This chart gives the average and maximum achieved scores for each question:
Most questions were multiple choice (MCQ) and worth one point (Q1.2e ii and Q1.2e I were worth 2 points). Almost every question (and every objective question) was awarded full marks to at least one person (see the red bar). To determine the difficulty of a question, look at the average (the blue bar). The MCQs ranged from very easy to quite difficulty (e.g., see Q1.2f ii).

Q1.1b ii-iv were the questions that had the error (no class “E” in the diagram). Given some feedback that these were likely easy questions, we decided to award the marks (uniformly) rather than deleting them. Thus, all scores were improved by 3 marks.

Performance on the essays varied widely. Several (Q1.3a, Q1.3b, and Q1.4b) had no one scoring the maximum. Essay specific comments are listed below. (Recall, each essay was worth a possible 5 marks.)

Q1.1a (Average: 3.27(65%) Max: 5(100%))
Few answers actually provided scenarios that might illustrate the application of techniques. Even a reference to the exercises that were conducted in class, or the courseworks would have been sufficient here. However, most provided at least two techniques along with an explanation.

The key feature of the 20 questions technique is that the domain expert asks the questions.

The 3-card trick is primarily for extracting implicit knowledge -- selecting the "different" card can force us to consider features not previously thought of.

Q1.3a (Average: 1.32(26%) Max: 3(60%))
In general, this question was answered poorly. The skos:broader relation is intended to capture a range of semantic relationships, and the decision made in the language design was that a transitive broader relationship may not reflect the intention in some vocabularies. SKOS allows us to then provide vocabularies without the same level of ontological commitment required in OWL. A number of answers made reference to SEP triples -- this is a mechanism for handling transitivity within a language like ALC. The design pattern used here however is transitive reduction, providing a transitive superproperty of skos:broader, skos:broaderTransitive. This can then be used to query the transitive closure of the broader/narrower relations.

Q1.3b (Average: 1.61(32%) Max: 3(60%))
Again, overall poor performance on this question. Few answers described appropriate application scenarios. Example scenarios would include applications that provide navigation based on a conceptual model or applications based on indexing and retrieval, rather than requiring rich
intentional descriptions of the domain. The downside would be lack of inference or the ability to provide concept definitions.

**Q1.4b (Average: 1.68(34%) Max: 3(60%))**

The key factor here is that a lack of standardisation here makes things difficult. We're left with 1/ URI names 2/ HTTP URIs 4/ Providing links to other things. Without a standardised format for delivery of information, it's hard to see how one can effectively deliver 4/ without clients having to do a lot of additional work in interpreting the information supplied, which could result in inconsistencies in the way that's done. Note the change in wording of the "principles". Early versions just said standards, while later versions mandate RDF/SPARQL. A Linked Data Web that used something else (JSON?) would still work, so it's the use of a standard rather than the standard itself that's key here.

Several answers simply stated "yes it can" or "no it can't" without providing adequate discussion or justification.

For the following essays, the rough schema was: was 1 point for coherence/intelligibility, 2 points for basic facts, and 2 points for strength of argument.

**2.2a i (Average: 3.37(67%) Max: 5(100%))**

There were 5 English sentences, thus a *transliteration* should contain 5 axioms. If all the content came over I was generally lenient as long as there was some parallel structure. If you had too much more than the 5 axioms, you lost a point.

Note that there were *no properties* needed in the transliteration. "hasSex some Female" is much worse than "Female".

There were ABox statements! Both involving Sally. "Sally" should not have appeared in any TBox axiom.

Generally, most of the TBox statements are naturally read as equivalences (except for the explicit if-the!) I was generally generous unless there was radical inconsistency in the pattern or if the if-then was read as an equivalence.

Witches are things that burn i.e., can be burned, not which burn other things.
For this question, it was really important to state benefits for *both* runtime and development time.

It’s not enough to say what you can do in an ontology (e.g., define terms) without saying why it’s useful for a terminology (e.g., verifying our understanding of those terms, or inferring links so we don’t have to check them by hand). Similarly, it’s not enough to mention a cost of the use at runtime, but why it might be helpful.

In general, improved performance is *NOT* an advantage of using an ontology at runtime! On the contrary!

I was fairly generous about variants of the benefits of inference (e.g., consistency and "verification")

Post-coordination is a critical runtime benefit. However, discussions of the computational complexity vs. flexibility were worth a point.

"Being able to query the ontology" is not a specific benefit of using an ontology. The specific modalities of querying and the specific benefits they bring are required.

"Definition oriented development" is not, by itself, a benefit of using ontologies (over term oriented development). You had to *enumerate* the benefits.

"The open world assumption" is not, itself, a benefit, nor is it necessary for the benefits of using an ontology (for the most part).

---

You needed to understand that weak cognitive complexity roughly means "usability", and usability has many facets (e.g., readable vs. writable).

Undecidability is not the only way that computational complexity can be (significantly) worsened.

Greater expressivity doesn't necessarily mean that we can say what we want to say "in a way that's easier to understand". It might let us say things in ways that are very very hard to understand.

It was important to mention how expressivity affects cognitive complexity *via* affecting computational complexity.

Examples were required!
Getting both the positive and negative relationships was critical, but mere mention that there were both was not sufficient.

Semi-decidability does *not* mean "no reasoning services"! If so, there'd be no first order logic reasoners. It means, generally, reasoning services are less robust.

Syntactic sugar doesn't generally increase expressivity, but does decrease cognitive complexity.

Saying that "we need to keep a good balance" does not contribute to the answer of this question.

2.5 (Average: 2.32(46%) Max: 5(100%))

"Practical" does *not* mean merely decidable. You can have very impractical reasoners that implement decision procedures for OWL 2 DL.

Practical does not have to give best case performance on one's own problem set, but "good enough" performance.

Practical does not mean best case = worst case. (This could be HUGELY impractical!)

Unsound (or incomplete) reasoners might well give different results! (Esp. if you compare an unsound and an incomplete one.) In general, a practical OWL reasoner is sound *and* complete (e.g., Pellet, FaCT++, HermiT).

It is critical to acknowledge that the Worst Case Complexity of OWL 2 DL makes a *tractable* reasoner (i.e., one that performs well in all cases) impossible, hence the need to talk about *practicality*.

If all you said was that a practical reasoner provides answers in a "reasonable" amount of time for "your" inputs (and nothing else), you got 2 points. (You need a least why this would be challenging and how reasoners can be practical but die on each other's good input).