New York University
Department of Media, Culture, and Communication
Creative Coding
MCC-UE 1585

COURSE DESCRIPTION
“Creative Coding” is a practice-based course designed to teach basic programming skills in the context of critical and cultural media studies and the digital humanities. The course requires no prior programming experience, simply a willingness to explore code at a more technical level with the aim of using computation as an expressive, analytical, critical and visualizing medium. In other words, this is a coding class designed to teach students to make projects that extend inquiry and exploration in media, culture and communication. Students will learn basic coding techniques such as variables, loops, graphics, and networking, all within a larger conversation on the social, cultural, and historical nature of code and coding practices.

COURSE OVERVIEW:
The course is structured around a series of weekly discussions, coding assignments, group critiques, ending with a culminating final project. This project will be developed in stages over the course of the semester. Students will use Processing, but the knowledge acquired will be transportable to other languages and coding environments.

LEARNING OUTCOMES
After completing this course, students will be able to effectively:
- Produce simple programs in the Processing programming language (or similar language).
- Explain elementary concepts in computer programming such as data types, operators, control structures, and functions.
- Demonstrate problem-solving skills relating to the design and execution of coding projects.
- Participate in current debates on critical coding such as open source, networking, and graphics.
- Recognize, deconstruct and analyze existing computational projects exploring social, cultural and medial topics.
- Use coding as an expressive, scholarly, and meaningfully rich interpretive, interactive and data-visualizing medium.

REQUIRED TEXTS
Casey Reas and Ben Fry, Getting Started with Processing (Sebastopol, CA: O’Reilly, 2010). Additional readings are itemized in the schedule below.


ASSIGNMENTS
Students are responsible for weekly coding assignments (individual and group), plus a culminating project at the end of the term. The final project will be scaffolded during the semester into several intermediary and accumulating assignments and/or versions. Students will be asked to write an application. First, an idea will be developed, and then students will draft a
process plan. Finally, students will draft, test and execute the code for their final project. Students are encouraged to use weekly assignments to think through and practice aspects of the final project. Class participation will be based on attendance, diligent reading and practice, and active participation in all class activities.

**Evaluation**
Participation: 10%
Weekly assignments: 50%
Final Project: 40%

**Evaluation Rubric**

**A= Excellent**
This project is comprehensive and detailed, integrating themes and concepts from discussions, lectures and readings, and reflecting critical and technical topics covered in class. Students who earn this grade are prepared for class, synthesize course materials and contribute insightfully in every class meeting.

**B= Good**
This project meets the general requirements, offering contributions at a general level of understanding. Classroom participation is consistent and thoughtful in nearly all class meetings.

**C= Average**
This project is adequate but nothing more, meeting the minimum requirements but without significant original thought, reflection, or inventiveness, whether theoretically or practically. Classroom participation is inarticulate or infrequent.

**D= Unsatisfactory**
This project is incomplete, and evidences little understanding of the workshops and discussions. Critique and implementation demonstrate inattention to detail, misunderstand course material and overlook significant themes. Classroom participation is spotty, unprepared and off topic, or rare.

**F= Failed**
This grade indicates a failure to participate and/or incomplete assignments

<table>
<thead>
<tr>
<th>Grade</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>B+</td>
<td>87-89</td>
</tr>
<tr>
<td>C+</td>
<td>77-79</td>
</tr>
<tr>
<td>D+</td>
<td>65-69</td>
</tr>
<tr>
<td>F</td>
<td>0-59</td>
</tr>
<tr>
<td>A</td>
<td>94-100</td>
</tr>
<tr>
<td>B</td>
<td>84-86</td>
</tr>
<tr>
<td>C</td>
<td>74-76</td>
</tr>
<tr>
<td>D</td>
<td>60-64</td>
</tr>
<tr>
<td>A-</td>
<td>90-93</td>
</tr>
<tr>
<td>B-</td>
<td>80-83</td>
</tr>
<tr>
<td>C-</td>
<td>70-73</td>
</tr>
</tbody>
</table>

**COURSE POLICIES**

**Absences and Lateness**
Attendance is mandatory. More than two unexcused absences will automatically result in a lower grade. Chronic lateness will also be reflected in your evaluation of participation. Regardless of the reason for your absence you will be responsible for any missed work.

**Format**
Specific format requirements will be given for each assignment. In general assignments should be submitted electronically, as a zip file (using the naming convention StudentName_Assignment1.zip).

**Grade Appeals**
Please allow two days to pass before you submit a grade appeal. This gives you time to reflect on our assessment. If you still want to appeal your grade, please submit a short but considered paragraph detailing your concerns. Based on this paragraph, we will review the question and either augment your grade or refine our explanation for the lost points.

**General Decorum**
Slipping in late or leaving early, sleeping, text messaging, surfing the Internet, doing homework, eating, etc. are distracting and disrespectful to all participants in the course.

**Academic Dishonesty and Plagiarism** ([http://steinhardt.nyu.edu/policies/academic_integrity](http://steinhardt.nyu.edu/policies/academic_integrity))
The relationship between students and faculty is the keystone of the educational experience at New York University in the Steinhardt School of Culture, Education, and Human Development. This relationship takes an honor code for granted and mutual trust, respect, and responsibility as foundational requirements. Thus, how you learn is as important as what you learn. A university education aims not only to produce high-quality scholars, but to also cultivate honorable citizens.

Academic integrity is the guiding principle for all that you do, from taking exams to making oral presentations to writing term papers. It requires that you recognize and acknowledge information derived from others and take credit only for ideas and work that are yours.

You violate the principle of academic integrity when you
- cheat on an exam,
- submit the same work for two different courses without prior permission from your professors,
- receive help on a take home examination that calls for independent work, or
- plagiarize.

Plagiarism, one of the gravest forms of academic dishonesty in university life, whether intended or not, is academic fraud. In a community of scholars, whose members are teaching, learning, and discovering knowledge, plagiarism cannot be tolerated. Plagiarism is failure to properly assign authorship to a paper, a document, an oral presentation, a musical score, and/or other materials that are not your original work. You plagiarize when, without proper attribution, you do any of the following:
- copy verbatim from a book, an article, or other media;
- download documents from the Internet;
- purchase documents;
- report from other’s oral work;
- paraphrase or restate someone else’s facts, analysis, and/or conclusions; or
- copy directly from a classmate or allow a classmate to copy from you.

Your professors are responsible for helping you to understand other people’s ideas, to use resources and conscientiously acknowledge them, and to develop and clarify your own thinking.
You should know what constitutes good and honest scholarship, style guide preferences, and formats for assignments for each of your courses. Consult your professors for help with problems related to fulfilling course assignments, including questions related to attribution of sources. Through reading, writing, and discussion, you will undoubtedly acquire ideas from others, and exchange ideas and opinions with others, including your classmates and professors. You will be expected, and often required, to build your own work on that of other people. In so doing, you are expected to credit those sources that have contributed to the development of your ideas.

Avoiding Academic Dishonesty

- Organize your time appropriately to avoid undue pressure, and acquire good study habits, including note taking.
- Learn proper forms of citation. Always check with your professors of record for their preferred style guides. Directly copied material must always be in quotes; paraphrased material must be acknowledged; even ideas and organization derived from your own previous work or another's work need to be acknowledged.
- Always proofread your finished work to be sure that quotation marks, footnotes and other references were not inadvertently omitted. Know the source of each citation.
- Do not submit the same work for more than one class without first obtaining the permission of both professors even if you believe that work you have already completed satisfies the requirements of another assignment.
- Save your notes and drafts of your papers as evidence of your original work.

Disciplinary Sanctions

If a professor suspects cheating, plagiarism, or other forms of academic dishonesty, appropriate disciplinary action may be taken following the department procedure or through referral to the Committee on Student Discipline. The Steinhardt School Statement on Academic Integrity is consistent with the NYU Policy on Student Conduct, published in the NYU Student Guide.

Student Resources

- Students with physical or learning disabilities are required to register with the Moses Center for Students with Disabilities, 726 Broadway, 2nd Floor, (212-998-4980) and are required to present a letter from the Center to the instructor at the start of the semester in order to be considered for appropriate accommodation.
- Writing Center: 411 Lafayette, 4th Floor. Schedule appts at rich15.com/nyu/ or walk-in.
SCHEDULE OF CLASSES, READINGS AND ASSIGNMENTS

Week 1: Course Introduction
Why Processing? The development of a visual environment. Coding languages for particular communities. Who is served by Processing and what are its critical and technical affordances? Who developed Processing and what does the community offer? What might we offer the community? Some examples of Processing and Processing.js Projects—just to get us thinking about what we will build this semester.

Download and tour Processing: < http://processing.org/ >

Week 2: “Hello world” -- writing your first sketch in Processing.
Sketches are the operative idiom in Processing. How do we go about imagining ourselves sketching a project, solution or visualization? From bar napkins to code, thought-experiments to flow-charts, the work of coding starts with thinking it through, imagining the context and issues, and designing the process.

Reading: Getting Started with Processing, chapters 1-2.

Looking/interacting: We Feel Fine by Jonathan Harris and Sepandar Kamvar <http://www.wefeelfine.org/>

Week 3: Graphics and Shapes
Visualization is one of Processing’s most powerful affordances. What is a visualization? A data visualization? A graphical representation? A drawing? How do simulations, models and representations take on or express meaning? We will examine several types of simple visualizations in class to develop a vocabulary and approaches to think about receiving and composing computational visualizations.

Reading: Getting Started with Processing, chapter 3.

&

Homework #1 due: 2 sketches –
a) a drawn and written thought-experiment (ideas for projects) (text)
b) “Hello World! I am...” (code)

Week 4: Variables and Operators
Drilling down a level from visualizing data, let’s look a bit closer. Computers are made to crunch information, data, numbers and simply to switch the bits on and off. How does this work? How does a piece of data get stored? And then how does a variable or operator work in relation to that bit of data or in relation to a database. Let’s take some time to fully understand how the hardware and software are working, especially in relation to the code that we are composing. A little software and hardware studies and some network
studies thrown in for practical good measure.

Reading: *Getting Started with Processing*, chapter 4.

Looking/interacting: Fidg’t Visualizer by Protohaus
   <http://sourceforge.net/projects/fidgtvisual/>
   and
   *Newsweek* cover: “Fear” by Benedikt Gross
   <http://benedikt-gross.de/log/2007/12/newsweek-cover-fear/>

Homework #2 due: 2 sketches
   a) draft three ideas for a term-long project. Include: what questions you
      would like to pursue or investigations you would perform; an
      explanation of how your project ideas are good creative and critical
      coding projects; and, why you are passionate about the idea. (text)
   b) graphics and shapes visualization. (code)

**Week 5: Loops, Iterations and Control Structures**

So we understand that a computer does just what its name implies, *quantitative*
computation. How, then, can we use computers and especially coding and programming
to achieve *qualitative* results? What does qualitative--quantitative coding look like? How
can we engage the social, the cultural and the historical, full of nuance, complexity and
contingency, using precise and executable programming languages? Complex math?
Maybe. But what would it look like to put our close reading skills to work to compose a
bit of complex code? Multiple ways of approaching and authoring our text, not to
mention relational, conditional and logical modes of articulation.

Designers and Artists*, pp. 51-68.

Looking/interacting: My Little Piece of Privacy by Niklas Roy
   and
   The Emergence Project by Daniel Sauter, Mark Hereld
   <http://emergenceproject.org/blog/?page_id=180>

Homework #3 due: 1 integrated sketch
   a) Expand on the one idea that you are most passionate about pursuing in this
      class. Using *variables and operators* to generate a text of approximately
      750 words explaining in detail what, if you had all the resources and
      know-how that you needed, your project would look like and do. (code
      + text)

**Week 6: Keyboard and Mouse Input and Events**

Like an ancient orator or Google, it’s one thing for our creative critical coding projects to
deliver a “message” or to produce processed data, but how do we engage our audience, actively
and rhetorically? How do we structure a dialogue, interaction and perhaps interruption in the
flow of events in our code? In programming, how do we design our invitation to others to engage us computationally in dialogic and participatory constellations of input, event, process and response?

Reading: *Getting Started with Processing*, chapter 5.

Looking/interacting: NPZ Visualizer by Neil Banas
<http://coast.ocean.washington.edu/~neil/NPZvisualizer/>
and
Boston 3-1-1 by Benedikt Gross
<http://senseable.mit.edu/bos311/>

Homework #4 due: 2 sketches
a) Get your project idea “cleared” by today. This means that you have explicit permission to pursue your chosen project for the term.
b) *Loops & controls assignment: relational, conditional, logical and iterative.* (code)

**Week 7: Images and Pixels and Motion (part 1)**

So far we’ve taken a “look” at numbers and words with a little time spent on shapes. It’s time to visualize with computational imaging. If a picture is worth a 1000 words (and we might think about what this means in computer languages), why and how might we choose to process and design our images or computationally produce images to articulate meaning in addition to what a still image can do? How do images integrate rhetorically with text and data?

Reading: *Getting Started with Processing*, chapter 6.

We will also take a look at digital imaging software such as Gimp and Photoshop.

Looking/interacting: The Sheep Market by Aaron Koblin
<http://www.aaronkoblin.com/work/thesheepmarket/>
and
Spies by Lisa Straussfeld and James N. Sears
<http://jamesnsears.com/applets/spies/>
and
Cop15: Generative Identity by OKDeluxe/Mikkel Koser
<http://www.okdeluxe.co.uk/cop15/>

Homework #5 due: 2 sketches
a) Come to class with at least 5 super interesting images that you have captured this week and that evoke, articulate, express or describe your semester project visually. (images)
b) Using your project idea as the basis for an inquiry, compose a set of 10 requests for input (keyboard and mouse) by posing a series of critical, cultural, historical, and/or design questions about your project. (code + text)

**Week 8: Functions, Expressions, Statements**
Just as paragraphs articulate a sequenced process of thought and meaning, functions allow us to sequence processes of thought and meaning computationally. They cascade with a carry-over of internal meaning and logic from one instance to the next. And also like some natural language sequences, machine language functions perform and do things by design and execution, retaining a constellation of elements (ideas and information) and moving through these elements in a particular manner. In addition functions “make” or generate elements. Functions are where the word “processing” becomes quite literal.

Reading: *Getting Started with Processing*, chapter 8.

Looking/interacting: MYPOCKET by Burak Arikan
<http://turbulence.org/Works/mypocket/>
and

Homework #6 due: 1 sketch and 1 project preparation
a) Using your project images as the basis for an introductory “visual essay,” compose code that manipulates your images in such a way that the meaningful relevance of your images in relation to your project increases, offering a greater critical, cultural, historical, and/or design understanding about your project. (code + images)

b) Begin to draft a thought map that lays out your project program.
(pseudocode & flowchart)

**Week 9: Objects**
The “thing-iness” of objects in computation is largely determined by their conversion into data structures. We might think of acts of translation here. To make something an object in programming is to make it a data type captured in a given syntax. How might we think about this in terms of representation and the world of material objects. What are the many ways in which we might carefully consider what the phrase “object orientation” means? How does this affordance of computation allow for a consideration of the powers of simulation and modeling of the ‘real world?’ How will you model your project with these super powers?

Reading: *Getting Started with Processing*, chapter 9.

Looking/interacting: Champs d'Ozone by HeHe
<http://hehe.org.free.fr/hehe/champsdozone/>
and
In the Air by Victor Vina, Nerea Calvillo
<http://www.intheair.es/>
and
News Knitter by Ebru Kurbak & Mahir M. Yavuz
<http://casualdata.com/newsknitter/>
Homework #6 due: 1 sketch and 1 project preparation
   a) Complete a thought map that lays out your project program. (pseudocode & flowchart)
   b) Compose a sketch that articulates 2 functions that will make its way into your final project. (code)

Week 10: Arrays
Data. Base. Think for a moment about the term “satellite array,” which offers us a set of dynamic objects (with a multitude of capacities and purposes), a configuration (or if in motion, a set of configurations) of these objects in space and at least one, if not many, means of communication between them. Collectively, we can engage them to request, say, weather information. Individually, they provide a slice of information or a vantage point of view. Taken together in a productive configuration we can interrogate them, make requests and demands. But the information they return is only as valuable as the arrangement of the data and the arrangement of the array in addition to the design of the communication that occurs between the array and the interlocutor. What does this mean? Databases and arrays (despite the misleading objectivism of their names) are highly social and filled with cultural values and assumptions, interpretive strategies and world views.

Reading: Getting Started with Processing, chapter 10.

Looking/interacting: They Rule by Josh On <http://www.theyrule.net/>
and
Terre Natale (Exits 2) by Stewart Smith, Robert Gerard Pietrusko, Aaron Meyers, Michael Doherty, and Hans-Christoph Steiner <http://stewd.io/w/exit/>

Homework #7 due: 2 sketches
   a) Compose your object orientations sketch, an exercise in translation and things. (code)
   b) Begin your project sketch. (code)

Week 11: Video, Audio and Motion (part 2)
We have arrived at the 20th century and the last of our medial modes in convergence with computation, video and motion graphics. From video to animation to gaming, computational motion graphics are the lenses, looking glasses and specula for our moment. Let’s think about what this pixelated and gaseous visuality offers, structures and grounds in terms of perception and affect as well as our sense of what makes something appear “really real” inside and outside of computational frameworks.

We will also take a look at video editing and encoding software such as Final Cut and Premiere Pro.

and
City Symphonies by Mark McKeague
<http://markmckeague.com/work/city-symphonies/>

Homework #8 due: 2 sketches and video
a) Compose a basic array (at least 10 data points), using data for your project. (code)
b) Bring 3 video scratch tracks to class (at least one minute each) that you have captured this week and that evoke, articulate, express or describe your semester project visually. (video)
c) Continue your project sketch. (code)

Week 12: Network Visualization (Carnivore)

Looking/interacting: NYTE by SENSEable City Lab
<http://senseable.mit.edu/nyte/>
and
Telekom Realtime Information Graphics by Zum Kuckuck
<http://projects.zumkuckuck.com/realtime/>
and
Metrography – London Tube Map to large scale collective mental map by Benedikt Gross
<http://benedikt-gross.de/log/2012/02/metrography-london-tube-map-to-large-scale-collective-mental-map/>

Homework #9 due: 2 sketches (and video)
a) Using your project video clips, compose code that manipulates your video in such a way that the meaningful relevance of your video in relation to your project increases, offering a greater critical, cultural, historical, and/or design understanding about your project. (code + video)
b) Continue your project sketch. (code)

Week 13: Lab, Lab, Lab: Final project preparation and finalizing project sketches

Homework #10 due: 2 sketches (and video)
a) Net Viz assignment
b) Rough draft, fully executable though perhaps not “finished”, of your semester project sketch. (code)

Week 14-15: Final project presentations
Here we go!!