Introduction, Programming, Python

Lecture 1

Based on the slides by Prof. Michael Mandel

Slides adapted & reused from Rachel Rakov's ‘Methods in Computational Linguistics 1’ course, Fall 2016

Some slides in this presentation have been adapted from the following sources:
Python Programming: An Introduction to Computer Science (third edition)

textbook slides, by John Zelle, Franklin, Beedle & Associates,

http://mcsp.wartburg.edu/zelle/python/

Outline

- Class Information
- Introduction to Computer Science
- Programming Languages
- First Look at Python
- Inside a Python Program

Website, GitHub classroom, BlackBoard

- The full syllabus, PowerPoint lecture slides, and PDFs of assignments will be posted on the class blackboard
  - I will try to keep the class website updated accordingly in case the Blackboard is down
- Assignments will be posted on the class blackboard; Assignment submission will be set up via the class blackboard.
- Assignments and corpora for the assignments will be also distributed via the class GitHub (more details will follow).
- To keep the class video recordings private for the LING78100 only we will be using the Blackboard Collaborate Ultra.
  - If CUNY blackboard is down we will use Zoom

Pre/Co-requisites

- Practicum (LING 73600) Session: Fridays, 11:45am – 1:45pm
  - Video meetings for LING 73600 will be set up using LING 78100 class blackboard collaborate
  - Arundhati Sengupta, PhD student, Linguistics
  - Programming practice, review of topics
  - Required co-requisite
- LING 78100 is a prerequisite for “Methods 2”
  - Usually taught in the spring semeste

NLP / Computational Linguistics at CUNY

- Computational Linguistics Faculty:
  - William Sakas, Virgina Teller, Martin Chodorow, Jia Xu, Rivka Levitan, Michael Mandel, Elena Filatova
- CUNY Computational Linguistics Lecture Series
- Comp Ling program at GC CUNY Linguistics:

  https://www.gc.cuny.edu/Page-Elements/Academics-Research-Centers-Initiatives/Doctoral-Programs/Linguistics/About-the-Program/Specializations/Computational-Linguistics

Computational Linguistics

- The study of what goes into getting computers to perform useful and interesting tasks involving human languages.
- The insights that such computational work gives us into human processing of language.
- Other Names for (or in addition) Computational Linguistics:
  - Natural Language Processing (NLP)
  - Human Language Technology (HLT)
  - Or just “Language Technology” (LT)
  - Natural Language Engineering (NLE)
  - Speech and Text Processing

Why is NLP important?

- Lots and lots of language data (in machine-readable form)!
  - Too much for humans to read and process on our own
- Improve digitally mediated communication
  - Both human-to-human and human-to-computer
- Language is complicated!
  - Linguists can test theories of language and see if they work on a larger scale
  - Computer scientists can build more useful structures if they know how to handle language
- These tasks are best handled by a combination of linguistic and probabilistic knowledge.

Knowledge, Language, and NLP

- “Natural Language” (NL) is the preferred way for humans to store and exchange information.
  - NL = “real” language people grow up using (e.g. French, English, Korean).
  - People don’t generally record or send information in the form of logical predicates or formal/computer languages.
  - Algorithmic languages: Python, Java, C, etc.
  - Why cannot we use NL for programming?!?!
- Computers normally store/input/output information in ways not closely related to human language.
  - Binary numbers… Data Structures… Logic Languages…
- Natural Language Processing (NLP) is the study of how computers can process natural languages.
  - People could interact with computers using language.
  - Allow computers to use information written in language.

Applications of NLP

- What are some useful things NLP can do?
  - Analyzing sentences and words
    - parsing, part-of-speech tagging, morphology
  - Information Extraction, Named Entity Recognition
  - Summary Generation
  - Machine Translation
  - Document Organization/Classification
  - Information Retrieval
  - Database Interfaces, Question Answering
  - Text Editing, Grammar/Style Checking
- And that’s not even getting into speech…

This Course

- Programming “Boot Camp”
  - Accelerated introduction to programming
  - Using the Python Programming Language
    - Useful language for Computational Linguistics
    - Valuable for many other forms of research
- Introduction to a few computational linguistic concepts through the use of examples.
- Main focus: computer programming, good style, data structures, key computer concepts, key math concepts, etc.

Introduction to Computer Science

Objectives

- To understand the respective roles of hardware and software in a computing system.
- To learn what computer scientists study and the techniques that they use.
- To understand the basic design of a modern computer.
- To understand the form and function of computer programming languages.
- To begin using the Python programming language.

The Universal Machine

- A modern computer can be defined as “a machine that stores and manipulates information under the control of a changeable program.”
- Two key elements:
  - Computers are devices for manipulating information.
  - Computers operate under the control of a changeable program.

The Universal Machine

- What is a *computer program*?
  - A detailed, step-by-step set of instructions telling a computer what to do.
  - **NO AMBIGUITY**
    - There are randomized algotih
  - If we change the program, the computer performs a different set of actions or a different task.
  - The machine stays the same, but the program changes!

The Universal Machine

- Programs are *executed*, or *carried out.*
- All computers have the same power, with suitable programming, i.e. each computer can do the things any other computers can do

Program Power

- Software (programs) rule the hardware (the physical machine).
- The process of creating this software is called programming.
- Why learn to program?
  - Fundamental part of computer science
  - Having an understanding of programming helps you have an understanding of the strengths and limitations of computers.

Program Power

- Helps you become a more intelligent user of computers
- It can be fun!
- Form of expression
- Helps the development of problem solving skills, especially in analyzing complex systems by reducing them to interactions between simpler systems.
- Programmers are in great demand!

What is Computer Science?

- Design
  - One way to show a particular problem can be solved is to actually design a solution.
  - This is done by developing an *algorithm*, a step-by-step process for achieving the desired result.
    - REMEMBER? NO AMBIGUITY

What is Computer Science?

- Analysis
  - Analysis is the process of examining algorithms and problems mathematically.
  - Some seemingly simple problems are not solvable by any algorithm. These problems are said to be unsolvable.
  - Problems can be intractable if they would take too long or take too much memory to be of practical value.

What is Computer Science?

- Experimentation
  - Some problems are too complex for analysis.
  - Implement a system and then study its behavior.

Programming languages

- Natural language has ambiguity and imprecision problems when used to describe complex algorithms.
  - Programs expressed in an unambiguous, precise way using programming languages.
  - Every structure in programming language has a precise form, called its syntax
  - Every structure in programming language has a precise meaning, called its semantics.

Programming languages

- Programming language like a code for writing the instructions the computer will follow.
  - Programmers will often refer to their program as *computer code*.
  - Process of writing an algorithm in a programming language often called *coding*.

Programming languages

- *High-level* computer languages
  - Designed to be used and understood by humans
- Low-level language
  - Computer hardware can only understand a very low level language known as *machine language*

Programming languages

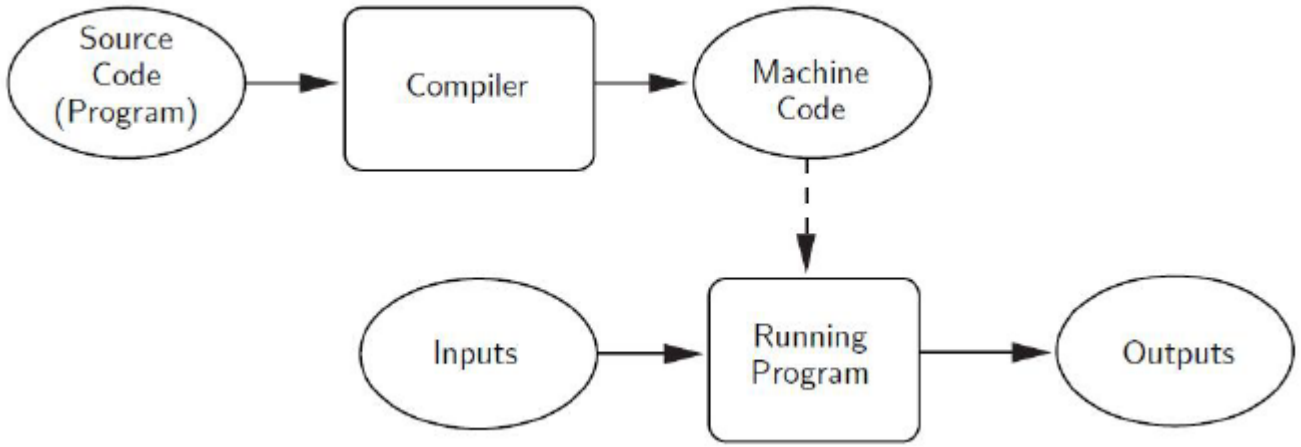
- Add two numbers:
  - Load the number from memory location 2001 into the CPU
  - Load the number from memory location 2002 into the CPU
  - Add the two numbers in the CPU
  - Store the result into location 2003
- In reality, these low-level instructions are represented in binary (1’s and 0’s)

Programming languages

- High-level language

  c = a + b

- This needs to be translated into machine language that the computer can execute.
- *Compilers* convert programs written in a high-level language into the machine language of some computer.



First look at python

The Magic of Python

- Shell Python prompt with >>>
- Write one command per prompt
- Put several commands within a file

- Or use IPython notebooks