ANSWERS

F3.6 Records

Strings, lists and records · GCSE · OCR J277 2.2.3, AQA 8525 3.2.6, Edexcel 1CP2 6.3.1 · about 15 min

What this lesson is about

Grouping related fields: a sensor reading as a record, and a list of records as a table.

- A record as a dictionary
- A list of records
- Searching a table of records
- Records as their own type

Questions

5 marks in all

1. What makes a record different from an array? [1 mark]

- ☐ **A** A record's fields can hold different data types, reached by name
- ☐ **B** A record can only hold numbers
- ☐ **C** A record is always sorted
- ☐ **D** An array can hold different types, a record cannot

Answer: A. An array holds items of one type by index; a record groups related fields of any types by name.

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

```
reading = {"heading": 90, "distance": 51.0}
reading["heading"] = 180
print(reading["heading"], reading["distance"])
```

Answer:

```
180 51.0
```

A field is reached by its name, and can be changed like a variable.

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

```
log = [{"h": 0, "d": 31}, {"h": 90, "d": 51}, {"h": 180, "d": 15}]
best = log[0]
for r in log:
    if r["d"] > best["d"]:
        best = r
print(best["h"])
```

Answer:

```
90
```

Keeping the whole record means the heading of the largest distance comes with it.

4. A robot log stores a time, a heading, a distance and a note for each reading. Which data types suit the fields? [1 mark]

Tick every answer that is true.

- ☐ **A** time: real
- ☐ **B** heading: integer
- ☐ **C** distance: real
- ☐ **D** note: string

> **Answer:** A, B, C, D. All four are sensible: that mix of types in one record is exactly what records are for.

5. In a table of records, what is each row? [1 mark]

- ☐ **A** One record
- ☐ **B** One field
- ☐ **C** One data type
- ☐ **D** One array index

> **Answer:** A. Each record is a row; each field is a column.

The task: reading records

The robot is boxed in on three sides. Look in four directions, 90 degrees apart, storing each look as a record with the fields `heading` and `distance` in a list. Then print one line per record as `heading 0: 31.0 cm`, and finally print `closest heading: <heading>` for the record with the smallest distance.

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

log = []
for i in range(4):
    turn_right(30, angle=90)
```

The hint students can ask for: Store each reading as a record with the direction it was taken in, and add it to a list before turning. Afterwards, loop over the list to print, and keep the record with the smallest distance.

A solution

```
from bugbot import *
connect()
log = []
for i in range(4):
    log.append({"heading": i * 90, "distance": distance()})
    turn_right(30, angle=90)
for r in log:
    print(f"heading {r['heading']}: {r['distance']} cm")
closest = log[0]
for r in log:
    if r["distance"] < closest["distance"]:
        closest = r
print("closest heading:", closest["heading"])
```

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