CS230 Tutorial 3

This week we will look at classes, interfaces and abstraction in more detail. We'll use UML to understand and answer questions about a system and then use ArgoUML to model a case study.

This is an answer sheet, answers are marked in RED.

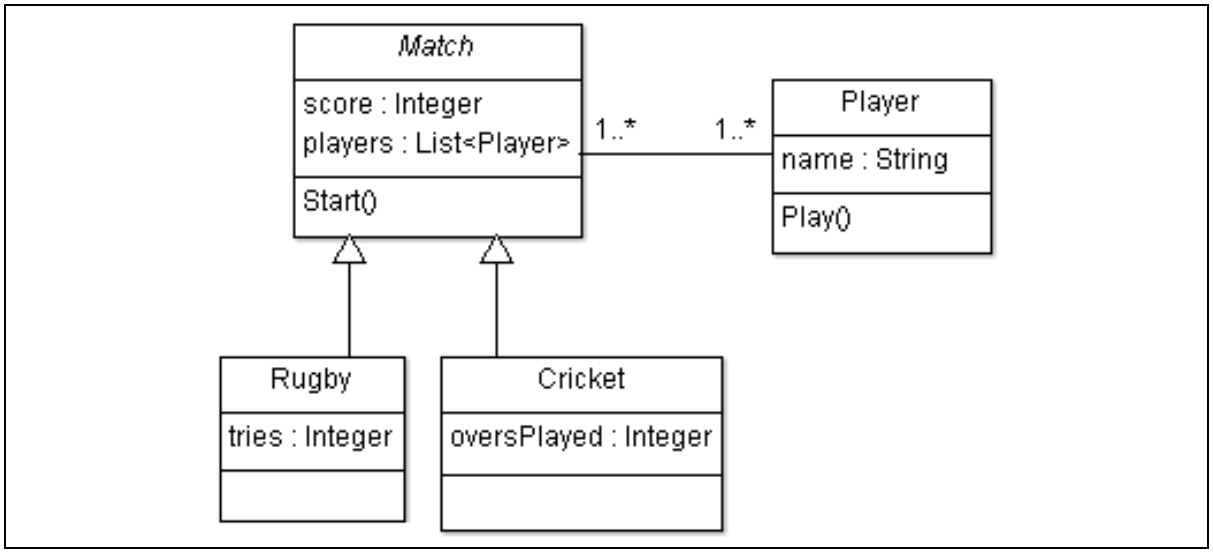
Quiz dates:

- **Quiz 2**, available from 9am Friday 3 Apr to 11:30pm Monday 6 Apr
- **Quiz 3**, available from 9am Friday 8 May to 11:30pm Monday 11 May
- **Quiz 4**, available from 9am Friday 29 May to 11:30pm Tuesday 2 Jun.

Review

Last week we looked at designing UML diagrams with ArgoUML. Test your understanding by answering the following question.

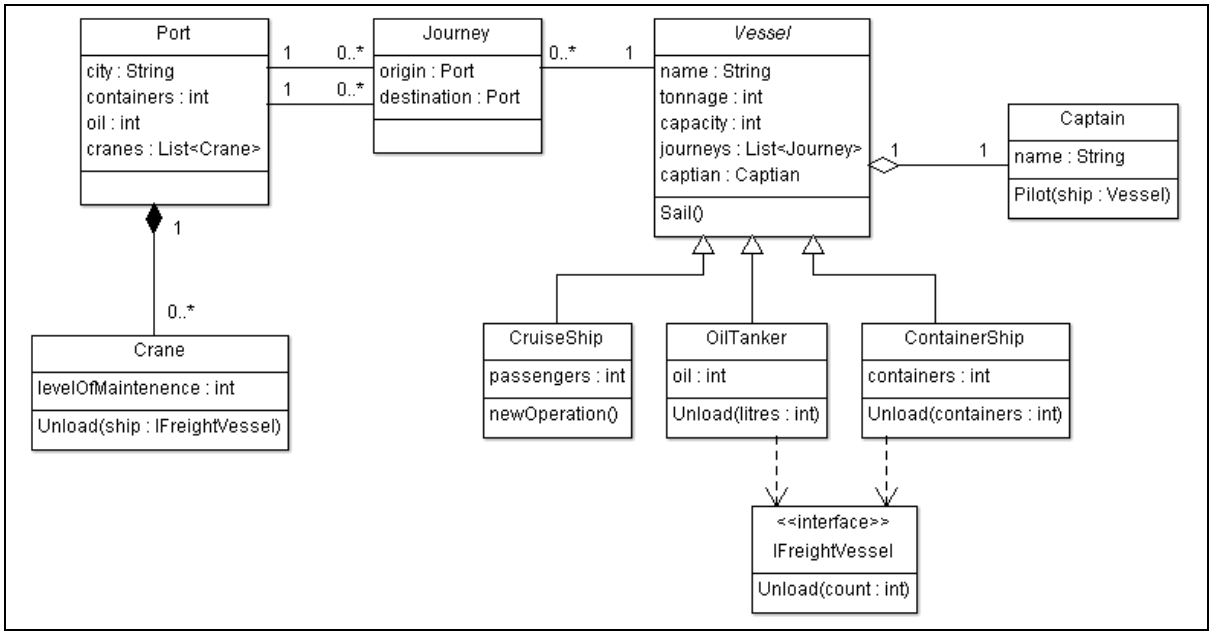
Question 1: Examine the following simple relationship in the following diagram:



a. What type of diagram is this?	The above diagram is a *UML Class Diagram*.
b. What is the relationship between *Rugby* and *Match*?	Rugby is a subclass of Match. In other words, Match *generalises* Rugby.
c. What is the multiplicity between *Match* and *Player*?	A match has between 1 and many players; a match has many players. A player can play many matches. You can't have a Match without a Player and vice versa.
d. What *type* of class do you think Match is likely to be?	Match could be an *abstract* class. Rugby and Cricket, conversely, would be concrete classes.

Part A – Expressive Relationships in UML

The following UML class diagram describes a way to model sea ports and large ships. We'll be examining this example in detail as it includes the association, aggregation, composition and interface relationships.



Question 2: Name two classes that have an *Association* relationship

Vessel and *Journey* (or *Port* and *Journey*)

Question 3: Name two classes that have an *Aggregation* relationship

Port and *Crane*

Question 4: Name two classes that have a *Composition* relationship

Vessel and *Captain*

Question 5: Name a class that *Implements* an interface

OilTanker (or *ContainerShip*)

Refer to the class diagram on page 2 for the following questions.

Question 6: Name a class that *extends* another class (i.e. name a subclass)

CruiseShip (or *OilTanker* or *ContainerShip*)

Question 7: Identify one abstract class

Vessel is an abstract class. In UML class diagrams, abstract classes are often represented by making the title of the class *italicised*.

Question 8: List all Attributes of the OilTanker class (including inherited ones)

OilTanker has the attribute 'oil' and the inherited attributes: 'name', 'tonnage', 'capacity', 'journeys' and 'captain'.

Question 8: Reference types are the different variable types that can be used to refer to an object and are related to polymorphism. What reference types can be used to refer to a *ContainerShip* object?

ContainerShip, IFreightVessel, Vessel and Object are valid reference types for a ContainerShip object.

Part B – Modelling a Domain

You have been hired by a restaurant to build them a point of sales system. The restaurant makes both food and drink by using local ingredients. Your manager tells you they want to keep track of all the different food and drink items they sell on their menu, as well as the ingredients required for each item. Both food and drink should have dietary notes (such as "may contain traces of nuts") as well as a price. Drinks carry an extra service charge which must be captured separately. Food needs to have a course (entrée, main, dessert). The menu changes periodically and the system you design will need to keep track of this. You will also need to keep track of stock levels for ingredients. An ingredient could be used for either a drink or food.

Next, your boss asks if you can extend the system to handle reservations. They don't care too much about who has made the booking apart from their phone number. They want to keep track of the table number for the reservation. At the end of the meal, your boss needs to be able to calculate the bill for the table, based on what the customer ordered.

Question 9: Solution UML Class Diagram