2.7 Object-Oriented Programming: 10-minute Pack

AQA 7517 · Calculators not allowed · about 10 minutes

Total Marks

Name: _____ Date: ____ / ____ / _____

Answer ALL questions. Show all your working.

1 A junior programmer has written the following class, but has not applied good object-oriented design principles.

class Student
public studentName
public examScore

public new(name, score)

```
    studentName <- name
    examScore <- score
endnew
```

endclass

Elsewhere in the program, another part of the code contains the line:

```
student1.examScore <- 150
```

(a) Identify two problems with this class and the line of code shown above. **(4)**

(b) Rewrite the class using AQA pseudocode conventions so that studentName and examScore are properly encapsulated. Include a constructor, a getExamScore() method, and a setExamScore(newScore) method that only updates examScore if newScore is between 0 and 100 inclusive. **(6)**

(Total for Question 1 is 10 marks)

© Revision Library | revisionlibrary.co.uk

Mark scheme · 2.7 Object-Oriented Programming: 10-minute Pack

Question 1

(a) problem 1 identified: both attributes are public, so they can be accessed and changed directly from anywhere in the program, breaking encapsulation **B1**

(a) problem 1 developed: this allows the line student1.examScore <- 150 to set examScore directly with no check at all **B1**

(a) problem 2 identified: the class has no validation logic (e.g. no setter method) to enforce rules on new values **B1**

(a) problem 2 developed: so an invalid score, such as 150 (presumably above a maximum possible mark of 100), can be stored without being rejected **B1**

(a) Answer: Problem 1: studentName and examScore are declared public, so any part of the program can access or change them directly, breaking encapsulation - this is why the line student1.examScore <- 150 is allowed to run at all. Problem 2: the class has no method that validates a new exam score before storing it, so an invalid value such as 150 (assuming scores should be 0 to 100) is accepted with no check.

(b) private studentName and private examScore declared **M1**

(b) constructor new(name, score) correctly assigning both parameters to the attributes **M1**

(b) getExamScore() method that returns examScore **M1**

(b) setExamScore(newScore) method header defined **M1**

(b) IF newScore ≥ 0 AND newScore ≤ 100 THEN condition used (oe, e.g. two nested IFs) **M1**

(b) examScore <- newScore placed inside the IF, so an out-of-range value is simply not stored cao **M1**

(b) Answer:class Student
private studentName
private examScore

public new(name, score)

```
    studentName <- name
    examScore <- score
endnew
```

public getExamScore()

```
    return examScore
endgetExamScore
```

public setExamScore(newScore)

```
    IF newScore >= 0 AND newScore <= 100 THEN
       examScore <- newScore
    ENDIF
endsetExamScore
```

endclass

© Revision Library | revisionlibrary.co.uk