CMPUT 175 (LEC B1 B2 B3 B4 Winter 2022)
Participants
Grades
General
Lectures Video streaming with Zoom
Labs
Week 1 (January 5-7)- Introduction and back to programming (Chapter 1)
Week 2 (January 10 - 14) - Hands-on Python with fun simple programs
Dashboard
Site home
Calendar
My courses
Defensive Driving
WRS 101 (SEM X54

EC B1 B2 B3 B4 - Winter 2022

CMPUT 175 (LEC B1 B2 B3 B4 Winter 2022) / General / Course Outline

nation

0 (Lec B1) and MWF 10:00-10:50 (Lec B2)

:50 (Lec B3) and MWF 14:00-14:50 (Lec B4)

r details about your instructor and teaching assistants.

ssions is permitted only with the prior written consent of the professor, or if the recording is part of an approved accommodation plan.

Course Policies

CMPUT 175 is subject to the Department of Computing Science policies. In particular, pay attention to these Computing Science Course Policies.

In particular, you may want to read:

- our Grading Policy,
- the Lab and Assignment Policy,
- the Excused Absence Policy, and
- the Collaboration Policy.

Labs

All labs are in CSC 153 except H4, H8, H12 and H16 are in CSC 129, and H18 is in CSC B10. Labs begin in the **second week** of classes.

Labs are for more in-depth study of certain topics, obtaining help on assignments, and additional assistance in general.

There will be lab tutorials and additional exercises in the labs. You have to implement solutions to the lab exercises during the lab time and demo them to the TA in order to get marks for this component of the course.

Lab Information

Overview

CMPUT 174 and 175 provide an alternative path to the study of Computing Science. A problem-driven approach is used to introduce the fundamental ideas of computing. Emphasis is on the underlying process behind the solution, independent of programming language or style used to implement the solution.

Basic notions of state, control flow, data structures, recursion, modularization, and testing are introduced through solving simple problems in a variety of domains, such as text analysis, map navigation, game search, simulation, and cryptography. Students learn to program by reading and modifying existing programs as well as writing new ones.

Objectives

At many institutions, there are departments of Computer Science or Computer Engineering. At the University of Alberta, the founders of our department deliberately chose to call it Computing Science. They wanted to emphasize that the foundation of our discipline is computing, not computers. Broadly speaking, computing science asks these kinds of questions:

- how do we think about problems to solve them computationally?
- what kinds of computation are there?
- are some computations impossible?
- how do we know our computations are doing what we intend them to do?
- what are good ways to structure computations so that others can understand them?
- what are the ways to specify and organize computations so that machines can perform them?
- how do we build the machines that make computation possible?
- what are good ways for humans and computations to interact?
- how can we apply computing ideas to other areas of human activity?

This two course sequence is a small introduction to the foundations of a major part of Computing Science: expressing problems precisely, solving them algorithmically by showing how to construct a solution, and then implementing that solution by writing a program.

Our approach in this course is problem-driven. We will take a problem and attempt to solve it. In the process of developing an algorithmic solution, we will introduce key computing concepts. Our intention is that every abstract concept should be grounded in concrete examples.

Our emphasis will be more on computation and problem solving than all the details of a particular programming language for implementing a solution. Our goal is to develop your intuitions for a variety of complementary styles of computation.

What Programming Language?

If we want to actually compute, then we need to program, which means we need to use a programming language. There are many programming languages used in Computing Science (e.g., C, C++, Java, etc.). The basic programming concepts of this course are commonly seen in other languages. We will be using Python 3.

There are many reasons why Python is a good choice for this course. Python is easy to learn. It is a sufficiently high level language that enables us to focus on the key computing ideas. We can quickly get a running implementation, and not get bogged down in language details.

Python lets us program in many different styles such as imperative, functional, and object-oriented. It is portable, running on virtually every hardware platform and operating system in existence. And if you are not going to be a Computing Science major, it is probably the only programming language you will ever need to learn.

Pre-requisites

CMPUT 174, SCI 100, or consent of the instructor. Basic knowledge of Python is assumed.

Course Topics

These are some of the core programming concepts that we will study. They will be introduced within the context of a variety of problems.

- Program flow control: e.g. when to use a ***for*** loop or a ***while*** loop?
- Passing arguments to functions, defining custom functions
- Storing non-trivial data in lists and dictionaries
- String processing and formatting
- Reading from files, writing to files
- Classes, objects, and methods
- Encapsulation and Abstraction
- Algorithm Analysis
- Implementation and application of Stacks, Queues, Linked Lists, Doubly Linked Lists
- Recursion
- Searching and Sorting
- Trees and binary search trees
- Hash functions and hash tables.

Course Work and Evaluation

Course Work	Date	Weight
Assignments (3)	TBA	20%
Lab exercises (10)	most labs	10%
In-class assessment	any class	5%
Video on-line assessment	any class	5%
Midterm 1 (online)	February 14th	15%
Midterm 2 (online)	March 14th	20%
Midterm 3 (online)	April 8th	25%

In light of the COVID-19 crises and the uncertainty vis-à-vis the progression of the pandemic this winter and the public health recommendations, We have made the following arrangements:

- All lectures will be given in person in class but a recording of the lecture content will be made available on *eClass* for at least a week after the lecture.

- All midterm exams will be administered online.

- After a midterm, some students will be selected for an additional online oral test to complement the Midterm. The weight of the oral test will count with the Midterm

- All office hours and lab exercise demos will be conducted online via teleconferencing.

Assignments are normally due by 11:55 PM on the due date. When submitting on *eClass*, make sure that you click the *Save Changes* button, and then verify that your submission has been accepted by the system before you navigate away from the page. **Late assignments will not be accepted.** No make-ups, alternatives or supplementals will be given for missed course components. A missed course component gets a mark of zero.

CMPUT 175 assignments are normally done under the department's CONSULTATION model (see Course Policies above). That means you may discuss the assignments with others, but you **must write up the solution on your own**. If you consult with other students, you must list their names in a comment at the top of your submission, along with a brief description of the part(s) of the assignment you discussed. All sources used must be cited. Moreover, each student is responsible for what is handed-in and must be able to explain it.

Textbooks are not allowed on the (midterm) exams. No electronic or living organic aids are allowed. That is, devices such as calculators, cell phones, computers, or anything with a network connection are not allowed.

Special Dates

Date	What
Jan 5	Classes begin
Feb 22-25	Reading Week
Apr 8	Classes end

Grading System

Your final *grade* will be based on our interpretation of the grading system as defined in Section 23.4 of the Academic Regulations. We do not use a pre-defined function of your final mark to compute your final grade, but instead use our judgment of how the class final marks reflect mastery of the course material. We believe that this produces a fair evaluation, and our extensive past experience supports this.

See University of Alberta Assessment and Grading.

Course Materials

The following textbook is suggested for this course:

- Bradley N. Miller & David L. Ranum, *Problem Solving with Algorithms and Data Structures using Python*, Franklin, Beedle and Assoc., 2011.

Installing the Python Environment

If you want to use Python on your own computer, you need to obtain one of the various distributions of Python. We will be using **Python 3** (not 2). We recommend that you download and install the current 3.x.x version of Python from

- http://www.python.org/download/

Python code can be written with any plain text editor and executed from the command line. We will be using the free version of WingIDE, WingIDE 101, which can be found at:

- http://wingware.com/downloads/wingide-101

We will review how to install and use these in the initial lab sessions.

Academic Integrity

The University of Alberta 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 Code of Student Behaviour and avoid any behaviour which could potentially result in suspicions of cheating, plagiarism, misrepresentation of facts and/or participation in an offense. Academic dishonesty is a serious offense and can result in suspension or expulsion from the University.

Last modified: Tuesday, 4 January 2022, 11:28 AM

Jump to... ⬍

Contact Information ►