McGill University

www.cs.mcgill.ca

Course Outline - Draft

Course Name: Computer Programming for Life Sciences
COMP 204 Fall 2024
Lectures: MWF: 09:35 - 10:25 - LEA26

Instructors: David Becerra
Office: Trottier 3107
Office Hours: M & W: 16:45 - 17:45
Email: david.becerra@mcgill.ca
https://www.cs.mcgill.ca/~dbecer/

Course Objectives:

Welcome to COMP-204! 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 email 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 memorise. 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.

Primary learning objectives:

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

- Design and describe precise, unambiguous instructions that can be used by a computer 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 and structure conventions to make programs easy to understand, debug and modify;
- Learn independently about new programming-language features and libraries by reading documentation and by experimenting;

What this course is NOT about:

This course is not about how to use a computer. It will not teach you how to send email, browse the Web, create word processing documents or spreadsheets, setup 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.

Pre-requisites: BIOL 112 and CEGEP level mathematics course.

Restrictions:

Credit can be given only for one of COMP 202, COMP 204, or COMP 208. COMP 204 cannot be taken for credit with or after COMP 364, COMP 250 or COMP 206. COMP 202 is intended as a general introductory course, COMP 208 is intended for students with engineering and physics background and COMP204 is intended for students in life science fields.

Required Software:

- Python IDE https://wingware.com/downloads/wing-101
- myCourses
- edStem

Textbook:

There is no primary mandatory textbook for this course. We will be using a flipped approach with a diverse set of materials and resources, which will be made available on myCourses.

If you would like to consult a free, online textbook, you can use the following:
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/

Teaching Method / Course Delivery:

The resources for the class are composed of the following items.

- Weekly Video + Quiz x 12
- In-class live coding + examples (flipped class) x 12
- In-class problem solving (flipped class) x 12
- Weekly mini assignment x 9 = Programming x 9
- Assigned lab time with designated lab material x 12

This course will follow the flipped classroom format. What this means in our case is that lectures will be distributed to the student via videos. Each video will be posted on MyCourses every Friday at noon and

will present the material for the following week. The student will watch the video. Students can ask questions to the professors and TAs and help each other using Ed Discussion. Each video will end with a short quiz and a mini-assignment. The quiz must be completed before midnight on Wednesday, for grades. The mini-assignment is due within 8 days, for grades. Class time will be problem solving and Q&A experience using an in-person experience. On Monday, optionally, you will have a laboratory whose problems are very similar to the ones asked in the mini-assignment. On Wednesday, David will present additional expanded material related to the video and then a couple of problems (examples) which will be solved together during class. There will also be live coding led by the instructor to show practical implementations of the theory explained in the video. On Friday, you will work in groups to solve problems and questions. The instructor and T.As will be there to help you with your questions. Since the course is conducted in-person, to enrich the student experience, we will use additional tools, like Zoom (when needed), ed-Lessons, slido, along with traditional methods like myCourses quiz tool, assignments and tests.

It is important to mention that I will assume that the majority of our COMP204 students have never programmed before as we introduce you to the wonderful and exciting world of computer programming.

Grading

Option I

Work	Weight	Comment
Lecture Quiz (12)	9%	At the start of weeks 1 to 12, watch the videos and complete the quizzes. 0.75% each.
Mini-Assignments (9)	32%	By the end of weeks 3 to 12, complete a small exercise (assignment) to demonstrate your understanding. Programming x 9 (I will consider the best 8 marks)
Exercises (12) + **Surveys** (2)	4%	To attend and be able to find the slido password in at least 10 exercise (Friday) sessions AND answer the two surveys of the class. These surveys provide information about your experience, and allow us to adjust the course to make your experience better.
Midterm exam (1)	21%	Midterm exam, just after the reading week.
Final exam (1)	34%	Final exam, during the exam week.

Option II

Work	Weight	Comment
Lecture Quiz (12)	9 %	At the start of weeks 1 to 12, watch the videos and complete the quizzes. 0.75% each.
Mini-Assignments (9)	36%	By the end of weeks 3 to 11, complete a small exercise (assignment) to demonstrate your understanding. Programming x 9
Midterm exam (1)	21%	Midterm exam, just after the reading week.
Final exam (1)	34%	Final exam, during the exam week.

Tentative Course Outline

Week	Video (Friday 11:59 AM)	Flipped Class (MWF)	Readings	Work (due Sat)
0) Aug 26	N/A	(Wednesday) Introduction Comp204 (Friday) Python installation Lab	N/A	Get account for edStem, myCourses Install Python, Install IDE
1) Sep 2	Python basics, Shell and Prompt, Statement, values, Expressions 1, Operator, Types, Errors (syntax), functions 1.	Exercise Arithmetic Operators. Exercise Types. Exercise Variables (swapping). Exercise Errors.	Binary numbers. Operator precedence. Memory model.	
2) Sep 9	Strings 1, variables, expressions 2, functions 2 (built-in and customized).	Exercise built-in functions. Exercise design functions. Exercise reuse functions (call) Exercise nested functions	Design of functions. Docstring. Debugging.	
3) Sep 16	Functions 3, boolean, conditionals, Strings 2, indexing, slicing,	Exercise booleans. Exercise if statements. Exercise strings	String built-in functions	Mini 3 - Program

	How to debug.	operations Exercise strings methods.		
4) Sep 23	Iteration (while, for, recursion)	Exercise for loop over strings. Exercise while loops. Exercise, same problem using for, for range and while. Exercise nested loops.		Mini 4 - Program Survey #1
5) Sep 30	Lists (normal, parallel, nested), mutable VS immutable, Dictionaries 1	Exercise for loops over lists. Exercise for List operations and methods. Exercise for parallel strings and lists. Exercise for nested lists.	Aliasing	Mini 5 - Program
6) Oct 7	Dictionaries 2, Tuples and Sets	Exercise for dictionaries. Exercise for tuples. Exercise for sets.		Mini 6 - Program
Oct 14	FALL READING BREAK			
Oct 21	Midterm exams	Midterm (Short answers) Midterm (Open) Midterm (SA + Open)	Midterm Exams Week	Individual and Group exam.
7) Oct 28	Exceptions, Files IO, Libraries	Exercise for files. Exercise for exceptions.		Mini 7 - Program
8) Nov 4	Functions 4, Matplotlib, Numpy 1, Sorting, Search	Exercise Sorting. Exercise Matplotlib. Exercise Numpy		Mini 8 - Program
9) Nov 11	Numpy 2, OOP and Coding a full project	OOP exercise creating a “big” project.		Survey #2 Mini 9 - Program
9) Nov 18	Numpy 2, OOP and Coding a full project	OOP exercise creating a “big” project.		Mini 9 - Program
10) Nov 25	Introduction to image processing in Python	Malaria Exercises		Mini 10 - Program
11) Dec 2	Introduction to machine learning in Python	Prostate Cancer Exercises		Mini 11 - Program Laboratory session replaced by exercises.
TBD	Final Exam			

Learning and Performance Descriptors for programming-based assessments.

Hybrid (Autograded-Human) Programming-based assessments - MiniAssignments

Trait	Mastery	Proficient	Developing	Beginning
Correctness	The solution works correctly on all inputs and meets all specifications.	The solution meets most of the specifications. It is incorrect only in a few instances.	The solution is incorrect in many instances.	The solution does not run at all or produces incorrect results almost always.
Readability	The solution is well organised according to course expectations and it is very easy to follow without additional context.	The solution is mostly organised according to course expectations and overall easy to follow for someone with context.	The solution is readable only by someone who knows what it is supposed to be doing.	The solution is poorly organised and very difficult to read.
Algorithm Design	The choice of algorithms, data structures, or implementation techniques is very appropriate to the problem.	The choice of algorithms, data structures, or implementation techniques is mostly appropriate to the problem.	The choice of algorithms, data structures, or implementation techniques is mostly inappropriate to the problem.	Fails to present a coherent algorithm or solution.
Reusability	The entire solution is composed of reusable units.	Most of the solution is composed of reusable units.	Some parts of the solution are composed of reusable units.	The solution is not organised for reusability.
Documentation	The solution is well documented according to the course expectations.	The solution is well documented according to the course expectations.	The solution documentation lacks relevancy or disagrees with course expectations.	The solution lacks documentation.
Performance	The solution meets all performance expectations	The solution meets most performance expectations	The solution meets few performance expectations	The solution is not performant.

General grading information.

For mini-assignments.

Each student will receive an overall score for each mini-assignment. This score is called the Combined score and it is the combination of the error and style category scores. The weights for each category in each assignment can vary (but it will be clearly communicated in the instructions of each assignment). This variation is explained given that the learning objectives of each assignment are different.

- Error score: This score refers to the "**correctness**" category/trait shown in the Autograded Programming-based assessments table (see above). This score will be computed based on how many test cases the student's code passed. The test cases will be composed of open and blind cases. The open test cases are instances of the problem that will be run with-in the student submissions and the student will receive automated test results (i.e., the autograder output) for them. Normally, the blind test cases are inputs that students have not seen and they will test the correctness of the student's algorithm on those inputs once the deadline of the assignment is over. For some assignments, students will know how many test cases their submission is passing even for blind instances.
- Style score: This score refers to the "**Readability, Algorithm Design, Reusability, Documentation and Performance**" categories/traits shown in the Autograded Programming-based assessments (see above). The submissions of the students will be reviewed by one or more Teaching Assistants. The T.A will evaluate your coding style based on the reported categories and they will give you feedback on it. The feedback will be provided using the manual grading feedback form of Ed-Lessons. From there, the student will be able to review the feedback (which will be presented as annotated comments inside the comment of the students). We (i.e., the teaching staff) expect the student to incorporate the feedback in the next submission of their assignments supporting a better interactive learning experience

Human Programming-based assessments - Exams

Trait	Mastery	Proficient	Developing	Beginning
Understanding of the problem	Demonstrates a deep understanding of the problem, identifying all key components and potential challenges.	Shows a good understanding of the problem, identifying most key components and challenges.	Shows a basic understanding of the problem but may miss some key components or misunderstand certain aspects.	Struggles to understand the problem and its equirements, leading to significant inaccuracies in the solution.
Strategy	Articulates a clear strategy to correctly solve the problem, according to their understanding of the task at hand.	Articulates only parts of an appropriate strategy, but most key elements are present.	Articulates a strategy that is only partially useful, key elements might be missing, or inappropriate steps are provided.	Fails to provide a strategy to address the problem.

Readability	The solution is well organised according to course expectations and it is very easy to follow without additional context.	The solution is mostly organised according to course expectations and overall easy to follow for someone with context.	The solution is readable only by someone who knows what it is supposed to be doing.	The solution is poorly organised and very difficult to read.
Algorithm Design	The choice of algorithms, data structures, or implementation techniques is very appropriate to the problem.	The choice of algorithms, data structures, or implementation techniques is mostly appropriate to the problem.	The choice of algorithms, data structures, or implementation techniques is mostly inappropriate to the problem.	Fails to present a coherent algorithm or solution.
Validation	mplements a structured strategy to validate the satisfaction of all the problem's requirements.	Implements a good strategy to validate the satisfaction of some of the problem's requirements.	Implements a weak strategy to validate the satisfaction of few of the problem's requirements.	The validation strategy s weak and incomplete.
Correctness	The solution works correctly on all instances of the problem and meets all specifications.	The solution meets most of the specifications. It is incorrect only in a few instances.	The solution is incorrect in many instances.	The solution produces incorrect results almost always.

General grading information.

For programming questions in exams.

Each student will receive an overall score for each programming question in exams. The weights for each programming question in each exam can vary (but it will be clearly communicated in the instructions of each assessment). This variation is explained given that the learning objectives of each question are different.

- Overall score: This score refers to the "**Understanding of the problem, Strategy, Readability, Algorithm Design**, **Validation and Correctness**" category/trait shown in the Human Programming-based assessments (see above). The solutions of the students will be reviewed by one or more Teaching Assistants and/or the instructor. The T.A or instructor will evaluate your coding style based on the reported categories and they will give you feedback on it.

General Information

Communication:

- **General Policy:** The University is committed to maintaining teaching and learning spaces that are respectful and inclusive for all. To this end, offensive, violent, or harmful language arising in course contexts may be cause for disciplinary action under the Article 10 of the Code of Student Conduct and Disciplinary Procedures and Section 2.7 of the Policy on Harassment, Sexual Harassment, and Discrimination Prohibited by Law.
- **My Courses**: All official communication, including announcements, lecture material, assignments, grades will be found on My Courses.
- **Course Discussions**: The online tool, edstem.org, is used as our course discussion board. Please make sure to enroll in the Fall 2024 COMP 204 course on edstem. Use this as your primary communication medium, since your questions are public and can help other students.
- **Private Email**: The professor and TA have private email accounts that you may also use, however these communication channels are for personal queries. For example: if you have a problem with your grade then email the TA who graded you directly, do not email the prof and do not use the course email address.
- **Appointments**: Please email directly the one you want to communicate with to book an appointment outside office hours.
- **Office Hours**: Please take a look at all posted office hours. Come to those times without appointment.
- **After lecture**: Some optional time will be available just after class to ask questions. I do not guarantee the length of this time since other constraints may interfere.
- **Email Policy**: E-mail is one of the official means of communication between McGill University and its students. As with all official University communications, it is the student's responsibility to ensure that time-critical e-mail is accessed, read, and acted upon in a timely fashion. If a student chooses to forward University e-mail to another e-mail mailbox, it is that student's responsibility to ensure that the alternate account is viable. Please note that to protect the privacy of the students, the University will only reply to the students on their McGill e-mail account.

```
CommunicationAlgorithm() :
    if (public) edstem(); // all will benefit
    else if (about marks) emailTAPrivately();
    else if (medical or special) emailProfPrivately();
```

Assignments & Tests:

- **Assignments Delivery**: All assignments are picked-up from myCourses and edstem.
- **Late Policy**: You will be notified in advance of assignment due dates. All assignments are due on myCourses or edstem at the indicated time and date. Late assignments will lose 20% of its grade per day. Assignments beyond 1 day late will not be accepted. You may not submit assignments via e-mail without the permission of the instructor.
- **Additional Work**: Students (regardless the grades) will not be given the opportunity to complete additional work to upgrade their grade.
- **Grading Policy**: No make-up tests or make-up assignments are allowed in this course. If you are not satisfied with the grading of an assignment or test, you may request the instructor for an explanation of the grade within 10 working days of the date of the return of the graded materials to the student. Please in the communication clearly indicate where and why you feel the marks are unjustified. Mistakes can occur

when grading. If I identify a possible error, I will re-grade a question on an exam or assignment. I reserve the right to re-grade other questions as well.

- **Cheating/Collaboration**: Collaboration is encouraged but your discussions should be public in the sense that anyone including the professor should be allowed to listen in. Assignments are original works created by the student alone. You are permitted and encouraged to have conversations with other students concerning the contents of the assignments and how to do them, but your work must be original. It is completely forbidden to show or share your code. If two or more assignments are found to be identical (or portions of assignments) then all parties will lose points. This includes the student who permitted their assignment to be copied. This includes written solutions and software source code.
- **Use of Generative AI**: Students are not encouraged, unless otherwise stated, to make use of artificial intelligence tools, including generative AI, to help produce assignments. We believe that working through the assignments on your own will help you gain a better understanding of the course material and will better prepare you not only for the other course examinations, but also for the subsequent CS courses, internships, research opportunities, and jobs. However, students are ultimately accountable for the work they submit. Any content produced by an artificial intelligence tool must be cited appropriately. Many organisations that publish standard citation formats are now providing information on citing generative AI (e.g., MLA: https://style.mla.org/citing-generative-ai/).
- **Exam Policy**: Students are responsible for all materials for the tests and exams. Exams will be a combination of all types of questions based on all sources, and students may be required to integrate theoretical concepts from the text to substantiate their arguments. Crib sheets, programmable calculators, dictionaries are not permitted during an exam or test unless specifically stated by the professor.
- **Calculators:** Only non-programmable, no-tape, noiseless calculators are permitted. Calculators capable of storing text are not permitted in tests and examinations.
- **Dictionaries:** Dictionaries are not permitted, but translation dictionaries are.
- **Handheld Devices:** Handheld devices capable of storing text and having calculator functionality (e.g. Palm, etc.) are not permitted.

Additional Information:

The course slides are not meant as a complete set of notes or a substitute for a textbook, but simply constitute the focus of the lecture. Important gaps are left in the slides that are filled in during class, thus **l**ecture attendance should be considered essential.

The material covered in the classroom will be used to supplement textbook readings.

Every chapter should be read twice. The first reading should be done prior to attending class and the second reading should be done after the class discussion of the chapter. The questions at the back of each chapter follow directly from the reading. Students should be able to answer these questions after a thorough reading of the material.

Right to submit in English or French written work that is to be graded.

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.

Academic Integrity: *Code of Student Conduct*

McGill University values academic integrity. Therefore all students must understand the meaning and consequences of cheating, plagiarism and other academic offences under the Code of Student Conduct and Disciplinary Procedures (see www.mcgill.ca/integrity for more information).

L'université McGill attache une haute importance à l'honnêteté académique. Il incombe par conséquent à tous les étudiants de comprendre ce que l'on entend par tricherie, plagiat et autres infractions académiques, ainsi que les conséquences que peuvent avoir de telles actions, selon le Code de conduite de l'étudiant et des procédures disciplinaires (pour de plus amples reseignements, veuillez consulter le site www.mcgill.ca/integrity).

Final Exam Policy: *Regulations*

Students should not make other commitments during the final exam period. Vacation plans do not constitute valid grounds for the deferral or the rescheduling of examinations. See the Centre Calendar for the regulations governing Examinations: https://www.mcgill.ca/exams/regulations

Students are required to present their I.D. Card (with photo) for entrance to their examination.

Final Exam Policy: *Conflicts*

If you are unable to write your final examination due to scheduling conflicts, you must submit a Final Exam Conflict Form with supporting documentation at least **one month** before the start of the final examination period. Late submissions will not be accepted. For details, *https://www.mcgill.ca/exams/dates/conflicts*

Final Exam Policy: *Exam Timetable*

Examination schedules are posted at the Centre and on the following page approximately 6-8 weeks before the examination period commences https://www.mcgill.ca/exams/dates

The Centre cannot provide examination dates over the telephone.

Student Rights and Responsibilities:

Regulations and policies governing students at McGill University can be downloaded from the website: https://www.mcgill.ca/students/srr/

Students Services and Resources:

Various services and resources, such as email access, walksafe, library access, etc., are available to McGill students: https://www.mcgill.ca/studentservices/

Various services and resources are offered to computer science students: https://mcgill-csus.ca/

Minerva for Students: http://www.mcgill.ca/minerva-students/

Important Note:

In the event of extraordinary circumstances beyond the University's control, the evaluation scheme in a Course is subject to change, provided that there be timely communications to the students regarding the change.

Land acknowledgement:

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 people of the world now gather. Please see here for more details: https://www.mcgill.ca/edu4all/other-equity-resources/traditional-territories .