COMP 698 Special Topics: Data Structures Fundamentals (replaces COMP 505, to become COMP 525)

BASIC INFORMATION

Course

Name:	COMP 698 Special Topics: Data Structures Fundamentals (to become COMP 525)
Credits:	4
Prerequisites:	COMP 425 or equivalent
Term:	Spring 2016
Class Meetings:	Tuesday, 9 - 11:50
Location:	P132
Course Site:	https://goo.gl/4ar5sc

Instructor

Name:	Mihaela Sabin, Ph.D., Associate Professor of Computer Science
Program:	Computing Technology, Division of Science and Technology, UNH Manchester
Office:	P137
Phone:	603 641 4144
Email:	mihaela.sabin@unh.edu
Office hours:	Tuesday 12:00 PM - 1:00 PM, or by appointment.

How to get in touch with me

There are three ways to get in touch with me:

1. Course-related communication outside class takes place exclusively on the class forum.
2. For in-person, one-on-one communication, see me **after class** or **during office hours**, or **by appointment**.
3. If you have personal issues, questions, or concerns see me or email me at mihaela.sabin@unh.edu. I'll reply no later than next weekday.

Course Description

Data structures and algorithms are fundamental to developing solutions for computational problems. In this course students design and implement data and functional abstractions; analyze and select appropriate data structures to solve computational problems; practice programming and software development techniques to implement computational solutions.
4 cr. Prerequisites: COMP 425 or equivalent.

Resources

Textbooks

Think Python: How to think like a computer scientist by Allen Downey

- http://www.greenteapress.com/thinkpython/
- Available through the UNH Library, Databases, Safari Books online
- Free PDF version adapted to Python 3 by Michael Karp and Lyle McGeoch is available at

http://www.cs.amherst.edu/lam/thinkpython.pdf (D) or in the course repository

Learning with Python 3 by Peter Wentworth, Jeffrey Elkner, Allen B. Downey, and Chris Meyers

- http://openbookproject.net/thinkcs/python/english3e/ (W)

Interactive edition of **How to think like a computer scientist** gives practice with solving computational problems using Python:

- http://interactivepython.org/runestone/static/comp525/index.html, Brad Miller & David Ranum (M)

Development Tools

- **Python 3.5.1** from https://www.python.org
- **Sublime Text 2** editor (cross-platform)
- **cmdr** terminal utility (Windows), **bash** (Linux), **terminal** (OS X and Linux)

Communication and Collaboration Tools

Because of the highly collaborative nature of the course, which values sharing and openness, we'll be using a variety of online tools that support these values: collaboration, sharing, and openness.

- **Google Drive** for teaching resources and student learning portfolios
- **Piazza** for the class forum (until everybody is on board with the new UNH course management system)

Tech Consultants

Computing Technology department has tech consultants who are available to help with software configuration and other technical questions you might have. You'll find them in the Tech Consultancy Workroom 124.

Teaching Resources

Teaching materials and resources will reside in a publicly shared Google Drive folder, unh-2016-spring-comp525. It includes syllabus, schedule, reading and homework assignments, and practice exercises and problems.

Learning Portfolios

All your work is uploaded to your student learning portfolio. You will use Google Drive and GitHub to create and maintain your portfolio for everything you create in this course: documents, presentations, and code.

Class Forum

The class forum is hosted by Piazza. All outside class participation takes place in the forum. Piazza is highly catered to getting you help fast and efficiently from peers and myself. Rather than emailing questions, I encourage you to post your questions on Piazza.

- Sign up at https://piazza.com/unh/spring2016/comp525

Find our class forum at https://piazza.com/unh/spring2016/comp525/home

Instructional Approach

"*For the things we have to learn before we can do them, we learn by doing them*", Aristotle, ~350 BC

Learning in this class depends heavily on ***active participation*** and ***open collaboration*** in and outside class. The

course has 15 weeks with Tuesday 2:50 hour class meetings. You are expected to study 6-8 hours outside class every week.

Learning activities are structured by **blending *in-person* and *online time*** to engage with the course content:

- During class time we participate in discussions, presentations, solution review, and guided lab activities.
- Outside class time is for studying concepts and techniques, applying them to solve problems, doing homework, givinging feedback to peers, reflecting on one's work, collaborating with peers, and working on team projects.
- Online means of communication and collaboration include the class forum, teaching resources repository, and learning portfolios.
- In-class collaboration uses pair programming and group deliberations for designing, coding, and discussion.

GOALS and LEARNING OBJECTIVES

Course Goals

The goal of the course is to combine the best practices in doing math, engineering, and science to develop ***computational thinking practices***: how to solve problems with computational approaches and computing devices and tools.

Solving computational problems means:

- expressing mathematical ideas;
- designing and building computational systems that apply mathematical and computer science concepts, meet human needs, and comply with all sorts of constraints; and
- observing the system behavior, forming hypotheses about its behavior, and debugging, testing, and improving the system.

Problem solving with computations is about formulating problems, thinking creatively about solutions, and expressing those solutions using programming and computing tools.

Learning Objectives

Upon completion of this course students should be able to:

1. Design and implement data and functional abstractions;
2. Analyze and select appropriate data structures to solve computational problems;
3. Practice programming and software development techniques to implement computational solutions.
4. Communicate and collaborate with others to achieve a common goal or solution.

Attitudes and Dispositions

"Success is going from failure to failure without loss of enthusiasm", Winston Churchill

Achieving the course learning objectives will, hopefully, form the following attitudes and dispositions:

- Tolerate uncertainty and ambiguity
- Persist in working with difficult problems
- Adapt, adjust, change course, be flexible as needed
- Have confidence in dealing with complexity

COURSE REQUIREMENTS

Learning Portfolios and Practice

All your work is uploaded to your student learning portfolio for this course. You will use the Google Drive (and possibly GitHub services) to create and maintain your portfolio. Portfolio work includes: **labs**, **homeworks**, **reflections**, **practice**, and the **team project**. Portfolios will be evaluated weekly during the semester. Portfolio evidence of practice outside class with lab assignments completion counts **3 points** to the final grade.

Homework Assignments

There will be **four** outside class homework assignments, each **4 points**, for a total of **16 points** of the final grade. Homework assignments must be completed and uploaded to your portfolios **before** class (no later than midnight, 11:59 PM, the day before the class). Homework assignments require completion of assigned readings, programming practice, solving problems, Python coding, asking questions, and giving feedback.

Learning Reflections

Reflecting on your learning experience is facilitated by **weekly reflections** you write in your portfolio. There are **12** required reflections (and 3 optional) for a total of **12 points**. Weekly reflections must be completed and uploaded to your portfolios **before** class (no later than midnight, 11:59 PM, the day before the class). Reflections require full sentences to four prompts and guiding questions. No credit is received for big, general statements that say little about your specific experience. To receive credit you must describe very specific instances, whether a challenge, an achievement, or collaboration situation. Descriptions must be followed by explanations.

Collaboration and Practice

You are required to collaborate **in class** with your partner using **pair programming**, with your peers through **discussions** and **demonstrations**, and with your team members while working on the project. Practice outside class is facilitated by the **class forum**. Class forum contributions are questions, answers, follow-ups, or edits to other contributions.

Team Projects

There will be a **team project** to investigate an important category of software developed with the Python programming language: ***web frameworks***. A web framework is a collection of modules that allow developers to write web applications by focusing on the application logic instead of the intricacies of how the web app's components communicate among themselves, with the web server, and with the user. A non-trivial web app uses different kinds of abstractions, stacked upon each other. That's why web frameworks are known as **full-stack frameworks** for supplying components for each layer in the stack .Typical components in full-stack frameworks are: HTTP application server, storage mechanism, template engine, request dispatcher, authentication module, and more. Python has been used to also create **non full-stack frameworks**, which, at minimum, provide just an HTTP application server.

Teams will be formed to work on projects that study and evaluate web frameworks written in Python, and use a particular Python-based web framework to support the development of a web app. Project artifacts include project logs (**6 points**), codebase (**6 points**), URC poster (**3 points**), presentation (**3 points**), and final report (**3**

points), for a total of **21 points**. All project artifacts must be completed and uploaded to your portfolios **before** class (no later than midnight, 11:59 PM, the day before class). Timely communication and open collaboration with your team members factor into the grading of your project work.

Exams

There will be a **midterm exam** (7th week of the semester) and comprehensive **final exam** (14th week of the semester). Each counts **24 points** to the final grade, for a total of **48 points.** The exams are closed books, paper and pencil based, and assess problem solving skills, application of concepts, and demonstration of programming techniques.

GRADING AND EVALUATION OF STUDENT WORK

To learn in this class you do homework assignments, work on the team project, collect your work in the learning portfolio, take a midterm and final exam, contribute to the class forum, and reflect on your weekly progress.

Final grade is calculated as follows:

- **4 homework assignments @ 4 points** each, for a total of **16 points**
- **12 weekly reflections @ 1 point** each, for a total of 12 points.
- **Lab practice, 3 points**
- **Team project, 21 points**, broken down into:
    - **3 logs @ 2 points** each, for a total of **6 points**
    - **codebase, 6 points**
    - **URC poster, 3 points**
    - **final presentation, 3 points**
    - **final report, 3 points**
- **Midterm exam** and **final exam @ 24 points** each, for a total of **48 points**.

COURSE POLICIES REGARDING STUDENT BEHAVIOR

Attendance

Attendance is taken every class. You are responsible for attending all classes and expected to abide by the **University Policy on Attendance** (as stated in the UNH Student Rights, Rules, and Responsibilities).
If you miss a class, you have the responsibility to:

- Email me about the circumstances for missing the class within a week of the absence.
- Contact your peers to find out what you've missed.
- Make up the absence by including in your weekly reflection what you've done to make up the missed work.

Except for absences due to serious medical reasons or circumstances beyond your control, no more than two such makeups will be accepted. Each additional absence will lead to a reduction of 5 points from the final grade.

Late submissions and make-up exams

Policy for late submissions and make-up exams is very strict and applies only in exceptional cases of student illness, accident, or emergencies that are properly documented. A late submission or make-up exam may be

granted ONLY IF:

- You email me prior to the deadline AND
- You explain and provide evidence for the circumstances that have prevented you from meeting the class requirement.

Failing to comply with these rules results in no credit for the late submission or missed exam.

Student use of computing devices

In-class use of any computing device is not allowed unless needed for lab activities and with my permission. Use of computing devices for non-class activities is not allowed. You will be asked to leave the class if you fail to comply with these rules. Students with a learning disability that requires the use of a computing device must provide evidence from the Disabilities Services office.

STATEMENT ON ACADEMIC HONESTY

No collaboration is allowed while taking the exams. Cheating on the exam is penalized with failing the course.

Assignment submissions should be entirely your work and may not include work done by others. Collaboration on assignments is encouraged, but does not include preparing and submitting the final artifacts that are uploaded to your learning portfolio.
Failing to comply with these rules is considered a violation of academic honesty policy.
See http://www.unh.edu/vpsas/handbook/academic-honesty for more information. There are very serious repercussions if you deviate from the academic honesty policy:

- The penalty for the first occurrence of an instance of academic dishonesty and plagiarism is no credit for the assignment in question. The Associate Dean will be immediately notified of the incident.
- The second attempt is penalized with failing the course.

STUDENTS WITH DISABILITIES

UNH Manchester is committed to providing students with disabilities with a learning experience which assures them of equal access to all programs and facilities of the University, which makes all reasonable academic aids and adjustments for their disabilities and provides them with maximum independence and the full range of participation in all areas of life at UNH Manchester. Students who need to document their disability and determine any accommodations, services, or referrals should schedule an appointment with the UNH Manchester Disability Services Coordinator by calling 641-4170. For more information, please see http://manchester.unh.edu/student/disability.

TENTATIVE COURSE SCHEDULE

This is a **tentative** schedule, subject to change depending on the class pace, student learning needs, and/or unforeseen circumstances, such as school closing due to inclement weather. Check the posts on the **class forum** for up-to-date information.

Week #	Date	Computer Science "big ideas"	Due next week
1	1/26	Core concepts: data, expressions, statements	R1
2	2/2	Functions	H1, R2
3	2/9	Iterative development. Turtle graphics. Iteration with **for** loop.	R3
4	2/16	Conditionals. Django framework. Project ideas.	PL1, R4
5	2/23	Fruitful functions. Incremental development. Iteration with **while** loops. Algorithms.	H2, R5
6	3/1	String data structure. Project teams, deliverables, timeline. Project titles. ***Midterm review.***	R6
Wednesday, 3/2, Deadline to register for the Undergraduate Research Conference (URC)			
7	**3/8**	**Midterm exam**	PL2, R7
Spring Break			
8	3/22	List data structure. Project proposals	H3, R8
9	3/29	File data structure. More on lists and strings.	R9
10	4/5	Dictionary data structure. More on files.	H4, R10
11	4/12	Map, reduce, filter algorithmic patterns. Project work.	PL3, R11, URC poster
12	4/19	Project alpha release presentation. Project work.	R12
Wednesday, 4/22, 4:00 - 6:00 PM - Undergraduate Research Conference Poster Session			
13	4/26	Project work. ***Final exam review.***	R13
14	**5/3**	**Final exam**	R14, Project presentation, codebase, & report
15	5/10	Project presentations and demonstrations	