ANSWERS

F13.6 Long answer questions

Exam preparation · GCSE · OCR J277 1.6.1, AQA 8525 3.8, Edexcel 1CP2 5.1.1 · about 15 min

What this lesson is about

Point, because, so: a structure for discuss and evaluate questions, and a conclusion that decides.

- What the examiner is looking for
- The structure: point, because, so
- Plan before you write
- Words that earn marks

Questions

5 marks in all

1. How are long answers marked? [1 mark]

- ☐ **A** In bands, for the quality of the argument
- ☐ **B** One mark per fact
- ☐ **C** By how much you write
- ☐ **D** By handwriting

> **Answer:** A. Developed points and a conclusion reach the top band.

2. What does a top-band answer always finish with? [1 mark]

- ☐ **A** A conclusion that decides, with a reason
- ☐ **B** A summary of the question
- ☐ **C** A list of key words
- ☐ **D** An apology for running out of time

> **Answer:** A. Discuss and evaluate both want a judgement.

3. The question names a village school with slow broadband. What should your points do? [1 mark]

- ☐ **A** Refer to that school and its broadband
- ☐ **B** Talk about schools in general
- ☐ **C** List every fact you know
- ☐ **D** Only give advantages

> **Answer:** A. Relevance to the situation is what separates the bands.

4. Put the parts of one developed point in order. [1 mark]

Number the lines 1 to 3 to put them in the right order.

___ `Because: the reason it matters`

___ `The point itself`

___ `So: what follows from it`

> **Answer:**
>
> ```
> The point itself
> Because: the reason it matters
> So: what follows from it
> ```
>
> Point, because, so. Repeat three times, then conclude.

5. Which is worth more marks? [1 mark]

- ☐ **A** "The cache holds data the processor uses most, so fewer slow fetches from RAM are needed"
- ☐ **B** "It is faster"
- ☐ **C** "Cache is good"
- ☐ **D** "It improves performance"

> **Answer:** A. Name the thing and say why: development is where the marks are.

The task: mark a long answer

Write `mark(answer)` that scores a draft: one point for each phrase from `FOR_WORDS` it contains, one for each from `AGAINST_WORDS`, and two if it contains a phrase from `CONCLUSION_WORDS`, all ignoring capitals. For each answer in `answers`, print `<name>: <score>` and then, on the same line, `needs both sides` if it has nothing from one of the two sides, or `needs a conclusion` if it has no conclusion phrase, or `top band` if it has all three. At the end print `best: <name>`. Watch out: `disadvantage` contains `advantage`, so it counts on both sides.

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

FOR_WORDS = ["benefit", "advantage", "because it helps"]
AGAINST_WORDS = ["drawback", "disadvantage", "risk"]
CONCLUSION_WORDS = ["on balance", "overall", "in conclusion"]
answers = [
    ("Ada", "A benefit is that students learn to program, and another advantage is
engagement. A drawback is the cost. On balance the school should buy them."),
    ("Bolt", "There is a clear advantage for practical work and another benefit for
engagement."),
    ("Cog", "The risk is e-waste and the disadvantage is the cost, but the benefit is
real."),
]
```

The hint students can ask for: Count how many phrases from each list appear in the lowered answer, with the conclusion worth two. Then decide the note: no phrase from one of the two sides, no conclusion, or all three.

A solution

```
from bugbot import *
connect()
FOR_WORDS = ["benefit", "advantage", "because it helps"]
AGAINST_WORDS = ["drawback", "disadvantage", "risk"]
CONCLUSION_WORDS = ["on balance", "overall", "in conclusion"]
answers = [
    ("Ada", "A benefit is that students learn to program, and another advantage is
engagement. A drawback is the cost. On balance the school should buy them."),
    ("Bolt", "There is a clear advantage for practical work and another benefit for
engagement."),
    ("Cog", "The risk is e-waste and the disadvantage is the cost, but the benefit is
real."),
]

def mark(answer):
    text = answer.lower()
    fors = sum(1 for w in FOR_WORDS if w in text)
    againsts = sum(1 for w in AGAINST_WORDS if w in text)
    ends = 2 if any(w in text for w in CONCLUSION_WORDS) else 0
    return fors, againsts, ends

best_name = ""
best_score = -1
for name, answer in answers:
    fors, againsts, ends = mark(answer)
    score = fors + againsts + ends
    if fors == 0 or againsts == 0:
```

```
        note = " needs both sides"
    elif ends == 0:
        note = " needs a conclusion"
    else:
        note = " top band"
    print(f"{name}: {score}{note}")
    if score > best_score:
        best_score = score
        best_name = name
print("best:", best_name)
```

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