Coláiste na Tríonóide, Baile Átha Cliath
Trinity College Dublin
Ollscoil Átha Cliath | The University of Dublin

Faculty of Engineering, Mathematics and Science

School of Computer Science & Statistics

Integrated Computer Science
Computer Science and Business
Year 4 Annual Examination

Hilary Term 2018

CS4012: Topics in Functional Programming

Wednesday, 3rd January Goldsmith Hall 14:00–16:00

Mr. Glenn Strong

Instructions to Candidates

You may not start this examination until you are instructed to do so by the Invigilator.

Attempt **three** questions.
(all questions are worth equal marks)

Q 1. Domain Specific Languages

(i) A domain specific language "RPN" based on a reverse-polish notation calculator has these operations (which only work on integers):

- Stack operations: push, pop, clear
- Arithmetic operations: Addition, subtraction, multiplication, (integer) division (each works on topmost two stack elements, and places the result on the stack).

Define a Haskell datatype `RPN a`, and make it an instance of the Monad class. Include functions which implement the full RPN calculator language shown above. Calculations can produce a numeric answer *or* they may fail with an error message (for example if an attempt is made to pop from an empty stack, or to divide by zero).

60 Marks

(ii) Explain the terms *deep* or *shallow* embedding with respect to Domain Specific Languages and explain which applies to your implementation of the RPN calculator.

15 Marks

(iii) A DSL shallow embedding "allows transformations like optimizations before translating to the target language". Explain what is meant by this statement and how we might take advantage of it in an implementation (you may refer to your RPN monad if it is appropriate, but do not be limited to discussing only that).

25 marks

Q 2. (i) Explain each of the following terms as it is used in Haskell, and comment on what each can be used for:

i. Existential type

ii. Generalised Algebraic Data Type (GADT)

iii. Type Kind

39 Marks

(ii) Consider a simple command language containing two IO actions (named "peek", and "poke") which can be used to model direct memory read and write operations through pointers.

Explain the principle by which GADT's could be used to define a "type-safe" implementation of these actions in which the compiler tracks the types associated with memory locations (it is not necessary to give an implementation for "peek" and "poke" in your answer, but you should provide suitable type signatures).

26 Marks

(iii) It is sometimes said that GADTs allow Haskell to simulate some features of dependent types. Explain what is meant by this statement, and include an explanation of what is meant by the term "dependent type".

35 Marks

Q 3. (i) What problem do Monad Transformers solve? Include in your answer a discussion of what issues might arise with an "IO" Transformer.

25 Marks

(ii) Give an implementation for the State Transformer Monad (that is, give the code for a monad transformer that can add stateful 'get' and 'set' operations to any monad).

45 Marks

(iii) What is the role of the Identity Monad in the context of monad transformers?

15 Marks

(iv) What does the "lift" function do in the Monad Transformer class?

15 Marks

Q 4. (i) The standard Haskell libraries offer both Parallel (through 'par' and 'pseq') and Concurrent (through 'forkIO') approaches to programming. Explain how these approaches differ semantically, and state when a programmer might want to take advantage of either approach.

30 Marks

(ii) Explain how `MVar` values are used in Haskell to enable communication and locking between threads.

20 Marks

(iii) Software Transactional Memory (STM) provides an alternative to lock-based communication.

i. What does it mean in STM for a block to be atomic?

15 Marks

ii. What is the effect of the `retry` function in the STM monad?

20 Marks

iii. What mechanism is used to prevent IO activity within STM atomic blocks, and why is this important?

15 Marks