What is Computer Science?

- It’s not really about computers.
- It’s not really a science.
- It’s about imperative ("how to") knowledge as opposed to declarative ("what is") knowledge.
- Imperative knowledge is expressed via algorithms = computational recipes.

"A computer language … is a novel formal medium for expressing ideas about methodology, not just a way to get a computer to perform operations. Programs are written for people to read, and only incidentally for machines to execute.”

-- Harold Abelson and Gerald J. Sussman

Four big ideas

- Four important concepts are at the core of this course:
  1. Abstraction;
  2. Modularity;
  3. Divide, Conquer and Glue;
  4. Models
- These interrelated ideas are important in almost every discipline, but they’re at the core of CS.
- We will illustrate these ideas in several ways.
- Our goal is to help you think about problem solving in new ways.

Big Idea #1: Abstraction

User / Client

Implementer / Designer

Contract / API
Big Idea #2: Modularity

- Large systems are built from components called modules.
- The interfaces between modules are designed so they can be put together in a mix-and-match way.
- In computer programming, the goal is to design packages for maximum reusability.

Big Idea #3: Problem Solving Strategies

Example: Divide-Conquer-Glue

Divide
- problem P into subproblems.

Conquer
- each of the subproblems.

Glue (combine)
- the solutions to the subproblems into a solution S for P.

Big Idea #4: Models

- Need simple models to understand complex artifacts and behaviors.
- We’ll draw lots of diagrams to predict what programs will do.

What will you build with these ideas?

We’ll start with numerical calculations and graphics

What is your name? Valentine
How many classes are you taking this semester? 5
How is the average time in class per week this semester? 2.5
How many hours per day do you spend on extracurricular activities (including jobs)? 15
How many hours per day do you sleep on average? 8
Weekly time profile for Valentine:
37.5 class hours: ################################################################################
15.0 extracurricular hours: ####################################################################
59.5 free hours: ###################################################################
56.0 sleep hours: ################################################################################

Harry Potter

Lovely Day
Define functions to capture common patterns

Apply problem solving strategies like Divide/Conquer/Glue to draw quilts

Write text-based games with conditionals

Tic-tac-toe

Use the power of iteration to generate graphics and write games

Rock, paper, scissors
Learn Data Structures: Lists and Dictionaries

In[16] : from wellesleyfresh import *
In[15] : getMeme(getWellieMemeFreshData), \'Thursday\', \'Bates\'
Dell Egg Salad
Daily Soup: fish chowder
Breakfast: scrambled eggs or white veg
Homestyle Lunch: wild rice
Homestyle Lunch: green beans
Dell Turkey Ham
Homestyle Dinner: steamed corn
Breakfast: hard boiled eggs veg
Fusion: turkey ala king
Homestyle Dinner: beef ribs
Breakfast: Waffle station veg
Homestyle Dinner: crusted spinach
Breakfast: steel cut oatmeal
Global Grill: grilled kalbi korean fusion: vegetarian ala king
Homestyle Lunch: baked chicken bruschetta
Pasta: pasta
Homestyle Lunch: garlic toast
Homestyle Dinner: baked potato bar
Global Grill: potato perogies
Pasta: marina
Dell: cheese veg
Global Grill: potato faworkes

Perform Data Analysis and Visualization

Problem solving with Recursion

Object-Oriented Programming
On to Python! Unlearn what you have learned

I don’t think that word means what you think it means

On to Python! Unlearn what you have learned

Python Intro Overview [Continues on Friday]

- **Values:**
  - 10 (integer),
  - 3.1415 (decimal number or float),
  - 'wellesley' (text or string)

- **Types:** numbers and text: `int`, `float`, `str`
  - `type(10)`
  - `type('wellesley')`

- **Operators:** + - * / % =

- **Expressions:** (they always produce a value as a result)
  - `len('abc') * 'abc' + 'def'`

- **Built-in functions:** `max`, `min`, `len`, `int`, `float`, `str`, `round`, `print`, `raw_input`

The First Model: Variable as a Box

- Variables are names we make up (but, there are rules for creating these names)
- A variable name should appear for the first time in an **assignment statement**.
- A value is stored in a “box”.
- The variable “labels” the box.
- When a variable is used in expressions, we lookup for the “box” with that name and read its value.
- We can reassign a (new) value to a box.
- If we use a name in an expression without using it in an assignment first, we get a **NameError**.
### Example with cs1graphics

Example with cs1graphics

Version annotated to display coordinates for reference points.

---

**cs1graphics – A library for graphics**

```python
from cs1graphics import *

paper = Canvas(400, 550, 'yellow')

head = Circle(50, Point(200, 150))
paper.add(head)

# The torso
torso = Rectangle(100, 200, Point(200, 300))
torso.setFillColor('gray')
paper.add(torso)

message = Text('If I only had a heart!', 20)
paper.add(message)
messagel.setFontColor('red')
messagel.moveTo(200, 25)

# SOME CODE OMITTED
```

---

### Expressions vs. Statements

- **Expressions**
  - They always produce a value:
    - `10`
    - `10 * 20 - 100/25`
    - `max(10, 20)`
    - `int("100") + 200`
    - `fav`
    - `fav + 3`
    - "pie" + " in the sky"

- **Statements**
  - They perform an action (that can be visible, invisible, or both):
    - `print(10)`
    - `age = 19`
    - `paper = Canvas(400, 550, 'yellow')`
    - `paper.add(head)`
    - `print('She is ' + str(age) + ' years old.')`

Expressions are composed of values, operators, variables, functions, and any combination of them.

Statements may contain expressions, which are evaluated **before** the action is performed.