

Practical Computational Methods for Economists (496-05 and 895-012)

Innovation Hall, Room 223; 3:00 – 4:15 (Tuesday, Thursday)

Instructor: Professor Kevin McCabe, kmccabe@gmu.edu, (o) 703-993-944.
All EMAIL correspondence should be sent to classes@brains.gmu.edu,
Undergraduates should use the subject line: ECON 496 FALL 2017.
Graduate students should use the subject line: ECON 895 FALL 2017.

Office Hours: Wednesdays, 2:00-4:00, Room 5016, Vernon Smith Hall, Arlington.

Prerequisite: Introductory Microeconomics

Course Description: The field of economics is becoming increasingly computational. This means if you want to be competitive in the job market you must learn how to think in terms of algorithms and be able to program algorithms on computers. In this course, you will learn how to program economics algorithms on a computer using the general-purpose computer language called Python. This course assumes that you have no previous programming experience.

Required Materials: Your first exposure to Python will be through the [Socratica Python Programming Tutorials](#), and the free “Learn Python” unit lessons on the [Code Academy](#) website. This materials represents around 15 hours of homework for the first twelve weeks.

In addition to the code academy lessons we will be using Jupyter notebooks. These notebooks will be available on blackboard. You should have the appropriate notebook on your computer and ready to go before each class.

Suggested Materials:

Textbook:

John Guttag, [Introduction to Computation and Programming using Python](#).

This textbook is highly recommended. It serves as a reference and explores more advanced topics once you have learned the basics. There is a free MIT course using the book which will give you access to the video lectures by the author.

Web:

Tutorials Point [website](#).

“How to Think Like a Computer Scientist”, [website](#).

Follow up Materials:

Once you have learned how to think computationally and build computer programs you will likely want to continue learn more python and how to apply it to economics.

More advanced material for economists can be found at the Quantitative Economics site hosted by Thomas J. Sargent and John Stachurski. (<https://lectures.quantecon.org/>)

Additional practice in Python programming can be found at these two websites. (<http://www.pythonchallenge.com/>) and (<https://projecteuler.net/>).

Blackboard: All additional materials will be made available on Blackboard.

How to Take This Class

In the first twelve classes, you will be introduced to Python programming. We assume you have no previous computer programming experience. Therefore, these twelve classes teach you the Python programming skills you need for the later projects. You will be given a study plan for each of these classes. It is imperative that you follow the study plan. The minimum preparation you should do is the Socratica video units and the Code Academy “Learn Python” units. You will be given a quiz on this material at the start of each class to test your basic understanding of the material. In class, we will work as a group through a Jupyter notebook to reinforce what you have learned and apply your programming skills to program economics computations. Once we have finished the group work you will continue to work on a programming exercise with a partner(s). After class, the study plan will offer several optional homework assignments from the web resources and the Guttag book to deepen your understanding.

In the remaining fifteen classes, we introduce you to a substantial framework involving several thousands of lines of code. Within this framework, you will work on three exercises to design and build a market system that can be used for both simulation and human subject experiments. This will introduce you to a substantial project involving several thousands of lines of code. From this project, you will learn the importance of designing, documenting, modularizing, and testing your software. For pedagogical reasons, we have broken the working code so you can learn how to debug code in larger projects. We have also written a new requirement for you and your team members to work on to add functionality to the framework.

Project One: You will design a Graphical User Interface (GUI) to build and look at supply and demand curves to be used in your market system.

Project Two: You will work on a spot market simulator. This project will build on project one by adding a double auction institution and programmed traders. You will build your own market agent to compete in a Double Auction tournament.

Project Three: You will design and build a market experiment in mTree using the code you built in projects one and two. You will then replace the programmed traders with human traders and run the market experiment in class.

Grading

Quizzes (40 points): At the beginning of each class there will be a short ten-minute quiz on the homework assigned before class. You may use the CSN Python Review notes during the quiz. Each quiz will be worth 5 points. There will be a total of ten quizzes and your two lowest quiz score will be dropped.

Projects (60 Points): During the semester you will be assigned three projects. Projects are built cumulatively from the assigned exercises in the Jupyter notebooks which we will be work on during class.

Undergraduates: Must complete all the basic exercises for project one and project two.

Graduate Students: Must complete all of the basic exercises for projects one, two, and three.

Expectations

The material in this course is cumulative and learning to think about programming has a relatively steep learning curve. This means you cannot binge learn this material at the end of the semester. This is true of the project exercises as well. I strongly recommend that you do the basic exercises for each project at the time they are covered in class. The quizzes are designed to give you an added incentive to learn at the proper pace.

Course Schedule: This schedule might undergo some further revision.
Unit # refer to the Code Academy (CA) “Learn Python” unit.

8/29 Variables and Assignment Statements, CA Unit One
8/31 Anaconda3, Spyder IDE and Jupyter notebooks
9/5 Strings and Console Output, Quiz on CA Unit Two
9/7 Conditionals and if statements, Quiz on CA Unit Three
9/12 Functions, Quiz on CA Unit Four
9/14 Lists and Dictionaries, Quiz on CA Units Five and Six
9/19 Lists and Functions, Quiz on CA Unit Seven
9/21 Loops, Quiz on CA Units Eight and Nine
9/26 List Comprehensions, Iterators, and lambda functions, Quiz on CA Unit Ten
9/28 Testing using assertions and exceptions
10/3 Classes, Quiz on CA Unit Eleven
10/5 File Input/Output, Quiz on CA Unit Twelve
10/10 **Columbus Day Break**
10/12 Project One: Supply and Demand Builder using the MVC pattern
10/17 Project One: Learning Tkinter
10/19 Project One: Learning Matplotlib
10/24 Project One: Debugging the Builder Code
10/26 Project One: Finishing the Builder Code
10/31 Project Two: Market System Design
11/2 Project Two: Debugging the Double Auction Institution
11/7 Project Two: Building Zero-Intelligence Traders
11/9 Project Two: Building your own Traders
11/14 Project Two: Running an Agent Based Experiment
11/16 Project Three: Building a Human Subject Market Experiment in mTree
11/21 **Thanksgiving Break**
11/23 **Thanksgiving Break**
11/28 Project Three: Building the Human Subject Interface, HTML guide
11/30 Project Three: Writing Instructions for Human Subjects
12/5 Project Three: Testing and Debugging the Experiment
12/7 Project Three: Running the Experiment

Final Rules: There are no makeup quizzes or assignments. You have been given enough leeway to be able to miss two classes and still get full credit for the course. Work must be submitted at the assigned time, and in class, or you will receive a zero grade for that work. No late work will be accepted. If you know you will not be in class arrange with Professor McCabe to turn your work in during class at an earlier date. If you have an emergency notify Professor McCabe as soon as possible and email your work to him. Upon receiving satisfactory documentation of your emergency your email work will be graded if it was emailed within twelve hours of the end of the assigned class.