ECS 15

Introduction on Python

Acknowledge: Prof. Nina Amenta
A touch on programming

- For people with no programming experience
- Learn the computer language Python
- Write three programs

What is a program?
Why learn to program?

- Understand computers
- Computers are used in almost every career
- The programs you have are never exactly what you want. Example:
  - UCD Student Health Center
    - UCD Medical Center Hospital medical record, OR...
    - Extension of current system
    - Which to choose?
    - How to connect them?
How does programming help?

- You can handle little problems yourself
- You know when you hire someone how big a job it is
- You have some idea of what is possible.
- A good programmer has more job options (you’ll need more than this one class, though)
It teaches you to think!

- The computer is your genie in a bottle
- It does exactly what you tell it to
- Your job is figuring out what to tell it
- Learning the language is the easy part; learning to give exact directions is the hard part
- Imagine telling a Martian how to tie their shoes....
- Some people really like it!
Why Python?

- Great for interfacing one program to another
- Free!
- Used in industry – Google, Yahoo!, NASA, CIA, Civilization 4, Walt Disney Animation,....
- Easy to get started with and good for going deeper.
- Lots of “libraries” (add-ons) that do things like sound editing, computational biology, Web database access.....
Other options

- ECS 10 – More Python programming (the whole quarter)
  - And you can get full credit for that too.

- ECS 30 – More intensive class for those with some programming experience. Learn C++. Required for ECS majors.
Lab assignments

- Three programs
- Tentative schedule:
  - “print your ID, decimal and binary”
    - Start your program, variables
  - “Jeopardy”
    - Input/output, for loop
  - “Wheel of fortune”
    - If, while, for loop, sequence
Computer Classrooms

- We have two “computer classrooms” with Python.
  - SLB 2020
  - 2060 SciLab
  - Sometimes in use by other classes
Install Python on your computer

- Go to Python download page
  - [http://www.python.org/](http://www.python.org/)
  - Choose version Python 3.4.1 or 3.4.2
- We can help during lab hours if you have a laptop
What to do?

- In class, I will exchange between lecture slides and Python programming examples.
- You are welcome to bring your laptop and type along.
- Class notes (slides) will be available before class.
- You are welcome to print them out and write your notes.
What to do?

- Install Python on your computer OR go to the computer classrooms and log on
Software

- Software is written in programming languages.
- A programming language is an artificial language that can be used to control the behavior of a computer.
- Programming languages are used to facilitate communication about the task of organizing and manipulating information, and to express algorithms precisely.
- An algorithm is a list of well-defined instructions for completing a task; that is, given an initial state, it will proceed through a well-defined series of successive states, eventually terminating in an end-state.
  - Deterministic or random

Acknowledgement: Prof. Koehl
Three main levels of programming languages:

- **Machine languages**: refers to the "ones and zeroes" that processors use as instructions. Give it one pattern of bits (such as 11001001) and it will add two numbers, give it a different pattern (11001010) and it will instead subtract one from the other. Often known as binary object file.

- **Assembly languages**: Alternative form of machine language using letters and normal numbers so people can understand it. Ex: `ADD 20, 40, 24`

- **High level languages**: A vocabulary and set of grammatical rules for instructing a computer to perform specific tasks. Each language has its own set of keywords and its own syntax.
High-level programming languages

- Most modern software is written in high-level notation, which is then translated into binary.
- Have special statement forms to help programmers give complicated instructions.
  - Example: Three-part if statement
    - Yes/no question to test
    - Instructions to operate if test is true
    - Instructions to operate if test is false
- Examples: Java, c, c++, perl, fortran, matlab, html, and of course Python.
Execution: Interpret or Compile?

Regardless of what language you use, you eventually need to convert your program into machine language so that the computer can understand it. There are two ways to do this:

- interpret the program through an interpreter
- compile the program through a compiler

The main disadvantage of interpreters is that when a program is interpreted, it runs slower than if it had been compiled.
An interpreter is a program that translates source code into some efficient intermediate representation or precompiled code to execute.
A compiler is a program that translates source codes into object codes. The compiler derives its name from the way it works, looking at the entire source code and collecting and reorganizing the instructions.

Thus, a compiler differs from an interpreter, which analyzes and executes each line of source code successively, without analyzing the entire program.
Programming languages: Examples

**Interpreted languages:**
- Perl,
- Python,
- Matlab
- Java

**Compiled languages:**
- Fortran
- C, C++
- Pascal
- Basic
- Cobol
- ADA
Program Logic

- Specification: what the program should do
- Design: how to achieve its goals
- Algorithm: determines the program logic using an algorithm, the idea behind, in English
- Program: the implementation, in programming language
- Control flow: the order in which the program instructions are executed.
Sequential execution

- "Pre-party grooming" algorithm
  - Shower
  - Brush teeth
  - Apply hair product

- Can order be changes?
Conditional execution

- “party dress” algorithm
  - Put on boxers
  - If party in downtown
    - Put on dress shirt
  - Otherwise,
    - Put on sweater
  - Put on leather belt
  - If Marta at the party
    - Put on the tie she bought
  - Take cellphone

- Are all instructions executed?
- Conditions
Repetitive Execution

- **Counting loop**
  - “Mingle with Marta” algorithm
  - Repeat (three times):
    - Locate Marta
    - Bring her a drink
    - Chat for ten minutes
  - End repeat

- **Conditional loop**
  - Repeat these steps
    - Sidle up to new group
    - Introduce self
    - Chat for ten minutes
  - Until 10pm
Procedure

- “breaking the ice” algorithm
  - Fill shaker with ½ cup ice
  - Add 2 ounces vodka
  - Shake
  - Pour into glass
  - Add olive

- Dump ice and rinse shaker

- Fill shaker with ½ cup ice
- Add 2 ounces gin
- Shake
- Pour into glass
- Add olive
Procedure (function)

- “breaking the ice” algorithm (new)
  - Perform martini procedure (alcohol is vodka)
  - Dump ice and rinse shaker
  - Perform martini procedure (alcohol is gin)

- Martini procedure (alcohol as parameter)
  - Fill shaker with ½ cup ice
  - Add 2 ounces alcohol
  - Shake
  - Pour into glass
  - Add olive
Software Development Phases

- Specification
- Design
- Implementation
- Testing
- Maintenance