Spring 2019

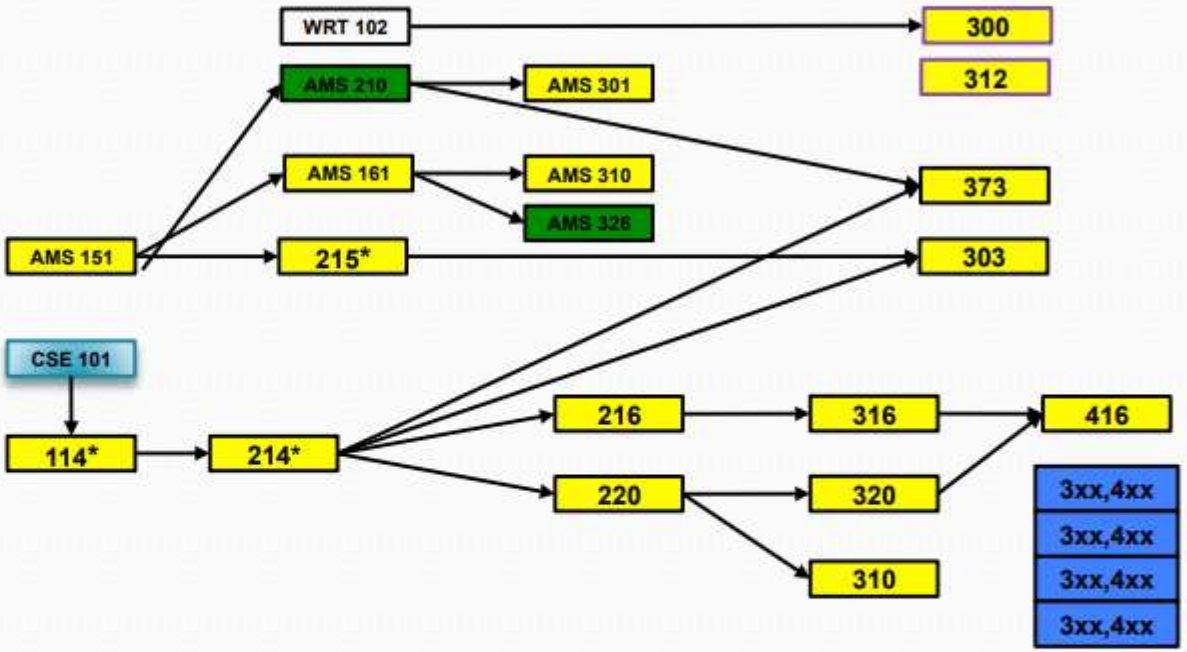
CSE 216 : Programming Abstractions

LECTURE 0 – COURSE INTRODUCTION

CS curriculum (Fall 2018)

CSE 101 Intro to Computational Thinking
CSE 114 CS I
CSE 214 CS II
CSE 215 Foundations of CS
CSE 216 Programming Abstractions
CSE 220 System Fundamentals I
CSE 300 Technical Communications
CSE 303 Intro to Theory of Computation
CSE 310 Computer Networks
CSE 312 Legal, Social, and Ethical Issues in Information Systems
CSE 316 Fundamentals of Software Development
CSE 320 System Fundamentals II
CSE 373 Analysis of Algorithms
CSE 416 Software Engineering

AMS 151 Applied Calculus I
AMS 161 Applied Calculus II
AMS 210 Applied Linear Algebra
AMS 301 Finite Mathematical Structures
AMS 310 Survey of Probability and Statistics
AMS 326 Numerical Analysis



Required

3xx, 4xx electives (4 courses)

→ Prerequisite

* Must be taken at SUNYK

One of the two

CSE 216 first offered in Spring 2019

Plan: 1xx, 2xx: offered every semester
300, 312: every other year
316, 320, 416: at least once a year
303, 310, 373: once a year
3xx, 4xx electives: whenever we can

2

Course Information

CSE 216 : Programming Abstractions

Course webpage: https://ppawar.github.io/CSE216-S19/index.html

Meetings: Lecture: Tue/Thu 3:30-4:50 PM

Recitation: Mon: 12:30-1:50PM

place: B207

Prerequisites: C or higher in CSE214, CSE major

Staff

Instructor

- Pravin Pawar
- Office: B424
- Email: Pravin.pawar@sunykorea.ac.kr
- Phone: +82-032-626-1227
- Office Hours: *Tue/Thu 10:30 AM - 12:00 PM, Wed 5:00 PM - 6:30 PM*

Teaching Assistants

- Graduate (grading TA):
- Undergraduate (tutoring TA):

Discussion Forum

As a technical discussion forum, we will be using Piazza on an experimental basis. Following is the signup link of CSE216 on Piazza:

http://piazza.com/sunykorea.ac.kr/spring2019/cse216

Piazza tutorial for students:

https://rutgers.instructure.com/courses/35/pages/piazza

Everyone is expected to:

- Use this platform responsibly, and maintain social decorum at all times, and
- Not use this platform for non-technical (especially non-course related) issues.

Course Overview

Course description:

- Intermediate-level programming language concepts and paradigms, including functional programming, object-orientation, basics of type systems, event-driven programming, program and data abstractions, and modularity. Includes weekly recitations, which provide students with experience in the practice of programming in a variety of high-level languages such as Java, Scala, Haskell, Python or Javascript.

Course outcomes:

- An understanding of programming paradigms and tradeoffs
- An understanding of functional techniques to identify, formulate and solve problems
- An ability to apply techniques of object-oriented programming in the context of large-scale software development
- An ability to construct user interfaces using event-driven programming

Expectations

Ability to write programs of a few hundred lines of code in the Java programming language

Understanding of fundamental data structures, including lists, binary trees, hash tables, and graphs

Ability to construct simple command-based user interfaces, and to use basic I/O for input and output of data

A solid foundation of basic mathematical and geometric reasoning

Major Course Topics

Programming Paradigms
- Paradigms
- Scope and bindings
- Parameter passing
- Type systems

Functional problem-solving using a high-level functional language
- Recursion
- Higher-order procedures
- Streams and Lazy Evaluation
- Modularity

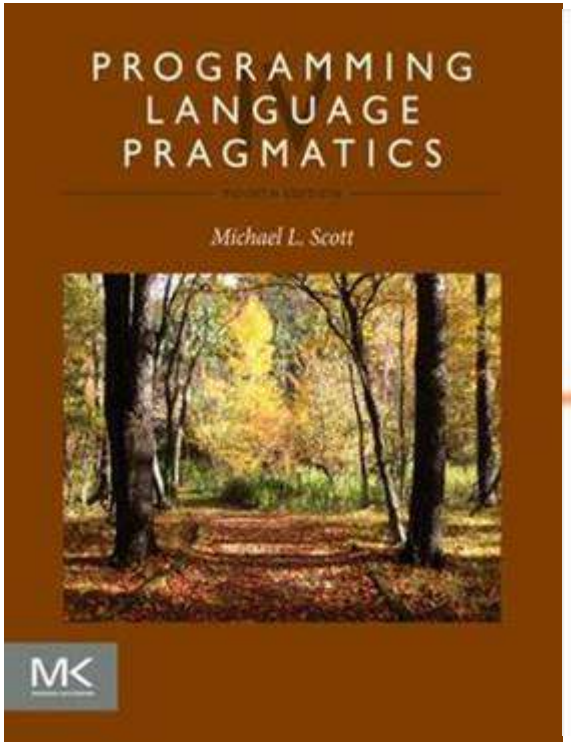
Cross-cutting concepts
- Unit testing and Integration testing
- Multi-paradigm programming

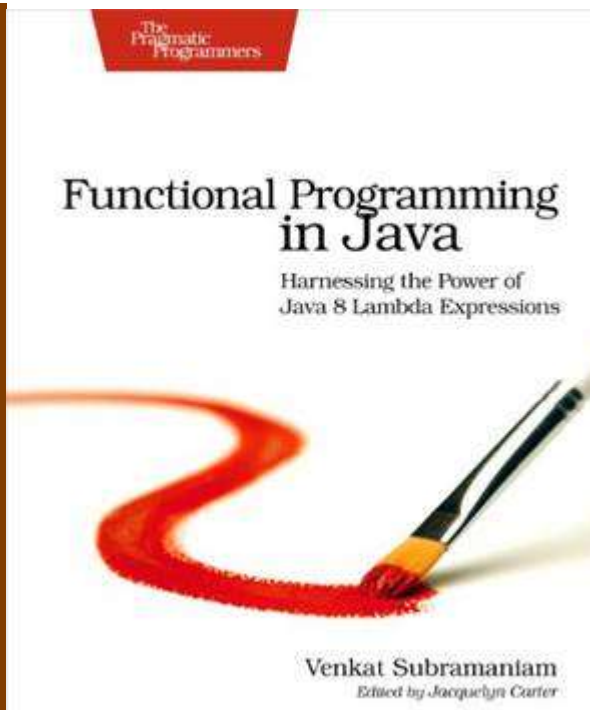
Object-oriented Design and Programming
- Abstraction and encapsulation
- Polymorphism and Inheritance
- Class hierarchy
- OO design principles (is-a, has-a, etc.)

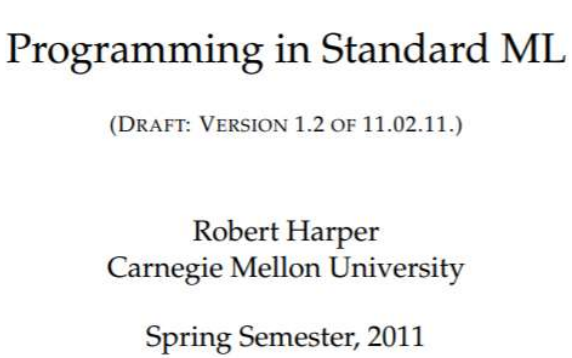
Event-driven and reactive programming
- Graphical User Interface Programming
- Multi-threading

Languages to be used
- Java
- SML
- Python
- Javascript

Textbook


PROGRAMMING
LANGUAGE
PRAGMATICS
Michael L. Scott
MK


The Pragmatic Programmers
Functional Programming in Java
Harnessing the Power of Java 8 Lambda Expressions
Venkat Subramaniam
Edited by Jacquelyn Carter


Programming in Standard ML
(DRAFT: VERSION 1.2 OF 11.02.11.)
Robert Harper
Carnegie Mellon University
Spring Semester, 2011

Assignments

Over the course of the term you will be required to solve computational problems by writing software in Python

These homework assignments will reinforce concepts from class and have you explore new concepts, too

All work will due on fixed dates and times

All work will be completed on an individual basis (write your own code) *unless otherwise instructed*!

You will use **Blackboard** to submit your completed assignments

Please start early on the assignments!

Late-work Policy

Assignments must be turned in by the due date and time.

- Any part of an assignment that's late means the entire assignment is late.
- If your assignment is incomplete or not entirely working by the due date, turn in what you have to get some partial credit.

If you have an emergency situation, email me before the due date and I may be able to work something out

Bottom line: Plan ahead, start early!

Cooperation vs. Copying

Cooperation (talking over problems) is a good way to learn and is encouraged

Do not copy code. Do not let others look at or copy your code.

Copying is not allowed on homework or exams no matter the source (written or verbal)

- When you submit your homework or tests, **you are pledging that the work is your own and you have not copied it.**
- **You are also pledging that you have not allowed others to copy it**.

DO NOT COPY! (*Software tools catch it easily*)

Examinations

Examination dates will be posted on the schedule page of the course website. Tentative dates are:

- Midterm exam: Thu, 18 April
- Final exam: June 13, 3:15 – 5:45 PM

Do not miss exams

Arrange your work and travel schedules as needed to be present for examinations

Makeup exams will only be given for verified, officially sanctioned university activities

All examinations will be closed-notes and closed-book, except one sheet of notes (A4 or 8.5x11), both sides, handwritten

Grading

Programming assignments (40%, 5 assignments given, lowest grade dropped) = 40% (200 points)

Quizzes (10%, 3 quizzes given, lowest grade dropped) = 10% (100 points)

Recitations (~10 recitations, 1% each) = (100 points)

Mid-term exam = 20% (100 points)

Final exam = 20% (100 points)

Policies:

- Makeup exams will only be given for verified, officially sanctioned university activities
- All makeup exams may be oral

Re-Grading

To promote consistency of grading, questions and concerns about grading should be addressed first to the TA and then, if that does not resolve the issue, to the instructor.

You are welcome to contact the TA by email or come to his/her office hour. If you would like to speak with the TA in person, and have a schedule conflict with his/her office hour, you are welcome to make an appointment to meet the TA at another time.

For the assignments, quizzes and mid-term exams, request for re-grading must be made within one week from after the announcement of grades.

TA Assistance

TAs are available almost every day each week

- Schedule is forthcoming (posted on course web)
- In “CS Commons” (next to CSD office)

Come with specific questions and/or code with which you need help

- TAs strive to spend time with everyone that comes to a session so be courteous and share the TA’s attention

Electronics in Class

Cell phones should be put away during class

Laptops may be used during periods where you are asked to work on an exercise during class

Lecture slides are available on the course website for study before class

Talk to me after class if there's an issue with this policy

Disability

If you have a physical, psychological, medical or learning disability, please contact the Student Services and Career Team.

- Location: Academic Building A208
- Phone: 626-1190

The DSS will determine with you what accommodations, if any, are necessary and appropriate

All information and documentation of disability is confidential

How to Succeed in this Class

Attend class and be on time!

- Not all information is in my lecture notes or in the book
- I sometimes do in-class demos that emphasize non-obvious details

This is an introductory course, true, but we're going to cover a lot of ground and move quickly starting from scratch

The assigned work will take a lot of your time, so practice good time management

Read the reading assignments and review the lecture notes and try out example code

- Practice is the only way to become proficient at coding
- Very often your first, second, or third attempt at solving a problem will not be successful. It is **essential** that you give yourself enough time to try different ideas, taking breaks along the way!
- Those who write extra code for problems not assigned ("for fun") generally do best in this class
- Learning to code involves learning to read other people's code

Ask questions right away if confused. Ask in class, ask a TA, come to my office hours or send email. Don't stay confused and don't get behind!

Welcome and I hope you enjoy the class!