Concepts in Programming Languages
Supervision 1 – 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 the different between a static and dynamic typing. Give with justification which languages and static or dynamic.
2. What is the different between strong and weak typing.
3. What is type safety. How are type safety and type soundness distinct?
4. What type system does LISP have what are the basic types?, where have you seen a similar type system before.
5. What is a abstract machine, why are they useful? Where have you seen or used them before?
6. Name at least six different function calling schemes and which languages use each of them.
7. Give an overview of block-structured languages, include which ideas were first introduced in a programming language and how they influenced each other. Making sure to list the main characteristics of each language. You should talk about *three* distinct languages, then for each language make sure to make distinctions between different version of each language. **Make a timeline using pen and paper or a diagram tool**.
8. Using the type inference algorithm given in the notes (a typing checker with **constraints**) find the most general type for

$$\textbf{fn}\ x \Rightarrow \textbf{fn}\ y \Rightarrow y\ x$$

9. What are the main principles of OO languages.
10. What is the different between subclassing and subtyping?
11. There is a trade off in terms of extensibility between functional and OO languages, Describe this trade off. How could this trade off be addressed in a functional and OO languages, give an example? Maybe consider representing a grammar $e ::= n \mid e + e$
12. Why can variants cause an weak type system. How could this be addressed.
13. What possible programs could be given this type `'a -> 'b` when using a Hindley–Milner like type inference algorithm.
14. Is Java's type system sound? If not give an example of where is falls down.
15. How is type equality handed in C and SML. Talk about why these notions of equally are useful. Lets say that we want to express physical quantities in our program in a type safe manor but the program must handle both SI Units and Imperial Units. *Without getting them mixed up.*

16. How is abstraction handed in in an OOP language and a functional language?
17. What features of a type system do we want for a functional vs a OO language give examples of both.
18. What does it mean for a type system to be safe. What languages have safe and unsafe type systems.
19. What is your favorite programming language(s), why is it? Consider the type system, the goals of the languages, and any other positive and negative features. Try to relate it to this course.