Tulane University

CMPS 1500: Introduction to Computer Science I (4 credit hours)

Fall 2024

Section 01: Monday/Wednesday/Friday 11:00am – 11:50am

Section 02: Monday/Wednesday/Friday 1:00pm – 1:50pm

Stanley Thomas hall room 302

Instructor	Prof. Matthew Toups
Office	Stanley Thomas hall room 310
Office Hours	Mon/Wed/Fri 4:00-5:00pm
Teaching Assistants	Kiran Shrestha, Keith Mitchell
Course Assistants	Dylan Murray, Hubert Mendez, Michael Weild

1 Course Summary

This course is an introduction to techniques, ideas, and problem-solving approaches that are used in Computer Science. At a high level, we focus on developing "computational thinking", which is the practice of using abstraction to design and implement algorithms and software to solve problems that arise in many different areas of our daily lives, such as networks, social media, and scientific computing, to name just a few. At a practical level, students will design, implement, test and document their programs to learn introductory programming concepts, such as: data types and data structures (e.g. lists, dictionaries, trees); programming techniques (modular design using functions, recursion, object-oriented programming); performance analysis via theoretical estimate, profiling and timing.

Most assignments in this course are programming assignments aimed to teach the students to express their ideas in efficient and elegant code; no prior programming experience is necessary to join and succeed in the course. Lecture periods are dedicated to introducing new material, discussions, individual and group activities. Lab periods are used for programming practice.

CMPS 1500 is the first course for the Coordinate Major in Computer Science.

Corequisite course: CMPS 1501 (lab sections), one 75-minute meeting each Tuesday

2 Objectives

2.1 Course Learning Objectives

Upon completion of this course, successful students will be able to:

- Develop computational solutions to problems:
  - Break a computational problem into manageable subproblems.
  - Write an algorithm to solve a specific problem, and then translate that algorithm into a program in a specific programming language (Python).
  - Reason about a solution's performance and measure its efficiency.
  - Write clear, concise documentation for their code and use tests to verify proper program operation.
- Explore the breadth of Computer Science as a discipline and how it exists in the world:
  - Reason how computational tools, and computational thinking, are used to solve problems in a diverse set of application areas.
  - Explain the basic theory and organization of a computer system, basic operation of modern software tools, and limits of computation.

2.2 Program-Level Outcomes (Major/Minor)

By the end of this course, students will have a practical and contextual foundation for future studies in Computer Science. Students will be able to write software to solve computational problems; they will have an introduction to topics that will be explored more deeply in future courses - abstract data types, algorithmic analysis, the limits of computation; and they will have explored computer science in the context of society - applications, problems, and impact.

3 Required Student Resources

We will be using ZyBooks, an online interactive textbook as the main source for our material.

A subscription is $92. Refunds are provided if the class is dropped within official withdraw deadline. You need to get the book and start completing the readings on the first day of class. If you're testing waters and are not sure whether you will stay in the class, get the book now and get a refund if you decide not to continue with the class.

4 Evaluation and Grading

Students may monitor scores throughout the semester using the Canvas gradebook. The tables below detail the weights and scale used.

Grades are computed using the following weights:	
Exams (midterm and final)	40%
Lab assignments	30%
Zybook readings	20%
Attendance and quizzes	10%

Grading Scale	
100 to 93	A
<93 to 90	A-
<90 to 87	B+
<87 to 83	B
<82 to 80	B-
<80 to 77	C+
<77 to 73	C
<73 to 70	C-
<70 to 67	D+
<67 to 63	D
<63 to 60	D-
<60	F

4.1 Late assignment policy

4.1.1 ZyBooks readings

Readings are due at 10:00am on the due date given, which is to say they should be completed either the evening before, or, at worst, the morning of a lecture on that topic. There are over 20 readings throughough the semester. To account for minor errors, at the end of the semester the 2 lowest reading scores will be dropped.

4.1.2 Lab assignments and grace days

There is a **25% penalty for lab assignments turned in late** within the 24 hours following the deadline. Each subsequent day late will incur another 25% penalty. In other words, assignments 0-24 hours late have a 25% penalty, 24-48 hours late has a 50% penalty, and so on. (Students will note this policy also means that assignments over 72 hours late will receive no points)

To allow for minor schedule disruptions, each student will be allowed 4 **Grace Days** for the semester (a maximum of 2 may be used on a single lab assignment). This grace day is a free extension of 24 hours, during which no penalty is assessed. After a student's grace days are exhausted, the 25% penalty will automatically be assessed as described above for the rest of the semester.

Grace Days are applied automatically by the course staff. Students do not need prior permission to use a grace day; this policy is in place to eliminate the need for such requests. Students may track their remaining grace days via the Canvas grade book. (Grace Days show up in the Canvas grade book, but they have no point value and are not factored in to final grades. In other words, unused grace days do not count for anything at the end of the semester.)

Further extensions are available in extreme cases only. Such requests must be made in writing via email, and must include documentation of illness, school-related travel, or other extraordinary situation.

5 Course Outline

Module	Component topic	Assessments
Intro to Programming	Sequences Conditionals Loops Iteration Functions Dictionaries File I/O	Hello-World Lab If-Else Lab Strings-Lists Lab Wordle Lab Class practice with functions
Intro to Systems and Architecture	Binary numbers Negative numbers Character encodings Other binary data formats	Functions Lab binary conversion practice
Midterm exam	all above topics	all above assessments
Intro to CS Theory	Algorithms Searching Sorting Recursion Complexity	Dictionary Lab Recursion Lab Algorithms vocabulary quiz
Intro to Object Oriented Programming	Classes and Objects Methods Constructors and special methods	Boggle Lab OOP quiz OOP class activities
Intro to Data Structures	Abstract Data Types Arrays and Lists Linked Lists Trees Graphs	Data Structures Lab Class practice with Trees Class practice with Graphs
Final Exam	review all lecture topics	review all lab topics

6 University policies

6.1 Attendance Statement

Students are expected to attend all classes unless they are ill or prevented from attending by exceptional circumstances. Students who find it necessary to miss class are responsible for obtaining notes on material covered in lectures or other class sessions.

Students are responsible for notifying instructors about absences that result from serious illnesses, injuries, or critical personal problems. Medical excuses are issued by the Student Health Center in the following instances: illnesses or injuries that involve hospitalization, a partial or complete withdrawal due to medical reasons, or a missed final examination for a medical condition being treated by the Student Health Center. In all of these instances, medical information will be released only with the student's written permission.

With the approval of the Newcomb - Tulane College dean, an instructor may have a student who has excessive absences involuntarily withdrawn from a course with a WF grade after written warning at any time during the semester.

6.2 Equity, Diversity, and Inclusion Statement (EDI)

Equity, diversity, and inclusion (EDI) are important Tulane values that are key drivers of academic excellence in our learning environments. In our drive for academic excellence, we seek to ensure that students, faculty, and staff across diverse social identities, cultural backgrounds, and lived experiences can thrive - especially those from from underrepresented and underserved communities (e.g., race/ethnicity, gender identity and expression, sexual orientation, disability, social class, international, veterans, religious minorities, age, and any other classification protected by applicable law - see Tulane's Nondiscrimination Policy). In order to build a supportive culture and climate for every member of our community, we recognize that we each of have unique EDI strengths to share with others and that we also have areas for EDI growth, learning, and change. This EDI commitment and cultural humility helps us collectively build a university community and culture where everyone experiences belonging.

6.3 Religious Accommodation Policy

Per Tulane's religious accommodation policy, we will make every reasonable effort to ensure that students are able to observe religious holidays without jeopardizing their ability to fulfill their academic obligations. Excused absences do not relieve the student from the responsibility for any course work required during the period of absence. Students should notify the instructor within the first two weeks of the semester about their intent to observe any holidays that fall on a class day or on the day of the final exam.

6.4 ADA/Accessibility Statement

Your instructor and course staff are committed to supporting all students throughout this course, and this is especially true for students with disabilities. Please rest assured that we will honor Tulane's commitment to accommodating students' needs. That commitment says:

> Any students with disabilities or other needs, who need special accommodations in this course, are invited to share these concerns or requests with the instructor and should contact Goldman Center for Student Accessibility at `http://accessibility.tulane.edu` or call 504-862-8433.

6.5 Code of Academic Conduct

The Code of Academic Conduct applies to all undergraduate students, full-time and part-time, in Tulane University. Tulane University expects and requires behavior compatible with its high standards of scholarship. By accepting admission to the university, a student accepts its regulations (i.e., Code of Academic Conduct Links to an external site. and Code of Student Conduct Links to an external site.) and acknowledges the right of the university to take disciplinary action, including suspension or expulsion, for conduct judged unsatisfactory or disruptive.

6.6 Academic Integrity in this course

Each exam and lab assignment must be the sole work of the student who turns it in. Violations will result in serious consequences. Some students have trouble distinguishing between *collaboration* and *cheating*. First of all, no collaboration is permitted on exams. None. For non-exam assignments, the following (non-exhaustive) list will provide examples of what is considered cheating and what is an acceptable type of collaboration.

What is cheating?

- Copying an answer to a test question or lab assignments (from a classmate, the web, anywhere)
- Simply looking at a solution from any of these sources is forbidden
- Searching Google or other sources for answers
- Verbally communicating answers or solutions
- Coaching: helping a friend with their lab line-by-line
- Cutting and pasting, retyping... it doesn't matter exactly how
- All are *easily* detected!

What is *not* cheating?

- Clarifying ambiguities or vague parts of assignment handouts or the text
- Using hints given in class (*must* be cited in code comments)
- Using help given by the instructor or textbook (again, must be cited)
- Asking a classmate for help logging in, using a text editor, or asking what an error message means (basic troubleshooting)
- Explaining how to use specific functions or tools

You should also be aware that the course staff may use automated means to flag possible plagiarism, including MOSS from Stanford University.[1]

[1] For more information on MOSS see `http://theory.stanford.edu/~aiken/publications/papers/sigmod03.pdf`