Concepts in Programming Languages
Supervision 2 – 18/19 v1

Supervisor: Joe Isaacs (josi2).

All work should be submitted in **PDF** form 24 hours before the supervision to the email `josi2@cam.ac.uk`, ideally written in LaTeX. If you have any questions on the course please include these at the top of the supervision work and we can talk about them in the supervision. **Note:** there is a revision guide `http://www.cl.cam.ac.uk/teaching/1718/ConceptsPL/RG.pdf` for this course.

1. What is duck typing, what benefits does it provide and why? Why does this integrate well with a prototype based programming language.

2. What are value and references types, give an example of each then compare the properties of both.

3. What is the motivation behind gradual typing?

4. When does an ML modules signatures match a strucutre?

5. In what ways do programming languages present parallelism to the programmer, talk about data level parallelism all the way up to cloud computing, do we need different constructs for different scales of parallelism.

6. What is a monad, define them in two different ways and show how they can use used to define the other.

7. Write a ML style module for monads include a `sig` and a few choice implementations `structs`. Define a signature for both monad definitions as in the questions above, then write a functor taking a module of the less common monad signature and convert it to the more common monad interface (the one used in Haskell).

8. Why are monads used in programming languages and what do they achieve? Why not allow functions with side effects, Haskell could be used to provide a motivation.

9. Download GHC, and write a simple program using monads, which you think is interesting, using both bind and return.

10. What are GADTs and could they be used write programs without a certain class of bugs. How could one write a `head` function which doesn't error or throw an exception when applied to an empty list, you need no write a program just an idea of how it could be done.