COMP 202 - Foundations of Programming

McGill University, Winter 2019

1 Course Details

Section 2	Instructor:	Elizabeth Patitsas
	Office:	McConnell Engineering Building (MC) 231
	Office hours:	Wednesday 11:30–13:30 or by appointment
	Contact info:	`elizabeth.patitsas@mcgill.ca`
	Lecture room:	MCMED 522
	Class times:	MWF 10:35–11:25
Section 3	Instructor:	Giulia Alberini
	Office:	McConnell Engineering Building (MC) 233
	Office hours:	Monday 4pm–6pm in McConnell 103 or by appointment
	Contact info:	`giulia.alberini@mcgill.ca`
	Lecture room:	STBIO S1/4
	Class times:	MWF 9:35–10:25

1.1 myCourses Webpage

- myCourses: `http://www.mcgill.ca/lms/`

1.2 Teaching Assistants (T.A.)

TAs will be available for office hours, on the third floor of the Trottier building room 3110, to help you with your assignments and answer questions about the course material. You can also contact TAs by e-mail. Each TA's office hours and e-mail address will be posted on myCourses.

1.3 Contacting Instructors and Teaching Assistants

Post all your questions about the course on the myCourses message boards (including about assignments and the midterm/final) so that everyone can see both the questions and the answers. You may freely answer other students' questions as well, with one important exception: you may not provide solution code (although you are permitted to provide one or two lines of code to illustrate a point).

For private matters only, you can send e-mail to a teaching assistant or instructor directly with "COMP 202" in the subject header.

Students are expected to monitor both their McGill e-mail account and myCourses for course-related news and information.

2 Course Description

Welcome to COMP-202! Please read this document carefully and keep it for reference throughout the term.

This course introduces students to computer programming and is intended for those with little or no background in the subject. No knowledge of computer science in general is necessary or expected. On the other hand, basic computer skills such as browsing the Web, sending e-mail, creating documents with a word processor, and other such fundamental tasks will be necessary in this course.

The course uses the Python programming language. Python is an example of a programming language (as are Java, C++, and many others). A large part of this course will focus on the basic building blocks of programming, which provide the foundations to learning other languages such as Java or C++.

Learning how to program is not easy; it is not a set of facts that one can simply memorize. In principle, a computer program is simply a set of instructions that tells a computer to perform a task. However, finding the right set of instructions can be quite challenging. For that, one has to learn how to structure a larger problem into small subsets, and then find the solution to each particular subset. This course aims to teach students *a way of thinking* that will enable them to build non-trivial programs.

2.1 Primary Learning Objectives

By the end of this course, you will be able to:

- Design and describe precise, unambiguous instructions that can be used to solve a problem or perform a task;
- Translate these instructions into a language that a computer can understand (Python);
- Write programs that solve complex problems by decomposing them into simpler subproblems;
- Apply programming-style conventions to make your programs easy to understand, debug and modify;
- Learn independently about new programming-language features and libraries, as you encounter them, by reading documentation and by experimenting.

2.2 What This Course is *Not* About

This course is not about how to use a computer. It will not teach you how to send e-mail, browse the Web, create word processing documents or spreadsheets, set-up and configure a computer, use specific software applications (except those needed to complete coursework), design Web pages, or deal with operating system or hardware problems. However, the course offers introductory tutorials that provide instruction in aspects of computer usage necessary to complete coursework.

2.3 Course Prerequisites

- A CEGEP-level mathematics course or equivalent. For students who did not attend CEGEP, any upper-level mathematics course is sufficient. However, attention to detail, rigour, and the ability to think in an abstract manner is much more important than knowledge of calculus, algebra, or trigonometry.

2.4 Recommended Textbook:

- *Think Python 2e*, by Allen B. Downey.

  Available at no cost under the terms of the Creative Commons Attribution-NonCommercial 3.0 Unported License at:

  `https://greenteapress.com/wp/think-python-2e/`

2.5 Other References

- *How to Think Like a Computer Scientist: Interactive Edition*, by Brad Miller and David Ranum. This is an interactive textbook based on the work by Jeffrey Elkner, Allen B. Downey, and Chris Meyers. Available at no cost at:

  `https://runestone.academy/runestone/books/published/thinkcspy/index.html`

- *The Python Tutorial*. You can browse or download this from Python Software Foundation's Web site.
  - `https://docs.python.org/3/tutorial/`
- *The Python 3 Documentation*. You can also browse or download this from Python Software Foundation's Web site.
  - `https://docs.python.org/3/`

3 Lecture Recordings

We will record both sections, and make the recordings available to all students on mycourses. Note that the two sections will be merged on mycourses, and they will be treated as one single course (same assignments, same exams).

4 Courses Discussion Board Rules

Please help out by answering each other's questions on the Discussion Board. The instructorsand TAs will try to moderate the Discussion Board. But the Discussion Board works best when students help each other out. When posting to the Discussion Board, please obey the following. *Posting that do not conform may be deleted*.

- Choose the appropriate folder (Topic).
- Use the search feature to see if your question has been asked before.
- Choose a suitable subject line, so that readers know what the posting is about.
- If you have multiple questions that are unrelated, then use multiple postings so people can more easily follow the thread.
- Proofread before posting. Take an extra minute to ensure that what you write makes sense.
- If you would like your posting to be deleted, just add a request within the thread.
- It is nice for you to post a thank you note on mycourses when someone helps you out. However, please keep in mind that everyone subscribed gets notified, which can be a bit annoying if everyone sends them. So please use some discretion.

5 Grading Scheme and Deadline Policy

Your final grade in the course is calculated by taking the *maximum* of the following two options:

- **Assignments:** 35%
- **Midterm Examination:** 20%
- **Final Examination:** 45%

OR

- **Assignments:** 35%
- **Final Examination:** 65%

This means that students who perform better on the final than on the midterm exam will have the (automatic) option to make their grading scheme 32% assignments, and 65% final. However, the assignments are a key part of learning the material, and as such there is no 100 % final option.

When we calculate your final course grade, we will use a formula that rounds off to the nearest integer. If your grade is 84.4 then it rounds to 84 and you get an A-, whereas if it is 84.6 then it rounds to 85 and you get an A. If your grade is 84.5, our formula will round it up to 85. The same round off procedure holds for low grades. If your calculated final course grade is 49.4 then it rounds to 49 which is an F. We draw a very a hard line on this, so if you don't want to fail then you should stay far away from that line.

Students who receive unsatisfactory final grades will **NOT** have the option to submit additional work in order to improve their grades.

Official language policy for graded work: In accordance 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.

5.1 Midterm Examination

The midterm examination will take place in the evening at the following date and time:

- Tuesday, October 22nd, 6:00pm - 9:00pm

The room assignments will be announced in class and posted on myCourses when it is closer to the date.

5.2 Assignments

There will be **four** assignments consisting of writing Python programs. It is *very important* that you complete all assignments, as this is the best way to learn the material. By working hard on the assignments, you will gain essential experience needed to solve problems on the midterm and final examinations.

To receive full grades, assignments (as well as all other course work) **MUST** represent your own personal efforts (see the section on Plagiarism Policy and Assignments below).

Late Policy: Late assignments will be deducted 10% each day or fraction thereof for which they are late, including weekend days and holidays; that is, assignments that are between 0 and 24 hours late will be deducted 10 points, assignments that are between 24 and 48 hours late will be deducted 20 points, and so on. Assignments submitted more than 2 days after the deadline will not be accepted, nor graded, and will therefore receive a grade of 0.

Assignment submission will always take place on myCourses. Instructors and TAs will discuss how to use it during the lectures and tutorials, but every student is responsible for verifying that their submissions are successful. If you believe the content of your myCourses submission box is different from what you have

submitted, you must e-mail your section instructor within **5 days** of the assignment deadline in question to provide evidence of your correct submission.

Assignment marks will also be posted on myCourses. It is your responsibility to check that the marks are correct and to notify your section instructor of any errors or missing marks. If you believe that your assignment was graded incorrectly, you should first email the TA who marked your assignment. Their email should be in the feedback left on your assignment. If you and the TA cannot resolve the discussion, then you should contact your instructor.

The instructors reserve the right to modify the lateness policy for a particular assignment; any such modifications will be clearly indicated at the beginning of the relevant assignment specifications. **Plan appropriately and do not submit to myCourses only minutes before the assignment deadline**. Take care, programming assignments are notoriously time-consuming and individual exceptions to the lateness policy will not be granted without appropriate justification submitted in writing and supported by documentary evidence.

5.3 Supplemental/Deferred Exam

In exceptional situations, students may write a supplemental examination. However, ability to do so is not automatic, and depends on your exact situation; contact your Student Affairs Office for further information.

The Supplemental/Deferred exam will cover the same material as the Final Exam and it will replace the Final Exam grade, with the same grading policy as stated at the beginning of this section. For information on Supplemental Exams, see
`https://www.mcgill.ca/science/student/general/exams/supplemental`.

6 Land Acknowledgment

McGill University is on land which has long served as a site of meeting and exchange amongst Indigenous peoples, including the Haudenosaunee and Anishinabeg nations. We acknowledge and thank the diverse Indigenous people whose footsteps have marked this territory on which peoples of the world now gather.

7 Accommodations

7.1 Office for Students with Disabilities

If you have a disability and require accommodations, the Office for Students With Disabilities (`https://www.mcgill.ca/osd/`) is here to help you sort those out. OSD liaises with us (the instructors) on your behalf to ensure that your accommodations are met.

7.2 Pregnancy and Caregiving

Students who are pregnant and/or caring for a dependent also often may find it helpful to receive academic accommodations. McGill's guidelines for accommodations for students who are pregnant and/or caring for a dependent may be found at `https://www.mcgill.ca/study/2018-2019/university_regulations_and_resources/graduate/gi_accommodation_pregnancy_caring_dependants`

8 Scent Free Environment

This classroom and associated office hours are a scent free environment. Please refrain from wearing perfume, cologne and body spray in these spaces out of respect for people with neurological respiratory issues who may be affected by these scents.

9 Campus Computer Laboratories

Using the SOCS computer laboratory facilities: All students registered in COMP-202 may use the SOCS computer laboratory facilities to do their work regardless of the program in which they are registered. These facilities are located on the third floor of the Trottier building.

Refer to `https://www.cs.mcgill.ca/about/facilities/` for more information on the SOCS computer laboratory facilities.

Other computer laboratory facilities: You may also use other computer laboratory facilities on campus to do your work. Most facilities are available to all McGill students, but there are facilities which grant usage privileges only to students registered in a course or program offered by the faculty or department which manages the facility.

Students should contact the work area of their choice to inquire about access requirements, opening hours, or any further information such as software availability.

10 Required Software

We will be using Anaconda, which bundles together many different Python libraries and related software. This means you only have one thing to install! :)

You can install Anaconda through going to `https://www.anaconda.com/distribution/` and following the instructions to install Anaconda for **Python 3.7**. Anaconda supports Windows, Mac and Linux.

Typically when programmers write code they used what is called an integrated development environment (IDE) to write programs. IDEs provide an editor that allows you to type your program, commands to compile and run it, and many other useful tools, all in one application.

Anaconda comes with an IDE caled IDLE, which is what we recommend for writing your Python programs with. There are many others out there, so if you prefer another IDE (such as Spyder or PyCharm) or using a text editor (such as Atom or Gedit) you are welcome to do so. Note that if you use a different editor, the teaching staff may not be familiar with your choice of editor.

11 Plagiarism Policy

Official policy: 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 `www.mcgill.ca/integrity/` for more information).

11.1 Plagiarism Policy and Assignments

You must include your name and McGill ID number at the top of each source code file that you implement and submit. By doing so, you are certifying that the program or module is entirely your own, and represents only the result of your own efforts.

Work submitted for this course must represent your own efforts. Assignments **must** be done **individually**; you **must not** work in groups. Do not rely on friends or tutors to do your work for you. You **must not** copy any other person's work in any manner (electronically or otherwise), even if this work is in the public domain or you have permission from its author to use it and/or modify it in your own work (obviously, this prohibition does not apply to source code supplied by instructors explicitly for this purpose). Furthermore, you **must not** give a copy of your work to any other person.

The plagiarism policy is not meant to discourage interaction or discussion among students. You are encouraged to discuss assignment questions with instructors, TAs, and your fellow students. However, there is a difference between discussing ideas and working in groups or copying someone else's solution. A good rule of thumb is that when you discuss assignments with your fellow students, you should not leave the discussion with written notes. Also, when you write your solution to an assignment, you should do it on your own.

11.2 Gilligan's Island Rule

You are free to meet with fellow students(s) and discuss an assignment with them. Writing on a board or shared piece of paper during the meeting is acceptable; however, you should not take any written (electronic or otherwise) record away from the meeting. Everything that you derive from the collaboration should be in your head.

After the meeting, you must engage in at least a half-hour of mind-numbing activity (like watching an episode of the television show *Gilligan's Island*), before starting to work on the assignment. This will assure that you are able to reconstruct what you learned from the meeting by yourself.

11.3 Getting Help and Partial Credit

Students who require assistance with their assignments should see a TA or instructor during their office hours. If you have only partially finished an assignment, **document the parts that do not work**, and submit what you managed to complete for partial credit.

11.4 Plagiarism Detection

We will be using automated software similarity detection tools to compare your assignment submissions to that of all other students registered in the course, and these tools are very effective at what they have been designed for. However, note that the main use of these tools is to determine which submissions should be manually checked for similarity by an instructor or TA; we will not accuse anyone of copying or working in groups based solely on the output of these tools.

You may also be asked to present and explain your assignment submissions to an instructor at any time.

Students who put their name on any code that are not entirely their own work will be referred to the appropriate university official who will assess the need for disciplinary action.

12 Course Content

Note that minor changes in content, reading material, and times for tutorials and assignments may occur. It is your responsibility to attend class and be aware of what content is being covered.

12.1 Tutorials

Throughout the term, there will be several (optional) tutorials. These will be designed to help you with the material and assignments, and to give you a chance to ask questions in a smaller environment than lectures. Further information will be posted on myCourses. It is not necessary to register for tutorials.

The tutorials will review and practice material presented in class. For example, a tutorial in the fifth week might cover the `while` and `for` statements to ensure that everyone is caught up. As well, three special tutorials will be provided:

Tutorial	Title	Contents
T0	Basics of Course Software Tools	Anaconda: download, install the appropriate software, learn how to use the console, and how to run your first program.
TM	Midterm Review	Review of all material for the midterm.
TF	Final Exam Review	Review of all material for the final.

12.2 Approximate Schedule of Topics

The references to chapters in the table below are from the recommended textbook (`https://greenteapress.com/wp/think-python-2e`). Although our lectures will not follow the textbook exactly, especially later in the semester, reading the textbook is highly recommended. **The following schedule is only approximate and may/will change depending on how the semester unfolds.**

	Week	Topics	Reference	Events
Intro	1– Sept 1	What is programming? What is computer science? How does a computer work? Binary numbers	Chapter 1	
Fundamentals	2–Sept 8	Basic Python programs Variables and types Expressions and mathematical operators	Chapter 2	
	3–Sept 15	Input arguments MatPlotLib: read in csv, plot barplot, plot lineplot Functions	Chapters 3, 6	
	4–Sept 22	Turtle Modules Relational and logical operators	Chapters 4-5	
	5–Sept 29	Conditional statatements While loops For loops	Chapter 7	A1 due
	6–Oct 6	Nested loops Strings and loops Debugging	Chapters 8-9 Appendix A	
	7–Oct 13	Intro to lists Midterm Review	Chapter 10	A2 due
	8–Oct 20	Lists	Chapter 10	Midterm (Oct 22)
	9–Oct 27	Dictionaries Tuples	Chapters 11-12	
	10–Nov 3	Case study on data structure Exception handling File IO	Chapters 13-14	A3 due
OOP	11–Nov 10	Intro to Objects and classes Functions and Methods	Chapters 15-17	
	12–Nov 17	Inheritance Case studies	Chapter 18	
Special Topics	13– Nov 24	Recursion Numpy Student Choice Lecture	Chapter 5	
	14–Dec 1	Review		A4 due