ANSWERS

F13.1 How your exam works

Exam preparation · GCSE · about 15 min

What this lesson is about

The papers for each board, command words, and planning your time by the marks.

- Two papers, whichever board you sit
- Command words
- Marks and minutes

Questions

5 marks in all

1. A question starts with "Explain". What must your answer include? [1 mark]

- ☐ **A** A reason: why or how
- ☐ **B** Only the fact
- ☐ **C** A drawing
- ☐ **D** A conclusion and both sides

> **Answer:** A. Describe says what happens; explain says why or how.

2. A question is worth 4 marks. What does that usually mean? [1 mark]

- ☐ **A** Four separate points are wanted
- ☐ **B** Write for four minutes exactly
- ☐ **C** Four sentences on one point
- ☐ **D** It is the hardest question

> **Answer:** A. One mark, one point. Repeating a point four times scores once.

3. Which command word asks for both sides and a judgement? [1 mark]

- ☐ **A** Discuss
- ☐ **B** State
- ☐ **C** Identify
- ☐ **D** Describe

> **Answer:** A. Discuss and evaluate both want an argument and a conclusion.

4. A paper is 80 marks in 120 minutes. How many minutes a mark? [1 mark]

> **Answer:** 1.5. 120 ÷ 80.

5. How much coursework counts towards your GCSE computer science grade? [1 mark]

- ☐ **A** None: it is all exams
- ☐ **B** Half
- ☐ **C** A quarter
- ☐ **D** It depends on the school

> **Answer:** A. The programming work you do is preparation for the exams, not a mark.

The task: your timing plan

Take each paper in turn. Print `<name>: <n> minutes a mark` to one decimal place, then, for each question size in `sizes`, print `<name> <marks>-mark question: <n> min` rounded to the nearest whole minute. When both papers are done, print `total marks: <n>` and `total minutes: <n>`.

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

# name, marks, minutes
papers = [("Paper 1", 80, 90), ("Paper 2", 75, 120)]
sizes = [1, 3, 6]
```

The hint students can ask for: Minutes a mark is the paper's minutes divided by its marks. A question's time is that rate times its marks, rounded. Keep running totals of the marks and the minutes as you go.

A solution

```
from bugbot import *
connect()
papers = [("Paper 1", 80, 90), ("Paper 2", 75, 120)]
sizes = [1, 3, 6]
total_marks = 0
total_minutes = 0
for name, marks, minutes in papers:
    rate = minutes / marks
    print(f"{name}: {rate:.1f} minutes a mark")
    for size in sizes:
        print(f"{name} {size}-mark question: {rate * size:.0f} min")
    total_marks = total_marks + marks
    total_minutes = total_minutes + minutes
print("total marks:", total_marks)
print("total minutes:", total_minutes)
```

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