ANSWERS

6.5 Following a robot

Seeing more · Robot club · about 20 min

What this lesson is about

Two controllers that never finish: chase a moving tag at a set distance.

- Watch it go
- Two controllers, forever
- Corners
- Losing it

Questions

6 marks in all

1. What does this program print? [1 mark]

```
for dist in [50, 30, 20, 12]:
    speed = max(0, min(70, (dist - 20) * 4))
    print(dist, speed)
```

Answer:

```
50 70
30 40
20 0
12 0
```

50 gives 120, clamped to 70. 30 gives 40. At 20 the error is 0. At 12 it would be -32, and the clamp holds it at 0.

2. Why is the lowest speed in the follower 0, the `max(0, ...)`? [1 mark]

- ☐ **A** So the robot stops when too close instead of reversing
- ☐ **B** So it never goes faster than the leader
- ☐ **C** So it keeps moving when the leader stops
- ☐ **D** Because negative speeds are an error

Answer: A. Closer than 20 cm, the controller would ask for a negative speed. The clamp holds it still instead.

3. With `speed = max(0, min(70, (dist - 20) * 4))`, what speed does the follower use when the leader is 26 cm away? [1 mark]

Answer: 24. (26 - 20) * 4 is 24, which is inside the clamp.

4. How is following a robot different from docking at a marker? [1 mark]

- ☐ **A** The target keeps moving, so the loop never finishes
- ☐ **B** It uses a different detector
- ☐ **C** It needs no rotation controller
- ☐ **D** The distance is measured in pixels

Answer: A. The same two controllers, but the distance and bearing are new every tick and there is no end point.

5. When the leader goes out of view, the lesson just turns right. What is a better recovery? [1 mark]

- ☐ **A** Remember the last bearing and turn that way
- ☐ **B** Stop and wait for the leader to come back
- ☐ **C** Drive forward at full speed
- ☐ **D** Switch to the line detector

> **Answer:** A. The leader is most likely where it was last seen. Better still, lesson 6.7 predicts where it is going.

6. The task starter always drives at 60 towards robot 101. What goes wrong? [1 mark]

- ☐ **A** It rams the leader, because nothing slows it down as it gets close
- ☐ **B** It never catches up
- ☐ **C** It turns away from the leader
- ☐ **D** It loses the leader immediately

> **Answer:** A. It only has the rotation controller. Add the distance controller so speed falls as the gap closes.

The task: follow the leader

Stay 12 to 35 cm behind the green robot for at least 80% of the run, without touching it. The starter drives at the leader at a fixed speed and rams it. Add the distance controller.

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

# do this 390 times (tick counts from 0)
for tick in range(390):
    seen = [r for r in robots() if r[0] == "green"]
    if seen:
        # forward, sideways, rotation: -100 to 100 each, until the next command
        drive(60, 0, (seen[0][1] - 160) * 120 / 320 * 3)
    # pause 0.1 s (the robot keeps doing what it was told)
    wait(0.1)
# all motors off
stop()
```

The hint students can ask for: The leader drives a loop at a steady speed with its dome lit green. Stay 12 to 35 cm behind it for at least 80% of the run without touching it: steer from its bearing, speed from its distance.

A solution

```
from bugbot import *
connect()
for tick in range(390):
    seen = [r for r in robots() if r[0] == 'green']
    if not seen:
        turn_right(40)                    # lost it: turn until the green dome is back
    else:
        cx, dist = seen[0][1], seen[0][3]
        bearing = (cx - 160) * 120 / 320
        speed = max(0, min(70, (dist - 20) * 4))   # hold about 20 cm behind
        drive(speed, 0, bearing * 3)
    wait(0.1)
stop()
```

Any program that meets the task's checks is marked correct in the simulator; this is one way, not the only way.