INFO 1100
Intro to Media Programming

Instructor: Phoebe Sengers
Location: Phillips 219
Time: M, W 10:10-11:25

Online syllabus: http://www.cs.cornell.edu/people/sengers/Teaching/INFO1100/index.php

Topic

In this course, you will learn how to program a computer by manipulating media: generating images, producing animations, manipulating text, and making media that respond interactively to user input. We will use Processing, an artist-designed programming language designed for visual and interactive applications, as a basis for creating and developing software 'sketches' that allow you to express yourself visually. In the process, you will learn basic programming skills, including understanding and controlling how data is represented in computers (data types and structures), telling the computer how to make decisions on the fly (conditionals) and how and when to repeat instructions (loops), structuring and organizing your computer code (functions and objects), and techniques for debugging your code.

Learning objectives

Upon completion of this course, students will be able to:

- Program in an object-oriented paradigm using data structures, functions, conditionals, and loops
- Manipulate and generate media in image and text formats, in memory, in files and over networks
- Generate time-based and interactive media
- Imagine, design, and debug novel, creative programs
- Use APIs and libraries as resources in developing programs

Prerequisites

This course is designed for students with no technical background; students who have already completed CS 1110 or have similar programming background may take the course by permission of the instructor only.

For further information

If you have questions, please contact the instructor, Prof. Phoebe Sengers, at sengers @ cs.cornell.edu.

You can download the full syllabus with all information from this website in print-friendly format.

Textbook

The course uses the following textbook:

Casey Reas and Ben Fry. *Processing: A Programming Handbook for Visual Designers and Artists*. Published August 2007, MIT Press.

You will also likely find the processing.org website a useful reference.

Assignments

The assignments in this course are designed to help you develop and assess programming skills. Your work for the class will consist of the following components: *readings* from the course textbook and in-class handouts; brief, informal *quizzes* to help you and us keep tabs on how well you are understanding the course content; near-weekly *homeworks* that let you practice and develop your programming skills; and a *mid-term* and *final* exam. **Of**

all these assignments, the homeworks are most critical because they are where you will really be learning the course material.

Homework assignments are posted to, and can be submitted at, our Course Management System. If you do not have an account for this class, please contact the professor to be added.

Grading formula

- Quizzes: 5% (lowest 2 dropped)
- Homeworks: 45% (lowest *completed* homework dropped (grade D or higher))
- Midterm: 25%
- Final: 25%

Grading is not just a matter of numbers, but also of judgment. The instructor reserves the right to adjust grades by up to half a letter grade based on knowledge of your performance not summed up in this tidy formula.

Due dates

Homeworks are generally due on Sundays at 11:00pm, via submission to our Course Management System. Late assignments accrue a half-letter-grade penalty for every 24 hours they are late, starting directly after the deadline (i.e. submitting at midnight on the due date would make a one-day penalty). All students may submit up to 2 homeworks in the semester up to 2 days late without a penalty - please save these for emergencies such as sickness, as additional extension requests will only be granted in extraordinary circumstances.

Regrade requests

If you believe that there may have been an error in grading your assignment, you may request a regrade in the 7 days after an assignment has been graded by contacting the professor in person or by email, explaining the nature of the mistake. The professor reserves the right to lower the grade as well as to raise it.

Academic Integrity

Academic integrity is taken seriously in this course. Please be aware of the specific requirements of academic integrity at Cornell.

You may be wondering what 'academic integrity' means in the context of programming. In this course, **you may consult with other students for conceptual and debugging help while working on your code**, but unless otherwise specified on the assignment **the final code you submit should be written, tested, and documented by you**. This means that, with the exception of code developed in in-class exercises, if 2 students submit code that is substantially the same we will consider this a likely academic violation. All assignments will be automatically scanned for similarity.

It is a usual practice for real-world programmers to find and adapt publically available code written by others in their own projects, and you may also do this in this class. If you use 'found code' from on-line sources you must bring that code up to the standards expected in this course. **You must also identify which code is 'found' and document its source with a comment in your code**, just as you would for an academic citation in a written paper. Unless otherwise specified for a particular assignment, **found code may not take up more than 10% of the code that you submit** (counting by correctly formatted lines).

Introduction
Aug 24 - Introduction to Processing
Introduction to the class. How to install and use the Processing environment. What a Processing program looks like. Writing a simple program to create shapes on the screen.
Making Images
Aug 29 - Shapes
Reading:
Using Processing, pp. 9-11 only
Structure 1: Code Elements, pp. 17-21
Shape 1: Coordinates, Primitives pp. 23-35
Image layout. More on drawing shapes and varying what they look like.
Aug 31 - Variables

Reading:
Data 1: Variables, pp. 37-41
Using variables to make it easier to experiment with your sketches by changing their parameters.
Sep 1, 5pm (note unusual day and time): Homework 0
Sep 6, noon (note unusual day and time): Homework 1
Sep 5
Labor Day
Sep 7 - Images by the numbers
Quiz
Reading:
Math 1: Arithmetic, Functions, pp. 43-50
Color 1: Color by Numbers, pp. 85-93
Using math to manipulate the variables that control your images. Representation of colors in Processing. Controlling the color of your images. A brief introduction to dynamic code.
Sep 11: Homework 2
Responding to input
Sep 12 - Responding to mouse movement
Reading:
Control 1: Decisions
Input I: Mouse 1, pp. 205-210 only
Varying the behavior of your program based on what someone is doing with the mouse. Programs that decide to do different things based on input data.
Sep 14 - Using mouse movement to change program behavior
Reading:
Structure 2: Continuous
Input I: Mouse 1, pp. 210-215 only
Constructing and controlling programs that run continuously and respond differently in different situations. Using this to respond dynamically to mouse input data.
Sep 18: Homework 3
Sep 19 - Responding to text input
Reading:
Data 2: Text, pp 101-103 only
Input 2: Keyboard, pp. 223-227
Programs that vary their behavior based on what people type in at the keyboard. More practice in responding differently to different situations. A brief introduction to repetition.
Transforming images
Sep 21 - Repetition
Quiz
Reading:
Control 2: Repetition, pp. 61-68
Designing programs that 'loop,' or repeat the same instruction multiple times. Using loops to create patterns in an image.
Sep 25: Homework 4
Sep 26 - How images are represented
Reading:
Image 1: Display, Tint, pp. 95-97 only
Image 3: Pixels, pp. 321-325
Output 1: Images, pp. 367-368 only
Introduction to pixels. Altering images by accesing and altering pixels. Loading and saving images.
Sept 28 - Storing complex information
Reading:
Data 4: Arrays, pp. 301-313
Introduction to the array, or structured lists of values. Use of the array to allow for more complicated control of images (without making your code more complicated).
Oct 2: Homework 5
Oct 3 - Transforming images
Reading:
Image 5: Image Processing, pp. 355-360, pp. 364-365
Operating directly on arrays of pixels in order to transform and manipulate images
Oct 5 - Simplifying your code I: Functions
Reading:
Structure 3: Functions, pp. 181-190, 193-196
How functions can make your code easier to write and debug. Defining and using functions in your code. Using functions to modularize an image processing program.
Oct 10
Fall Break
Oct 12 - Image processing
Reading:
Image 5: Image Processing, pp. 355-360, pp. 364-365
Using for loops, arrays, and functions to read and alter images on the fly.

Oct 16: Homework 6

Pause and digest

Oct 17 - Review

Quiz

Oct 19 - Prelim

Transforming text

Oct 24 - Representing text

Reading:
Data 2: Text, pp. 103-104
Data 3: Conversion, Objects, pp. 105-109

How Processing stores text. The String object. Data conversion. Creating strings, printing them, and displaying them to the screen.

Oct 26 - Reading and visualizing text data

Using Processing to read and visualize plain text files. Looking for and visualizing patterns in text.

Oct 30: Homework 7

Oct 31 - Simplifying your code II: Objects

Reading:
Structure 4: Objects I, pp. 395-406 only

Using objects and classes to structure your code. Defining and instantiating classes.

Nov 2 - Objects continued

Reading:
Structure 4: Objects I, pp. 406-411 only

Arrays of objects. Modularizing files.

Nov 6: Homework 8

Nov 7 - Altering program behavior through text control

Quiz

Reading:
Input 6: File import

Reading text from a file to control your program's output. Finding patterns in text. Using text data to change what programs do.

Nov 9 - Creating text output

Reading:
Output 2: File export

Writing data from your program to a text file, which can be used as input later.

Nov 13: Homework 9

Nov 14 - Reading and responding to web data

Reading:
Handouts in class

Loading a web page, finding the data you want on it, and responding to that data in your program.

Deepening Processing

Nov 16 - Objects II

Reading:
Structure 5: Objects II, pp. 456-460 only

Using classes that inherit from other classes. Using these techniques to simplify your code.

Nov 20: Homework 10

Nov 21 - Deep interactivity

Reading:
Input 3: Events, pp. 229-236

How Processing programs can respond to things that happen outside of it. Real-time interaction using event loops. What is an 'event' and how can programs respond to one. Using events to create meaningful interactivity.

Nov 23 - Objects III

Reading:
Structure 5: Objects II, pp. 453-455 only

Using multiple constructors and creating composite objects.

Thanksgiving Break

Nov 28 - Deep iterativity

Reading:
Shape 3: Parameters, Recursions, pp. 201-204

Using recursion, or functions that call themselves, as a way to generate more complex structured images.

Wrap-up

Nov 30 - Review and wrap-up

Dec 2 (note unusual day): Homework 11