Section A

This section is COMPULSORY

Question A1.

a). How can UML diagrams be used in the context of Agile development? Your answer should mention at least two different uses. (4 marks)

All UML diagrams are ceremony, and hence optional [1]. Domain class diagrams can be used to help understand the domain and motivate questions for stakeholders [1] Design class diagrams can be used to discuss design choices [1] UML diagrams can be used at the start of a sprint to help determine architectures choices for that sprint. [1]

Marks for anything else sensible, e.g. they can be used at any point in a sprint to clarify the code structure.

This question was generally answered well, with no obvious systematic misunderstandings

b). State two similarities between the agile UP and SCRUM (2 marks)

They both use timeboxed iterations. [1] They both aim to minimise ceremony. [1]

As was this

c). State two differences between the agile UP and SCRUM (2 marks)

The UP starts off with a large number of possible artefacts, SCRUM does not. [1] SCRUM has features such a retrospectives which are not in the UP. [1]

And largely this. Some people identified things as differences that weren’t (the extent of the similarity between the two approaches was not appreciated in some answers).

d). The University of Southwest England is a fairly small university with a highly distributed campus and many distance learning students. Existing student records systems are expensive and insufficiently flexible for their purposes, so the University wishes to develop a new system and your team has provisionally been awarded the contract. The final go-ahead for the project will be given by the University Senior Management Team (SMT) based on the recommendation of the University’s Head of IT. Of course you intend to use agile development.

i. You have a problem. The head of IT is only familiar with Waterfall development, and when he consults Computer Science academics he discovers that they are sceptical of agile methods (which they refer to as “fragile
methods”). How would you convince him that agile development is indeed a good way of going about the project? (3 marks)

Agile methods deal much better with changes which occur throughout the project than waterfall development. [1] As the University has an unusually distributed structure, the requirements are likely to be complex and harder to elicit than for standard systems. [1] (Or something which relates to the specific scenario rather than being generic). Agile methods also produce early results allowing feedback from stakeholders and business value to the University at an early stage. [1]

This was answered very well, marks were mostly lost for saying too few things rather than saying the wrong things.

ii. Apart from the SMT, name four groups of stakeholders, and what their principal concern of each group would be (4 marks)

Students need to be assured that their records are correct and confidential
Academic staff need appropriate access to the records
Sys admin staff need to be able support the system without undue effort
Clerical staff need the system to be easy to use and to support existing procedure.

However here’s where it gets tricky. Many students failed to recognise the range of stakeholders involved, which has a knock-on effect...

iii Would you be willing to have a single stakeholder representative work with this project development team? Briefly explain why or why not (2 marks)

No, because no single person can represent such a divergent set of interests.

Answers which made a case for a single rep were given 1 mark iff they justified the choice of rep carefully. A number of people were under the impression that the University Senior Management Team would be good customer reps. Although are paying for the system they are unlikely to ever use it.

iv. Together the Head of IT and the Computer Science academics are not convinced by the idea of an agile method such as Scrum. They are prepared to accept the use of agile UP. Name, and motivate the choice of, three UP artefacts that you would like to use in the project. (1 mark for each named AND motivated choice) (3 marks)

Main point is that the use of the artefacts is motivated. E.g. A domain model is good because the domain is complex and not familiar to the developers. [1] Design class diagrams will be used to discuss design choices as the design is non trivial. [1] Use cases would be used, generally brief or casual, although where clearly defined clerical procedures are required, fully dressed UCs might be appropriate. [1]

A number of answers confused SCRUM artefacts (e.g. user stories) with UP artefacts (eg. use cases).
Section B

Answer ONE question from this section

Question B1

Note: in answering this question you should ignore any actual knowledge you may have about the domain and work purely from the question text.

a) Briefly explain how the Requirements disciple in the Agile UP fits into the process overall. In this respect is there any significant difference between the Agile UP and SCRUM? (3 marks)

Requirements are gathered throughout the process (although the proportion of requirements work is higher toward the beginning of the project) and are expected to change. [1] Requirements are gathered through interaction with stakeholders, including feedback on partial systems produced at the end of each iteration. [1] In this regard there is no significant difference [1].

Lots of answers got bogged down in details and missed the main point. There was lots of reciting of the phases of the UP (which some people thought occurred during each iteration!)

b) You have been hired to develop a system which will help with the teaching of Biology in secondary schools (ages 11-16). You have a 20 page “requirements specification” written by a junior government minister. It includes contact details for an experienced Biology teacher who has agreed to help in the project (in a way not yet specified). None of your team have any significant knowledge of biology or teaching experience. Describe the first few steps you would take in determining the requirements for the system (before writing any use cases or drawing any UML diagrams). (5 marks)

Read the document but assume any requirements stated there will be wrong. [1] Instead check it for contractual and funding information [1] Arrange to meet the biology teacher as soon as possible [1] Clarify how much time and effort s/he will be able to devote to the project [1] Ask her to outline in broad detail what the real requirements are [1]

Other good points are:

Talk to students about their experience of learning Biology [1] Find out about what sort of computing infrastructure a typical school has [1]

A lot of answers involved interviewing the govt minister, I allowed this if context was sensible, but finding out the real requirements from the biology teacher is much more important.
c). It is agreed that the Biology teacher will be seconded to you for 50% of her time for the duration of the project. However, her salary for that time will come out of the project budget, which means that you cannot afford to pay for your team members to work overtime. Briefly discuss the advantages and disadvantages of this arrangement. (3 marks)

In this case (unlike in A1) the teacher can be considered a good stakeholder representative, so she will be invaluable in helping to define requirements and giving feedback on partial systems. Working overtime is generally considered a Bad Thing in agile development anyway, as tired developers are ineffective developers. However, this means you have no extra resource to deploy if a crisis occurs, so the team needs to be well organised and avoid being “agile” in the bad sense of doing things at the last minute. [1]

Several answers mentioned the demotivating affect of not having the opportunity to work overtime – accepted if clearly stated. Several (surprisingly) understated the advantages of having an ideal stakeholder present.

d). The teacher tells you:

“One aspect of Biology we want the system to help with is Phylogeny, the study of relationships between different species, the ‘tree of life’. Each species is related to every other species by a common ancestor if you go back far enough. We want to be able to display and manipulate these ‘phylogenetic trees’. Any such tree will contain at least three species, a common ancestor and two descendents. The way I think about it, there are three kinds of species; modern ones like Humans, extinct ‘dead end’ species like Dodos, and common ancestors such as whatever was the common ancestor of Humans and Dodos about 200 million years ago.”

Draw a domain class diagram which shows the important concepts in this description. (5 marks)

There will be much variation in the diagram. Clearly the notions of Tree and Species are essential, and they are related by Tree 1..* contains 3..* Species. (The teacher does not think in terms of singleton or empty trees as a Computer Scientist would). The phrase “The way I think about it, there are three kinds of species;” mandates that there should be three subclasses of Species, with names such as CommonAncestor, ModernSpecies and DeadEndSpecies. There’s little justification for adding much else to the diagram (a timeOfDivergence) attribute for a common ancestor, maybe). Likely errors are

Adding extraneous classes detail not justified by the description.
Notation abuse – the model includes both containment and inheritance relationships and they need to use the right kind of arrows.
(Very weak students) showing instances like Human rather than classes.

By that criterion most of the students were very weak! Many answers had class diagrams which were phylogenetic trees, as opposed to modelling the components of such a tree according to the description. Class diagrams always show classes, not objects.
e). You ask what information needs to be recorded about individual species. The answer is

“Well that depends on the species - different species have different properties. For example for birds we might want to know the wingspan, for a fish the kind of water they live in. I imagine the database will contain lots of properties for each species, and we won’t want to show them all to the students.”

As a result of this information you might add a Database class to the domain class diagram. What other addition would you make? (2 marks)

A property class where a species has one or more properties. [1] Because of the variability in kinds of properties, we can only represent them in a very general way, for example as a name, value pair e.g. (wingspan, 2m). [1]

Almost nobody got the key idea here. I gave a mark for anything which sort-of makes sense.

f). Suggest two Pure Fabrications which might be added when you design this part of the system. (2 marks)

There is a database which suggests a database connection class. [1] Property handling sounds complicated, so some sort of PropertySet or PropertyHandler (or maybe PropertyFactory) is indicated. [1]. Marks for any sensible suggestion.

This was better though, a lot of people suggested database connectors or factories.

Question B2.

a). What is the fundamental relationship between a domain class diagram and a design class diagram? (1 mark)

Domain classes often inspire design classes, although this is not a 1-1 mapping. (No marks for answering any other questions about domain classes and design classes, e.g. replacing “relationship” with “difference”).

b). State three things which are shown on design class diagrams which do not appear on domain class diagrams. (3 marks)

Extra classes, pure fabrications which don’t correspond to domain classes. [1] Software oriented things like types and visibilities[1] Operations [1]

A number of answers had multiple examples of pure fabrications, I counted these as 1. Also types-and- visibilities were counted as 1 unless there was another marginal point to add in

c). Explain, with an example, the GRASP principle of Polymorphism. For full marks you should use an example which has not been given in the course, and state the kind of application for which such use of polymorphism would be appropriate. (3 marks)
The principle is “use polymorphism (inheritance) to improve cohesion and reduce coupling” [1]
A good situated example is. Classes Bus, Car etc. inheriting from a RoadVehicle class[1] in a
road traffic simulation [1]

Remarkably a number of answers gave exactly this example. I don’t remember mentioning it in the
course so I gave it full marks. It is an example I use elsewhere, and which other people have
borrowed from me.

d). You are working on a Computer-assisted assessment project. Like ABC and MELT, your software has different kinds of questions – for now just multiple choice questions and text questions. Also, like ABC but unlike MELT, questions can have sub-questions and sub-sub questions nested to any depth. This is achieved by having the notion of a composite question which contains other questions.

A junior developer joins your team. You know nothing about him, so you set him the task of designing the classes representing questions; (You won’t be using his code because those classes are already written.) You are rather surprised when he comes back with a single class, which called Questions, which he explains as follows:

“I chose to make it a single class to minimise coupling – the rest of the code has only one class to interact with. Also, I found some nice opportunities for code reuse; for example this list contains sub-questions if it’s a composite question and the options if it’s an MCQ. This public instance variable QT shows what kind of question it is.”

i. Explain, in terms of GRASP principles, why this is a truly awful design. Your answer should include an explanation why this design does not minimise coupling. (4 marks)

Note: this example is not a far-fetched as it sounds. I’ve noticed that even MSc students who are strong programmers often bung everything in one class, and I’ve been trying to discourage this in the course. MELT is the case study used in my lectures, ABC is that actual online exam software they use.

Firstly the Questions class is completely incohesive, it represents several different entities. [1] The design does not minimise coupling because any code using the Questions class must do so through a large and messy interface; first if must check the type of the question, then it must do different things depending on the result. [2 for a good explanation, 1 for the general idea] It also does not make use of the obvious Polymorphism in question types [1] Finally it has AAAGH! a public instance variable which AAAGH! doesn’t have a meaningful name and AAAGH! Doesn’t follow normal naming conventions. [1] – only one for this because they shouldn’t fixate on it, replacing it with a getQuestionType() method does nothing to fix the real problems with the design.

Generally answered ok but there was a tendency to ramble on about marginally (at best) relevant stuff rather than focussing on the specifics of the example. Always give specific answers to specific questions.

ii. What would be the practical consequences of implementing this design? (2 marks)
Code using the Questions class would be complex (lots of if-statements), and highly sensitive to changes in the Questions class. [1] That class would itself be complex and difficult to modify if different question types (e.g. diagrams) were added. [1]

A lot of people found this difficult. I appear to have given a number of generous marks

[iii] Draw a UML class diagram which shows a set of classes and representing a much better design. You do not need to need to show any attributes or operations, but you do need to show the relationships between the classes clearly. Hint: you need a CompositeQuestion class which is related to the Question class in two different ways. (4 marks)

Hurrah! This is of course the Composite design pattern (which I deliberately avoided in the MELT case study). They should have an abstract question class with subclasses TextQuestion, MCQ and CompositeQuestion. There should also be a relationship CompositeQuestion contains> 1..* Question (not the subtypes individually). Usual remarks about using the right kind of arrows and avoiding spurious junk apply.

I was pleased by how many people got this right or nearly right

[iii] Briefly explain why this design really does exhibit low coupling. (2 marks)

External code is still coupled to just one class, but this class has a much simpler interface. [1] There is low coupling within the design too, e.g. the specific question classes are coupled only to the Question class.[1]

Almost everybody considered one or other of the above but not both

[iv] Explain how a method to get the marks allocated to a question would be implemented (only an explanation is required, not the code). (1 mark)

Each atomic question (text or MCQ) would store its marks allocated. The CompositeQuestion class would recursively add up the marks of its subquestions. [1]

It’s a pity I only allocated one mark to this, it would have been much better out of 2. Many spot-on answers, but quite a lot who just didn’t; understand the question.

Section C

Answer ONE question from this section

Question C1
a) In agile approaches individuals and interactions are valued more than processes and tools. Discuss.

Individuals constitute teams [1] and individuals in teams interact to perform work [1]. Processes and tools are secondary in the process of performing work [1], but that does not mean that we do not value them too [1]. For example in a TDD/BDD model of development the only way that the development process can proceed is because tools make automated testing possible [1].

Nobody distinguished between processes and tools but then the marking scheme above does not require them to. Some answers didn’t say anything processes and tools at all and I gave full marks only to answers which did at least mention them in some sensible way.

b) In the course retrospective someone wrote on a sticky words to the effect of “The course was about not coming to the point” (i.e. a single conclusion) Explain why one valid view of an agile development course is that it should not reach such a point?

The agile manifesto contains only four statements of value [1] and is motivated by 12 principles [1]. These are not prescriptive [1] and given the breadth of possible interpretations and subtleties of project organisation and context [1 for anything sensible] it is impossible to be definitive to the extend of reaching a single point in the course [1].

Some answers talked purely about agile development and ignored the two instances of of the word “course” in the question. Others were confused between the two. As a result some quite similar-looking answers got very different marks.

c) Name ten things that must happen in a pair programming situation involving the two people

Eye contact, continual communication, trust, respect, keyboard and mouse squarely in front of typist, pointing at things, asking questions until understanding reached. Nothing about testing, except work out test result before running the test. Common cause/intent. .... breaks, aka anything sensible

Many answers included more technical things such as refactoring. Mark and I agreed that these got marks too, but that for full marks an answer must include some of the human-interaction stuff above.

d) Burndown charts are used in Scrum. Draw a burndown chart (labelling the axis) for a team that has underestimated its velocity. In such a situation what should the team ask/do, and why?
Chart including correct axis labels [3] ask product owner/customer to move a sprint backlog item/user story back to the product backlog [2 for perfect terminology]

*The above is the wrong way round! People could either do this (the majority) or they couldn’t so the marks were fairly binary*

**Question C2**

a) In the course retrospective someone wrote on a sticky words to the effect that “The course was about not coming to the point.” However, the thinking student attending the course would be able to give several learning outcomes of importance from the course. Give three examples of learning outcomes (i.e. topics in agile development) to reach in such a course, and reasons why these outcomes are valuable.

The examples used in your answer should be non-trivial. Thus “learning about TDD” is not acceptable, a deeper understanding of the role TDD plays is required

(6 marks)

The creation of a learning culture [1] both [accept one] mirrors and gives experience of an ideal cultural aspiration for a team, and helps more learning to occur between team and class members [1].

Reaching the end of a sprint [1] with an understanding of sprint processes [1]

Understanding TDD to the point that it is understood as a means of design [1] so that students learn that big design upfront is not needed [any weak points only get 1]

*This question was too easy as it allows you to pick from anything in the course, so most people got high marks.*

b) Discuss the stages in a sprint retrospective, stating why each stage is important in the process. Note this is the team’s retrospective, not one involving the customer focused around acceptance testing and learning about the product.

(10 marks)

Setting the stage [1] scopes activity, aligns participants [1 for one of]

Gathering evidence [1] evidence is the basis of informed decision later [1].

*Time based then topic based clustering of information [1] Identification of problems/impediments [1]*


Close [1], important to build team cohesiveness, sprit, feeling [1]
Retrospective on the retrospective [1] enables learning for next time [1 must mention learning]

A lot of good answers, full marks only for getting things in the right order and motivating each step.

c) Name what you personally think of as the most important contributing factor in successful teamwork, and motivate this choice, indicating why it is a better choice than at least two competing choices

(4 marks for the motivation)

Sensible answers get 3, outstanding answers get 4

The wording of the question makes rating bananas over chocolate or coffee a valid answer. Full marks only if the chosen factor – usually communication within the team – is convincing.

END OF EXAMINATION