Introduction to Programming

Matthew X. Curinga

Table of Contents

**Computer Science 0145-602 Section, Fall 2018**

**Keywords:** computer programming, CS1, javascript, computational thinking, critical computational literacy, problem solving

**Description:** This course introduces students to programming and some core concepts of computer science, using a modern, object oriented programming language. Students learn concepts of variables, functions, selection, repetition/loops, basic data structures (arrays, lists, hashtables), and basic object oriented programming.

We are looking at a society increasingly dependent on machines, yet decreasingly capable of making or even using them effectively. ― Douglas Rushkoff, *Program or Be Programmed: Ten Commands for a Digital Age*

**Class meetings:** Wed. 4:30-6:20, SCB 227

**Instructor:** [Matthew X. Curinga](https://matt.curinga.com), [mcuringa@adelphi.edu](mailto:mcuringa@adelphi.edu)

**Office hours:**

* Monday 12-2pm, online
* Wednesday 2pm-4pm, Alumnae Hall Room 226A (Garden City campus)
* *office hours by appointment*

# Learning Goals

* understand the types of problems that can be solved using computational techniques
* understand the basic concepts of computation (CPU, RAM, permanent storage, GUIs, file systems, network connections)
* learn core computer programming concepts (abstraction, variables, conditions, functions, repetition, recursion)
* think algorithmically to design and test computer programs
* master the basic syntax and idioms of the Javascript programming language
* use technical documentation, APIs, and the internet to learn new technical concepts
* develop step-by-step problem solving and debugging practices

# Required Books

None

# Required Software/Online Accounts

* Software
  + [Slack Client](http://slack.com) (recommend desktop and mobile clients)
  + Firefox or Chrome web browser
* Accounts
  + [repl.it](https://repl.it)
  + [AU Ed Tech Slack channel #code](https://auedtech.slack.com/signup)

# Class Sessions

https://www.lynda.com/C-tutorials/Algorithms/167922/181556-4.html

|  |  |  |  |
| --- | --- | --- | --- |
| Week | Date | Topic | Due |
| 1 | Aug 29 | Turtle: Coding and algorithms | - |
| 2 | Sep 5 | Problem solving in computer programming | Lab 1 |
| 3 | Sep 12 | Variables and data | Lab 2 |
| 4 | Sep 19 | Functions and testing | Lab 3 |
| 5 | Sep 26 | Conditions | Lab 4 |
| 6 | Oct 3 | Repetition with while | Lab 5 |
| 7 | Oct 10 | Strings and repetition with for | *study!* |
| 8 | Oct 17 | Midterm | - |
| 9 | Oct 24 | Arrays | - |
| 10 | Oct 31 | Objects & Sets | Lab 6 |
| 11 | Nov 7 | JSON data | Lab 7 |
| 12 | Nov 14 | Networks and files | Lab 8 |
| 13 | Nov 21 | *no classes* | - |
| 13 | Nov 28 | Open Data | Lab 9 |
| 14 | Dec 5 | Graphs & Charts | Lab 10 |
| 15 | Dec 12 | Finals | Final project |

# Assignments and Grading

|  |  |  |
| --- | --- | --- |
| Assignment | Pct | Due date |
| Labs (10 total) | 50% | Sep 13 |
| Midterm exam | 25% | Oct 18 |
| Final project | 25% | Dec 19 |

## Labs (50%)

Most weeks there will be a programming lab due. Labs will consist of a single exercise that focuses on using the computer programming concepts we’re learning to solve a problem. Each lab will be worth 0-5 points. Roughly:

* *0 points*: for not turning in any work
* *1-2 points*: for a basic attempt, but code isn’t working or has fundamental flaws
* *4-5 points* (mostly) solution demonstrates mastery of relevant concepts
* *5 points*: solution works, demonstrates mastery of concepts, and is well formatted and clearly written

Everyone’s lowest grade will be thrown out (so your lab score will be the average of your 9 highest lab grades).

## Midterm (25%)

For the midterm, you will choose to write a program to solve 1 of 3 problems. Each problem will require demonstration of all of the concepts covered so far in the course, as well as a demonstration of the computer science problem solving approaches that we have been studying and practicing. The exam problems will be similar to the lab exercises.

Computer program scoring guide:

* *5 points,* **solution**: How well and correctly does the code solve the given problem?
* *5 points,* **elegance**: Does the program make efficient use of programming concepts? Are variables and functions used to create *abstractions*? Are the algorithms clear, straightforward, and concise?
* *5 points,* **robustness**: Does the program handle different conditions/states? Would it be easy to adapt if the specification changed?
* *5 points,* **style**: Is the code properly formatted? Do variable and function names follow our conventions? Is it written so that it can be easily read by a human?

## Final project (25%)

Students will work in teams of 2 or 3 to develop a final, group project. The project will be of the students own design, but it will involve:

* accessing a data set (such as real-time data feed)
* analyzing the data
* displaying results using text and graphical displays

The computer program scoring guide from the midterm will be used for the final project as well. All project members will receive the same grade.

## Javascript Documentation and References

* [Mozilla Developer Network](https://developer.mozilla.org/en-US/docs/Web/javascript)
* [W3 Schools](https://www.w3schools.com/js/default.asp)
* [OverAPI](http://overapi.com/javascript)
* [ES6 Lang Spec](https://www.ecma-international.org/ecma-262/6.0/index.html)

## Books & Tutorials

* [Javascript the Right Way](http://jstherightway.org/)
* [You Don’t know JS (book series online)](https://github.com/getify/You-Dont-Know-JS)
* [Eloquent JavaScript](https://eloquentjavascript.net/)
* [Understanding Programming through JavaScript](https://cs.stanford.edu/people/eroberts/CS106AJ-Reader.pdf)
* [Mastering Regular Expressions](http://shop.oreilly.com/product/9780596528126.do)
* [JavaScript & jQuery: Interactive Front-End Web Development Hardcover](http://www.wiley.com/WileyCDA/WileyTitle/productCd-1118871650.html), also J. Duckett, same series