CS 1501

Metaprogramming

Maxwell Patek
mtp4be@virginia.edu

Office Hours:
TBD
TBD

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:** Mondays 1:00 pm **Room:** Chemical *Engineering* Building 005

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. Understand homoiconicity
9. Conceptualize intepreters, compilers and specializers as metaprograms
10. Use compile-time computation

Grade Distribution:

Assignments	50%
Final Exam	10%
Attendance	40%

Course Policies:

- **Lecture**
  - Lectures will be integrated lecture-lab. ie mix of instruction and workshop style coding.
  - 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 most weeks.
  - Assignments are designed to be low-pressure and fun. Don't stress over them.
  - 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.
  - 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.
- **Grading**
  - Majority of grading will be by peer grading.
  - Purpose of peer grading is mainly to get feedback and to see what others came up with.
  - Peer grades will have redundancy to reduce noise.
  - Peer grades will be spot checked by the instructor.
  - Throughout the semester, students may be given the opportunity to provide feedback on their graders. The most helpful grading comments may be anonymously shown to the class as a way of reviewing the homework.
- **Late Policy**
  - To review the homework as a way of recapping the previous lecture, late submissions are not accepted.
- **Final Exam**
  - Will cover high-level concepts and overall paradigms.

Academic Honesty Policy:

I want the class to be a collaborative effort, meaning that I encourage students to work together. Each student should attempt the homework problems and submit their work. If you worked with someone else, please cite them as a resource and make sure that you understand the code that you submit. Please Google liberally!

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	Prisoners Dilema
2	Python Inheritance	Dependency Injection
3	Objects as Functions	Esoteric Print
4	Closures	Partial Function
5	Decorators	Fibonacci
6	The Metaclass	Meta Prisoner
7	Reflection	Restricted Function
8	Meta Java	none
9	Compile-Time Computation	none
10	Code Generation	Problem Generator part 1
11	Language Metaprogramming	Problem Generator part 2
12	Homoiconicity	TBD
13	Applications	TBD
14	Conclusion	none