CS 1501

Metaprogramming

Maxwell Patek
mtp4be@virginia.edu

Office Hours:
Sunday 5:00-6:00
Olsson 104

Course Description: Students will learn several implementations and applications of metaprogramming, starting with Python and eventually moving to other languages. Metaprogramming is the writing of programs that take in programs as input and output programs as a result. Metaprograms can sometimes even do this to themselves. Some languages, like Python, have built in features that facilitate this programming style. After this course, students will know these features, what they can accomplish, and the true power of python. More broadly, students will learn how to live DRY (don't repeat yourself).

Prerequisites: CS 111x and 2110, or Familiarity with Object Oriented Programming and Python
Credit Hours: 1 **Time:** Monday 1:00-1:50 **Room:** Thornton E316

Course Objectives:

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

1. Write object-oriented Python
2. Use functions as first-class objects
3. Use Python Inheritance
4. Write closures and decorators
5. Use and write metaclasses
6. Write dynamic classes and functions
7. Use reflection in several languages
8. Use homoiconicity
9. Develop domain specific languages

Grade Distribution:

Assignments	60%
Take Home Final Exam	20%
Attendance	20%

Course Policies:

- **Lecture**
  - Lectures will be integrated lecture-lab. ie mix of instruction and workshop style coding.
  - When we code in class, the code will be a form of attendance for the day.
  - When coding, students should have their computers out; however, I do ask that students keep computers away otherwise.
  - There will be minimal need for taking notes, as I will make all lecture materials available on Collab.
- **Assignments**
  - There will be an assignment for each week.
  - Assignments are designed to be low-pressure and fun. Don't stress over them.
  - Some assignments will be *explorations* designed to be creative, unique, and fun applications of course topics.
  - Other assignments will be *puzzles*. These puzzles are meant to be solved with metaprogramming, but no penalty will be incurred if students can solve them without metaprogramming. So that students do not stress too much over them, a certain number will be dropped.
  - Assignments will be made available as soon as possible, and students may start working as early as they wish. However, I reserve the right to make changes until the week that the assignment is officially assigned
  - Assignments will be due at the start of lecture the week after they are officially assigned.
- **Late Policy**
  - 25% off per week.
- **Final Exam**
  - Take home.
  - Will cover high-level concepts and overall paradigms.

Academic Honesty Policy:

In the real world, there is no such thing as cheating as long as you cite your sources and your sources agree to being cited. Same goes with this course. Students are encouraged to work together and google things.

Professor Sponsor

If a student has an issue with the course, grades, or instructor, he/she may contact the sponsoring professor, **Luther Tychonievich**.

Tentative Course Outline:
The weekly coverage might change as it depends on the progress of the class.

Week	**Topic**	**Assignment** (due the following week)
1	Object-Oriented Python	Exploration: Prisoners Dilema
2	Python Inheritance	Puzzle: Dependency Injection
3	Objects as Functions	Exploration: Esoteric Print
4	Closures	Puzzle: Partial Function
5	Decorators	Exploration: Decorating Contest
6	The Metaclass	Puzzle: Abstract Base Prisoner
7	Python Reflection	Puzzle: Restricted Function
8	Java Reflection	Puzzle: Breaking Visibility
9	Compile-time Computation	none
10	Homoiconicity	none
11	Domain Specific Languages	none

Assignment Descriptions:

- Prisoners Dilema

  Students will write an class that defines a strategy for the prisoner's dilemma. There will be a tournament with all on-time submissions, pitting instances of the students' classes against each other. Prizes for the best and worst strategies.

- Dependency Injection

  Students will use Python's Method Resolution Order (MRO) to make a subclass that overrides a 'super' dependency higher up in the inheritance tree. Overriding this dependency will prevent the "bomb from going off."

- Esoteric Print

  Students will override the built-in print function with an object that has some creative esoteric behavior. One requirement will be that print() should refuse to print the same thing 3 times in a row. Prize for the funniest print function.

- Partial Function

  Students will write a function that partially applies arguments to a function, returning a function bound to that partial list/dict of arguments.

- Decorating Contest

  Write a creative decorator. Prizes for the best. Bonus points for decorating your decorators.

- Abstract Base Prisoner

  Students will write a metaclass for their prisoners that will automatically add new prisoner classes to the prisoners list and will require that all prisoners implement the confess_or_no() method.

- Breaking Visibility

  I have a very secure Java gradebook application. Students will use reflection to change my private fields to public so that they can hack their grade to an A.

- Restricted Function

  Students will write a restricted function that I can't call.