

THEME B: COMPUTATIONAL THINKING · B3.2 · HL ONLY

Fundamentals of OOP for multiple classes

Original practice questions · 34 marks · about 55 minutes · covers B3.2.1–B3.2.5

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. **B3.2 is a higher level only topic**, so every question in this paper is HL. 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

The gym system is extended. A Person class holds the details common to everyone. Member and StaffMember both build on it.

```
Person    .....  name, dateOfBirth,  describe()
|
+-- Member      ....  membershipNumber,  describe()
+-- StaffMember ....  staffId, role,      describe()
```

(a) **HL** Explain how inheritance promotes code reusability in this system. [3]

(b) **HL** Construct the class header for StaffMember to show that it inherits from Person. [1]

(c) **HL** Explain the effect on StaffMember of declaring the name attribute of Person as private rather than protected. [2]

Each class provides its own version of the `describe()` method.

- (a) **HL** Explain what is meant by method overriding. [2]
-
-
-
- (b) **HL** Construct the overridden `describe()` method in the `Member` class so that it also reports the membership number. [3]
-
-
-
-
-
-
-
-
- (c) **HL** A variable declared as type `Person` holds a `Member` object. State which version of `describe()` runs when it is called, and explain why. [2]
-
-
-
-

Polymorphism can also be applied without inheritance.

- (a) **HL** Distinguish between method overriding and method overloading. [3]
-
-
-
-
- (b) **HL** Construct **two** overloaded versions of a method `addMember`: one taking a name only, and one taking a name and a membership number. [2]
-
-
-
-
-

4

Total: 6 marks

The developers make Person an abstract class.

- (a) **HL** Explain how an abstract class establishes a common interface for its subclasses. [4]

- (b) **HL** State **one** reason why an abstract class cannot be instantiated. [2]

5

Total: 5 marks

A Gym object holds a set of Locker objects, which are created when the gym is built and destroyed if it closes. It also holds a list of Member objects, who may leave and join other gyms.

- (a) **HL** Distinguish between composition and aggregation. [3]

- (b) **HL** Identify which of these two relationships applies between Gym and Locker, and justify your answer. [2]

Design patterns provide tested solutions to problems that occur repeatedly.

- (a) **HL** Explain the singleton pattern and a problem that it solves. **[3]**

- (b) **HL** Outline the purpose of the observer pattern. **[2]**

END OF QUESTIONS