On Programming Literacy

“Computers and networks finally offer us the ability to write. And we do write with them on our websites, blogs, and social networks. But the underlying capability of the computer era is actually programming—which almost none of us knows how to do. We simply use the programs that have been made for us, and enter our text in the appropriate box on the screen.”

—Douglas Rushkoff
Algorithm
Algorithm

A set of rules
An operation
A procedure
A process
A recipe
Precise step-by-step instructions
Computing in Context

Hardware
Operating system
Software
Computer Code
Introduction to Computer Programming
CSCI-UA 2

Class 1
Introduction and Overview
Introduction to Computer Programming
CSCI-UA 2

Class 1
Introduction and Overview

The graph shows the transistor count doubling every two years, starting from the early 1970s with the RCA 1802 to modern processors like the 16-Core SPARC T3.

Transistor count vs. Date of introduction.

- 1971
- 1980
- 1990
- 2000
- 2011

- 2,300
- 10,000
- 100,000
- 1,000,000
- 10,000,000
- 100,000,000
- 1,000,000,000
- 10,000,000,000
- 2,000,000,000

Key processors:
- RCA 1802
- MOS 6502
- AMD K5
- Pentium III
- AMD K6-III
- Pentium II
- Pentium III
- Pentium 4
- Barton
- AMD K8
- AMD K7
- AMD K6
- Itanium 2
- Itanium 2 with 9MB cache
- Dual-Core Itanium 2
- POWER5
- POWER6
- 8-core POWER7
- Quad-core x196
- Six-Core Opteron 2400
- Core i7 (Quad)
- Six-Core Opteron 7400
- Six-Core Core i7
- Six-Core SPARC T3
- 10-Core Xeon Westmere-EX
- 10-Core Xeon Nehalem-EX
- 8-Core Xeon Tukwila
- 8-Core Xeon Nehalem-EX
What is a program?
What is a program?

A sequence of instructions for a computer to follow

May be mathematical or symbolic

Basics include:
- input
- output
- math
- conditional execution
- repetition
Programming Languages

Low-Level

Low-level programming languages are closer to “machine language”

They are difficult (though not impossible) for humans to read and, as such, are more error-prone
Programming Languages
High-Level

High-level programming languages are closer to real syntax.

High-level languages are abstracted and therefore require interpretation.

We’ll be working with a high-level language.
Natural Languages and High-Level Programming Languages

Similarities

Syntax
Grammar
Parts of speech
Semantics
Syntax

Natural language syntax is the arrangement of words and phrases to create well-formed sentences.

Programming language syntax is the arrangement of words and characters to correctly structure programs.
Grammar

Natural language grammar refers to the whole system and structure of a language, such as sentences and paragraphs.

Programming languages also implement structure, such as blocks of code and statements within the blocks.
Parts of Speech

Natural languages incorporate different parts of speech, like nouns, verbs, and adjectives.

Programming languages also have parts of speech called “data types” that include different kinds numbers and characters.
Semantics

In natural languages, semantics refers to the meaning of a word. “Cat” brings something specific to mind.

In programming languages, certain symbols, like `+` and `=`, have specific meaning as well as some key words.
Key Words
A primary difference between natural and programming languages

Python keywords:
False, None, True, and, as, assert, break, class, continue, def, del, elif, else, except, finally, for, from, global, if, import, in, is, lambda, nonlocal, not, or, pass, raise, return, try, while, with, yield
Pseudocode

Determining the logic of a program without regard for the language it will be written in

Best written out on paper or in a plain text editor

Pseudocode describes the steps of an algorithmic process
Introduction to Computer Programming
CSCI-UA 2

Class 1
Introduction and Overview
Python
A general purpose, cross-platform programming language

Python 3
Conceptual and technical foundation
Freely available
Clear syntax
Robust programming language
Python

In Use

Web and Internet development

Scientific and numeric computing

Education

Desktop GUIs

Software Development
input()
### Introduction and Overview

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>a == b</th>
<th>a != b</th>
<th>a and b</th>
<th>a or b</th>
<th>not a</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>False</td>
</tr>
</tbody>
</table>
Introduction to Computer Programming
CSCI-UA 2

Class 1
Introduction and Overview
Introduction to Computer Programming  
CSCI-UA 2

Class 1  
Introduction and Overview
Introduction to Computer Programming
CSCI-UA 2

Class 1
Introduction and Overview
Introduction to Computer Programming
CSCI-UA 2

Class 1
Introduction and Overview
Course Content

Introduction to Computer Programming

Python Basics
Variables and Operations
Control Structures
Repetition Structures
Functions and Modules
Strings
Graphics
Lists
File Input and Output
Dictionaries
Introductions
Me

Joshua Clayton
jclayton@cs.nyu.edu
Room 420, Warren Weaver Hall
Office hours
• Monday, 9:30–11:00 a.m.
• Thursday, 11:00 a.m.–12:30 p.m.

cs.nyu.edu/cs/faculty/clayton
Introductions

You

• Name
• Where you’re from
• Describe your code literacy (if any)
• What interests you about this class
Syllabus

Prerequisites

No prior experience assumed

3 years of high school math required

For students considering a Computer Science major

For students considering or pursuing a Computer Science minor

For students interested in programming

C or better is required to take further CS classes as a major
Syllabus

Class Format

The course consists of three primary components

• Online learning modules

• In-class discussion and application of principles

• In-class workshops
Syllabus

Quizzes

There will be ten online quizzes that go along with each online learning module.

Questions are multiple-choice.

Quizzes are delivered via NYU Classes and can be attempted up to five times.

Your most recent score will be the one recorded.

Quizzes must be completed before class on the day in which they are due.

Quizzes are worth 5% of your grade.
Introduction to Computer Programming
CSCI-UA 2

Class 1
Introduction and Overview

CSCI-UA 2, Section 6
Tuesday and Thursday, 9:30–10:45 a.m.
Room 201, Warren Weaver Hall

Class site: cs.nyu.edu/courses/fall15/CSCI-UA.0002-006/
Syllabus

Attendance

You are expected to come to all classes and arrive on time.

Please let me know in advance if you will be out for any reason.

Please let me know if you miss class due to illness.

You are encouraged to bring a computer to class.

If you ever feel overwhelmed or need extra help, I will be available to you.
There will be ten assignments over the course of the semester.

Details of each assignment will be posted on the class website.

All assignments are to be submitted via NYU Classes.

Do your best to turn work in on time. 10% will be deducted for each class period after the deadline.

No assignments will be accepted after three classes or after the final exam.
Syllabus

Texts

Required textbook

*Starting Out with Python, 3rd Edition*
Tony Gaddis
ISBN: 978-0-133-58273-4

Optional textbooks

*A Byte of Python*
Swaroop C H

*How to Think Like a Computer Scientist*
Peter Wentworth, Jeffrey Elkner, Allen B. Downey, and Chris Meyers

(Both available online)
Syllabus

Grading Rubric

Assignments
20%

Quizzes
5%

Midterm Exam 1
20%

Midterm Exam 2
20%

Final exam
35%
For Next Class

• Review class website
• Complete learning module 1
• Take quiz
• Install Python 3
• Read chapter 1 of *Starting out with Python*
Introduction to Computer Programming
CSCI-UA 2

Class 1
Introduction and Overview

cs.nyu.edu/courses/fall15/CSCI-UA.0002-006/