48-784 Design Computation II: Algorithms

Fall Semester 2017 Mini A2 • 6 units • Tuesdays 1.30–4:20pm (CFA 317)

Open to upper-year undergraduate and graduate students

Prerequisite: 48-782 (or instructor's permission, which may take the form of an entrance assignment)

Instructor: Ramesh Krishnamurti • ramesh@cmu.edu

Teaching Assistant: Pedro Veloso • pveloso@andrew.cmu.edu

Office Hours: TBD

Syllabus

Course description

This is a fast paced intense half-semester technical object-oriented programming course targeted at design students with an emphasis on algorithms and on producing clear, robust, testable and effective code.

This course assumes familiarity with basic Python programming language as provided by the course, 48-782 Design Computation I: Programming.

Learning outcomes

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

- develop problem-solving skills using Python applied to new problems in design
- explore exemplar form-finding algorithms
- produce clear, robust, and effective code in Python by
  - employing modular, top-down design
  - using 'appropriate' Python datatypes, structures, constructs and libraries
  - proactively designing and writing test cases to effectively test and debug code
- develop an effective programming style based on established standards, practices, and guidelines
- write a long (> 500 lines) program in Python to implement a graphics-based solution to a problem originating in design

The course consists of lectures, computer cluster instruction and assignments.

References

There is no textbook but the following are useful references.

- Timothy A Budd. *Exploring Python*. McGraw-Hill. 2009.
- Mark Lutz. Learning Python, 5th Edition. O'Reilly. 2013.
- Marina von Steinkirch (bt3) *An introduction to Python & Algorithms*, 2nd Edition, 2013.
- Michael T Goodrich, Roberto Tomassia, Michael H Goldwasser, *Data Structures and Algorithms in Python*, Wiley 2013.

Other books that are useful:

- Kenneth Lambert, *Fundamentals of Python: From First Programs through Data Structures*, 2010.
- Allen B Downey, *Think Complexity*, Version 2.0.10, Green Tea Press, 2016.
- Jason Brownlee, *Clever Algorithms*: *Nature-Inspired Programming Recipes*, 2011.
- Russell C. Eberhart, Yuhui Shi, *Swarm Intelligence*, Morgan Kaufmann, 2001
- Stuart Rusell , Peter Norvig*, Artificial Intelligence: A Modern Approach*, (3rd Edition), 2009

Design related computation books that are useful:

- Lionel March, Philip Steadman, *The Geometry of Environment*, MIT Press, 1974
- Philip Steadman, *Architectural Morphology*, Pion, 1983
- Lionel March (ed), *The Architecture of Form*, Cambridge University Press, 1976
- Christopher Alexander, *Notes on Synthesis of Form*, Harvard University Press, 1964

Online resources

- Python >> general support for programming in Python
- Lynda >> great video tutorials on all things digital. Sign in through CMU for free access

Software platform

Repl, Sublime, …

Canvas

All course material will be on Canvas. We will be using Canvas for class discussion. The system is highly catered to getting you help fast and efficiently from classmates, the teaching team. Rather than emailing questions to the teaching staff, we encourage you to post your questions on Canvas.

Course Requirements

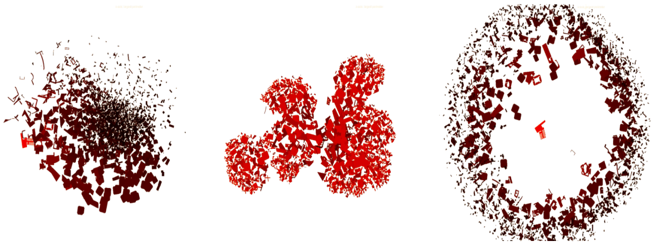
Two Assignments and a project.

Grading

Grades are based on two assignments, and a project. Each has two parts: a written part and a coding part.

The **written** part contains questions on data structures and algorithms. These questions may refer to excerpts from books, papers or websites. However, as this is not a seminar course, the questions will address concepts, analysis and implementation of algorithms.

NOTES ON THE SYNTHESIS OF FORM
CHRISTOPHER ALEXANDER
The Architecture of Form
Artificial Intelligence with Python
THE GEOMETRY OF ENVIRONMENT
Architectural Morphology
CLEVER ALGORITHMS
Data Structures & Algorithms in Python
SWARM INTELLIGENCE
Think Complexity
O'REILLY
Artificial Intelligence A Modern Approach
Packt



The **coding** part refers to the algorithms and data structures that will be implemented. It may consist of multiple exercises and challenges. The student will select a subset of these to implement. be graded (to be specified in the assignment). The code itself or an animation of the results will be graded.

Students will receive feedback on each assignment.

The breakdown for overall grade is as follows: **Project 50% Assignments 50%**

Grades are based on the following scale with a finer ± refinement:

A: 90% and over **B:** 80-89% **C:** 70-79% **D:** 60-69% **R:** < 60%

Students are not graded on a curve, however, we will consider the degree of difficulty experienced in the assignments in determining the final scores.

Prerequisite

48-782 (or instructor's permission, which may take the form of an entrance assignment to demonstrate familiarity with basic Python programming)

Policies

All university academic and student policies as set out in http://www.cmu.edu/graduate/policies/ and https://www.cmu.edu/policies/student-and-student-life/index.html apply to this course.

Specifically:

- You are expected to be on time at all lecture and lab sessions.
- Please backup your work in the cloud. We cannot accept hardware failure as a valid excuse.
- You may not copy code without citation. Copying code without citation is plagiarism.
- Late work may result in a reduced grade.

- Email should only be used for crucial queries and concerns. Please direct software related questions to Piazza or to Pedro during office/lab sessions.

Accommodations for students with disabilities

If you have a disability and have an accommodations letter from the Disability Resources office, I encourage you to discuss your accommodations and needs with me as early in the semester as possible. I will work with you to ensure that accommodations are provided as appropriate. If you suspect that you may have a disability and would benefit from accommodations but are not yet registered with the Office of Disability Resources, I encourage you to contact them at access@andrew.cmu.edu.

Student well-being and support

Carnegie Mellon University is deeply committed to creating a healthy and safe campus community including one that is free from all forms of sexual and relationship violence. To that end, University Health Services, the Office of Community Standards & Integrity, and the Office of Title IX Initiatives have partnered to expand their educational efforts for graduate students in this domain. There is an educational opportunity for all graduate students at Carnegie Mellon that reflects its commitment to sexual assault and relationship violence prevention as well as to your overall safety:

Haven Plus for Graduate Students. For more information follow the link:
https://shib.everfi.net/login/default.aspx?id=CarnegieMellonHavenPlus

Additionally, it is important to take care of yourself and try as best as possible to reduce, preferably avoid, stress. Maintain a healthy lifestyle by eating well, exercising, getting sufficient sleep and taking some time to relax. All of us benefit from support during times of struggle. There are many helpful resources available to all students on campus. Asking for support sooner rather than later is more often helpful.

If you or anyone you know is experiencing academic stress, difficult life events, or feelings like anxiety or depression, we strongly encourage you to seek support. Counseling and Psychological Services (CaPS) is here to help: call 412-268-2922 and visit their website at http://www.cmu.edu/counseling/.

Consider reaching out to a friend, faculty or family member you trust for help getting connected to the support that can help. If you or someone you know is feeling suicidal or in danger of self-harm, call someone immediately, day or night:

CaPS: 412-268-2922

Re:solve Crisis Network: 888-796-8226

If the situation is life threatening, call the police:

On campus: CMU Police: 412-268-2323

Off campus: 911

If you have questions about this or your coursework, please let us know.

Course Schedule*

* Schedule subject to changes

Week	Lecture & lab topics | design problem	Assignment
1 **Sequential data-structures and search**	**introduction to course** **introduction to analysis of algorithms –** modeling the growth of functions • asymptotic notation **sequential data-structures in Python** • **linked lists** – flexibility and references, implementing linked lists in OOP • **stacks and queues** – linked list FIFO and linked list LIFO	Assignment 1 out
10/24/17	Design – shape grammar	
2 **Trees**	**trees & applications of trees** – data representation • game states • arrangements (divisions) • layouts • binary trees **search in trees** – tree traversals: pre-, in-, post-order • backtracking • topological sort	
10/31/17	Design– rectangular dissections	
3 **Graphs**	**graphs & applications of graphs** – navigating a map • solving a maze • solving a game (min-max) • social networks **graph search** – breadth-first (BFS) • depth-first (DFS) • backtracking • [optional – A*, Djikstra & Kruskal algorithms] **graph algorithms** – graphic degree sequences, connected components, transitive closure	Assignment 1 due Assignment 2 out
11/07/17	Design – adjacency and connectivity	
4 **MVC**	**GUI** **a pattern for GUI: MVC** model-view-controller **overview of Python modules: AI** and **CG**	
11/14/17	PROJECT – Open session	

5 **Complexity** 11/21/17	**the idea of complexity** – bottom-up simulation **branching systems** – L-systems **cellular systems** – cellular automata (CA) • Game of Life (GoL) • reaction diffusion (RD) • diffuse limited aggregation (DLA) **agents** – boids Design – self-organizing city	Assignment 2 due Project out PROJECT – 5 min show
6 **Optimization** 11/28/17	an evolutionary model for optimization – searching for the best **metaheuristics** | simulated annealing • **generative algorithm • swarm intelligence** • … Design – the fittest form will survive	
7 **Project review** 12/05/17	PROJECT – a working prototype	
8 **Last day** 12/12/17	PROJECT – the final prototype	Project due