Programming Languages and Paradigms

Isak Karlsson, Tobias Fasth and Beatrice Åkerblom
(isak-kar | tobi-fas | beatrice)@dsv.su.se

December 21, 2011

Instructions

This exam is probably a lot of work. One reason is that the work is "unbounded" – there is no right answers and you can always go over your answer again. Knowing when to stop and what is "good enough" is an important quality. It is easy, but wrong, to think that quality is related to the time you spend on your answer.

You should use the literature, but you *may not copy* any answer, not discuss these questions with anyone else, or allow anyone to aid you in the answering of these questions. *Your answers must be products of your own mind.*

NOTE: The sum of your answers must not exceed 10 pages (LaTeX, article, 10pt) or 3500 words (whichever you prefer), *excluding* the bibliography. Divide this between your answers as you please. Don't exceed this limit!

There are no deliberate trick questions in this exam. However, in order to keep the questions reasonably short, we will assume the following:

1. We assume that you will read the questions with a critical mind. This means that there might be more to the question than what is explicitly stated. For example, if the question states "argue why X is better than Y", it might be the case that X is sometimes better than Y and sometimes the worse, depending on Z, or something else.
2. We assume that you understand that some of the questions have no single correct answer. A question can seldom be answered by a single statement. Rather, the questions are meant to be "thought provoking" – we hope that your knowledge and programming language skills will be visible in the *reasoning* in your answers.
3. We assume that you are able to determine if a question is very broad and make appropriate delimitations to be able to produce a sensible answer.

How to pass this exam

If you try to give the *correct answer* for each question, this exam is incredibly hard. Instead, aim for an answer that is *good enough*. While it may be hard

to determine what good enough is, it is much easier than covering all possible aspects. In your answers, we want *you* to think and to be able to make reasonable judgments and assumptions[1] – not give an answer that cannot possibly be questioned.

Core-dumping is not acceptable. A long answer with a lot of irrelevant remarks is worse than one that is short and to-the-point. Please note that we will also judge your ability to keep irrelevant information out of the answers.

A good strategy is to complete the exam the first or second day of the exam period, let it rest for an entire day, and then go back to polish your arguments. As you will most likely notice, this is a good strategy for any text you produce.

Be sure to properly reference all your sources. Also, keep a critical mind when searching for material.

Grading Every question is graded with the full grade scale. The final score is the average. The examiner may round up or down depending on the overall quality of the exam. For example, a strong E/D/B/C/A will count favourably in a "rounding situation".

To get a "pass" on a question, an absolute minimum requirement is that you answer all parts of the question. To score high, you should show that your view is well-informed and based on sound reasoning.

Handing In Hand in the electronic version of the exam in to Turnitin no later than 2012-01-09, 12:55.

First, go to `www.turnitin.com`

Create a user profile. You'll need the class ID and enrollment password to enroll in the class.

- **class ID:** 4657774
- **enrollment password:** iGNUcius

When you have created your profile, you just have to click on "PROP - Exam" and follow the instructions for submission of papers. Submit to the folder "Exam". You can choose file upload or cut & paste. Click "training" if you find the instructions unclear.

If you hand in late, expect it to negatively influence your score. The handed in file should be a pdf file, contain your name and be named after you, e.g. if Tobias were to hand the exam in, he would submit a file called *tobiasFasth.pdf*.

Hand in a paper version of your exam at the seminar 8, 2012-01-09, 13.00.

Good luck!!

Beatrice, Isak & Tobias

[1] By this, we implicitly say that answers produced by quoting extensively from different sources without criticizing, comparing, judging, etc. are not acceptable.

Questions

1. Paradigm? **(15%)**

What is a programming paradigm? Give four examples and explain what the differences beween them are. Do some of the paradigms overlap? Give examples of languages that can be sorted into each paradigm and explain why they should be categorised that way. Are there languages that can fit into several paradigms? Is that important for our understanding of what a paradigm is?

2. Referential Transparency **(30%)**

Referential transparency is considered an important property of a functional programming language. What is meant by this term, why is it important and what are the possible benefits from using it? Should it be a goal for functional programs to maintain this property in all parts? Motivate your answers and support your argumentation with examples.

Could referential tranparency be useful in the context of other programming paradigms, e.g. imperative or object-oriented? What are the possible benefits from using it? Would there be difficulties maintaining it? Illustrate your answer with examples from both an imperative or object-oriented language and a functional language with which you are familiar.

3. Logic? **(15%)**

Logic programming is based on some fundamental abstractions, which are these abstractions and how does the implementation of a practical logic programming language handle those abstractions? Do you think that these abstractions could be made more useful in a mixed paradigm language, e.g. combining logic and functional programming? Argue for and against this combination.

4. Class-based vs. Prototype-based **(15%)**

Explain the differences between class-based and prototype-based object-orientation. Use examples to illustrate your description. Which approach gives a more object-oriented language? Motivate and argue for your answer. You probably also need to provide your own definition of object-orientation to make the argumentation better.

5. Choosing Between Languages? **(25%)**

The software company you work for currently uses an object oriented (imperative core) programming language for its systems development. It is now considering moving to a functional programming language and you are asked to investigate the potential effects of the change. In your answer you should consider the possible benefits, overheads and disadvantages of making this change. Illustrate your answer with appropriate examples.