WORKSHEET

A15.1 How your A level is assessed

Exam preparation · A level · OCR H446 2.2.1, AQA 7517 4.4.1.2, Eduqas A500QS 1.8 · about 35 min

BugBotLab

Name _____ Class _____ Date _____

What this lesson is about

The papers and the project for OCR H446, AQA 7517 and Eduqas A500QS, the assessment objectives, and a timing plan built with integer arithmetic.

- Three boards, one shape
- The assessment objectives
- Marks are the clock
- The project runs alongside
- Using this module

Questions

5 marks in all

1. Which A level paper is sat on a computer, with a skeleton program released before the exam? [1 mark]

- ☐ **A** AQA paper 1
- ☐ **B** OCR paper 1
- ☐ **C** OCR paper 2
- ☐ **D** AQA paper 2

2. What percentage of the A level is the programming project worth, on every board? [1 mark]

3. A question says: "Write a program that returns the robot home using a stack." Which assessment objective does it mainly target? [1 mark]

- ☐ **A** AO3: design, program and evaluate solutions
- ☐ **B** AO1: knowledge and understanding
- ☐ **C** AO2: apply knowledge to analyse problems
- ☐ **D** None of them

4. An OCR paper has 140 marks in 150 minutes. To the nearest minute, how long should a 12-mark question take? [1 mark]

5. What does this timing calculation print? [1 mark]

```
start = 9 * 60
finish = start + round(35 * 1.4)
print(f"{finish // 60:02d}:{finish % 60:02d}")
```

The task: the timing plan

Make a timing plan for a mock paper. `START` is the start time as a string `"hh:mm"` in 24-hour time. `MINUTES` is the length of the paper in whole minutes, `CHECKING` is the whole number of minutes kept back at the end for checking, and `marks` is a list of the whole-number marks for each question in order. 1. Write `clock_time(minutes)`, which takes a whole number of minutes after midnight and returns the time as `"hh:mm"`, with two digits each. Use `//` for the hours and `%` for the minutes. 2. The minutes a mark are `(MINUTES - CHECKING)` divided by the total of `marks`. Print `<n> minutes a mark`, to one decimal place. 3. For each question, keep a running total of the marks so far. The question should be finished by the start time plus that total times the minutes a mark, rounded to the nearest whole minute with `round`. Print `Q<n> (<marks> marks): finish by <hh:mm>`, numbering from 1. 4. Print `checking from <hh:mm>`, the start plus `MINUTES - CHECKING`, and then `end <hh:mm>`, the start plus `MINUTES`. The robot stays still.

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

START = "09:00"
MINUTES = 150
CHECKING = 10
marks = [8, 12, 15, 5, 20, 10, 18, 12]
```

Plan your program here, then type it in and press Run.

Do it on the robot

www.bugbotlab.com/learn/a15-1-how-a-level-is-assessed/

The simulator checks it and tells you when it passes. Nothing to install, no account.

Challenges

1. Change the plan to your own board's paper, and find how many minutes you should spend on a 12-mark question.
2. Which assessment objective does each target: "State the purpose of the program counter"; "Explain why this robot needs a real-time operating system"; "Write a function to validate a room number"?
3. Make `clock_time` refuse a negative number of minutes with a `ValueError`.