CS-1410 *Object Oriented Programming*

Instructor Contact Information

Instructor: [George Rudolph (https://uvu.instructure.com/courses/489064/pages/instructor-information)](https://uvu.instructure.com/courses/489064/pages/instructor-information)

Office: CS 520j

Office hours: MTWR 10am-11am, W 4-5pm

Telephone: 801-863-8116

Email: george.rudolph@uvu.edu

Preferred method of communication: Email/Teams/Discord are all fine. Discord may be the fastest. I will answer within 1 business day or sooner.

Course Description

Teaches proper program structure using the core concepts of object-oriented programming: **classes, objects, encapsulation, inheritance** and **polymorphism.** Presents problems of **increasing size and complexity** requiring OOP techniques, standard libraries and other appropriate language constructs. Presents methods to identify, define and implement solutions to naturally recursive problems. May be delivered online.

This is a **required course** for CS majors.

You must earn a **C+** or higher in order to take the next course, **CS 2420** (Data Structures and Algorithms).

Matriculation into CS Programs

In order to take upper-division Computer Science classes, students are required to complete the following classes with grades at least as high as those listed below:

CS Matriculation Courses

Course	Minimum Grade

CS 1400	C+
CS 1410	C+
CS 2300 (Discrete Mathematical Structures I)	C+
CS 2420 (Data Structures)	C+
Math 1210 (Calculus I)	C
English 1010	C

You must earn a C+ in each of the CS classes above before moving on to the next class in sequence.

Additional Requirements:

- Overall GPA must be at least 2.5.
- Students may not repeat a class more than once

Screening:

Students are reviewed in CS 1400 and CS 1410. They are screened for matriculation while taking CS 2420. Students will be notified approximately halfway through the semester as to their status.

Course Outcomes

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

1. Implement object-oriented solutions to problems of increasing size and complexity, which exhibit use of appropriate classes, objects, encapsulation, inheritance, and polymorphism
2. Test programs to assure that solutions are correct and complete
3. Design readable, maintainable code, using a good, consistent programming styl
4. Use algorithms and data structures from standard libraries to solve problems

Mastery Rubric

This mastery rubric should help you gauge where you are and how you're doing in mastering the skills this course is intended to teach you mapped to a grade range. For example "C" represents "C+", "C", and "C-".

Course Outcomes	A (Mastered)	B (Near-Mastery)	C (Approaching Mastery)	D (Partially Mastered)	E (Not Yet Mastered)
Implement object-oriented solutions to	Consistently implements robust object-oriented	Implementations generally show understanding of object-oriented	Occasionally uses object-oriented principles	Rarely uses object-oriented principles correctly.	Fails to correctly use object-ori principles.

problems of increasing size and complexity, which exhibit use of appropriate classes, objects, encapsulation, inheritance, and polymorphism	solutions. Utilizes appropriate classes, objects, encapsulation, inheritance, and polymorphism. Handles complexity exceptionally well.	principles, but there may be occasional inconsistencies or errors. Handles complexity well, but struggles with some intricate problems.	correctly, but there are frequent inconsistencies or errors. Struggles with problems of moderate complexity.	Demonstrates poor or inconsistent use of classes, objects, encapsulation, inheritance, and polymorphism. Difficulty handling simple problems.	Cannot handle even simple problems effectively.
Test programs to assure that solutions are correct and complete	Develops comprehensive test cases that ensure program correctness and completeness. Regularly employs testing frameworks and methodologies.	Develops good test cases, but they may not be entirely comprehensive. Uses testing frameworks and methodologies inconsistently.	Develops some useful test cases, but they may be limited in scope. Rarely uses testing frameworks or methodologies.	Struggles with developing adequate test cases. Testing may be sporadic or ineffective, and fails to ensure program correctness and completeness.	Fails to develop meaningful test cases. Does not employ testing frameworks or methodologies.

Design readable, maintainable code, using a good, consistent programming style	Code is consistently well-organized, readable, and maintainable. Adheres to best practices for programming style, including appropriate naming conventions, documentation, and code structure.	Code is generally well-organized and readable, but may have occasional lapses in maintainability or style. May occasionally deviate from best practices.	Code is sometimes readable and maintainable, but often deviates from best practices for programming style.	Code is occasionally readable and maintainable, but generally lacks organization and adherence to best practices.	Code is rarely readable or maintainable. Programming style is inconsistent, lacks appropriate documentation, or fails to adhere to best practices.
Use algorithms and data structures from standard libraries to solve problems	Consistently uses appropriate and efficient algorithms and data structures from standard libraries to solve problems. Demonstrates a deep understanding of algorithm and data structure principles.	Generally uses appropriate algorithms and data structures, but may occasionally choose inefficient or unnecessary options. Demonstrates good understanding of algorithm and data structure principles, but may struggle with more complex problems.	Occasionally uses appropriate algorithms and data structures, but often chooses inefficient or unnecessary options. Struggles with problems of moderate complexity.	Rarely uses appropriate or efficient algorithms and data structures. Demonstrates poor understanding of algorithm and data structure principles.	Fails to use appropriate or efficient algorithms and data structures. Does not understand algorithm and data structure principles.

✓ Prerequisites and Needed Skills

Course Prerequisites: CS 1400 **and** (MATH 1050 or MATH 1055 with a C+ or better, or MATH above 1050)

Technology Expectations:

- Access to a computer and the internet
- Previous experience with the Python programming language in a development environment of your choice.

Materials, Fees and Technology Tools

Required materials, fees and technology:

- ***CS: Objects in Python*** ***(https://www.codio.com/resources/objects-python)*** - Introduction to Objects in Python with native Codio content promoting engagement and active learning with fully auto-graded assessments and minimal text.
- Access to Codio for textbook, assignments and exams.
  - paid software (see steps below)
  - Codio **Statement on Privacy Compliance and other Legal Stuff** **(https://www.codio.com/legal-stuff#:~:text=Codio%20will%20not%20sell%20Personal,third%20party%20for%20marketing%20purposes.)**
- **(https://www.codio.com/legal-stuff#:~:text=Codio%20will%20not%20sell%20Personal,third%20party%20for%20marketing%20purposes.)** To access Codio:
  1. Click on **Modules** in Navigation at the left.
  2. Scroll down to Module 1 or any linked assignment or reading, and click on it. This will take you to a screen to access Codio.
  3. Students who took CS 1400 at UVU within a year before CS 1410 will have previously used Codio and will have purchased a license that is good for one year. The same account should be active for CS 1410.
  4. If #3 does not apply to you, you will be asked to purchase access and set up an account. Use a credit card to purchase access. Use the same email that you use for UVU login credentials. Make sure that you have **popups** enabled in your browser.

Optional Materials:

- Access to **Python** 3.10 or later on your own local machine.
  - free software
  - **Python Software Foundation Privacy Statement** **(https://www.python.org/privacy)**
- Discord OnStudy server for CS 1410 discussion groups across all sections
  - free software
  - join link is **bit.ly/onstudy** **(http://bit.ly/onstudy)** .

- **Privacy Policy (https://discord.com/privacy)**

How This Course Works

Course Mode: Online

Description of How this Course Works:

This course is organized into 13 modules by topic, each 1 week long. Students will progress through the course week by week. A typical week includes readings, videos, practice labs, exercises, and programming projects. There are also a midterm and a final exam. The final exam is a programming project.

For this **three (3) credit online** course students should expect to spend up to **3 hours/week** reading materials, watching videos or interacting with the instructor and other students online, and up to **6 hours/week** completing assignments outside of class for a total of 9 hours of academic engagement each week. Practice Labs, Exercises, Projects and Exams are *typically* done outside "class" time.

"Expect" when completing assignments means generally "on average". Some students may finish things faster, and some students may take longer. If you are unhappy with your progress, ask for help, and do not cheat. There are no shortcuts when it comes to learning to program and problem solve.

All grades are in Canvas. Any assignments graded in Codio are automatically transferred from Codio to Canvas. This is the same as it was for CS 1400.

Note about Codio: Major assignments such as labs, exercises, and projects are in Codio and accessed through Canvas. **If you attempt to access Codio without going through Canvas, many items will not be correctly accessible and grades will not sync to Canvas correctly.**

How to Be Successful in this Course

Start Early--Think Ahead!

In a given module each week:

- Readings are due on Thursday
- Exercises and Projects are due on Friday
- Power Questions (or similar assignments) are due on Friday each week (if used: check with your instructor)

Why are Projects due on Friday instead of Sunday? If you need the weekend to get your project done, use the weekend BEFORE the due date, NOT the weekend AFTER. Rather than planning to be late if something unexpected happens, start sooner so that you have time to deal with questions and issues that WILL come up. We want to train you to make time for success, not live in crisis with failure. This is what we mean when we tell you "start early".

Student Responsibilities:

- Start class the first week of the term.
- **Be accountable by setting aside regular time each week to complete course activities and assignments on time as noted per the due dates**.
- Learn how to use Canvas for assignments, grades and course materials.
- Learn how to use Microsoft Teams to hold video/voice meetings, post chats, and retrieve files. If you have technology-related problems contact the **Service Desk (https://www.uvu.edu/servicedesk)** .
- Learn to use Discord to: communicate with your instructor and other students, ask questions when you need help sooner rather than later, be part of a community rather than going it alone.
- Abide by ethical standards. Your work must be your own. Do not claim mastery of skills you haven't mastered, nor ask for a grade you have not earned by submitting someone else's work, nor by allowing someone else to submit your work as their own.
- Contact your instructor as early as possible if an emergency arises. Do NOT wait until the last minute to ask for an extension.

Instructor Responsibilities:

- Respond to messages within ONE business day. If multiple messages are received regarding the same question or concern, they may be responded to with an announcement to the entire class.
- Provide timely, meaningful and constructive feedback on assignments.
- Facilitate an effective learning experience.
- Refer students to appropriate services for issues that are non-course content specific. For instance, technical issues, writing labs, accessibility services, etc.
- Mentor students through the course.

Inclusion Statement

“Come as you are. UVU has a place for you.”--President Tuminez.

UVU is committed to preparing all students for success in an increasingly complex, diverse, and globalized society. We value and promote collegial relationships and mutual respect among students, faculty, staff, and the community. We acknowledge and seek to address the needs of populations who are underserved as well as students with varying levels of academic preparation. Since your experience in this class is important to me, it is my intent to promote civility and respect the voice, dignity, and potential of each individual. I aim for an inclusive learning environment that provides equitable opportunities and fosters the understanding, appreciation, and recognition that diversity and individual differences are a source of strength. I aim for a course that is respectful of diversity including age, culture, disability, ethnicity, gender, nationality, race, religion, sexuality, and socioeconomic status.

Please contact me if you need to talk about any issues you are facing. I value any suggestions on how to improve the effectiveness of this course. If that feels uncomfortable to you, you can con the **The Office of Institutional Engagement and Effectiveness (https://www.uvu.edu/engagement-and-effectiveness/index.html)** to ask for help and support.

Grading and Late Work Statement

Grading

You earn a grade for completing assignments, you don't lose grades for not completing them. You start with and E and work your way up to a grade you want---you don't start with an A and lose it. *It is your job to demonstrate that you have mastered the material. It is not your instructor's or your IA's job to demonstrate that you have not.*

50% is the minimum score a student can receive on any assignment or project, keeping in mind that 50% is an E.

Grading Scale:

The following grading standards will be used in this class, correlate roughly to the mastery rubric above using the latter grades:

Percentages for Letter Grade

Grade	Minimum Percent
A	93
A-	90
B+	87
B	83
B-	80
C+	77
C	73

C-	70
D+	67
D	63
D-	60
E	0

Assignment Categories

Activity	Weight
Codio Readings	not graded
Watching Videos and other Supplements	not graded
Codio Practice Labs	not graded
Codio Weekly Exercises	17%
Codio Programming Projects	50%
Codio Midterm Exam	12%
Codio Final Exam	19%
Accountability	2%

Activity	Weight
Extra Credit	2%

Academic Dishonesty: Copying someone else’s code and submitting it as your own is asking for a grade you did not earn and claiming mastery of skills that you have not demonstrated. We will use a plagiarism tool on your code in this class. This includes studying in groups and writing code together--two students MAY NOT submit the same code that they wrote together, nor code "given to you" by an IA in the lab.

You are not alone, though we do expect you to learn to solve problems. If you find yourself stuck on something, contact the instructor or a classmate, your course I.A. or the CS Tutor Lab for help. Class time, Office Hours, Discord and Teams are there for your support; please use them. If you need to, set up an appointment for help. These projects should be rewarding and instructional, not frustrating and demoralizing—but I don’t know when or how to help unless you ask.

Late Work Statement: All work is due on the date assigned before the stoke of midnight: If an assignment is due on Sep 8, it is late at 12:00am on Sep 9. There is no grade penalty for late work, and assignments may be resubmitted with corrections for regrading. If you find yourself routinely submitting work 3-5 business days late, this will likely negatively affect your ability to learn later material and you may struggle to do well in the course.

Statement on using ChatGPT (and similar tools) for this class: Use ChatGPT as a Learning Assistant, not an Oracle. ChatGPT can significantly streamline the learning process for students in a programming class, but it's crucial to ensure that it doesn't inadvertently promote dependency or shortcut genuine understanding. The tool can provide immediate help on complex topics and code troubleshooting. Many companies are expecting students to graduate knowing how to use it to generate code, so using it can no longer be considered cheating or plagiarism. Nevertheless students are encouraged to independently problem solve and think critically. These most important skills in programming are best developed through a balance of self-led exploration and learning with an assistant.

Assignment and Assessment Descriptions

Assignments

There are two types of assignments for this course: **Reflective and Evaluative.**

Reflective Assignments

Reflective assignments give you an opportunity to reflect on your own learning--in this case the important questions revolve around whether your program design and problem-solving skills ar improving over time.

- **Pretest:** In week 1, you have Image Processing exercise as an assessment of general readiness for CS 1410. If you can complete it comfortably on your own within 1 hour, then you are likely OK. You don't have to pass this exercise to stay in the class, but if you cannot do it, you should have a serious talk with your instructor.
- **Self-test:** The level to which you feel comfortable experimenting with non-graded snippets given in reading material in Codio and how easy it is to complete non-graded Practice Labs gives you a low-stakes way of testing yourself and possibly stretching yourself in the class.
- **Practice Labs:** low stakes or additional coding practice within Codio. These labs are scored by Codio, but not included in your final grade.
- **Spiral Learning with Projects:** As in CS 1400, the curriculum in CS 1410 is not linear, but instead is a spiral. We introduce new ideas and revisit them over and over during the semester. The projects intentionally require you remember and use concepts covered earlier in the course. Most students realize that things which were difficult in week 3 are now easy in week 12 because that which they persist in doing often becomes easier.

Evaluative Assignments: Exercises and Projects

Coding Exercises (Module 1 - Module 6): Practice coding skills in Codio. These assignments are scored in Codio and count toward your final grade.

Projects (Module 7 - end): Projects include both single-session tasks and multi-week assignments where students revisit and enhance the same code using new skills. This spiral learning approach reinforces and builds upon prior coding knowledge.

Key projects include:

- **Dessert Shop Project:** Comprising 12 small parts, each designed to be manageable and progressively build coding skills.
- **GUI Project using Kivy**

Since each part of the Dessert Shop Project is small, it is manageable to complete two parts in a week.

Evaluation: Project code will be manually graded. Please *do not* change the names of any provided files, functions or classes within the code--doing so makes submissions harder to grade. Projects may be manually manually graded, autograded by Codio or a combination of both. All other exercises are autograded by Codio.

Discussions

Discussions are unique in this course and designed to facilitate chatting with other coders (and me as your instructor). Discussions take place within a common coding community such as Discord or

Teams. You will be accountable for these discussions by submitting a "quiz" Accountability Exercise stating you've participated.

We anticipate that many students joined Discord in CS 1400, but if not and you want to for 1410, the join link is **bit.ly/onstudy** (http://bit.ly/onstudy) . Go to the #course-list channel and click the box next to cs1410.

Discord and Teams: Do not post spoilers nor executable code.

Communication within the Course

Mechanisms for getting help, communicating with others:

- Message the instructor. Make sure you know what the preferred contact method is.
- Talk to the Instructional Assistant assigned to your class section. Make sure you know who that is.
- Go to the CS Tutoring Lab in CS 726
- Initiate discussions in MS Teams or Discord OnStudy server for help or topics outside of class time. This is for your convenience and enrichment but will not be graded.

Assessments

- A midterm exam
- A final project exam

All exams are given in Codio but linked in Canvas in the Exams module on the Module Page**.** Each exam has instructions for taking it.

Course Schedule

Course Schedule (https://uvu.instructure.com/courses/489064/pages/course-schedule)

UVU Policies and Resources

Policies and Procedures (https://www.uvu.edu/otl/students/policiesandprocedures.html)

Student Success Resources (https://www.uvu.edu/otl/students/index.html)

Students needing accommodations due to a disability, including temporary and pregnancy accommodations, should contact Accessibility Services at **accessibilityservices@uvu.edu (mailto:accessibilityservices@uvu.edu)** or 801-863-8747 located in LC 312. To request ASL interpreters, please contact Katie Palmer at **katiep@uvu.edu (mailto:katiep@uvu.edu)**

Withdrawal Policy

Accessibility Services

Academic Integrity

Title IX Statement

Religious Accomodations Statement

UVU Honor Code

Technology Support Services

For 24/7 technical support contact [Instructure's Canvas Support Live Chat (https://cases.canvaslms.com/liveagentchat?chattype=student&sfid=001A00000085cNxIAI)](https://cases.canvaslms.com/liveagentchat?chattype=student&sfid=001A00000085cNxIAI)

☎ (385) 204-4930 (Available 24/7)

Fair Use Act Disclaimer

This site is for educational purposes only.

Fair Use

Copyright Disclaimer under section 107 of the Copyright Act of 1976, allowance is made for “fair use” for purposes such as criticism, comment, news reporting, teaching, scholarship, education and research.

Fair use is a use permitted by copyright statute that might otherwise be infringing.

Fair Use Definition

Fair use is a doctrine in United States copyright law that allows limited use of copyrighted material without requiring permission from the rights holders, such as commentary, criticism, news reporting, research, teaching or scholarship. It provides for the legal, non-licensed citation or incorporation of copyrighted material in another author’s work under a four-factor balancing test.

UVU Department of COMPUTER SCIENCE
UTAH VALLEY UNIVERSITY