Introduction to:
Introduction to:
Computers & Programming

Adam Meyers

New York University

Outline

- What is computer science?
- What is a computer (hardware)?
- What is a computer program (software)?
- What is an algorithm?
- What is a programming language, e.g., Python?
- By 2020, why should all newly educated people know how to write computer programs?
  - Alternatively, do you believe that learning to program is unnecessary for the general population? I.e., should programming be left to specialists?

Some Quick Answers

- A (modern) **computer** is:
  - A mechanical device that makes calculations and solves problems, consisting of physical components, aka **hardware**. Examples: main frame, work station, (cluster/farm), desktop, laptop, tablet computer, gps, cell phone, etc.
- A **computer program (software) is**:
  - Sets of instructions which a computer can "interpret" to solve problems, make calculations, perform tasks, etc.
- A **programming language** (like Python) is:
  - A formal "language" that humans use to write programs.
- **Computer Science** is a field of study concerning:
  - The design of computer hardware and software
  - Modelling problems and using models to find solutions

History of the Modern Computer

- 1837: Charles Babbage's *analytical engine*
  - The first programmable computer
  - Though he himself never completed a working model, working models have been built based on his plans.
- Between 1842 and 1843: Ada Lovelace designed the first computer program (software)
  - Designed for Babbage's computer
  - She saw the non-number-crunching possibilities of Babbage's machine.

What is an Algorithm?

- A precise "recipe" for solving a type of problem
- A crude formalization of the PBJS Algorithm
  - Transfer **slice** of bread from loaf to plate
  - Repeat until enough peanut butter
    - Put knife into peanut butter jar and get peanut butter
    - Transfer peanut butter from knife to slice of bread
  - Transfer **other slice** of bread from loaf to plate
  - Repeat until enough jelly
    - Put knife into jelly jar and get jelly
    - Transfer jelly from knife to **other slice** of bread
  - Put **slice** of bread (pb side down) on **other slice** of bread

Program = Implementation of Algorithm

- The PBJS instructions in ACME sandwich maker 2000 implements the PBJS algorithm
- The instructions refer to specific levers, switches and gears that are part of the ACME sandwich maker 2000
- The instructions refer to specific types of knives, plates, breads, and jellies
- Other PBJS instructions use different, but mathematically equivalent methods
  - The PJ-maker Deluxe squeezes peanut butter out of a squeezy bottle and does not use a knife, but otherwise follows the same algorithm.

Sandwich Making Algorithms Online

- http://www.infopackets.com/news/technology/science/2002/20020919_algorithms_are_a_computer_programmers_best_friend.htm
- http://scienceblogs.com/gregladen/2008/11/please_step_away_from_the_cold.php
- http://library.thinkquest.org/06aug/02352/whataregeneticalgorithms.html

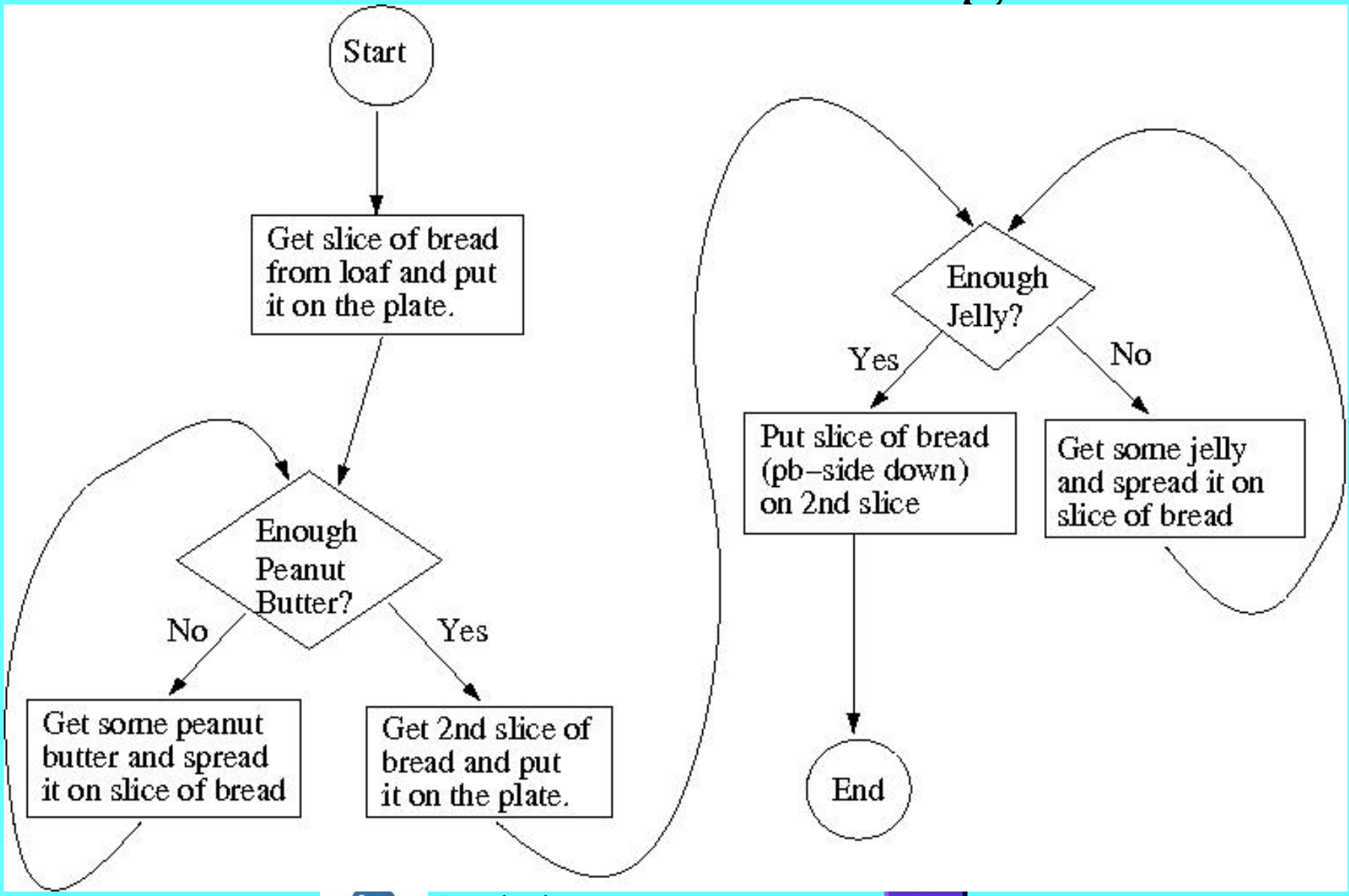
•ETC

- Most sandwich making algorithms online are the same
  - They involve getting ingredients and building a sandwich in essentially the same order
  - The McDonald's one is a little fancier (the 2nd one) because it deals with all different kinds of sandwiches
  - There are also some presentations of sandwich making algorithms designed to argue for the benefits of certain CS principles (object oriented, etc.)

Writing Algorithms

- Design algorithms independently of programs
  - Focus on "best" way to solve a problem
  - Ignore punctuation, syntax, etc. of programming language
- Use *Pseudo Code* (previous PBJ algorithm slide)
- A *Flow Chart* (next slide)
- One algorithm can be implemented as a computer program in many different computer languages

Flow Chart for PBJ algorithm



Some Flowchart Conventions

- Circles/Ovals are for Start and End
- Rectangles represent steps in processing
- Diamonds represent decision points
  - Yes or No
  - Multiple Choice
  - Etc.
- Arrows show the sequence in which steps are applied.

Algorithms Predate Computer Science

- The word *algorithm:* 12th century based on 9th century Persian mathematician Muhammad ibn Mūsā al-Khwārizmī, founder of algebra
- Formal processes in science/math have been described this way for centuries
- Music algorithms: 10th Century to Present (Guido of Arezzo, Bach, Cage, Feldman, Brian Eno, Nine-Inch Nails)
- Art algorithms: 8th century Islamic tile patterns, Escher, Sol Lewitt and modern computer artists
- Algorithms implemented on computers are not subject to some human limitations (e.g., speed of computation, memory size, emotional bias), but are more limited in other ways (e.g., imagination, intuition, real world knowlege)

What is a Programming Language?

- A formal language for encoding instructions that can be carried out by a computer
  - A formal language has a syntax, a strict set of rules for defining valid expressions in that language.
  - Syntax violations cause one of the main types of *bugs* (errors) in computer programs.
- Computer languages: **Python**, JAVA, C, C++, Perl, Lisp, Prolog, Fortran, …
- Some formal languages are not programming languages: propositional logic, arithmetic, HTML, XML, etc.

Low Level Programming Languages

- Machine Language
  - Written in 0s and 1s
  - Different CPUs use different machine languages
  - Very limited set of commands
  - Not very intuitive to use
- Assembly Language
  - Consists of a few commands
  - Easier than machine language, but still not intutiive
- See http://professorandpat.org/ for an introduction

An Off-the-Cuff Sampling of High-Level Programming Languages

- Ubiquitous: C, C++ and JAVA
  - Portable, object-oriented, large libraries, more than 15 years old
- Popular as scripting languages
  - Perl, Python, Javascript, Ruby, Sed, Awk
    - Although Python is becoming more general purpose
- Popular A.I. Languages
  - LISP, PROLOG, (increasingly JAVA, C and Python)
- Popular Languages for Art, Music, Multimedia
  - Processing, Arduino, Supercollider, Actionscript, …
- Popular Languages for Teaching 1st Classes in CS
  - Python, JAVA, SCHEME (related to LISP)

The Programming Language Python

- High Level Programming Language
  - Versus low level languages
    - machine code (commands written as 1s and 0s)
    - assembly language (short hard to read commands)
- Designed for humans to read & understand easily
- Low Syntactic Overhead
- Gaining traction as a 1st programming language
  - Previous 1st programming languages: Pascal, C, JAVA
- Does not scale up for computationally-intensive problems
- Started by Guido van Rossum in 1980 (who likes Monty Python's Flying Circus)
- We will be using Python 3.1

Sample commands in Python 3

- print('Hello World')
- 2 + 2
- Count to 5

```
for i in range(5):
    print(i+1)
```

- Syntax requires that
  - Strings like 'Hello World' be surrounded by quotes
  - '+' must be between two numbers
  - Special words are in lower case (for, range, print)
  - Certain parts of statements must be followed by :
  - Other parts must be preceded by (appropriate) indentation
- Computer scientists start counting at 0

Working with Models on Computers

- A model is a representation of something
  - A plastic model may represent an actual car
  - The letters *cat* may be a model of the sound |kæ t|
  - The word "cat" (sound, letters, ...) can represent a cat in some model (e.g., of taxonomy)
  - A set of formulas may be a model of gravity
- People use computer programs to work with models
  - To model scientific theories, processes, events, finance, audio-visual, games, etc.
  - Input/Output can be symbols, pictures, sound, etc.
  - Input/Output can be instructions from/to other machines

Computers are General Problem Solving Devices

- Scientists can test theories faster than any time in history
- Artists can implement ideas that were inconceivable fifty years ago
- In particular, tedious parts of all problems can be easily formulated as computer programs
- Many hard problems can be broken down into smaller tedious problems
- Programming is a game-changing innovation

Anyone Who Has Ever Played Civilization Knows...

- Literacy and Writing were game changers
  - They were necessary precursors to abstract thinking
- Like literacy, computer programming provides new ways to model parts of our world
- Currently, most newly trained scientists spend about ½ of their time writing computer programs
- Computer programming is also extremely important in law, medicine, art and other areas
- **I predict that within the next 10 to 20 years, every educated person will need to understand computer programming**

Why Should Everyone Know How to Program?

- Why not just use a computer like an appliance?
  - Programs created by others are sufficient for entertainment, office work, basic art, etc.
- Knowing how to program makes it possible to solve new types of problems
  - Whether one writes a program or hires a programmer
- Knowing how programs work makes one be able to use programs better.
  - Knowing what is and is not possible, easy/hard, etc.

What you should learn in this class

- How to write algorithms to solve simple problems
- How to write simple programs in Python
- What programming is and why it is important
- And if you are really into it
  - You will start thinking about how every day problems could be solved by writing computer programs
  - IMHO, the best way to learn to program well is:
    - Find an interesting problem that you care about
    - Write a program to solve that problem or part of that problem

Homework: Due in 2 Classes

- Install Python
  - Some instructions are on the class website
- Open and test Idle (to make sure it works)
  - Try
    - 2+2
    - 3-1
    - print(3-1)
    - print('Hello World')
- Read the 1st chapter of Donaldson book
- Write Simple Algorithm (next slide)

Homework

- Problem: Use pseudo code or a flow chart to describe how to combine a few items (like the PBJ algorithms) or do some other simple action
- Goal: To show that you understand what an algorithm is
- Examples:
  - Prepare a specific non-alcoholic or alcoholic mixed beverage: chocolate milk, ice cream soda, a Martini, ...
  - Preparing a bowl of cereal with milk and fruit
  - Instructions for gluing sticks on a piece of paper to make a stick figure
  - Building a house out of blocks
  - Put together a jigsaw puzzle
- Submit via blackboard – If drawn by hand, scan to pdf file and attach

Grading Criteria

- Is your description adequate? Would an idiot (a robot or computer) be able to follow your instructions?
- Does the algorithm make choices? Are steps repeated?
  - For example, the PBJ algorithm keeps checking if there is enough peanut butter. If there is enough the algorithm advances, if not it repeats the process. [Here there is a choice and a repeat]
- Cleverness, innovation, creativity is worth more points
- More detail is worth more points.