UNIVERSITY OF SASKATCHEWAN
Department of Computer Science
176 Thorvaldson Building
110 Science Place Saskatoon, SK S7N 5C9 Canada
Telephone: (306) 966-4886 Facsimile: (306) 966-4884

COURSE SYLLABUS
CMPT 141.3: INTRODUCTION TO COMPUTER SCIENCE

Catalogue Description:

An introduction to computer science and problem solving using procedural programming. This course introduces the basic computer science and computer programming principles of algorithms, abstraction, encapsulation, variables, conditional branching, repetition, functions, recursion, and elementary data structures. These concepts are applied to problem solving applications such as data analysis and visualization, simulation, text processing, and image processing. The programming skills acquired in this course are applicable in all fields of study, the work-place, and personal projects.

Prerequisite(s):	One of (Computer Science 30, CMPT 105, CMPT 140), and one of Mathematics B30 or Foundations of Mathematics 30 or Pre-Calculus 30; or MATH 110 or MATH 123 (can be taken concurrently). **Note: Recommended for students with Computer Science 30, or CMPT 105 or CMPT 140, or for students in programs that require MATH 110 (or equivalent).** Students with credit for CMPT 115 of CMPT 117 cannot take this course for credit. Students may not take CMPT 100 or 120 for credit concurrently with or after CMPT 141.
Class Time & Location:	Section 01 (Mark Eramian): MWF 9:30-10:20, PHYSICS 165 Section 03 (Michael Horsch): TR 1:00-2:20, ARTS 134 Section 05 (Jennifer Seaton): MWF 11:30-12:20, PHYSICS 103
Website:	http://moodle.cs.usask.ca/

Instructor Information

	Section 01	**Section 03**	**Section 05**
Instructor:	Mark Eramian	Michael Horsch	Jennifer Seaton
Contact:	eramian@cs.usask.ca	horsch@cs.usask.ca	jms815@mail.usask.ca
Office Hours:	**Location**: 3rd floor Spinks Lab **Hours**: TBA	**Location**: 3rd floor Spinks Lab **Hours**: TBA	**Location**: 3rd floor Spinks lab **Hours**: TBA

Who is this course intended for?

This course introduces the basic computer science and computer programming principles. These concepts are applied to problem solving applications such as data analysis and visualization, simulation, text processing, and image processing. The programming skills acquired in this course are applicable in all fields of study, the work-place, and personal projects. This course can be taken as a Science credit for Arts & Science majors, and is also a required course in Computer Science major programs, and a few other programs.

CMPT 141 is the starting point for two kinds of students:

- Students with CS30, or roughly equivalent experience, can start with CMPT 141. Some students may already have programming experience, so we recommend that they start with CMPT 141, instead of CMPT 140.
- Students who are pretty good in math should feel confident starting at CMPT 141, even if they've not done any computer science before. By "pretty good at math" we mean that you're taking MATH 110, or equivalent, or you already have credit for it.

  Important Note: CMPT 141 won't need any particular math knowledge; we're far more interested in the by-products of math class: good analytical skills, logical and deductive reasoning skills, attention to detail, which are all identified as useful in learning computer science. Of course, you'll need a little math (everyone always needs a little math!), but nothing to be intimidated by.

Course Overview

Lectures will be opportunities to apply the concepts covered in the course, discuss them, as well as to ask questions and receive guidance; we will not waste class time reading PowerPoint slides to you. Short readings will be assigned before each class, and you will be expected to be prepared to discuss, ask questions, and participate. Laboratory times are listed below; these are your opportunities to put into practice the week's material under the guidance of a teaching assistant.

Assignments and laboratory exercises are weekly, to ensure that all the relevant material is put into regular, consistent practice. Even a simple assignment can turn into a time-consuming affair, if you get stuck on something that blocks your progress. Working at the last minute is a guaranteed source of stress and burn-out. Start every assignment early, to allow yourself time to consult if you run into a problem. Please make use of the teaching resources (instructors' office hours, TAs, labs, lectures, discussion forums etc.) available to you.

The midterm examination (see above for the schedule) will have some multiple choice, and some programming and short answer questions. The final examination is scheduled by the university. The midterm and final will be written, without a computer. As you progress through the course, practice for exams by working on some assignment questions "by hand" before you approach a computer.

Solutions to assignments and grading schemes will be made available a week or two after the assignment is due. Your assignments will be graded by a team of markers. If you have concerns about an assignment grade, we ask you to read the solution set, and the grading scheme, so that you can understand how your work was assessed, and what we were looking for. If you feel there is a grading problem, we ask that you do not contact the marker directly. Talk to an instructor during office hours first. If one of us agrees that there was a problem, we will initiate a review of the grading.

Course Learning Objectives

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

- Apply Python data structures including tuples, lists, and dictionaries, in programs requiring basic data organization.
- Apply basic computer science strategies relating to practical forms of abstraction, encapsulation, generalization, and specialization.
- Demonstrate problem solving skills applied to simple simulations, data analysis, text and image processing.
- Design and implement recursive functions in Python.
- Compare and contrast linear search and binary search in terms of runtime and memory costs.
- Compare and contrast insertion sort, quick sort and merge sort in terms of runtime and memory costs.

- Apply skills in elementary software testing, debugging, and tracing.
- Apply their basic programming skills to build programs that solve simple computational problems that arise in science and engineering, and sometimes in other fields of study requiring data analysis.
- Extend their knowledge of Python programming language by self-study.
- Apply their knowledge of computer science to learn other programming languages, or scripting environments such as Matlab or R.
- Continue their formal study of computer science in courses such as CMPT 145.

Lecture Schedule

A detailed schedule of lecture topics is available on the course webpage. The following is a rough outline.

- Introduction to Python (9 hours): variables, expressions, console I/O, conditionals, loops, file I/O.
- Data structures (7 hours): strings, tuples, lists, dictionaries, sets, arrays.
- Functions, Abstraction and Recursion (7 hours): defining functions, encapsulation, abstraction, generalization, problem vs algorithm, recursion.
- Software skills (2 hours): testing, debugging, tracing.
- Searching and Sorting (4 hours): linear and binary search, insertion sort, quick sort, merge sort, elementary complexity analysis.
- Applications and problem solving (5 hours): simple simulations, data analysis and visualization, text processing, image processing.
- Binary numbers and computer architecture (1.5 hours): adding and multiplying binary numbers, converting between binary and decimal, organization of computer hardware, and how computer instructions are executed.

Student Evaluation

Grading Scheme

9 Assignments	(4% each)	36%
7 Lab Exercises (CodeLab)	(1% each)	7%
Midterm Exam		17%
Final Exam		40%
Total		100%

Midterm Exam

The midterm is scheduled for October 19, in the evening. There will be 3 exam seatings, tentatively starting at 5pm, and 5:30pm, 6pm; students in any section may attend any of the seatings, though there will be a sign-up procedure about a week before the exam. Locations will be announced on the course Moodle webpage (http://moodle.cs.usask.ca) closer to the actual date. The midterm will consist of multiple choice questions, some short answer questions and some programming questions.

Criteria that must be met to Pass

Students must write the final exam. A student who does not write the final exam will receive a grade of at most 49 in the course.

Attendance Expectation

- Attend every class, and participate actively. There will be short reading assignments for all classes (see Moodle webpage http://moodle.cs.usask.ca), and students are expected to come to class having completed the readings. There is no penalty for missed lectures.
- Attend all laboratory sessions. These are opportunities to practice the course material with the guidance of a teaching assistant. There is no penalty for missed lab sessions, provided that the lab exercises (CodeLab) are completed by the due date (Fridays, 6pm).
- Attend the midterm examination. If you have part-time work, or other responsibilities, please try to make arrangements ASAP that will allow you to write the midterm. We will make alternative arrangements for students who cannot attend the evening seatings, but obviously, we would like to keep the special arrangements to a minimum. A missed midterm will be counted as a score of zero, excluding reasonable situations such as health or compassionate grounds.

Final Exam Scheduling

The Registrar schedules all final examinations, including deferred and supplemental examinations. Students are advised not to make travel arrangements for the exam period until the official exam schedule has been posted.

Note: All students must be properly registered in order to attend lectures and receive credit for this course.

Textbook Information

Required Text

- Course Readings are provided free of charge, and are available on the CMPT 141 course Moodle webpage (http://moodle.cs.usask.ca).
- CodeLab: Students are required to purchase access ($25 USD) to CodeLab for CMPT 141. Please see the CodeLab Setup section on the course webpage for instructions.

Optional Reading:

- Dierbach, Charles. Introduction to Computer Science Using Python: A Computational Problem-Solving Focus. Wiley, 2012. **ISBN-10:** 0470555157

CodeLab

CodeLab (turingscraft) is a web-based interactive programming exercise system for intro programming classes. We will be using it for all of your lab exercises. You are required to purchase access to CodeLab to be able to complete all of the lab exercises. The procedure for registering for CodeLab will be posted on the course Moodle webpage (http://moodle.cs.usask.ca).

Note that CodeLab is an external company that is not affiliated with the University of Saskatchewan. If you have troubles/problems with your CodeLab account, then you must contact CodeLab to have them resolved; your instructors are not able to assist with CodeLab account problems.

Registration for CodeLab will cost $25 (US$). This registration is NON-REFUNDABLE, so only register before Sept 19 if you are sure that you will not be withdrawing from CMPT 141 before then. If you do not have a credit card with which to purchase access, then you can purchase a pre-loaded credit card from most grocery stores in the city. We suggest purchasing a $50 card if you have to use a prepaid credit card. Note: The balance on a prepaid credit card can be used at most/all local stores to partially pay for something; for example, if you have $4.21 on a prepaid credit card, and buy something for $10 then you can use the card to pay for $4.21, and then pay the remainder by cash/debit/etc.

Policies

Recording of Lectures
Lecture videos will be available through a link on the course Moodle site.

Late Assignments
Unless otherwise noted, all *Codelab assignments* are due Fridays at 6pm, and *regular assignments* are due Wednesdays at 6pm. Because of the weekly nature of assignments, late lab exercises or regular assignments cannot be accepted. Yes, that's harsh, but we have a schedule to keep. We may make exceptions, but only rarely for emergencies or exceptional circumstances; please contact your instructor in such cases. Only **your** instructor can grant an extension.

Missed Assignments
Students are expected to attempt (and hopefully complete!) all assignments, and all laboratory exercises. It's better to submit partially completed assignments than to submit nothing at all. A missed assignment will receive a score of zero. If you miss an assignment for medical or compassionate reasons, contact your instructor as soon as possible.

Missed Examinations
1. Students who miss an exam should contact the instructor as soon as possible. If it is known in advance that an exam will be missed, the instructor should be contacted before the exam.
2. "A student who is absent from a final examination due to medical, compassionate, or other valid reasons, may apply to the College of Arts and Science Undergraduate Student's Office for a **deferred** exam. **Application must be made within three business days of the missed examination** and be accompanied by supporting documents." (http://artsandscience.usask.ca/students/help/success.php)

Incomplete Course Work and Final Grades
"*When a student has not completed the required course work, which includes any assignment or examination including the final examination, by the time of submission of the final grades, they may be granted an extension to permit completion of an assignment, or granted a deferred examination in the case of absence from a final examination.*

Extensions past the final examination date for the completion of assignments must be approved by the Department Head, or Dean in non-departmentalized Colleges, and may exceed thirty days only in unusual circumstances. The student must apply to the instructor for such an extension and furnish satisfactory reasons for the deficiency. Deferred final examinations are granted as per College policy.

In the interim, the instructor will submit a computed percentile grade for the class which factors in the incomplete coursework as a zero, along with a grade comment of INF (Incomplete Failure) if a failing grade.

In the case where the student has a passing percentile grade but the instructor has indicated in the course outline that failure to complete the required coursework will result in failure in the course, a final grade of 49% will be submitted along with a grade comment of INF (Incomplete Failure).

If an extension is granted and the required assignment is submitted within the allotted time, or if a deferred examination is granted and written in the case of absence from the final examination, the instructor will submit a revised assigned final percentage grade. The grade change will replace the previous grade and any grade comment of INF (Incomplete Failure) will be removed.

A student can pass a course on the basis of work completed in the course provided that any incomplete course work has not been deemed mandatory by the instructor in the course outline and/or by College regulations for achieving a passing grade." (http://policies.usask.ca/policies/academic-affairs/academic-courses.php)

For policies governing examinations and grading, students are referred to the Assessment of Students section of the University policy "Academic courses: class delivery, examinations, and assessment of student learning" (http://policies.usask.ca/policies/academic-affairs/academic-courses.php)

Academic Honesty

The University of Saskatchewan is committed to the highest standards of academic integrity and honesty. Students are expected to be familiar with these standards regarding academic honesty and to uphold the policies of the University in this respect. Students are particularly urged to familiarize themselves with the provisions of the Student Conduct & Appeals section of the University Secretary Website and avoid any behavior that could potentially result in suspicions of cheating, plagiarism, misrepresentation of facts and/or participation in an offence. Academic dishonesty is a serious offence and can result in suspension or expulsion from the University.

All students should read and be familiar with the Regulations on Academic Student Misconduct (http://www.usask.ca/secretariat/student-conduct-appeals/StudentAcademicMisconduct.pdf) as well as the Standard of Student Conduct in Non-Academic Matters and Procedures for Resolution of Complaints and Appeals (http://www.usask.ca/secretariat/student-conduct-appeals/StudentNon-AcademicMisconduct.pdf) Academic honesty is also defined and described in the Department of Computer Science statement on Academic Honesty (http://www.cs.usask.ca/students/academic-honesty/index.php).

For more information on what academic integrity means for students see the Student Conduct & Appeals section of the University Secretary Website at:
http://www.usask.ca/secretariat/student-conduct-appeals/forms/IntegrityDefined.pdf

Examinations with Disability Services for Students (DSS)

Students who have disabilities (learning, medical, physical, or mental health) are strongly encouraged to register with Disability Services for Students (DSS) if they have not already done so. Students who suspect they may have disabilities should contact DSS for advice and referrals. In order to access DSS programs and supports, students must follow DSS policy and procedures. For more information, check http://students.usask.ca/health/centres/disability-services-for-students.php, or contact DSS at 966-7273 or dss@usask.ca.

Students registered with DSS may request alternative arrangements for mid-term and final examinations. Students must arrange such accommodations through DSS by the stated deadlines. Instructors shall provide the examinations for students who are being accommodated by the deadlines established by DSS.

Student Supports

Student Learning Services (SLS) offers assistance to U of S undergrad and graduate students. For information on specific services, please see the SLS web site https://www.usask.ca/ulc/.

The Student and Enrolment Services Division (SESD) focuses on providing developmental and support services and programs to students and the university community. For more information, see the SESD web site http://www.usask.ca/sesd/.

Appendix: ACM-2013 Learning Outcomes

This course achieves the following learning outcomes (listed alphabetically) from the ACM-2013 Computer Science Curriculum Guidelines (https://www.acm.org/education/CS2013-final-report.pdf):

- Discuss the importance of algorithms in the problem-solving process. [Familiarity, SDF-AD-1]
- Discuss how a problem may be solved by multiple algorithms, each with different properties. [Familiarity, SDF-AD-2]
- Analyze and explain the behavior of simple programs involving the fundamental programming constructs variables, expressions, assignments, I/O, control constructs, functions, parameter passing, and recursion. [Assessment, SDF-FPC-1]
- Identify and describe uses of primitive data types. [Familiarity, SDF-FPC-2]
- Write programs that use primitive data types. [Usage, SDF-FPC-3]
- Modify and expand short programs that use standard conditional and iterative control structures and functions. [Usage, SDF-FPC-4]
- Design, implement, test, and debug a program that uses each of the following fundamental programming constructs: basic computation, simple I/O, standard conditional and iterative structures, the definition of functions, and parameter passing. [Usage, SDF-FPC-5]
- Choose appropriate conditional and iteration constructs for a given programming task. [Assessment, SDF-FPC-7]
- Describe the concept of recursion and give examples of its use. [Familiarity, SDF-FPC-8]
- Identify the base case and the general case of a recursively-defined problem. [Assessment, SDF-FPC-9]
- Discuss the appropriate use of built-in data structures. [Familiarity, SDF-FDS-1]
- Write programs that use each of the following data structures: arrays, records/structs, strings, linked lists, stacks, queues, sets, and maps. [Usage, SDF-FDS-3]
- Compare alternative implementations of data structures with respect to performance. [Assessment, SDF-FDS-4]
- Explain what is meant by “best”, and “worst” case behavior of an algorithm. [Partial, AL-BA-1]
- Implement basic numerical algorithms. [Usage, AL-FDSA-1]
- Implement simple search algorithms and explain the differences in their time complexities. [Assessment, AL-FDSA-2]
- Be able to implement common quadratic and O(N log N) sorting algorithms. [Partial, AL-FDSA-3]
- Discuss the runtime and memory efficiency of principal algorithms for sorting, searching. [Partial, AL-FDSA-5]
- Explain and give examples of the benefits of simulation and modeling in a range of important application areas. [Partial, CN-MS-1]
- Demonstrate the ability to apply the techniques of modeling and simulation to a range of problem areas. [Partial, CN-MS-2]
- Analyze simple problem statements to identify relevant information and select appropriate processing to solve the problem. [Assessment, CN-P-2]
- Explain how simple data is represented in a machine. [Partial, CN-P-6]