UG Exam Performance Feedback
Third Year - Semester 1

COMP 30001 High Performance Microprocessors — John Gurd

Comments:

Overall: The average mark for this year's paper is significantly lower than in previous years. By far the most common reason for losing marks was the absence of any answer to some part or parts of a question (including, in one case, a failure to answer an entire question). The paper followed the style of previous year's papers by posing a mixture of bookwork and problem-solving parts. Candidates appeared to avoid answering the problem-solving parts much more than the bookwork parts. Despite these problems, two candidates (10% of the class) attained first class marks overall.

Question 1: This was the question entailing the most problem-solving and, presumably as a result (although it was also a long question to read), it was the least popular to answer. Answers obtained an average mark below the average for the paper as a whole.

Question 2: This question was second-best answered on average, and was quite popular to answer.

Question 3: This was the second most popular to answer. It entailed problem-solving using loop-unrolling, which many candidates did not understand.

Question 4: This question was the second least popular to answer. It entailed cache bookwork and related problem-solving, but many candidates did not show understanding of how a cache memory works. As a result, answers to the question got the lowest average mark. In previous years, this material has been understood much better.

Question 5: This question contained the least problem-solving and was the most popular to answer. Answers achieved the highest average mark of all questions, although still less than 50%.

Comments:

Original question text in normal font, comments in italics. Extracts from my original marking scheme are in quotes. I've included a few comments about real-world issues which would arise in the situations mentioned. These are tagged <Real world> and are not part of the exam feedback as such, although you would be wise to read them.

In general the answers were very disappointing, both compared to previous years and to results on the tests from this year. One factor was a small "tail" of people who knew almost nothing, but that's normal with such a large group. Of much more concern was the number of marks which were thrown away by poor exam technique (particularly failing to read the questions properly) and other forms of carelessness. Of just over 6000 marks I had at my disposal, I reckon at least 500 were lost unnecessarily – the difference between an average which just crawled over 50 and the healthy high-50s of previous years.

The rubric says that the answers need to be concise as well as accurate. While there were some overly verbose answers which lost marks, a bigger problem was answers which were so short and cryptic that there was little to give marks for.

Question 1: Compulsory.

1.1: State four reasons why design patterns are important in object-oriented software development. 4 marks

A number of people seemed to be thrown by the slight rewording of the question from previous years, My marking scheme said:

"Provide tried and tested solutions to common software design problems
Provide a language which allow experiences developers to communicate efficiently
Provide a learning aid to inexperienced developers
Provide a framework for discussing design tradeoffs
(and any other reasonable points)"

I also gave a mark for clearly explaining what a design pattern was. Quite a lot of people failed to say four different things and so lost marks.

1.2Assume that there is a class ExamServer, and there should only ever be one instance of this, which must be created immediately when the server program runs. Show how the Singletom design pattern can be used to guarantee this (you can assume that the ExamServer constructor has no parameters).

Well done to those who spotted that since the ExamServer instance is created at startup, you don't need the testing for null stuff. (If you think about it, it makes sense to set up the exam server before doing anything else, rather than waiting for somebody to try to use it). Hence the solution is simply:

```
public class ExamServer {

private static ExamServer instance = new ExamServer(); // Eager evaluation

private ExamServer() {  // Constructor must be private
…
}

public static ExamServer getInstance() {
    return instance; // No need to check for null
}

}
```

Solutions which accurately reproduced the lazy instantiation version with the check for null got 3 /4.

<Real world>

Although the lazy instantiation trick is neat, it's not usually all that useful. Often, you need to set the thing up at the start to avoid delays later on, as in this example, or it's so small you might as well do so and save messing about. Lazy instantiation is only necessary if the Singleton takes up significant resources and is likely not to be needed. Also, any time you're thinking of using a Singleton, ask yourself whether you need to create an instance of the class at all – a factory for instance can often just be a class with static methods.
</Real world>

1.3Clearly the Observer pattern (in the form of the Java Delegation Event Model) would be used in the user interfaces of the project. Suggest a more fundamental role it could perform in the implementation of the Take Exam use case. 2 marks

What I was initially looking for here was:

"To organise the communication between the clients and the exam server. Either way round, or both, is plausible: the clients can be observers, registering themselves with the server and being notified of events or the server can be an observer, looking for changes of state in the clients"

However, the way the question is worded makes a number of other answers valid. In particular a lot of people applied it to the monitoring tool, making the question identical to that from a previous paper. I accepted this, or any other reasonable suggestion.

However, in reading the answers it was often unclear whether the writer actually knew what the Observer pattern was. I gave the marginal cases the benefit of the doubt… until I saw the diagrams.

1.4 Draw a UML diagram to illustrate the Observer pattern as applied to this situation. 6 marks

Then it became clear that many people had no idea what the Observer pattern was, and those which did often muddled things up. Suppose for example you have the monitoring tool watching for events from the clients. Then the Monitoring tool is the Observer (aka Subscriber, aka Listener), and it implements an Observer Interface (this allows for e.g. different kinds of monitoring tools). That interface defines methods which will be executed when a particular state change occurs in the client, e.g. onBackupFailure(). The client is the Observable (aka event source, aka Publisher). It has a method to register Observers (which takes an Observer Interface as it's parameter – you can also have a de-register method) and a notify() method which is used to notify all observers when an event of interest happens. Very few people drew a diagram which accurately reflected this.

In addition, a lot of people threw marks away by simply not using the UML notation correctly - e.g. using inheritance "fatarrows" for all sorts of things other than inheritance.

1.5 Suggest how two other design patterns might be used in implementing Take Exam 4 marks

The sort of thing I was looking for was:

"e.g. in a randomised multiple choice test the questions could be generated from a Factory which would be a Singleton. The exam client could have a Façade controller. Strategy could be used for different end-of-test behaviours for different types of assessments."

i.e. specific GoF design patterns. A lot of people waffled about cohesion, coupling, Pure Fabrication etc. which are not design patterns they are general GRASP principles. (I did allow Controller as although it's classified as a GRASP pattern it's specific enough to be considered as a design pattern in its own right).

Question 2

2.1. A year into the project, you give a successful demo to various users and stakeholders, including Simon Smartsuit.

Well done if you stopped and thought about the consequences of this before you started to answer the question. If you didn't, here's why you should have done. This scenario is much

later in the project than the ones you considered in the earlier tests. A year in you will be well into Construction, and what you demonstrated would be very substantial – it could easily be a complete working system ready for students to use. Furthermore, you will have had a year of continuous interaction with users and stakeholders (remember how you justified the User Support Officer at the beginning) so the requirements relating to what you demonstrated will be fairly stable. (Although you'll have a whole stack of additional requirements outstanding, many of which you hadn't dreamt of when you started). So if you even mentioned elaboration (or even worse inception) in your answer you were probably on the wrong track, and similarly if you made a big deal of requirements change (a brief mention relating to the diagrams was ok).

<Real world>
Giving a software demo in the real world has a completely different purpose from giving a demo of your third year project for example. In the latter case you are trying to get as many marks as possible for what you've done in the past. This means you show everything you can (well everything which works anyway). In the real world the purpose is to secure funding for the future. This means you will probably choose not to show everything you have, for a number of reasons:
-If management perceive that the project is "complete" they are likely to chop your budget, and if you're lucky your next project will have greatly reduced funding as they can see how much you can do for how little.
-You need to tailor your demo to the particular interests of the people you're demonstrating to, so it gets their attention and doesn't confuse them.
-The penalty for showing something which doesn't work is high, so you want to try hard to avoid that. (When something goes wrong in an important demo – even if it's a small thing – it's a horrible feeling).
-You may want to keep something back so if you're challenged you can pull it out.
Anyway, back to the exam…
</Real world>

He then asks for a copy of all your documentation for his records, which you provide. Unfortunately he's been reading up on the UP a bit and as usual has misunderstandings. You get the following email from him – how do you respond?

"Thanks for all the gen old boy, but surely there's a lot of stuff missing. You seem to have gone to a lot of effort to produce a comprehensive glossary, but otherwise there's not a lot there. There seem to be hardly any fully dressed use cases (Invigilate Exam and a few others). A lot of the class diagrams seem to be incomplete – some of them don't even show any operations and none of them seem to document the code in full. Also they're just photos of diagrams drawn on whiteboards - could I have the final versions, please? Oh and I'd particularly like to see your Vision document."

Your answer should be concise, while addressing all the points raised (the standard answer contains 10 sentences, each worth a mark). 8 marks

The standard answer was:

"Simon, we're using the UP in an agile way, which means we don't produce any more documentation than necessary.[1] That's why we had so much to demonstrate the other day. [1] The glossary is low cost and high value – it helps make sure we and the users are understanding each other correctly.[1] Producing fully dressed use cases takes a lot of time and we decided only to do this for situations where the user would have a fixed procedure to follow, as for invigilating exams.[1] The class diagrams without operations are domain models which again are quick to produce and help make sure our system fits with existing procedures. [1] We don't use class diagrams to document the code as that would be very expensive. [1] Also the two could easily got out of synch. [1] It would also be redundant since we use javadoc for that (you're welcome to see it but there's a lot of it.) [1] The diagrams you have are the final versions – there was no point taking a lot of time to make them look nice. [1] We don't have a Vision document – if you look back at our original funding proposal you'll see that it contains the same information. [1]"

Note that this is all about being agile, not about requirements change, although talking about the latter in moderation was ok. Some people made the very good point that drawing diagrams on a whiteboard was good for collaborative working. Almost nobody picked up on

the point of the diagrams not documenting the code in full. A lot of people promised a vision document – if you don't already have one there's no point producing one now (although there was one very good answer which said something like "you can have a vision document if you like but I'll have to take somebody off development work to do it – do you really want me to do that?") Worse, a number of people said the Vision document would only be available at the end of the project – if you're going to have a vision for a project it had better be at the beginning!!

2.3. Each of the following terms is ambiguous: suggest two or more alternative meanings for each, and note any other properties of these which should be included in a glossary.

2.3a. Exam paper 2 marks

This could mean either a question paper or an answer paper. Script is a common term for the latter.

<Real world>
This exact ambiguity caused us a problem with one of the other ABC exams this time round. Dr X had split up his exam so that he and another marker could mark in parallel, and then uploaded the marked fragments back to the server, where they were automatically recombined At this point he didn't know which buttons to press to download the final version, but he noticed that Dr Y had a lock which he thought was preventing him from doing this. The day before the marks were due in we got an anguished email "The exam is locked by Dr Y". In fact Dr Y had a lock on the question paper which was irrelevant to the downloading of the answer papesr. The latter happened as normal once I showed him which buttons to press.
</Real world>

2.3b Set of answers 3 marks

"This could be the set of answers for on student for one exam (a script) the set of answers for all students to a whole exam (this could be a script set; in the real project we call it an answer set) or the set of answers for all students to a particular question. A possible term for this is 'atomic answer set'."

There were a number of other reasonable suggestions here.

2.3c Marks 3 marks

"Marks allocated to a question vs. marks awarded to a particular answer. There are various constraints on these for instance marks awarded <= marks allocated. There is also an issue of units; in the real software we require that the marks are non-negative integers."

The last bit was an example of what I meant in the question by noting properties in the glossary.

2.4 Give two other examples from the domain of commonly used terms which need to be clarified in this way. 4 marks

"The classic example is "rubric", which in the US means a marking scheme. "question" and "answer" are not trivial as we need to distinguish between atomic questions/answers and composite ones. "examiner" covers a multitude of sins."

There were lots of other good examples given, but also quite a lot like "Question – could be a text question or an MCQ etc" The fact that there can be different kinds of a think doesn't make the thing ambiguous it just means it's an abstract thing (often, as in this case corresponding to an abstract superclass)

Question 3

3.1Give three tests which can be applied to determine whether a potential Use Case is a good one. In each case give an example from the exam software domain of a potential use case which would pass the test and one which would not.

" Boss test – does it describe something which a user could tell the Boss they had been doing

recently , e.g. Mark Exam but not Log On.
Size test – when the UC is written will it be substantial, or does it look like one or two steps?
Elementary Business Process (EBP) test. Does the UC correspond to a single business process which leaves the data involved in a consistent state. E.g. Set Test Conditions but not Examine Student"

Note that the Boss test and Size test are checking for UCs which are too small, while the EBP test checks for ones which are too big. Most of the answers got two ok but couldn't remember or correctly reproduce the third. There was also the option of the extra test I suggested:

"User Manual Test. Would the UC correspond to a non-trivial part of the user manual. E.g. Invigilate Exam but not Backup Answers."

But I don't think anybody used that test.

3.2Write the main success scenario of the Mark Exam use case in casual format: no more than 150 words, in paragraphs, not bullet points. You should assume that the examiner gets the data from the exam server at the beginning and returns the marks there at the end, but that the process of marking itself is done offline, using a marking tool like that you've seen in the course. Also assume that an exam may contain a combination of multiple choice, text and diagram questions. 10 marks.

"The examiner logs on to the exam server and is authenticated. He downloads the question paper, the complete set of student scripts and the standard answer paper from the server and loads them into the marking tool. He then logs off from the server.

The marking tool automatically marks the multiple choice questions and displays the structure of the exam to the examiner. For the remaining questions, the examiner works though them, marking the all answers to each individual part-question together. The examiner can review existing marks and change them – including the marks for MCQs – at any time. When all answers have been marked, the examiner requests a summary of all the marks, which is saved, as is the answer set containing the marks.

The examiner logs back on to the server, is authenticated, uploads the results, and logs off."

Note how I've tried to make it flow by combining related actions into one sentence. A lot of the answer read in a rather disjointed fashion with lots of short sentences.

Obviously the above makes a number of assumptions, and if you made other reasonable assumptions that was fine. I though this was quite difficult and I was surprised how many people did it. Generally the answers were pretty good, and of the right sort of length – only a few were penalised for being much too long.

A remarkable number of people had the marker manually marking MCQs. In any sensible system, either the answers are specified in advance, so marking is fully automatic, or the marker selects the correct answer once and the system does the rest. (We do both: the right answer is specified in advance, and marking is automatic. However, the marker can override this, if for example a question turns out with hindsight to be ambiguous he can give credit for an answer which was not the original "right" one – in which case this automatically applies to all instances of that answer). Even more remarkably a number of people had the marker adding up marks by hand!!

3.3. Suggest four important non-functional requirements related to this use case, in at least three "URPS+" categories. 4 marks.

"e.g.
Usability: the user interface should be optimised for efficient operation by experienced users (c.f. the simple-as-possible interface for students taking exams).
Usability: the user interface should be accessible to visually impaired examiners.
Reliability: the marking tool should be extremely reliable (there is no excuse for flakiness in a stand alone tool).
Performance: during marking, the next question should be presented to the examiner near-

instantly, even for large answer sets (some delay at startup is acceptable).
+ The marking procedures should be consistent with the university's quality assurance procedures."

A lot of people said that the UI should be simple to use as many markers are not technically aware. I gave a mark for this but the first point about is more relevant – many academics spend 100+ hours a year marking, and as a colleague once remarked "every hour of marking is an hour less of life". Generally the answers were sensible except that…

… a scary number of people disengaged brain at this point and on autopilot typed "The students' answers must be continually backed up as they are taking the exam…" This and similar things are irrelevant to the Mark Exam UC and got 0.

Question 4

4.1 What is the purpose of domain modelling? Your answer should include the relationship between domain classes and design classes, and the factors which influence the amount of effort worth spending on domain modelling. 5 marks

"Domain modelling both helps us understand the domain and suggests objects and interactions which may have close parallels in the software. Design classes are not the same as domain classes, but they are often motivated by them, and often have the same names and attributes (mapmaker principle). This lowered representational gap is what makes OO software development the One True Way The effort indicated in building a domain model depends on common-sense factors such as the size of the project, the degree of familiarity of the developers with the domain, and whether something similar already exists, always bearing in mind that you understand the domain less well than you think you do."

Answers were generally good, not surprising since this is bookwork which comes up every year. The main failing was not answering the last part of the question, or answering the different question "why is it worth spending time on domain modelling."

4.2. Consider the following description of the process of marking traditional paper exam scripts.

"The examiner is given a set of exam scripts and a mark sheet on which to record the marks. Each script contains the answers to the exam from one student, identified by a library card number on the front cover. The front cover also holds the real name of the student (hidden under a sticky flap), the title and course code of the exam, and a table for the examiner to enter the marks. The mark sheet has a row for each student, identified by library card number, with columns for the marks for each question and the total. The examiner marks all the answers for a complete question from all the students. Marks for each sub-question and the question total are written in the margins of the pages of the script. The question totals are transcribed onto the table on the front page of the script and onto the mark sheet. This process is repeated for each question, checking carefully that every page of the script has been inspected. The examiner adds up the marks on the mark sheet to give the totals. All the additions and transcriptions of marks are checked by clerical staff. Once checking is complete, they are entered into a central database."

Draw a class diagram to represent the significant domain classes suggested by this description and their important relationships and attributes. 7 marks

This is quite tricky. A good way to go about it is to look for actors, and the information which they manipulate, and what other concepts are important. The former are Examiner, ClericalStaff, and CentralDatabase (not Student – some people put Student in the middle as usual). Information is contained in ExamScript, FrontCover, MarkSheet and StudentID (or LibaryCardNumber which could just be an attribute). Other important concepts are Question (and hence QuestionPaper)and Answer. In my solution I also broke MarkSheet down:

MarkSheet 1 -> 1..* MarkShettRow -1 > 1..* Mark

So that's roughly the set of classes you should have had. There's rather a lot of them so if the diagram is to be clear it should have very little information about each. Common errors were:

•Not identifying enough classes a lot of solutions only had about three
•Abusing the notation – there is no inheritance in this example, so anybody who drew a fatarrow lost a mark for that.
•Not showing relationships between classes clearly.
•Worst of all, drawing a design class diagram cluttered with lots of attributes, types, operations etc. This makes no sense as were' describing a paper system. These solutions tended to include the other mistakes above too.

4.3 To what extent would these domain classes correspond to design classes? 4 marks.

Obviously this would depend on what classes you had. My standard solution said:

"Very well in general. Script, QuestionPaper, Question, Answer etc. will all have direct equivalents in the software. Otherwise it depends what the have in the model but e.g. if there is no StudentID in the model one is needed in the software, while there is probably no direct equivalent of FrontCover. CentralDatabase will be a whole separate software system, not a design class, of course. MarkSheet is interesting – whether such a notion appears in the software depends on whether we expect it to talk directly to the central database."

A specific answer, such as this was required to the specific question. Quite a lot of people got 0 because they waffled on about the relationship between domain classes and design classes in general.

4.4. Suggest ways in which the dynamic aspects of the process could be improved in the software implementation. 4 marks.

There are numerous possibilities, e.g.

"Marking all the answers to each part-question together (rather than a whole question).
No need to check for stray bits of answer in silly places
No need for multiple transcription and clerical checking.
Examiner can have additional options, eg. Keyword highlighting."

Again, a specific answer to the specific question was required, and in a number of cases not provided.

Question 5

In an early iteration you have decided to implement a cut-down version of Take Exam with two kinds of questions: text questions which are answered by typing text into a box, and composite questions which contain other questions (which themselves may be composite questions). You therefore have design classes TextQuestion, CompositeQuestion and correspondingly TextAnswer and CompositeAnswer. In addition you have classes to represent the complete documents, QuestionPaper and AnswerPaper.

5.1
According to the GRASP principles of Polymorphism and Protected Variations, two further classes should be added to this design. State what these classes are, why the resulting design conforms to these principles and what the advantages of doing this are. 6 marks.

"Abstract classes Question and Answer, with TextQestion and CompositeQuestion subclases of Question etc. – i.e. the Composite design pattern. Makesu se of Polymorphism to capture commonalities between classes and provide a single interface to code using these classes. Protects against variation because new question types can be added without affecting existing code. E.g. a getMarksAwarded() method would be abstract in Answer, return a number in textAnswer and recurse in CompositeAnswer."

Most people got this right. Some suggest a Paper class which abstracted QuestionPaper and AnswerPaper. This is a reasonable idea is missing the main point.

5.2

Based on the appropriate GRASP patterns, suggest which of these classes should implement

the following operations:

5.2a
String GetQestionText() – get the text for a question. 1 mark

“By Expert, Question, as all questions have text (not TextQuextion!)”

As predicted a lot of people said TextQuestion. A text question is defined above as one which is answered by typing text into a box. As far as the question text is concerned, there is no difference between a text question and any other question, the all have question text, including composite questions – such as question 5 here.

5.2b
String getAnswerText() – get the text of a particular student answer

“This time it is TextAnswer!”

A lot of people said Answer – wrong because not all answers are text – and various other classes which didn’t make sense.

5.2d
QuestionPaper constructQuestionPaper(String storedRepresantation)
Construct a question paper from the String representation in which is is stored on Disk. Suggest THREE options 3 marks

“The options are:
- Entirely within the QuestionPaper class (GRASP Creator principle)
- Distributed between the QuestonPaper class and the Question classes (GRASP polymorphism principle)
- Separately using a Factory (GoF factory pattern)”

Most people got the third of these and one or other of the others. The snag with the first is that the QuestionPaper class needs to know about all the classes which make up a paper. The snag with the second is that the logic gets very distributed. So the third is probably the best (although what we actually do is the second).

5.3. Explain the notion of a Controller, and the different types of controller, using the Client side of the application as an example. Suggest what sort of controller, if any, would be most appropriate here. 5 marks

“A controller is the first object beyond the GUI layer which controls system operations. A façade controller represents the overall “system” or “root object”, or a specialised physical device (not relevant here). We might call this ExamClient. This makes sense if we think of the client as a message handling system, communicating with the server in an asynchronous way. A use-case or session controller represents the control required to manage a use case. We might call this ExamController. This makes sense if we consider the client primarily as stepping through the sequence of steps for an exam. Either is plausible. Some sort of controller object is indicated as the possible sequences of operation are complex.”

The answers for this were generally good. As the standard solution suggests, you can make a case for either a façade controller or a UC controller, and indeed the answers were split about 50/50 on this. Either was fine so long as it was justified.

5.4. What additional classes, required to implement Take Exam are suggested by GRASP patterns? 4 marks.

“ e.g. Pure Fabrication could be applied to handling of messages between client and server, or reading data from persistent storage etc.

Polymorphism and Protected variations suggest UI counterparts to the Answer classes, e.g. AnswerDisplay, TextAnswerDisplay CompositeAnswerDisplay etc. “

Again note that the answer needs to be specific, in this case to the Take Exam UC. A Lot of

people suggested sensible classes which weren't directly related to the UC. Some answers talked about classes without relating them to the GRASP principles, and some did the opposite – didn't actually suggest any classes at all!

To summarise: the first rule of exam technique is to read the questions carefully.

Comments:

Question 1
Generally this question was well answered, but many marks were thrown away by simply not reading the question properly. In the early part of the question a specific scenario is described, and the question asks that this scenario be used to form a context for the remainder. Many answers ignored this, and instead gave general answers. The question specifically asks for components to be illustrated with diagrams -- again many marks were lost by not providing these diagrams. Overall, however, answers showed a good grasp of the principles.

Question 2
This question was generally well answered, with most marks being lost for lack of detail throughout.

Question 3
The main source of problems with this question was a confusion between regular and irregular hierarchical mechanisms; octrees are a regular form of spatial enumeration, bounding volumes are an irregular form, each with their pros and cons. The main 'novely' for binary space partitioning is the ability to preserve an 'in front of' / 'behind' relationship between components of the scene -- this was rarely mentioned. Also, in many cases candidates omitted the discussion on space / time complexity for each of the spatial enumeration techniques.

Question 4 (photon tracing)

This was generally well answered. A couple of points seemed to cause confusion.

1. There were some references to iteration towards a solution. This was a consequence of confusing radiosity, which does use iteration and convergence, with photon tracing which doesn't converge towards a solution. Instead, iteration is employed to loop over the photons during shooting, and also to loop over primary rays during the final ray tracing pass, but no convergence is involved.

2. The use of the 'n' closest photons to estimate the irradiance at a point was almost invariably described in terms of interpolation, or averaging. This was so common that I have realised that I probably didn't explain it clearly. It has not been penalised during marking. The point about estimation -- if using Russian Roulette, which almost everyone described -- is that each photon carries a more or less equal amount of energy, so we need to sum the energies to get the estimate. One can think of this like a Geiger counter -- we count the number of clicks (in this case photons) to get the strength of the signal. We do actually scale the energy associated with each photon according to its distance from the point of interest (e.g. using a Gaussian filter). It should be recalled that the estimate takes account of the area covered by the 'n' closest photons, which takes care of the energy density.

One other error occurred several times. The photon map stores irradiance, not radiance. The radiance (energy leaving a point) is computed by applying the BRDF to each photon and summing the results, suitably weighted, to evaluate the energy along the ray during ray tracing. (This is why the incoming direction is also stored for each photon -- it's needed for the BRDF.)

Overall average for this question was 54.3%, with a standard deviation of 18%.

Question 5 (modelling different materials)

There were some excellent answers to this, but also a significant number of very poor ones with low marks. Perhaps the commonest misunderstanding was the use of the BRDF for all incoming energy to compute the radiance exiting in a particular direction -- hence the integral. Quite a few suggested that the photon map stores the radiance (the outgoing energy), which is wrong. Most people managed to suggest (correctly) using bump mapping or displacement mapping, and also use of the BSSRDF and BTF. Many assumed that materials must be

diffuse, which is not true -- what about satin, for example? A common problem was that, having suggested the correct method, many answers lacked any explanation, or detail, so it was difficult to award many marks.

Overall average was low: 49.8%, with a standard deviation of 24.5% (high!) The latter resulted from too many weak answers.

The material for question 5 was taught early, while that for question 4 came in the final lectures. I wonder if this explains the different level of performance between the two? They shouldn't really be any different in terms of difficulty, but perhaps the photon tracing was fresher in people's memories? If true, that suggests a worrying lack of attention during revision. I received only a single email query about any of this material before the exam.

COMP 30151 Understanding Programming Languages Dave Lester

Comments:

The results this year are broadly in line with previous years' results. This is for two reasons: the conceptual difficulty with a non-While language was mercifully absent this year. This was counteracted by the allegedly unexpected appearence of an axiomatic semantics as the reference semantics. Overall, there were a gratifyingly large number of over 90% results -- including one 100% -- and a lower than expected number in the range 0-10%. As usual, well-prepared students did well, and those who didn't attend lectures did badly. I did notice that a relatively large numer of students did not attempt any part (c), thereby throwing away 25% of the potential marks available.

The questions in detail:

Q1: Matching a natural semantics to the axiomatic semantics proved tricky for many students. Presumably those who felt that -- despite all warnings -- selective revision would be adequate.
Q2 Generally well done.
Q3 Those that did this question did well.
Q4 Relatively few attepted this. Those that did, did well.
Q5 Actually the easiest question, but only two (out of a possible 49 attempts).

Comments:

Overall Average: 58%

Q1
Average: 56%

a) Almost everyone calculated the NA correctly. However, many rounded the answer to 0.3, stated this was the same as the fiber NA and hence all the light would propagate into the fiber efficiently. However, unfortunately Physics does not round numbers up, so in fact some light was lost!
b) Marks were generally lost for not identifying that the logic rules are different between the 1st and 2nd/3rd stages. A number of scripts discussed the wrong number system!
c) The marks were awarded for identifying rotation of polarisation AND magnetic material. A number of answers simply stated change in polarisation without stating how.
d) On the whole answered well.
e) The discussion of the operation of the device was limited in a number of cases. The main disadvantage when using the device to implement a logic device is the need for a holding beam to enable consistent logic levels.
f) On the whole answered well.
g) A lot of answers simply stated the range as 0-272, and didn't point out that it can also be -136 to +136 - I was generous and didn't mark this down! A number of answers didn't mod the result when calculating the negative representation, so stated 3 6 11, not 0 6 11.
h) No problems
i) No problems
j) On the whole Ok. The lens performs a Fourier Transform, not the 4-f system which contains lenses. I required a discussion of how the spatial frequency distribution varies for low spatial frequency and high spatial frequency images, I didn't want a discussion about how the properties of the actually image may differ!

Q2
Average: 60%

This was the most popular optional question. It was generally answered well, although some diagrams were very poor!
Some answers discussed the 3-level system in detail when discussing the stimulated emission process ... not necessary.
Marks were lost for not calculating the numerical answer correctly and in a large number of cases, not stating units (J or eV).

Q3
Average 63%
This question was a gift. Most was bookwork and required limited discussion. On the whole marks were lost for silly mistakes or missing information.

Q4
Average 60%

On the whole those who attempted all the question answered it well. There were a number of incorrect inverter implementations: the variable forms the control and the inputs are '1' and '0' respectively.
In some cases the discussion of the directional coupler was limited and didn't state the key points of operation, hence some marks were lost.

Q5
Average 56%

On the whole those who attempted all the elements of the question did well. Marks were generally lost for not arriving at the correct lens and laser selection in the numerical part of the question, or not checking all combinations. The discussion of the astigmatic focusing and 3-spot tracking techniques was limited in some cases.

COMP 30291 Digital Media Processing — Barry Cheetham

Comments:

COMP 30311 Advanced Databases — Goran Nenadic, Sandra Sampaio

Comments:

Question 1: About the use of SQL extensions and SQL embedded in Programming Languages for access to Relational Databases.
This was the most popular question in the exam as almost 100% of the students solved it. This topic was covered in detail for the period of 3 weeks (i.e. 6 lectures) and was explored in the coursework, where students had to solve a number of questions in practice using the Oracle DBMS.
The marks for this question were high, with an average of 60%. The students scored higher in the Bookwork Questions and in straight forward Problem Solving Questions. They scored significantly lower in the Analytical questions. I believe the mind set of students is to learn how to use a technology for solving the most common problems, rather than analysing the pros and cons of the technology when compared to alternatives. In addition, most part of the material that discusses the pros and cons of technologies, comparing them to alternatives, is described in the suggested material and discussed during the lectures, but are not in the lecture notes.

Question 2: About the use and implementation of Triggers in Relational DBMS.
This was also a popular question in the exam (about 60% of students solved it), probably because the students had the chance to implement a number of Triggers using PL/SQL blocks for solving exercises of the coursework.
Same remarks made for Question 1 apply in the case of Question 2.

Question 3: About technologies for access to Object-Oriented and Object-relational DBMS.
This was the least popular question (only 20% of students solved it), probably because of the limited number of lectures covering this topic and the small fraction of the coursework relating to this topic.
Same remarks made for Questions 1 and 2 apply in the case of Question 3.

Question 4: About access and implementation of XML Databases.
This was a very popular question (more than 80% of the students solved it). This topic was extensively covered in 4 lectures and was also well explored in their practical coursework.
Same remarks made for Questions 1 and 2 apply in the case of Question 4.

Q5 This question was taken by a third of students, with the average mark in the region of 60%, with few students having extremely low marks. Part (a) turned to be most difficult - while most students have articulated some of the main issues in content-based retrieval, many struggled to link these to the case study. There were very few discussions on why controlled vocabularies/ontologies may not be suitable for a social website. Part (b) was a success for many, in particular for XML design - there were occasional problems with modelling a choice of elements or assigning attributes to an element. Part (c) i.e. XPath quering was also well done - the main issue was referring to attributes.

COMP 30341 Model-Based Software Design — Andy Carpenter

Comments:

Question 1
a)A common mistake by students was the need to allow overdrafts on current accounts. It was expected that this would result in a constraint on savings accounts that prevented the balance from going negative; the specification did not require the addition of an overdraft attribute for current accounts. Several students also lost marks by not using containment to insure that there was a single root object, and because they did not use opposites when bi-direction associations were required.
b)The question asked for an evaluation of the usefulness of invariants and the style of mechanism used in Emfatic, several students described OCL and gave its history which was not what the question asked for.
c)The question asked for an evaluation of approaches to adding detail during the PIM to PSM transformation. Although the question asked for this evaluation to be illustrated by examples of the sort of implementation detail added, it did not just want information on the sorts of information added which is how several students answered the question. Several student also gave information about the process used for model-to-model transformations, again this was not answering the question.
Question 2
a)Too many of the student answers for this question gave generic answers that did not consider the specific case described in the question.
b)Again too many students failed to use details from the case described in their answers.
c)Answers often failed to identify that there are multiple aspects to a complete system (for example, concepts, UI and execution) and that different modelling notations are appropriate for these different aspects.
Question 3
a)In the problem described, there were two specific types of UI. Hence, can not simply transform to three PSMs as many students suggested; each UI technology needs its own PSM. The question asked for identification of the models and meta-models required, many students omitted to identify the meta-models needed. Several of the answers were generic and did not consider the details of the case described.
b)The question asked for reasons why features given in answers were required. Many students failed to give reasons why the features that they described were required.
c)A model-based tool is application just like any other application. Thus, a model-based approach to developing this tool will be the same as the model-based approach to develop any application. Many students failed to appreciate this in the answers that they gave.
Question 4
a)A meta-model that could be used by application developers during the production of their own models was required by this question. This meant that the classes in the answers needed to have a collection of attributes and a collection of relationships. Most answers did not provide this flexibility; the models given in answers dictated that application developers would have to use the particular attributes and relationships given in the answers.
b)Bookwork that was generally answered well. Some points were lost for not giving the purpose of each level
c)Several answers described what generic tooling was without discussing the proc and cons of it.

COMP 30351 Engineering Web Applications — Thierry Scheurer

Comments:

This exam has been very satisfactory. All registered students attended, and the results were excellent. The quality of answers from the best scripts were often impressive. There were no problems in the production of the paper and its delivery.

The lab work (seven programming exercises) also was commendable. The only problem was that one hour of lab a week, supported by demonstrators, was just sufficient. On the basis of this experience, THE COURSE SHOULD BE ALLOCATED TWO LAB HOURS A WEEK.

COMP 30421 Natural Language Engineering — Sophia Ananiadou

Comments: The exam results and coursework showed that the students understood most of the concepts of natural language processing and text mining. The exam results were overall above 55. Most students gained a good level of understanding.
Q4 on parsing was attempted by 2 students (out of 15) with moderate results. A distinct improvement over last year's results. Coursework: average of sixty plus.

COMP 37321 Software Engineering 3 — Chris Harrison

Comments: Overall: The average mark for this year's paper is similar to than in previous years. Fewer students tacked the question/part question on executable equational specifications than in previous years. A small number of students did not answer all of the parts of a question, in some cases for more than one question, obviously losing the associated marks. Some students provided very good answers indeed, and some of those students did so for all the answers they gave, and gained appropriately high marks, presumably due to diligent revision.