRAP?

A red danger zone with a warning band

The **dark red** middle is the trap. A colour sensor on it: **−20**.

Around it is a **10 cm light-red band**. Driving on the band is safe: it is a warning.

React to the **band**, and the robot never reaches the trap.

COLOUR SENSORS · LIGHT RED

The band in numbers

	R	G	B
Floor	about 226	about 232	about 238
Warning band	250	175	175
Trap	185	20	30
Range we use	230 – 255	145 – 205	145 – 205

Red as high as the floor, but green and blue clearly lower. The range comes from the simulator’s own tip.

PART 2

The plan

Back off, then get back on course

THE PLAN

Three rules, most important first

1. A sensor sees the **warning band** → **back away** and turn away from it.
2. Off course → **curve back** towards FINISH (compass and position).
3. Anything else → **drive forward**.

RULE 2 · TWO SENSORS WE KNOW

Which way? How high?

Compass

270 = facing FINISH (east). Below 270 = turned right, towards the bottom. More in Presentation 11.

PositionY

START and FINISH are at about **Y = 90**. Below 80 = too low; above 100 = too high.



PART 3

Step by step in Python

Six branches, at most 20 lines

STEP 1 · THE PLAN

Safety, then course, then forward

Branch	Condition	Wheels	Duration
1 WarnLeft	isWarn(left): R ≥ 230, G and B 145–205	−1 / −4	500 ms
2 WarnRight	isWarn(right)	−4 / −1	500 ms
3 Low	PositionY <= 79 and Compass 240–299	3 / 4.5	60 ms
4 High	PositionY >= 101 and Compass 241–300	4.5 / 3	60 ms
5 TooRight	Compass 90–262	3.5 / 4.5	60 ms
6 Forward	else	4.5 / 4.5	60 ms

STEP 2 · LINES 3–12

Back away, nose turning away from the band

```
3  def isWarn(R, G, B):
4      return R >= 230 and 145 <= G <= 205 and 145 <= B <= 205
5
6  def game1():
7      if isWarn(CSLeft_R, CSLeft_G, CSLeft_B):          # WarnLeft
8          WheelLeft = -1; WheelRight = -4
9          Duration = 500
10     elif isWarn(CSRight_R, CSRight_G, CSRight_B):     # WarnRight
11         WheelLeft = -4; WheelRight = -1
12         Duration = 500
```

isWarn: R 230–255, G and B 145–205.

WarnLeft: −1 / −4, nose swings right.

WarnRight: −4 / −1, nose swings left.

STEP 3 · LINES 13–16

Too low or too high: aim up or down a bit

```
13      elif PositionY <= 79 and 240 <= Compass <= 299:   # Low
14          WheelLeft = 3; WheelRight = 4.5
15      elif PositionY >= 101 and 241 <= Compass <= 300:  # High
16          WheelLeft = 4.5; WheelRight = 3
```

Low: Y below 80 and facing roughly east → curve left.

High: Y above 100 → curve right.

They stop at **299** / **241**, so the robot heads diagonally, not back into the traps.

STEP 4 · LINES 17–20

Turned too far right? Curve back.

```
17      elif 90 <= Compass <= 262:                    # TooRight
18          WheelLeft = 3.5; WheelRight = 4.5
19      else:                                         # Forward
20          WheelLeft = 4.5; WheelRight = 4.5
```

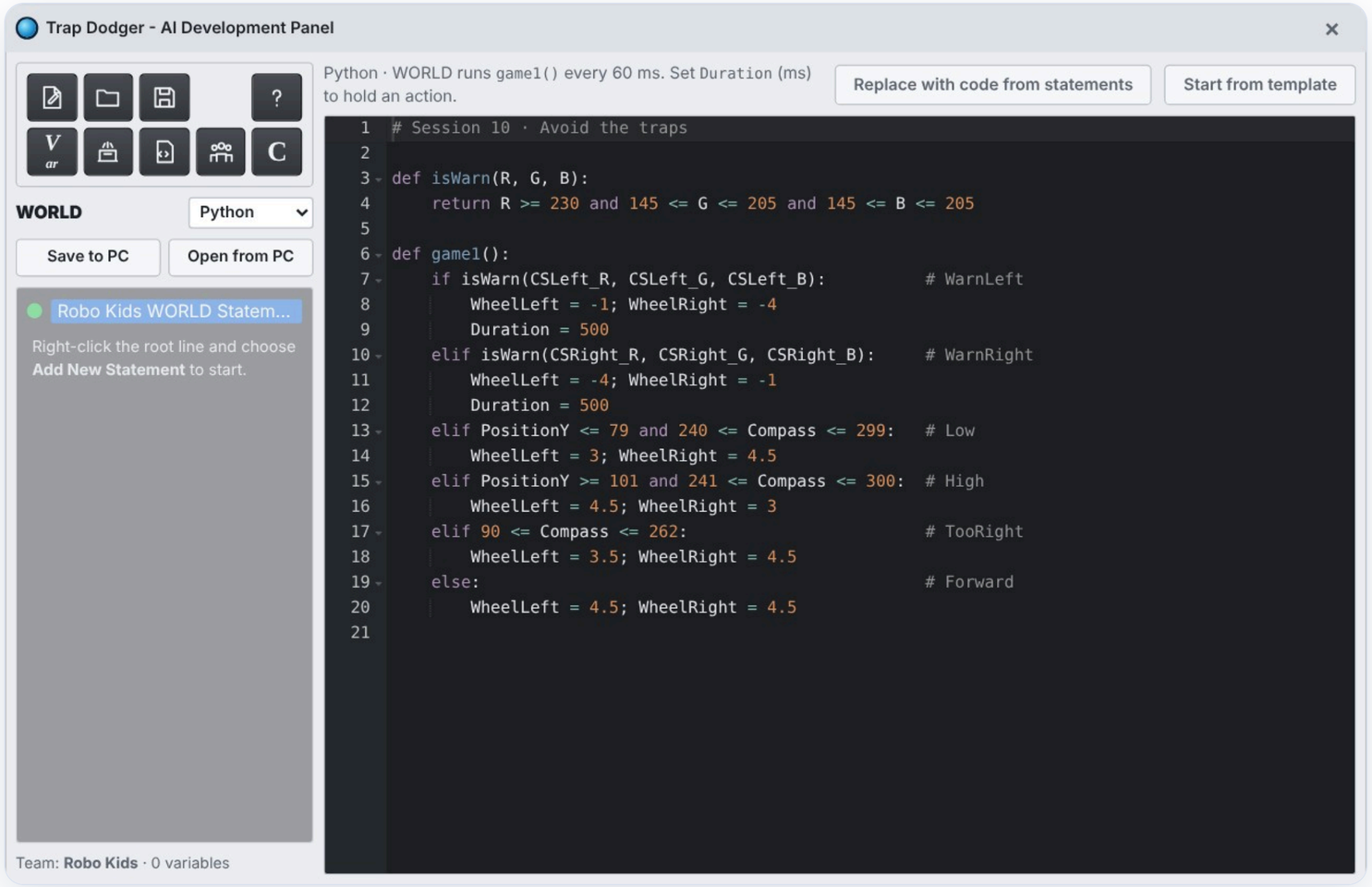
TooRight: Compass 90–262 → curve left, 3.5 / 4.5.

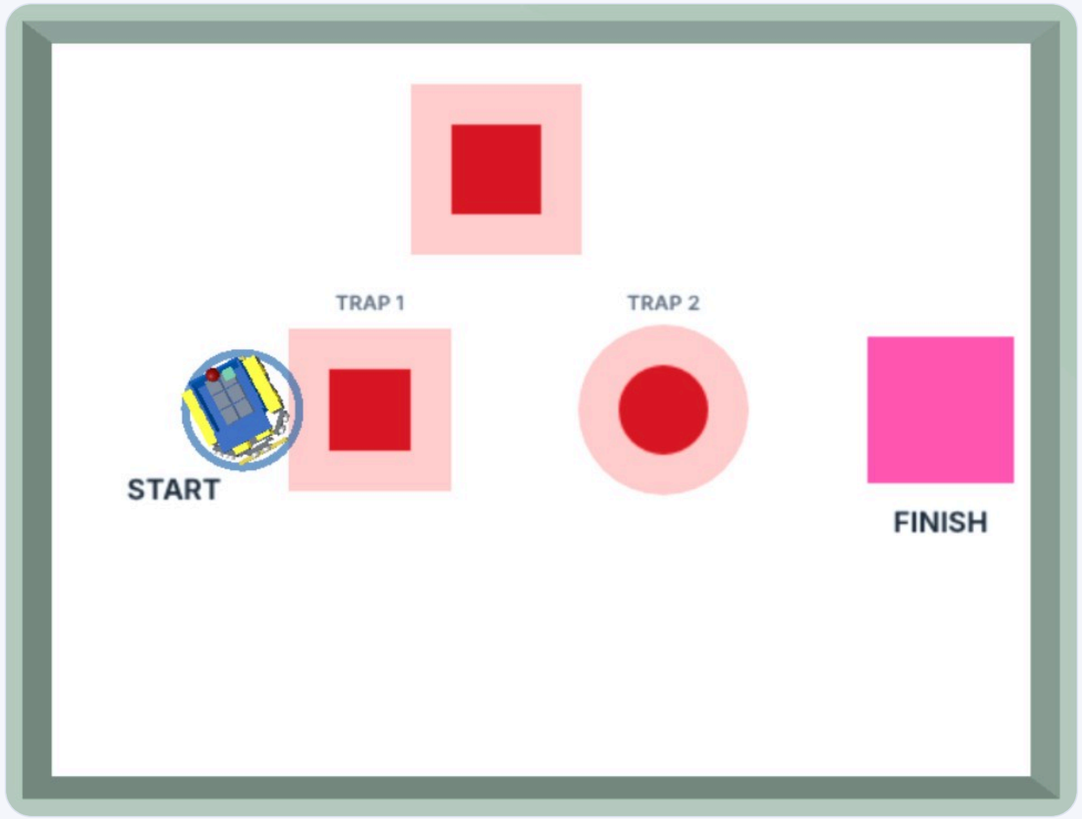
Forward: the else, always last.

STEP 5 · BUILD AND LOAD

The whole program, built

1. WORLD menu: **Python**.
2. Type the program, name it **Trap Dodger**.
3. Press **Build**, then **OK**.
4. Press **Load** on the BLUE side.





STEP 6 · RUN · THE BAND

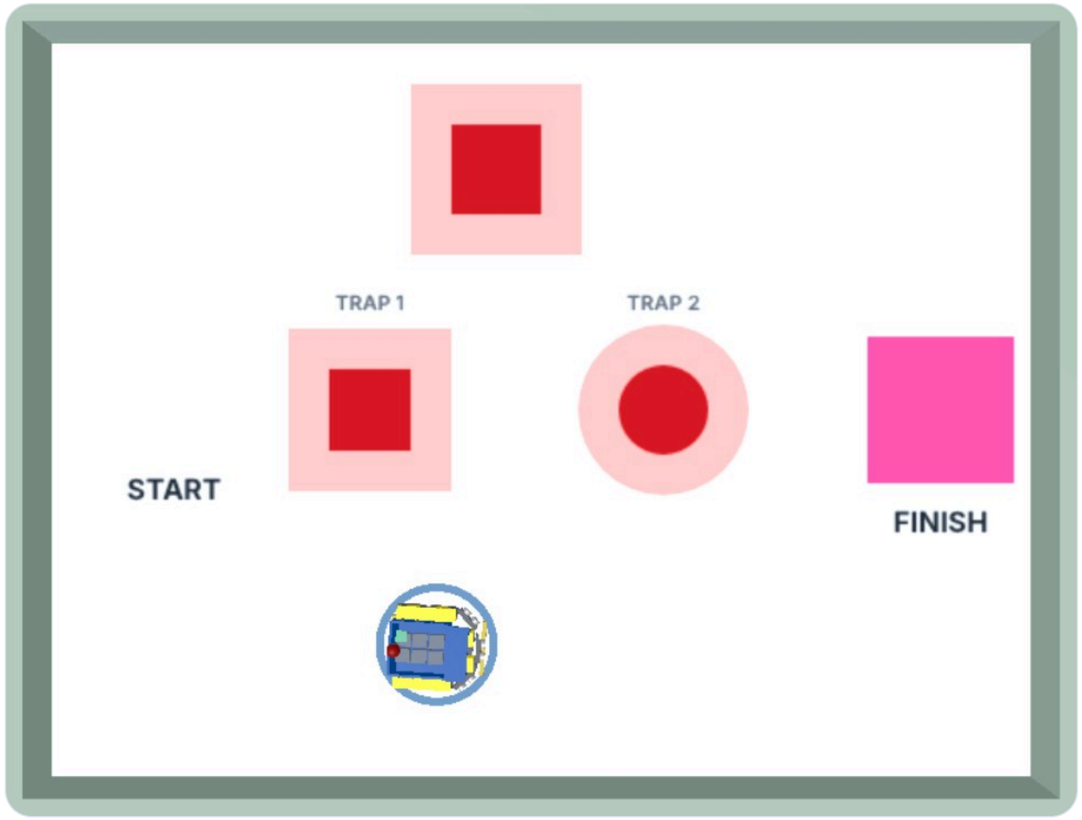
Band ahead: WarnLeft

After about 1 s the sensors reach the band of trap 1.

WarnLeft backs away, and again if needed.

At 2.6 s the compass reads **213**: turned towards the bottom.

The trap itself is **never touched**.



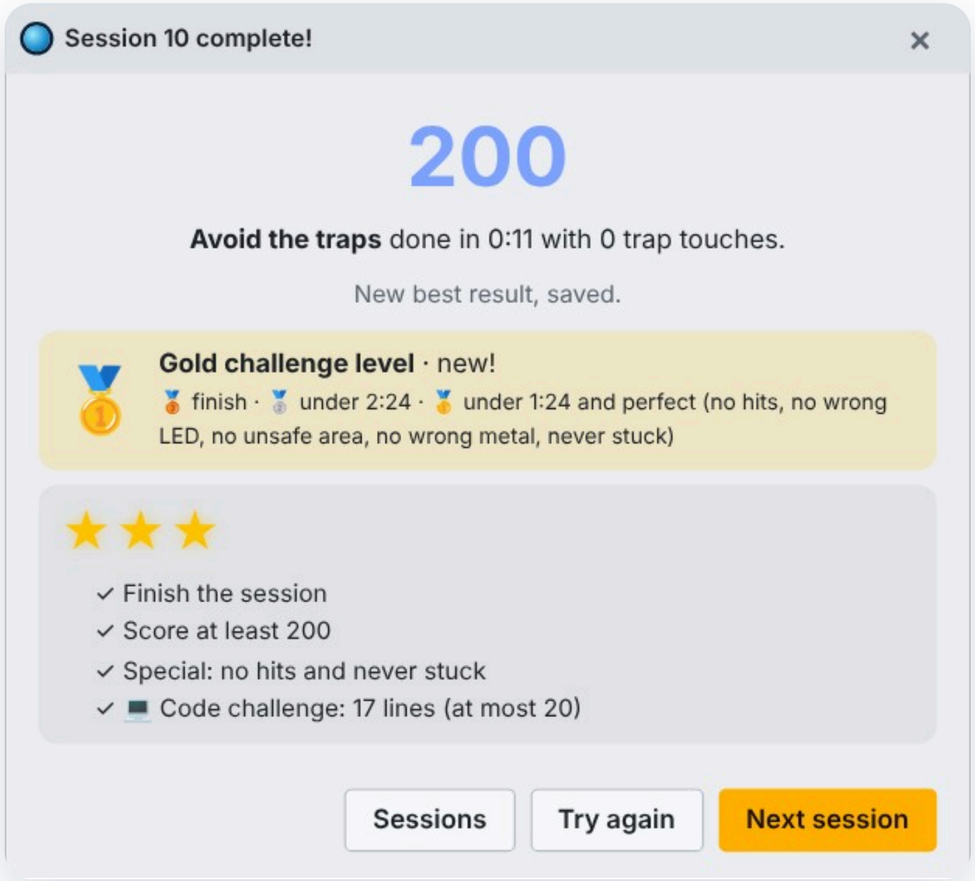
STEP 7 · RUN · BACK ON COURSE

TooRight, then Low, then Forward

TooRight first curves the robot back towards east.

At 7.4 s: PositionY **35**, Compass **293**: **Low** curves it up.

Then Forward runs diagonally up to FINISH, below trap 2.



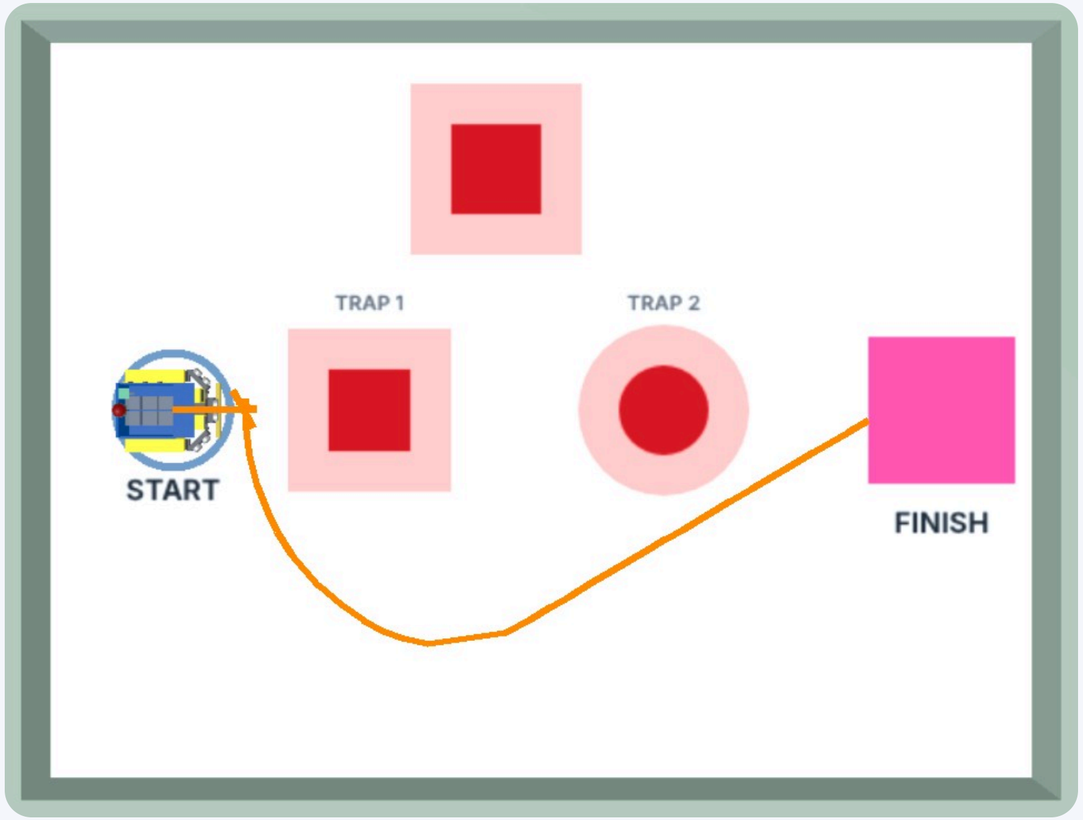
RESULT · TESTED

200 points in 0:11 · Gold

Score · trap 1 +50 · trap 2 +50 · FINISH +100. 0 trap touches.

Medals · Silver: under 2:24 · Gold: under 1:24 and perfect.

Code challenge · 17 lines, at most 20.



THE PATH · RECORDED

Round the bottom, then a diagonal

One bounce off the band of trap 1. Then the course rules did the rest.

React to the **warning**, not the danger.

DID NOT WORK?

Check first, then tune carefully

What you see	What to do
Build error with a line number	Colon : after def / if / elif / else, and indentation
Drives into the trap (−20)	isWarn ranges: R ≥ 230, G and B 145–205
Spins on the band for a long time	Back away (both wheels negative), don’t spin on the spot
Ends in a corner	Low and TooRight missing, or Compass ranges wrong
Misses FINISH by a little	Check Low: PositionY <= 79 and Compass 240–299
Old program still on the robot	Build again, then Load on BLUE

NEXT TIME

Presentation 11 · Compass targets

What a compass is, how it works, and how to turn to **any heading** in Python.