48-724 Scripting and Parametric Design

Fall Semester 2020 • 10 units • Tuesday + Thursday 1.30–2:50pm (REMOTE)

Instructors Ramesh Krishnamurti • ramesh@cmu.edu
Jinmo Rhee • jinmor@andrew.cmu.edu

Office Hours TBD

Course open to upper-year undergraduate and graduate students

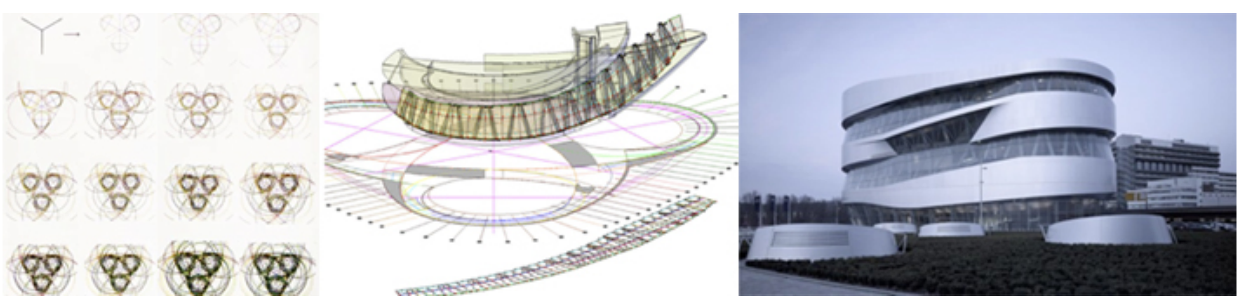
Prerequisite 62-225, 48-624, 48-783, 62-275, or any equivalent courses with basics of parametric modeling

COURSE DESCRIPTION

This course aims to prepare students to modeling geometry through scripted development of parametric schemes for architecture applications — that is, to introduce students to basic scripting with a focus on algorithms relating to form making and to reinforce and extend basic concepts of parametric modeling.

> "Parameters express architectural values in rational, functional and objective terms. As the evolution of the chosen parameters is traced over time, the project emerges as of its own accords."
>
> Ben van Berkel and Caroline Bos, UNStudio

Contemporary approaches to modeling architectural geometry are computational — this is reflected in designers wanting much more control over the generative process by varying the parameters; in turn, this enhances the efficiency with which they can navigate design variations, analyze design artifacts and explore design manifestation. The complexity of constructing geometry sometimes can be simplified as sequences of machining instructions, which allow geometry to be built by designated fabricating techniques and materials. In other words, designers not only generate the conceptual ideas but also design the fabrication process.



Mercedes Benz Museum, Stuttgart, 2005

Computing and fabricating non-simple and sometimes intricate geometric forms that go beyond straight-forward modeling exercise. Below we see an international competition entry based on the *Lípa*, the lime tree or *linde* (tilia), the national tree of the Czech Republic, and reflects the myth of queen *Libuše* as the female founder of the Czech nation.

New Czech National Library, Prague, 2006, Ocean North and Scheffler + Partner

In order to realize the goals set forth above, this course has two parts: firstly, to supply the basics of object-oriented programming (using Python), and secondly, to supply the basics of scripting parametric geometric constructions (using GH Python in conjunction with Rhino/Grasshopper objects).

Students are expected to be familiar with the basics of parametric modeling – i.e., Rhino/Grasshopper, since the course is an intermediate course to bridge advanced parametric design to the generative design system in the frame of computational design. Through this course, students can start to pragmatically expand their basic knowledge in the parametric design into the generative design.

The course consists of lectures, computer instruction and assignments.

LEARNING OUTCOMES

In this course the student will:

- write (simple) programs in Python to implement visually motivated solutions, viz., form generators
- become familiar with the scripting syntax, program flow, and geometry manipulation in Rhino
- model complex forms and relationships using geometric concepts and parametric tools
- model complex data flows toward desired design outcomes
- apply algorithmic thinking to design problems
- develop a sensibility for generative modeling

COURSE TOPICS

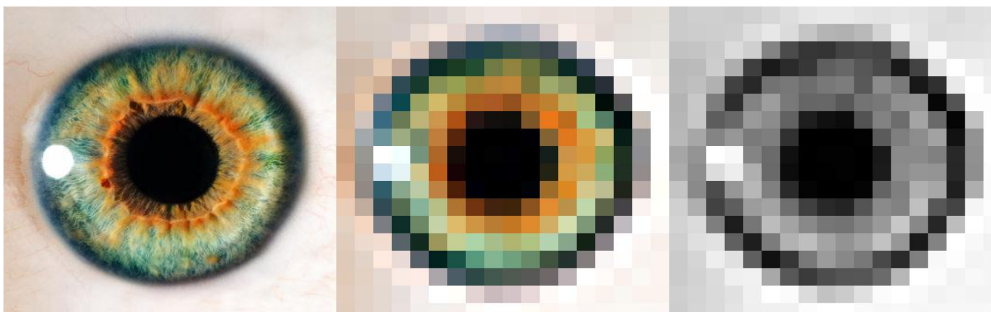
The **first part** of the course introduces object-oriented programming and algorithmic thinking through the following Python language constructs:

- sequential, conditional, and loop statements
- basic data types – strings, lists, tuples, and dictionaries
- objects and classes
- recursion and recursive functions
- simple event-driven graphics program
- simple form generators

We will use **Jupyter Notebook** for this part of the course. We illustrate these topics with two problem examples typical of those considered in this part of the course.

Problem 1– illustrating the use of strings, loops and conditionals

[Create] the function **printImage (image, width, height)** which receives a string image containing a grayscale representation of the image. Each character in the image is a digit from 0 to 9, representing a value in gray scale (0 is black and 9 is white). The following ascii characters represent the colors: " .:-=+*#%@", from black to white (the background of the terminal is black). An image is represented as an uni-dimensional sequence of characters, but of two dimensions: width and height. Each line is organized as a sequence of width characters separated by " " (the image does not come with these spaces between characters).



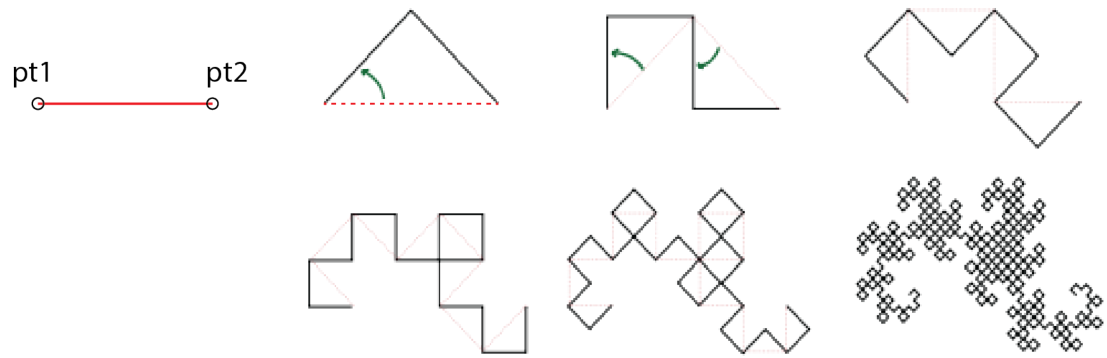
The function should generate the following result:

```
image =
"778788888888888888777788776677888888777775322233578888777622224443237999776325545545422899774256544455661399773466410036554179764555200004666169768855100001665259757865100002665269753565200004676279863456532246664389885246665566752799888425666666525999888731245443258999888888633334799999888889987789999999888899999999999999
"
width = 18
height = 18
print(printImage(image, width, height))

# # % # % % % % % % % % % % % % % %
# # # # % % # # * * # # % % % % % %
# # # # # + - : : : - - + # % % % %
# # # * : : : : = = = - : - # @ @ @
# # * - : + + = + + = + = : : % @ @
# # = : + * + = = = + + * * . - @ @
# # - = * * = .     - * + + = . # @
# * = + + + :         = * * * . * @
# * % % + + .         . * * + : + @
# + # % * + .         : * * + : * @
# + - + * + :         = * # * : # @
% * - = + * + - : : = * * * = - % @
% % + : = * * * + + * * # + : # @ @
% % % = : + * * * * * * + : + @ @ @
% % % # - . : = + = = - : + % @ @ @
% % % % % * - : - - - = # @ @ @ @ @
% % % % % @ @ % # # % @ @ @ @ @ @ @
% % % % @ @ @ @ @ @ @ @ @ @ @ @ @ @
```

Problem 2 – illustrating the use of recursion and vectors

[Implement without using loops] a recursive function **dragon (line, n, isLeft = True)**, to develop a general version of the Heighway Dragon fractal. In the Heighway dragon, one starts from a base line, then each segment is replaced by 2 segments with a right angle and through a rotation of 45° alternatively to the left and to the right.



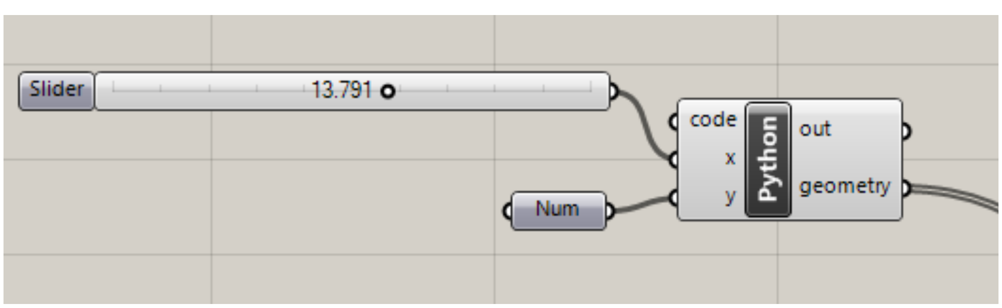
The function should produce results like

```
print(dragon([[0, 0], [100, 0]], 1))
>[[[0, 0], [50.0, -50.0]], [[50.0, -50.0], [100, 0]]]

print(dragon([[0, 0], [100, 0]], 2))
>[[[0, 0], [0.0, -50.0]], [[0.0, -50.0], [50.0, -50.0]], [[50.0, -50.0], [50.0,
0.0]], [[50.0, 0.0], [100, 0]]]

print(dragon([[0, 0], [100, 0]], 3))
[[[0, 0], [-25.0, -25.0]], [[-25.0, -25.0], [0.0, -50.0]], [[0.0, -50.0], [25.0,
-25.0]], [[25.0, -25.0], [50.0, -50.0]], [[50.0, -50.0], [75.0, -25.0]], [[75.0,
-25.0], [50.0, 0.0]], [[50.0, 0.0], [75.0, 25.0]], [[75.0, 25.0], [100, 0]]]
```

The **second part** of the course focuses on generative geometry construction — customizing procedures for generative design via scripting and/or the use of specific plug-ins. The primary graphical scripting construct is the **Grasshopper Python component**, which take inputs and produces graphical output, for purposes of display, modeling or making.



Grasshopper Python Component

Constructs may be simple or more involved; both kinds are considered in this course where students learn:

- the basics of RhinoCommon SDK
- ways to handle geometry through the scripted development
- difference between the visual computing in Grasshopper and the scripted development
- algorithms for geometric constructions based on
    - rule-based modeling by fractals, L-systems and grammars
    - simulation-based modeling by agents, swarm intelligence, evolutionary algorithms, genetic algorithms

Students will produce patterns such as

L-System, Surface Subdivision, Aperture Openings **v**arying **a**ccording to Height, Various Growth Patterns

COURSE RESOURCE

Course material will be on Canvas. All subsequent references to 'Canvas' refers to the course Canvas website.

Canvas has a section for Discussion which will be used for Q&A sessions. Students can post questions about concepts and assignments. Other students can reply to the post to share their experience or ideas and logic about a problem. However, **uploading any file which includes code is prohibited**.

Class will be **conducted entirely remotely** through Zoom. The meeting link will be announced in Canvas.
Links for each class in this course are provided in the Syllabus section on Canvas.
'Jump to Today' tab in the section will be helpful in finding the class link.
Please refer to the section below on Using Zoom.

COURSE REQUIREMENTS

We will work with Jupyter Notebooks and Grasshopper in Rhinoceros. Rhinoceros version should be 6.0 or higher. For this course, students are required to have Rhinoceros and Grasshopper installed on their laptops. However, instructions will be provided in class on installing Jupyter Notebooks.

GRADING

Grades are based on thirteen class assignments and a project which are weighted as follows:

Class assignments: 55% Project: 45%

The precise breakdown is given in the course schedule. Students will receive feedback on each assignment.

- Grades are based on the following scheme:

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

- Students are not graded on a curve.
- Any late submission will have 5% deduction a day. For example, if a student submitted an assignment two days late, then 90% will be the best score that students have.
- All assignment should be submitted on Canvas.

TEXTS AND READINGS

There is no textbook for this course but reading material will be provided.

The following are useful resources.

- Arturo Tedeschi. *AAD – Algorithms-Aided Design*. Len Penseur Publisher, 2014.
- Robert Woodbury. *Elements of Parametric Design*. Routeledge, 2010.
- Helmut Pottmann, Andreas Asperl, Michael Hofer, and Axel Kilian. *Architectural Geometry*. Bentley Institute Press, 2007.
- Wassim Jabi. *Parametric Design for Architecture*. Lawrence King Publishing, 2013.
- Rajaa Issa, *Essential Mathematics for Computational Design* – Third Edition. http://www.rhino3d.com/download/rhino/5.0/EssentialMathematicsThirdEdition/
- Andrew Payne & Rajaa Issa, *The Grasshopper Primer – Third Edition* https://modelab.is/download/grasshopper-primer-third-edition-2/#pkg_10515_57c89cfd7be4d
- Zubin Khabazi, *Generative Algorithms*. https://labdigifab.files.wordpress.com/2014/03/generative-algorithms.pdf
- Jinmo Rhee and Eddy Man Kim. *Digital Media Series, Grasshopper*. 2020
- Jinmo Rhee and Eddy Man Kim. *Digital Media Series, Rhinoceros*. 2019

Other readings will be added to this list.

ONLINE RESOURCES

- GH Gateway >> for access to tutorials, videos, and other resources
- GH Forum >> for answers to specific GH questions from the GH community
- Grasshopper Primer >> great reference material to have on hand
- Rhino Scripting Gateway >> for access to tutorials, example code, etc.
- Rhino Python Forum >> for answers to specific scripting questions in Rhino
- RhinoCommon SDK >> have this open for reference while scripting in Rhino
- Python >> general support for programming in Python
- Lynda >> great video tutorials on all things digital. Sign in through CMU for free access
- Stack Overflow >> an all-purpose online forum to ask programming questions

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 (remotely).
- 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.
- Email should only be used for crucial queries and concerns. Please direct software related questions to Jinmo Rhee during office/lab sessions or use Canvas Discussion section.
- The Canvas Discussion is not a place to catch up on missed classes. In necessary circumstances where you are unable to attend class, please make sure to inform us via email and we will address the situation accordingly.

REMOTE INSTRUCTION

This semester involves regular use of technology during class. Research has shown that divided attention is detrimental to learning, so we encourage you to close any windows not directly related to what we are doing while you are in class. Please turn off your phone notifications and limit other likely sources of technology disruption, so you can fully engage with the material, each other, and me. This will create a better learning environment for everyone.

IN PERSON INTERACTIONS: Although this semester, this is a remotely conducted class and there may be occasions which require face-to-face interactions with Jinmo, in which case all university recommendations on face-to-face contact must be observed.
See https://www.cmu.edu/coronavirus/students/tartans-responsibility.html.

USE OF ZOOM IN THE CLASS (INCLUDING USE OF VIDEO)

In our class, we will be using Zoom for synchronous (same time) sessions. See the Zoom link on Canvas.

Please make sure that your Internet connection and equipment are set up to use Zoom and able to share audio and video during class meetings. (See https://www.cmu.edu/computing/start/students.html for information on the technology you are likely to need.) Let Jinmo know, as early as possible, if there is an issue with your technology set-up to sort it out.

SHARING VIDEO: In this course, being able to see one another helps to facilitate a better learning environment and promote more engaging discussions. Our default will be to expect students **to have their cameras on** during lectures and discussions. However, we also completely understand that there may be reasons students would not want to have their cameras on. If you have any concerns about sharing your video, please email us as soon as possible and we can discuss possible adjustments.
Note: You may use a background image (preferably static image) in your video if you wish; just check in advance that this works with your device(s) and internet bandwidth.

- During our class meetings, **please keep your mic muted** unless you are sharing with the class or your breakout group.
- If you have a question or want to answer a question, please use the chat or the "raise hand" feature (available when the participant list is pulled up). We will monitor these channels in order to call on students to contribute.
- Our synchronous meetings may involve breakout room discussions, and those will work better if everyone in your small group has their camera turned on. During large group debriefs, you may keep your video off.

RECORDING OF CLASS SESSIONS

All synchronous classes will be recorded via Zoom so that students in this course (and only students in this course) can watch or re-watch past class sessions. Please note that breakout rooms will not be recorded. However, chats are recorded. We do not encourage private chats during recorded Zoom sessions, instead we recommend that you send us emails. We will make the recordings available on Canvas as soon as reasonably possible after each class session. Recordings will live on Canvas. **Please note that you are not allowed to share these recordings**. This is to protect your FERPA rights and those of your fellow students.

RESPECT FOR DIVERSITY

This has long been a graduate course in the School of Architecture, and as such has always had students from many diverse backgrounds. It is my intent that all students irrespective of background or perspective continue to be well served by this course, that students' learning needs are addressed both in and out of class, and that the diversity that you bring to this class be viewed as a resource, strength and benefit. It is my intent to present materials and activities that are respectful of diversity in all its forms. Your suggestions are encouraged and appreciated. Please let me know ways to improve the effectiveness of the course for you personally or for other students or student groups. In addition, should any of our class meetings conflict with religious events, please let me know so that we can make suitable alternate arrangements for you.

ACCOMMODATION FOR STUDENTS WITH DISABILITY

If you have a disability and are registered with the Office of Disability Resources, I encourage you to use their online system to notify me of your accommodations and discuss your 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:

It is important to take care of yourself. Do your best to maintain a healthy lifestyle this semester by eating well, exercising, avoiding drugs and alcohol, getting enough sleep and taking some time to relax. This will help you achieve your goals and cope with stress.

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 Service (CaPS) is here to help: please call 412-268-2922 or 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 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

COURSE SCHEDULE*

* Schedule subject to change

Week	Topic	Tuesday	Thursday	Assignment	Grading
1	Data Type	9/1 Getting Started Intro to course Syllabus What is Python? Installing Jupyter	9/3 Basic Data Type Lecture Data type Recitation Boolean, Number, String, List, Tuple Exercise Simple Plan Drawing	A1 5 practice 1 Exercise **9/3** A1 Released	 1 3 **4**
2	Operation	9/8 Operation Part 1. Recitation Boolean, Number, String	9/10 Operation Part 2. Recitation List Exercise Plan Type	A2 5 practice 1 Exercise **9/10** A1 Due A2 Released	 1 3 **4**
3	Conditionals Functions	9/15 Conditionals. Lecture Conditionals | Functions Exercise Letter Grade Converter, Discriminant, Perpendicular	9/17 Functions Recitation Basic and Advanced Functions Exercise Relation Between Two Circles	A3 5 practice 1 Exercise **9/17** A2 Due A3 Released	 1 3 **4**
4	Function with Loops	9/22 Loop Part 1 Lecture Loop Recitation For statement	9/24 Loop Part 2 Recitation While statement Exercise Circle Repulsor	A4 5 practice 1 Exercise **9/24** A3 Due A4 Released	 1 3 **4**

Week	Topic	Tuesday	Thursday	Assignment	Grading
5	Set and Dictionary	9/29 Set and Dictionary Lecture Set and Dictionary Recitation Set	10/1 Set and Dictionary Exercise Recitation Dictionary Exercise Diffusion Limited Aggregation	**A5** 5 practice 1 1 Exercise 3 **10/1** A4 Due A5 Released	**4**
6	Recursion	10/06 Makeup Session Q&A Session Makeup for Remote Assignment Review	10/8 Recursion Lecture Recursion Recitation Recursion Exercise Sierpinski Triangle	**A6** 5 practice 1 1 Exercise 3 **10/8** A5 Due A6 Released	**4**
7	OOP	10/13 OOP Recitation Concept, Class, Inherit	10/15 OOP Exercise Recitation OOP Example Exercise qGrowth simple version	**A7** 5 practice 1 1 Exercise 3 **10/15** A6 Due A7 Released	**4**
8	Data Structure	10/20 Elementary Data Structure Recitation Stack, Queue, Tree, Graph	10/22 GH Python Recitation Intro, GH Python Rhino and GH Review	**A8** Install Rhino 1 Read Grasshopper 1 Reference **10/22** A7 Due A8 Released	**2**

Week	Topic	Tuesday	Thursday	Assignment	Grading
9	Point and Curve	10/27 Point Lecture Point Recitation Construct Point, Point Grid, Vector Exercise Points on Parabolic Curve, Points in a Rectangular 3D Point Grid	10/29 Curve Lecture Curve Recitation Line, Polyline, Rectangle, Circle Exercise Random Triangles	**A9** Koch Curve **10/29** A8 Due A9 Released	5 **5**
10	Surface and Brep	11/3 Surface Lecture Surface Recitation Boundary Surface, Loft, Sweep Exercise Trimmed Boundary Surface Parametric I-Beam	11/5 Breps Lecture Breps Recitation Bounding Box, Joined Surface Exercise Box Representation Random Massing in GH	**A10** Responsive Facade System for CFA Addition **11/5** A9 Due A10 Released	5 **5**
11	Rule-based models	11/10 Transformation Lecture Transformation Recitation Euclidean, Parametric Transformation Exercise Circle Repulsor in GH Gravel in GH	11/12 Rule-based Modeling Part 1 Lecture Shape Grammar Exercise Line Algebra Binary Tree in GH	**A11** Ice-rays in GH **11/12** A10 Due A11 Released	5 **5**
12	Rule-based Agent-based models	11/17 Rule-based Modeling Part 2 Q&A Session Makeup for Remote Assignment Review Exercise qGrowth template	11/19 Agent-based Modeling Part 1 Lecture Agent Based Modeling Recitation with Grid Settings Exercise Cellular Automata	**A12** qGrowth in GH **11/19** A11 Due A12 Released	5 **5**

Week	Topic	Tuesday	Thursday	Assignment	Grading	
13	Agent-based models	11/24 Agent-based Modeling Part 2 Introduction of Final Project Recitation with cartesian settings Exercise Boids	11/26 **Thanksgiving Day Holiday** No Class	A13 Islamic City	5	
				11/24 A13 Released		**5**
				11/28 A12 Due		
14	Optimization	12/1 Simple Optimization Recitation Genetic Algorithm Hill-Climbing and Gradient Descent Exercise Genetic Algorithm, Hill-Climbing	12/3 Final Project Idea Presentation Theme, Topic, Idea, Strategy, Representation	FP **12/03** A13 Due **Idea Presentation** Idea Strategy	 5 10	
15	Final Project	12/8 Final Project Meetup Individual Meeting for the final project	12/10 Final Project Meetup Individual Meeting for the final project	**Final Presentation** Clear Problem and Goal Solution Quality	 5 5	
16	Final Project	12/15 Final Project Meetup Individual Meeting for the final project	12/17 Final Project Presentation	**Final Deliverable** Code PDF Video	 10 5 5	**45**
				12/19 Final Project Due		