Programming Languages and Paradigms
COMP 302

McGill University, Winter 2018

Course Details

Time: Tuesday, Thursday, 13:05–14:25
Place: ENGMC 204 (and temporary overflow in Birks 205)

Instructor: Professor Clark Verbrugge
Office: McConnell, room 230
Office hours: Tuesday 14:30–16:00, Friday 10:00–11:30, or by appointment.
Phone: 514 398-2411
Email: `clump@cs.mcgill.ca`

Teaching Assistants: TBA

Email, Website

McGill's MyCourses will be used for course announcements, to manage assignments and for online discussions. Students are expected to monitor their McGill email account for course-related news and information. The external course website is: `http://www.sable.mcgill.ca/~clump/comp302`

Pre-requisites

- COMP 250 (Introduction to Computer Science)

Note: students registering without the pre-requisite may find the course removed from their transcript by their Faculty. Please consult the instructor if you do not have all the pre-requisites.

Textbook

There is no required text for this course. The following texts, however, provide useful additional information on many of the topics covered in class:

- *Scala for the Impatient 2nd Edition,* Cay S. Horstmann, 2017. We will be using Scala as a base language in this course. This text provides a good, relatively quick introduction/reference for the Scala language.
- *Programming in Scala 3rd Edition,* Martin Odersky, 2016. A different Scala reference; more comprehensive, but also less gentle. You really don't need both this and the above text; pick one, and if undecided, go with the Horstmann text above.
- *Types and Programming Languages,* Benjamin C. Pierce, 2002. A classic reference, somewhat expensive and covering much more than we will get to in-class, but useful if you find type systems interesting. You can also find earlier, draft versions of the text on the web.

Description

This course is designed to provide a thorough introduction to the foundations and paradigms of programming languages. Theoretical and practical aspects of language design and programming practice will be examined in a functional programming context. Topics additionally include basics of programming language design and construction, binding and scoping, parameter passing, lambda abstraction, data abstraction, and type checking.

Evaluation

4 Assignments:	40%	
Online quizzes:	5%	
Exam:	55%	(open-book)

A supplemental exam (55%) will be held if required. In accord with McGill University's Charter of Students' Rights, students in this course have the right to submit in English or in French any written work that is to be graded.

Assignment and Exam Policy: Assignments must be submitted on time. Late assignments will only be accepted in highly-exceptional circumstances and only with **written** permission of the instructor. This typically is only granted for documented medical reasons, and in particular your workload in other courses is **not** sufficient reason for an extension. No assignment submissions will be accepted after marked assignments have been returned, or after solutions have been discussed in class.
Students are expected to monitor their returned work. Corrections to assignment grades will not be made later than 2 weeks after an assignment grade is given out.
McGill University values academic integrity. Therefore all students must understand the meaning and consequences of cheating, plagiarism and other academic offenses under the Code of Student Conduct and Disciplinary Procedures (see `http://www.mcgill.ca/integrity/` for more information).
More specifically, all work in this course is individual work: **work submitted for this course must represent your own efforts.** Copying assignments or tests, or allowing others to copy your work, will not be tolerated. Note that introducing syntactic changes into a copied program or assignment is still considered plagiarism.

Assignments

Expected assignment distribution dates and due dates are listed below. Note that this is mainly to help you in general planning; topic descriptions are vague and non-exhaustive, and both the topic and the associated dates may change. Be sure to consult MyCourses for final, official due dates.

Assig.	Main Topic	Available	Due
1	Functional programming	Monday, January 22	Monday, February 5
2	Language construction	Monday, February 12	Monday, February 26
3	Language evaluation	Monday, February 26	Monday, March 19
4	Advanced construction, typing	Monday, March 26	Monday, April 9

Quizzes

There will be 2 online quizzes (within MyCourses). You will be given 1 week notice for each, and have 1 week to complete it once it is available.

Course Content

Lecture dates are approximate and topics will likely shift and/or span lectures.
You will achieve best results by attending class and taking notes. Formal notes are not provided; consult the recorded lectures, or obtain notes from friends, acquaintances, or friendly strangers.
Appropriate readings from Horstmann's book are indicated below as chapter.section entries; ones starting with **P** indicate topics found within the Pierce text. Note that class lectures are insufficient to fully cover Scala, and so some Scala resource is strongly recommended, whether from the Horstmann text, Odersky text (find chapter/sections yourself), or the multitude of online sources. Pierce readings are completely optional, and class lectures are sufficient for that material.

Lec#	DoW	Date	Topic	Textbook Readings (Horstmann)
1	**Tuesday**	**January 9**	Introduction, Scala language	**1.1–1.5**
2	**Thursday**	**January 11**	**No class!**	
3	**Tuesday**	**January 16**	Scala introduction	**2.1–2.10, 3.1–3.3, 11.1–11.4**
4	**Thursday**	**January 18**	Functional languages	**12.1–12.4, 14.1–14.5**
5	**Tuesday**	**January 23**	Higher order functions	**4.1–4.8, 12.5, 13.8–13.11**
6	**Thursday**	**January 25**	Scopes and environments	
7	**Tuesday**	**January 30**	Scopes and environments	**12.6, 12.8**
8	**Thursday**	**February 1**	Constructing a language	**9.10–9.11**
9	**Tuesday**	**February 6**	Parsing	**20.1–20.7**
10	**Thursday**	**February 8**	Example language: WML	
11	**Tuesday**	**February 13**	Control flow	**11.5–11.6**
12	**Thursday**	**February 15**	Functions and parameters	
13	**Tuesday**	**February 20**	Evaluation	**2.11**
14	**Thursday**	**February 22**	Lazy evaluation	**2.11, 13.12–13.13**
15	**Tuesday**	**February 27**	Continuation passing style	
16	**Thursday**	**March 1**	Lambda calculus	**P5**
17	**Tuesday**	**March 13**	Lambda calculus	
18	**Thursday**	**March 15**	Type systems	**8.1–8.11, 14.9, P1, P8, P11**
19	**Tuesday**	**March 20**	Type systems	**18.1–18.11, P22.1, P22.5**
20	**Thursday**	**March 22**	Type checking	**P9**
21	**Tuesday**	**March 27**	Type checking and inference	**P22**
22	**Thursday**	**March 29**	Unification	
23	**Tuesday**	**April 3**	Operational semantics	
24	**Thursday**	**April 5**	Operational semantics	
25	**Tuesday**	**April 10**	Declarative languages	
26	**Thursday**	**April 12**	Esoteric languages	