PRESENTATION 11 · LEARNING WITH PYTHON

Compass targets

No position sensor this time: the robot finds three targets by turning to the right heading and driving straight.

TheBotCup 2026 · AiBotCup Simulator



RECAP

Each sensor answers one question

Time
How long since Play?

Ultrasonic
How far is the wall?

Colour
What is on the floor?

Compass
Which way am I facing?

In Presentation 10 we used the compass a little to steer back towards FINISH. Today it is the main sensor.

PART 1

What is a compass?

A sensor for direction

HOW IT WORKS

The Earth is a giant magnet

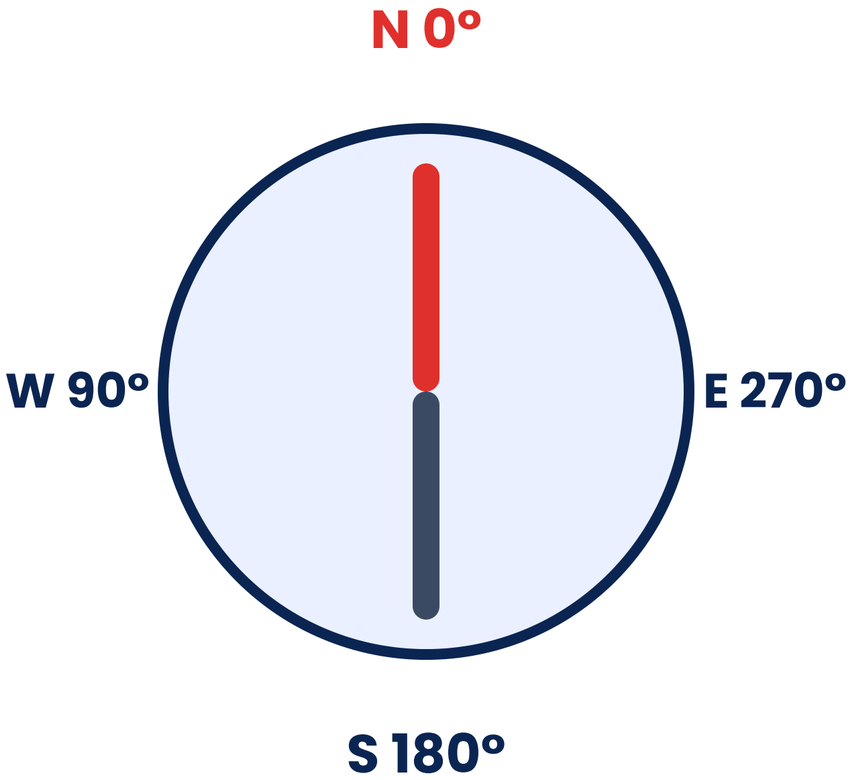
Needle compass

A small magnetic needle turns until it lines up with the Earth’s magnetic field, so one end always points **north**.

Electronic compass

A robot uses a tiny chip (a **magnetometer**) that measures the same field and turns it into a **number in degrees**.

That number is the robot’s **heading**: the direction it is facing, from 0 to 359.



DEGREES IN THE SIMULATOR

0 is north, and turning left adds

Robot faces	Compass
North (up)	**0**
West (left)	**90**
South (down)	**180**
East (right)	**270**

Careful: on a map compass east is 90. In the simulator the numbers go the other way round: **east is 270**.

PART 2

The mission: Session 11

Three targets by heading

SESSION 11 · COMPASS TARGETS

Visit 1, 2 and 3 in order

Target	Heading	Points
1	0° (north)	+50
2	270° (east)	+50
3	135° (south-west)	+100

PositionX and PositionY are off (they read 0). Only the next target in order counts. Time limit **4:00**.

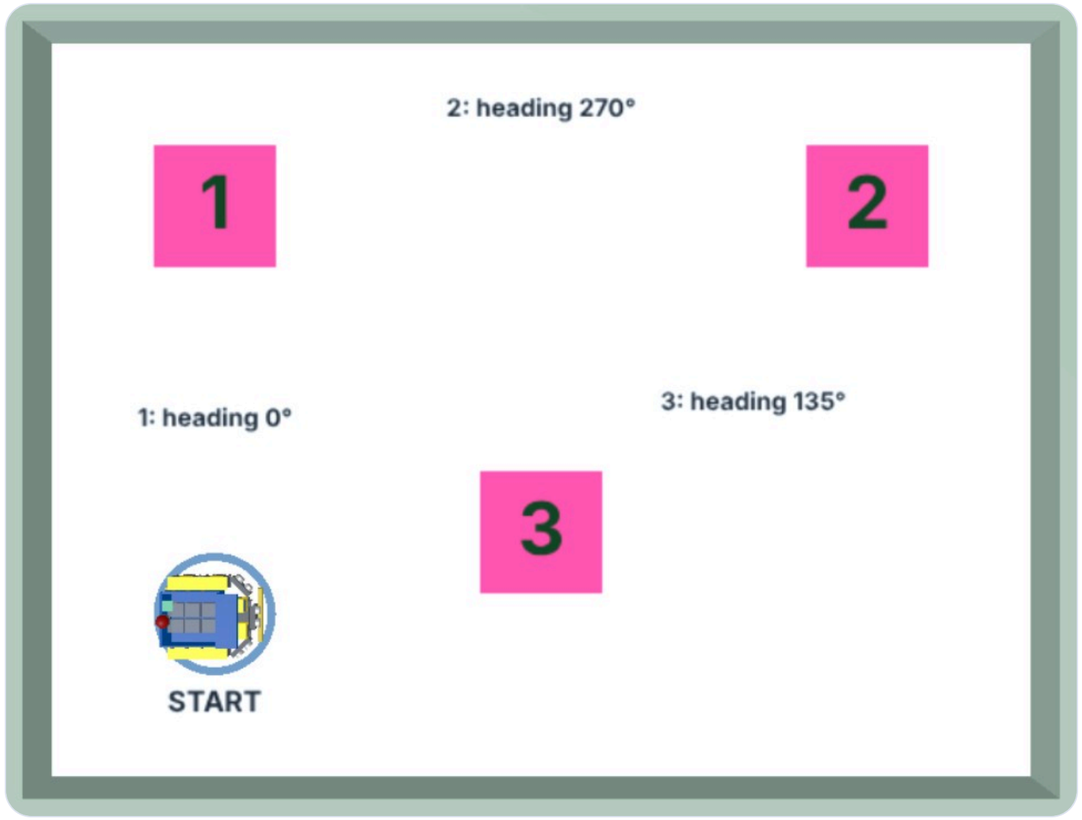
11

Session 11 · Compass targets

Use the compass to visit target 1, 2 and 3 in order. Position (PositionX, PositionY) is switched off: turn to each heading and drive straight.

Target 1 (heading 0°): +50. Target 2 (heading 270°): +50. Target 3 (heading 135°): +100. Only the next target in order counts. Time limit 4:00.

Not done yet



THE FIELD

Three pink pads, one route

START bottom left, facing **east** (Compass 270).

Target 1 straight **north** of START.

Target 2 straight **east** of target 1.

Target 3 **south-west** of target 2.

PART 3

The compass in Python

Read it, test it, turn with it

SENSORS PANEL

Read the compass before you code

At START: **Compass 270**, the robot faces east.

PositionX 0, PositionY 0: switched off in this session.

Time still works: we use it to know how far to drive.



WAY 1 · COMPARE THE COMPASS

A range means “facing about this way”

Easy: east

267 <= Compass <= 273 means “facing east, give or take 3°”.

Tricky: north

North is 357…359 **and** 0…3. One “between” test cannot hold both ends of the circle.

We also want to know **which way to turn**. That needs a helper.

WAY 2 · A BUILT-IN FUNCTION

angleDiff(heading, Compass)

It answers: **how many degrees do I still need to turn?** From −180 to 180.

Wanted heading	Compass now	angleDiff	So…
0 (north)	270	**+90**	turn **left** 90°
270 (east)	2	**−92**	turn **right** 92°
135 (south-west)	270	**−135**	turn **right** 135°
270	269	**+1**	close enough: drive

Positive → turn left · negative → turn right · near 0 → you are facing it.

THE PATTERN

Two branches per leg: Turn, then Go

Turn Same Time window and **angleDiff(H, Compass) > 3** → spin left (or **< −3** → spin right).

Go Same Time window only, no compass test → drive straight, 4.5 / 4.5.

Turn comes before Go, so it wins **until the robot faces H** (within 3°). Then Turn stops matching and Go takes over.

PART 4

Step by step in Python

Seven branches, three targets

STEP 1 · THE PLAN

Three legs, each with its own seconds

Leg	Time	Turn to	Then drive to
1	0 – 4	0° · turn left	Target 1
2	5 – 11	270° · turn right	Target 2
3	12 – 20	135° · turn right	Target 3
End	after 20	—	Stop (the else)

Time 0 – 4 means the whole of seconds 0, 1, 2, 3 and 4: five seconds.

STEP 2 · LEG 1 · LINES 3–7

Face north, then drive north

```
3  def game1():
4      if Time <= 4 and angleDiff(0, Compass) > 3:              # FaceNorth
5          WheelLeft = -2; WheelRight = 2
6      elif Time <= 4:                                          # North
7          WheelLeft = 4.5; WheelRight = 4.5
```

FaceNorth: Time ≤ 4 and angleDiff(0, Compass) > 3 → spin left, −2 / 2.

North: Time ≤ 4 only → drive 4.5 / 4.5. angleDiff(0, 270) = +90, so we spin left first.

STEP 3 · LEG 2 · LINES 8–11

Turn east, then drive east

```
8       elif 5 <= Time <= 11 and angleDiff(270, Compass) < -3:   # TurnEast
9           WheelLeft = 2; WheelRight = -2
10       elif 5 <= Time <= 11:                                    # GoEast
11          WheelLeft = 4.5; WheelRight = 4.5
```

TurnEast: Time 5–11 and angleDiff(270, Compass) < −3 → spin right, 2 / −2.

GoEast: Time 5–11 only → drive. On target 1 the compass reads 2: angleDiff(270, 2) = −92.

STEP 4 · LEG 3 · LINES 12–17

Turn south-west, drive, then stop

```
12      elif 12 <= Time <= 20 and angleDiff(135, Compass) < -3:  # TurnSW
13          WheelLeft = 2; WheelRight = -2
14      elif 12 <= Time <= 20:                                   # GoSW
15          WheelLeft = 4.5; WheelRight = 4.5
16      else:                                                    # Stop
17          WheelLeft = 0; WheelRight = 0
```

TurnSW / GoSW: Time 12–20, heading 135. From 270 that is a 135° turn to the right.

Stop: the else at the end. When every Time window is over, wheels 0 / 0.

WHY THESE SECONDS?

Distance = speed × time

27

cm per second at 4.5 / 4.5

≈ 1 s

to turn 90° at 2 / −2

±15

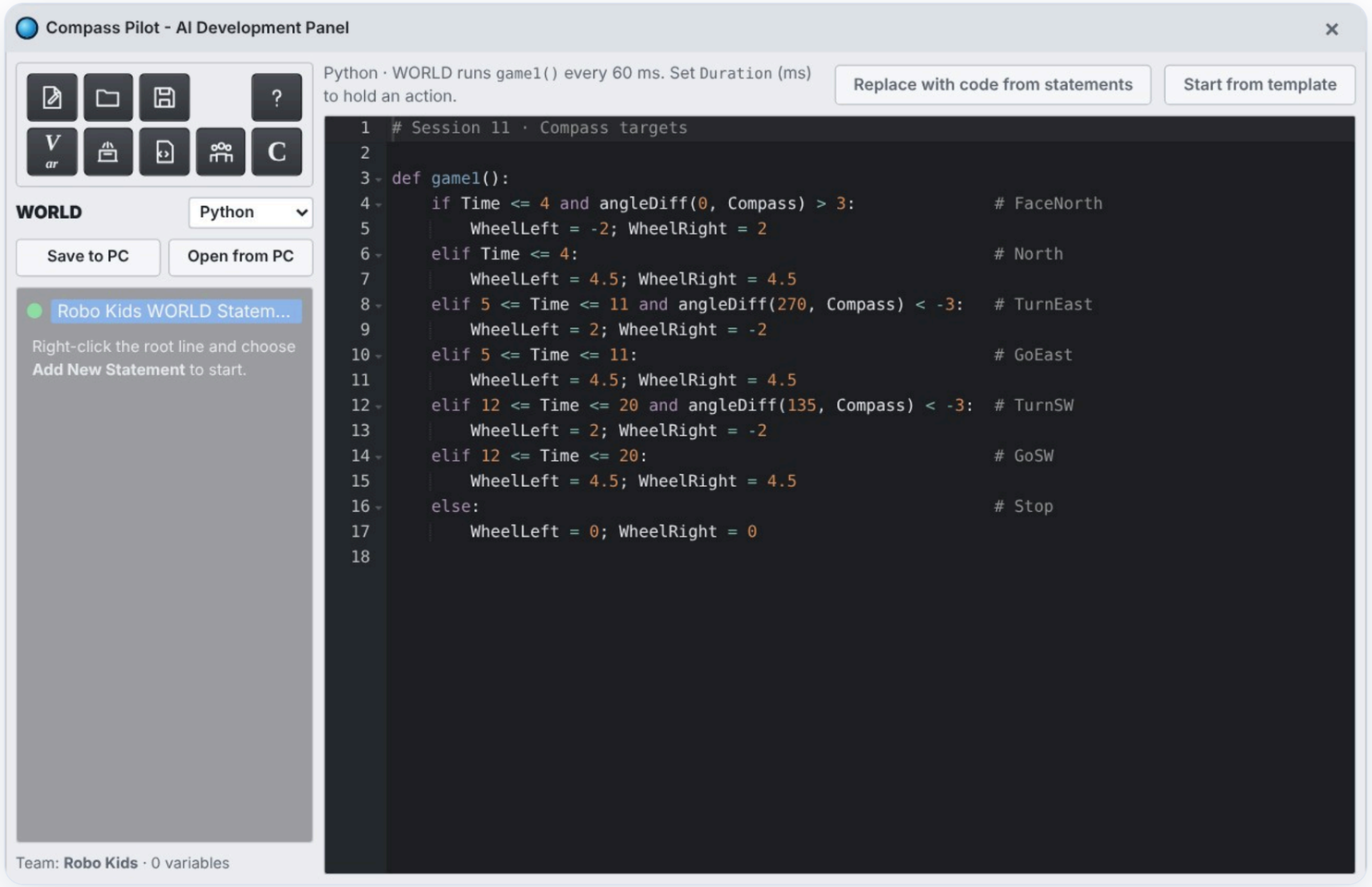
cm: room inside a 30 cm pad

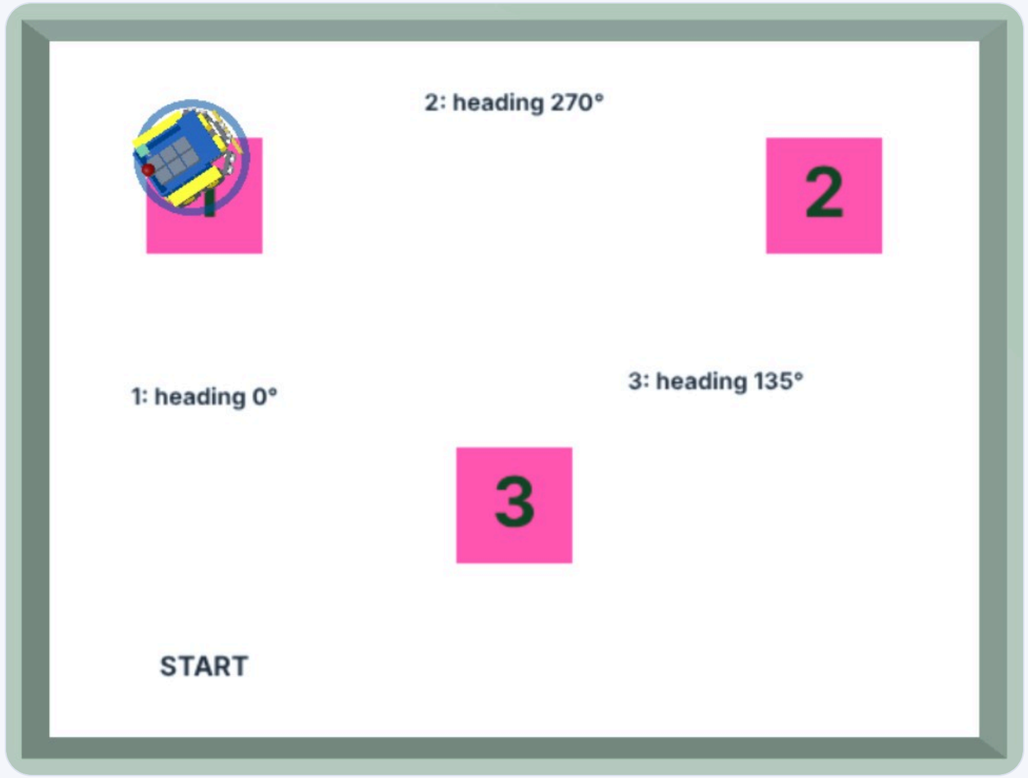
Leg 1: about 1 s to turn, then about 4 s × 27 cm = **108 cm** north, into the middle of target 1.

STEP 5 · BUILD AND LOAD

The whole program, built

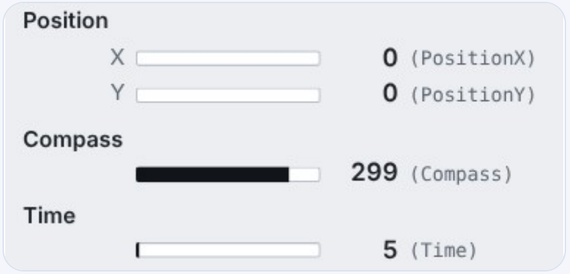
1. WORLD menu: **Python**.
2. Type the program, name it **Compass Pilot**.
3. Press **Build**, then **OK**.
4. Press **Load** on the BLUE side.



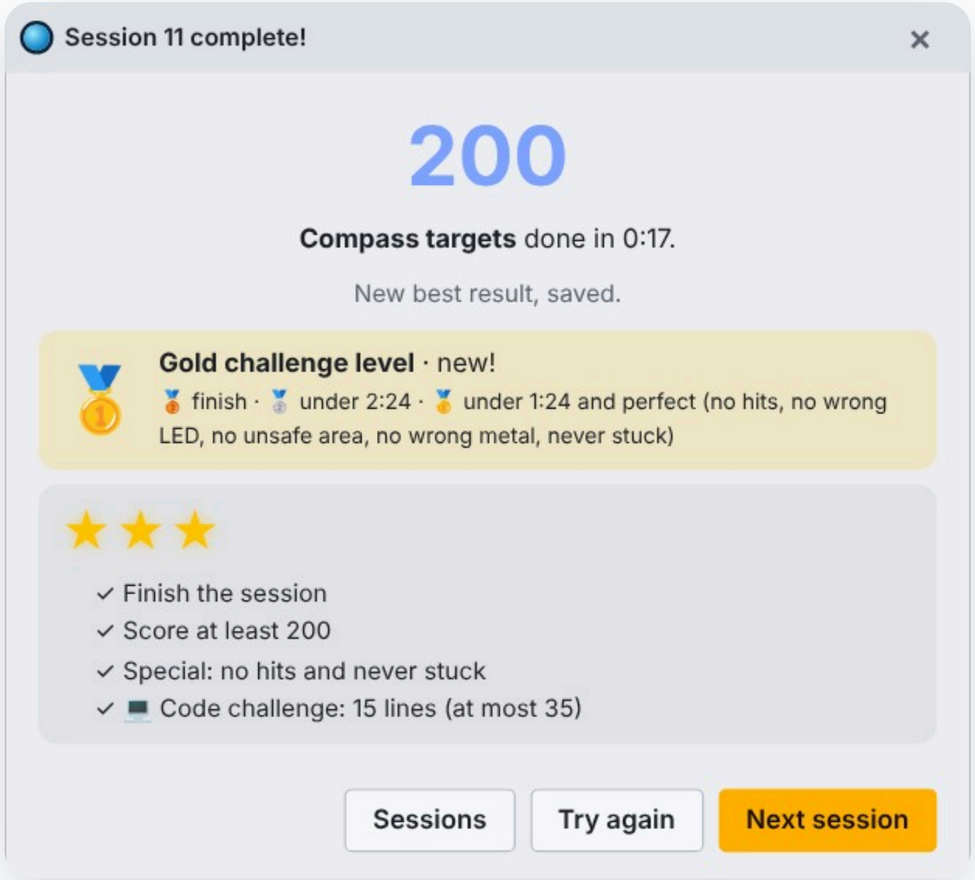


STEP 6 · RUN

Target 1 done, now turning east



Target 1 scored at 4.2 s. At Time 5 **TurnEast** spins right: Compass **296** and falling to 270.



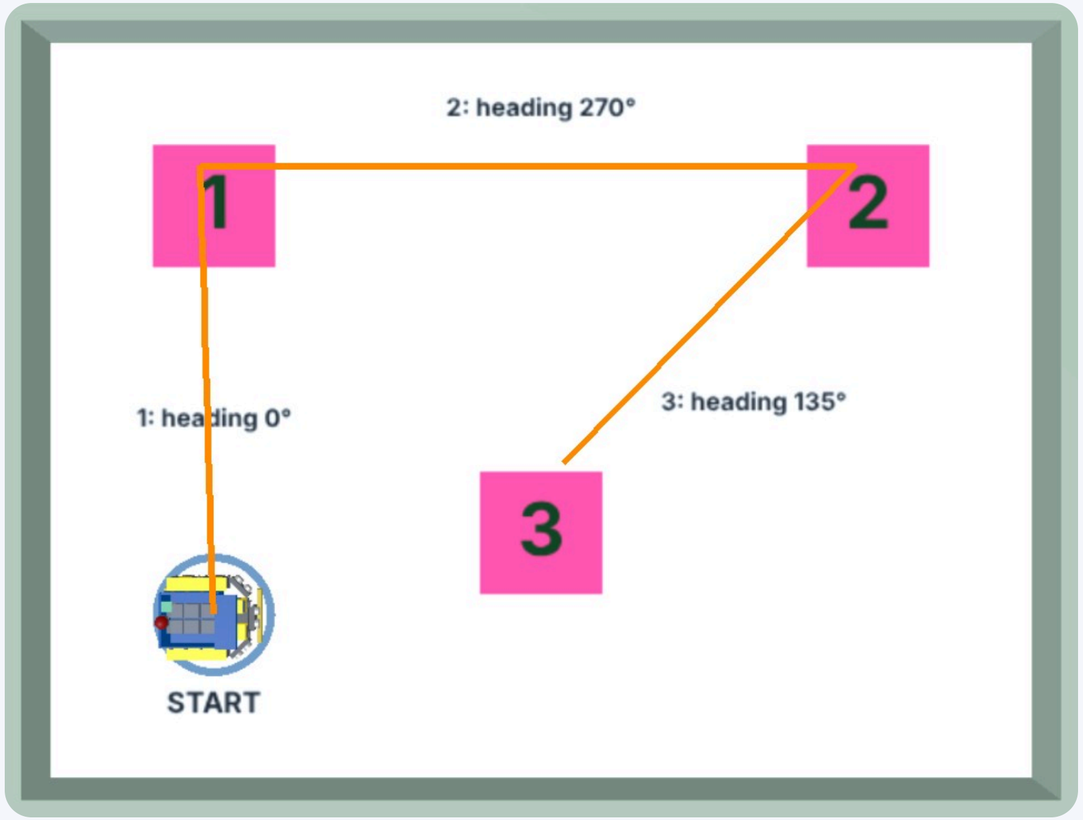
RESULT · TESTED

200 points in 0:17 · Gold

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

Stars · finish · score 200 · no hits and never stuck.

Code challenge · at most 35 lines. We used 15.



THE PATH · RECORDED

Three straight lines

North, east, south-west: each line starts with a turn that stops at exactly the right heading.

No position sensor needed: **heading + time** is enough.

DID NOT WORK?

Check first, then tune carefully

What you see	What to do
Turns the long way or ends facing wrong	Sign and wheels must match: **> 3** with −2 / 2, **< −3** with 2 / −2
Goes the wrong way	Check the heading: east is **270**, west is **90** in the simulator
Stops just before a pad	Make that leg’s Time window 1 second longer
Never turns, only drives	Turn branch must come **before** its Go branch
Old program still on the robot	**Build** again, then **Load** on BLUE

WHAT WE LEARNED

Turn to a heading, then go

Compass

0 north · 90 west · 180 south · 270 east

angleDiff

+ turn left · – turn right · ≈ 0 facing it

Turn + Go

Two branches with the same Time window