

THEME B: COMPUTATIONAL THINKING · B3.1

Fundamentals of OOP for a single class

Original practice questions · 34 marks · about 55 minutes · covers B3.1.1–B3.1.5

Instructions. Answer all questions. The maximum mark for this paper is [34 marks]. The marks available are shown in brackets. Parts badged **HL** assess higher level only content; standard level students may leave these out. Questions that ask you to construct code may be answered in **either Java or Python**. State which language you are using. Marks are for the logic, not for perfect syntax.

1**Total: 6 marks**

Object-oriented programming models real-world entities in code.

(a) Outline what is meant by a class and by an object.

[2]

(b) **Evaluate** the use of an object-oriented approach rather than a procedural one for a large program worked on by several developers.

[4]

A gym manages its lockers in software. Every locker has a number, a size (small or large) and a record of whether it is currently occupied. A locker can be assigned to a member and later released.

- (a)** Construct a UML class diagram for a Locker class, showing its attributes and its methods. [4]

- (b)** State the symbols used in a UML class diagram to show that a member is private and that it is public. **[2]**

The Locker class is now written in code.

- (a)** Construct the Locker class, including its attributes and a constructor that sets the locker number and size, with the locker initially unoccupied. **[5]**

- (b)** Construct a statement that creates a Locker object with number 12 and size "large". **[2]**

4

Total: 5 marks

The attributes of the Locker class are declared as private.

- (a) Explain what is meant by encapsulation and information hiding. [3]

- (b) Explain why the attribute recording whether a locker is occupied should be private rather than public. [2]

5

Total: 5 marks

The gym also wants to know how many lockers exist in total.

- (a) Distinguish between a static variable and an instance variable. [3]

- (b) Suggest how a static variable could be used in the Locker class, and justify your answer. [2]

A method is needed to assign a locker to a member.

Construct a method `assign()` that marks the locker as occupied only if it is currently free. It should return `true` if the locker was successfully assigned and `false` if it was already in use.

[5]

END OF QUESTIONS