AQA Computer Science A-Level

4.1.2 Programming Paradigms

Past Paper Mark Schemes

Additional Spec Qs Paper 1

03	1	**All marks AO1 (understanding)** Easier to test each class using unit testing // with composition each class can be tested separately but it is not possible to test a subclass independently from the base class; There can be unintended side-effects for derived classes if a method in the base class is altered; Composition is more flexible as if a new class is developed it can easily be used instead of the class that currently is used in the composition; **MAX 1**	1

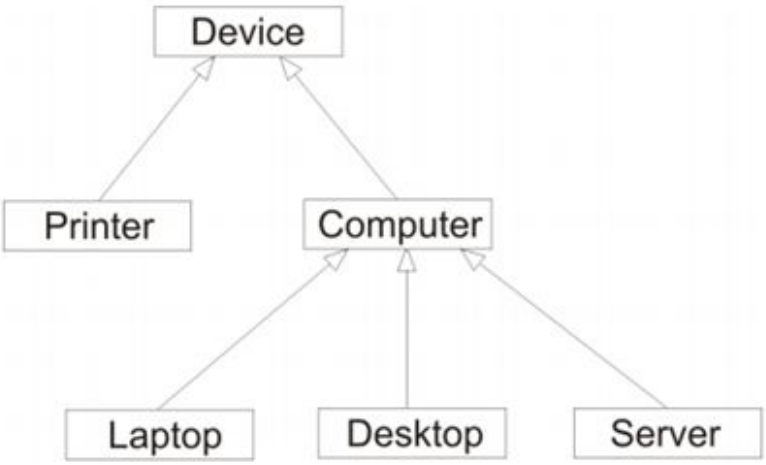
03	2	**All marks AO2 (apply)**
		
		`Buyer = Class(Client) {`
		`Private:`
		`  NoOfBedroomsRequired: Integer`
		`  OffStreetParking: Boolean`
		`  AreaDesired: String`
		`Public:`
		`  Function GetNoOfBedroomsRequired`
		`  Function GetOffStreetParking`
		`  Function GetAreaDesired`
		`  Procedure SetDetails (Override)`
		`}`
		
		Info for examiner: Accept answers that use different notations, so long as meaning is clear.
		
		Mark as follows:
		
		1 mark for correct header including name of class and parent class;
		1 mark for redefining the `SetDetails` procedure;
		1 mark* for defining all 3 extra functions needed to read variable values, all identified as being public (keyword public is optional if functions are declared before variables);
		1 mark** for defining all 3 extra variables, with appropriate data types and identified as being private;
		A. answers that indicate separately that each variable is private or each method is public
		R. do not award mark for declaring new functions if any of the functions have the same name as the variables
		I. parameters to methods, minor changes to names that do not affect clarity
			4

		* - Do not award this mark if any extra functions/procedures have been declared, except for functions that would set values e.g. `SetAreaDesired` an incorrectly named procedure to add e.g. `AddBuyer` ** -Do not award this mark if any extra variables have been declared	

03	3	**All marks AO1 (knowledge)** Composition and aggregation are both "has a" relationships – when an object contains another object; With composition if the containing object is destroyed so are the objects in contains, this is not the case with aggregation;	2

03	4	**One mark is for AO1 (understanding) and one mark is for AO2 (apply)** **1 mark for AO2:** the `Client` class contains an object of the class `Location`; **1 mark for AO1:** If composition had been used then if the client object is destroyed so is the address object – but in real life the address object would still exist;	2

June 2012 Comp 3

7	(a)		 **1 mark** for Device at top of diagram with Printer and Computer directly underneath it and linked to it and no other labels linked to it; **1 mark** for Computer with Laptop, Desktop and Server directly underneath it and linked to it, and no other labels linked to it (except Device above); **1 mark** for correctly styled diagram, i.e. lines drawn as arrows and boxes (any shape) around labels; - ***This mark is only available if candidate has already achieved at least one mark for correct contents of the diagram.*** **A** arrows drawn as:  **A** filled/empty arrowheads **A** diagram rotated by 90 degrees	3

7	(b)		Computer = Class/Subclass/Extends(Device) 1 (Public) Procedure AddDevice (Override) 1 Function GetProcessorName Function GetRAMCapacity } 1 Function GetHDDCapacity Private / Protected ProcessorName : String RAMCapacity : Integer } 1 HDDCapacity : Integer End Accept answers that use different notations, so long as meaning is clear. **1 mark** for correct header including name of class and parent class; **1 mark** for redefining the AddDevice procedure; **1 mark*** for defining all 3 extra functions needed to read variable values, all identified as being public (keyword public is optional if functions are declared before variables); **1 mark#** for defining all 3 extra variables, with appropriate data types and identified as being private; **A** any sensible numeric types for RAMCapacity and HDDCapacity, do not have to be whole numbers **A** answers that indicate separately that each variable is private or each method is public **R** do not award mark for declaring new functions if any of the functions have the same name as the variables **I** parameters to methods, minor changes to names that do not affect clarity * - Do not award this mark if any extra functions/procedures have been declared, except for functions that would set values e.g. SetProcessorName or an incorrectly named procedure to add e.g. AddComputer # - Do not award this mark if any extra variables have been declared	4

7	(c)		Laptop = Class/Subclass (Computer) 1 (Public) Procedure AddDevice (Override) 1 Function GetBluetoothInstalled Private / Protected 1 BluetoothInstalled : Boolean End **1 mark** for correct header including name of class and parent class; **MAX 1** of the following two marks: **1 mark** for redefining the AddDevice procedure; **1 mark*** for : • defining the GetBluetoothInstalled function needed to read this value, identified as being public (keyword public is optional if function is declared before variable) • defining the BluetoothInstalled variable with an appropriate data type as being private. **A** Boolean or whole number types for BluetoothInstalled but reject string, character or real number types **A** Different sensible name for GetBluetoothInstalled function e.g. CheckBluetoothInstalled, IsBluetoothInstalled **A** answers that indicate separately that each variable is private or each method is public **I** parameters to methods, minor changes to names that do not affect clarity **I** addition of any extra functions or variables * Do not award this mark if any extra functions / procedures / variables declared, except for a SetBluetoothInstalled procedure.	2