ANSWERS

F13.7 Where marks are lost

Exam preparation · GCSE · about 15 min

What this lesson is about

The ten commonest mistakes, checking on paper, and repairing a broken program.

- The ten that cost the most
- Checking a program on paper
- The slips that break code

Questions

5 marks in all

1. Which is the commonest way to lose marks you knew? [1 mark]

- ☐ **A** Answering the wrong command word
- ☐ **B** Bad spelling
- ☐ **C** Writing too neatly
- ☐ **D** Using a pencil

> **Answer:** A. Describe answered with one word scores one of two.

2. What is wrong with setting a total to 0 inside a loop? [1 mark]

- ☐ **A** It is reset every time round, so it never adds up
- ☐ **B** Nothing
- ☐ **C** It makes the loop slower
- ☐ **D** The loop never ends

> **Answer:** A. It belongs before the loop.

3. `if x = 5:` will not run. Why? [1 mark]

- ☐ **A** = assigns; a comparison needs ==
- ☐ **B** x must be a string
- ☐ **C** There is no colon
- ☐ **D** if needs brackets

> **Answer:** A. One of the commonest slips in a written answer.

4. Which three values should you trace a program with? [1 mark]

- ☐ **A** A normal one, a boundary one and an invalid one
- ☐ **B** Three normal ones
- ☐ **C** Only the boundary
- ☐ **D** Only what the question gives

> **Answer:** A. That is the same test data you met in F6.3.

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

```
total = 0
for n in range(1, 6):
    total = total + n
print(total)
```

> **Answer:**
>
> ```
> 15
> ```
>
> 1 + 2 + 3 + 4 + 5 = 15.

The task: repair the program

This program should drive 4 steps of 15 cm, printing `step <n>: <total> cm` after each one, then `total: 60 cm` and turn the LED green. It has three faults. Find them, fix them, and leave the rest alone.

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

for step in range(1, 4):
    total = 0
    forward(50, distance=15)
    total = total + 15
    print(f"step {step}: {total} cm")
print(f"total: {total} cm")
led(255, 0, 0)
```

The hint students can ask for: Three faults: how many times the loop runs, a variable that is set back to zero every time round when it should be set once before the loop, and the colour at the end.

A solution

```
from bugbot import *
connect()
total = 0
for step in range(1, 5):
    forward(50, distance=15)
    total = total + 15
    print(f"step {step}: {total} cm")
print(f"total: {total} cm")
led(0, 255, 0)
```

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