ANSWERS

1.4 Sideways

Driving · Robot club · about 15 min

What this lesson is about

Strafing: moving without turning, the thing wheels cannot do.

- left and right
- Why it matters
- Diagonals
- A gentle curve
- Sideways by distance, with a sensor

Questions

7 marks in all

1. What is strafing? [1 mark]

- ☐ **A** Sliding sideways or diagonally while facing the same way
- ☐ **B** Turning on the spot
- ☐ **C** Driving backwards
- ☐ **D** Driving in a curve

> **Answer:** A. BugBot's feet can push it sideways without turning, so it keeps the heading it had.

2. What does this program make the robot do? [1 mark]

```
forward(60, distance=20)
right(60, distance=20)
backward(60, distance=20)
left(60, distance=20)
```

- ☐ **A** Drive a square, facing the same way the whole time
- ☐ **B** Drive a square, turning at each corner
- ☐ **C** Drive forward 40 cm
- ☐ **D** Stay exactly where it is

> **Answer:** A. Forward, right, back, left is a square, and none of those commands turn the robot, so its nose never changes direction.

3. Why is strafing useful? Choose all that apply. [1 mark]

Tick every answer that is true.

- ☐ **A** It keeps the heading, so no turning error is added
- ☐ **B** The camera and distance sensor can stay pointed at something while the robot moves
- ☐ **C** It makes the robot drive perfectly straight
- ☐ **D** It uses no power

> **Answer:** A, B. Every turn is a little off, and strafing avoids turning. The sensors on the front keep looking where you want. It still drifts like any drive.

4. What does `drive(60, -60, 0)` do? [1 mark]

- ☐ **A** Drives diagonally forward and to the left
- ☐ **B** Drives diagonally forward and to the right
- ☐ **C** Drives forward and turns left
- ☐ **D** Stays still, because 60 and -60 cancel out

> **Answer:** A. The three numbers are forward, sideways and rotation. Negative sideways is left, so forward plus left is a diagonal.

5. Why does `drive` need a `wait` and a `stop` around it? [1 mark]

- ☐ **A** It returns at once and takes no distance
- ☐ **B** It only works inside a loop
- ☐ **C** It drives for exactly one second
- ☐ **D** wait and stop set its speed

> **Answer:** A. `drive` is the raw command the others are built from. It starts the motors and returns, like `forward(speed)` with no distance.

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

```
p = (12.5, 30.0)
print("moved right to x =", p[0])
```

> **Answer:**
>
> ```
> moved right to x = 12.5
> ```
>
> Like a list, `[0]` picks the first item, which for a position is `x`. `[1]` would be `y`.

7. What does `drive(60, 0, 20)` with a `wait(3)` make the robot do? [1 mark]

- ☐ **A** Drive forward in a gentle curve
- ☐ **B** Slide sideways
- ☐ **C** Spin on the spot without moving forward
- ☐ **D** Drive dead straight

> **Answer:** A. Forward plus a little rotation means it turns slowly as it goes, which draws a curve.

The task: reach the green square

There is a wall between you and the green square. Get the robot into the square within 20 seconds without touching the wall. The wall is 20 cm wide and the square 20 cm beyond it. Forward, sideways, forward, sideways, and no turning needed. Remember the margin for coasting.

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

# drive forward at 60 for 20 cm, then stop
forward(60, distance=20)
# your turn: go round the wall and into the square
```

The hint students can ask for: The wall is in the way. Drive round it, or use distance() to find it. The robot coasts a little after stop(), so leave a margin.

A solution

```
from bugbot import *
connect()
forward(60, distance=20)
right(60, distance=25)
forward(60, distance=45)
left(60, distance=25)
```

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