McGill University School of Computer Science Comp 321

Course Outline

Course Name: Programming Challenges
COMP 321 Fall 2024

Instructors: Section 1 - David Becerra
Lectures: MW: 2:35 - 3:55
Office Hour: F 15:30 - 17:00
Email: david.becerra@mcgill.ca
Office: TR 3107

Welcome to COMP-321! Please read this document carefully and keep it for reference throughout the term.

Course Format:

This course consists of twenty 1.5-hour lectures and three 3-hour programming contests. In addition to the ~40 hours of in-class work, the students will be given weekly training problems to exercise their skills at home. The student will need to invest time (~80 hours) to work on the assignments and study/review the topics covered in class. Then, all the activities bring the total amount of work for the course to approximately 120 hours.

- Lectures (20): M-W 14:35pm - 15:55pm

- Contest (3): TBD; however, they are planned to happen on the week of October 9, November 6 and November 27.

Course Goals:

- Give students the opportunity to test their algorithm design and programming skills on tricky problems and puzzles.
    - Given a problem, we want to:
        - Solve it efficiently by using algorithms and data structures.
        - Convert our solution into a program.
        - Do it as quickly as possible (under pressure)
        - And do it correctly (without bugs)
- Encourage students to join the McGill team for the programming contest.
- To have **FUN!**

Comp321 is right for you if:

- You are looking for a fun course that puts a fresh face on standard topics in programming and algorithms. OR

- You are planning to apply soon for a job interview in a big company. OR
- You are interested in joining the McGill team to participate in diverse programming contest competitions. OR
- You are simply motivated by the thrill of competition and learning.

Pre-requisites:

- COMP-250, and COMP-206

Co-requisites:

- COMP-251

Required Software:

- You can solve the proposed exercises in any of the following programming languages: Java, Python and/or C/C++.

Textbook:

There is no required material; however, I recommend the following material:

BOOKS

- Skienna, S; Revilla, M., Programming Challenges, Springer Verlag, 2003. ISBN: 0-387-00163-8.n
- T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms (Third Edition), MIT Press, Cambridge, MA, 2009
- Principles of Algorithmic Problem Solving. Johan Sannemo, 2018.
- Competitive Programming by Steven Halim (https://sites.google.com/site/stevenhalim/)
- Any other good textbook on algorithms and C++, Java and/or Python reference

ONLINE CONTESTS:

- TopCoder
- Google Code Jam
- CodeChef

TRAINING PROGRAMS:

- USACO training program.

ONLINE JUDGES:

- https://open.kattis.com/
- http://www.spoj.com/
- https://uva.onlinejudge.org/

COLLEGIATE CONTESTS:

- https://icpc.baylor.edu/

- http://www.ioinformatics.org/index.shtml

Course Outline:

Week	Activity	Date	Topic
1	**Lecture 1**	Aug-28	Introduction - Presentation of Comp321
2	**Lecture 2**	Sep-4	Preliminaries to Competitive Programming
3	**Lecture 3**	Sep-9	Data Structures I
	Lecture 4	Sep-11	Data Structures II
	Assig. 1 due	Sep-11	Ad - Hoc Problems
4	**Lecture 5**	Sep-16	Problem Solving Paradigms (Brute Force)
	Lecture 6	Sep-18	Problem Solving Paradigms (Divide & Conquer)
	Assig. 2 due	Sep-18	Data Structures Problems
5	**Lecture 7**	Sep-23	Problem Solving Paradigms (DP)
	Lecture 8	Sep-25	Problem Solving Paradigms (DP)
	Assig. 3 due	Sep-25	Brute Force Problems
6	**Lecture 9**	Sep-30	Problem Solving Paradigms (DP + Greedy)
	Lecture 10	Oct-02	Problem Solving Paradigms (Greedy)
	Assig. 4 due	Oct-02	DP Problems
	NAQ-Contest	Oct-05	Contest (voluntary)
7	**Contest 1***	Oct-09	180 mins Contest (L1 - L10) (mandatory)
	Assig. 5 due	Oct-09	Greedy Problems
8	**No-class**	Oct-14	Reading Week
	No-class	Oct-16	Reading Week
9	**Lecture 11**	Oct-21	Graph Theory I
	Lecture 12	Oct-23	Graph Theory II
	Assig. 6 due	Oct-23	Postmortem Contest 1 problems

10	**Lecture 13**	Oct-28	Strings I
	Lecture 14	Oct-30	Strings II
	Assig. 7 due	Oct-30	Graph Theory Problems
11	**Contest 2***	Nov-6	180 mins Contest (Lecture 11 - Lecture 14)
	Assig. 8 due	Nov-6	String Problems
	NENA-Contest	Nov-10	Contest (qualified)
12	**Lecture 15**	Nov-11	Computational Geometry
	Lecture 16	Nov-13	Computational Geometry
	Assig. 9 due	Nov-13	Postmortem Contest 2 problem
13	**Lecture 17**	Nov-18	Algebra, number theory and combinatorics I
	Lecture 18	Nov-20	Algebra, number theory and combinatorics II
	Assig. 10 due	Nov-20	Computational Geometry Problems
14	**Contest 3***	Nov-27	180 mins Contest (Lecture 15 - Lecture 18)
	Assig. 11 due	Nov-27	Algebra, number theory and combinatorics problems
15	**Lecture 19**	Dec-02	TBD
	Lecture 20	Dec-04	TBD
	Assig. 12 due	Dec-04	Postmortem Contest 3 problem

*The week of the contest there is no lecture (because ideally the contest lasts 3 hours)

Course Grading:

- **45%** for 12 assignments [3.75% each]. (9 home training problem sets + 3 postmortem contests).
- **55%** for 3 programming contests. (19% the first, 18% the second and 18% the third contest).
- **(Bonus) 0.1%** for each problem solved in NAQ, up to a maximum of 10 problems.

Learning and Performance Descriptors for programming-based assessments.

Autograded Programming-based assessments - MiniAssignments & Contests

Trait	Mastery	Developing
Correctness and Performance	The solution works correctly on all inputs of the judge and meets all the performance (time and memory) specifications.	The solution is incorrect in many instances of the judge and/or does not meet the performance (time and memory) for some instances.

Note about the grading:

The grading will be given by the online judges. Judging is relentlessly strict and the grading will be based on the online judge acceptance criteria. This criterion establishes that the student program will be judged as Accepted (i.e., `Mastery column in the descriptors') if and only if the program passess **all** the test files successfully (i.e., the program is correct and efficient in terms of time and memory). The following learning descriptors will help and support the student learning process on the creation of correct and efficient algorithms and implementations.

Trait	Mastery	Developing III	Developing II	Developing I
Readability	The solution is well organized according to course expectations and it is very easy to follow without additional context.	The solution is mostly organized 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 organized 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 organized 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.
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 requirements, 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.

The feedback given by the judge is limited[1] to one of the following categories: Accepted, Wrong Answer, Compile Error, Run Time Error, Time Limit Exceeded, Memory Limit Exceeded. The students are always welcome to attend office hours (during and after the development of the assignment) to get more information and feedback about their submission. During office hours, the teaching staff will also be able to provide advice about how to improve the algorithm design for future assignments.

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

[1] This is a common practice in competitive programming.

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 321 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.
- **Office Hours**: Please take a look at all posted office hours. Come (i.e., connect via zoom) 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.

Assignments & Programming Contests:

- **Assignments**:
  - All assignments are submitted to and picked-up from kattis.
  - After the lecture, you get a set of problems about that topic.
    - Each set has approximately 8 problems.
    - Each problem has a varying difficulty.
    - Each problem is assigned some amount of points based on its difficulty.
    - To get a perfect score you need to get at least a certain amount of points. Usually representing the solution of 2 to 4 problems, or a total of 3.75 points.
    - If you solve twice (or more) the number of required points to get a full mark, you will get a wild-card. You will be able to exchange your wild-cards by test cases in your next assignments.
    - The grade follows linearly from the number of points you get.
    - The deadline of a problem set is the following class (Wednesday of the next week).
    - Late submissions will not be accepted.
    - Individual solutions will not be partially graded (i.e., they must be accepted by the judge).

- **Late Policy**: Due date/time, location/mode for returning your solutions, and accepted formats are announced for each assignment. Failure to return your assignment in time will result in penalties or even absence of grading.
- **Programming Contests:**
    - The programming contests sessions allow the students to apply their problem solving skills on a set of problems and puzzles using either C, C++, Java or Python.
        - The lab sessions will consist of approximately 13 problems of varying difficulty. The idea of the contest is to solve as many problems as possible during 3 hours.
        - To get full marks you do not have to solve ALL the questions. This number will be defined later in the course when the instructor has more information regarding the current programming level of the registered students. The grade DOES NOT follow linearly from the number of exercises you solve.
- **Cheating/Collaboration**: I greatly encourage you to discuss the assignment with each other student in the class (except during the programming contests). However, these discussions should not so far be that you are sharing code or giving away the answer. A rule of thumb is that your discussions should be considered public in the sense that anything you share with a friend should be sharable with any student in the class. It is highly possible that you will find solutions to the contests on-line. Please do not copy the answers. You are registered in this course to train your problem solving skills and copying code from others does not help you in this goal.
- **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 organizations that publish standard citation formats are now providing information on citing generative AI (e.g., MLA: https://style.mla.org/citing-generative-ai/).
- **Accommodations:** Students who for valid documented reasons (such as extenuating illness or serious personal circumstances) cannot undertake or submit an assessment task in a course may request academic accommodations within a timeframe that is in accordance with local guidelines and procedures, and instructor approval. Students should submit their request for accommodations and supporting documentation directly to the course instructor. Students who wish to request an academic accommodation must do so before the assessment task is submitted, and no more than 5 working days after the due date.

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 lecture attendance should be considered essential.

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).

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 .