Welcome to CS50 section! This is Week 9.

- Final project official proposals: due next Friday at noon

- Add to your calendar:
  ○ Final project status report: Due Monday, Nov 28 at noon (Halfway point; be at least ⅓ done)

- The “quiz” (aka the second midterm) is Monday 11/14 through Thursday 11/17
  ○ Same “take home, no collaboration” policy
Welcome to Python

Python lets us write smarter programs, faster.

Course timeline:
Raw C code
Distribution C code
Raw Python code
Framework Python code (Flask)
HTML/CSS
JavaScript
JavaScript frameworks (jQuery)
Before starting pset 7

- Conceptual basics of Python
- Flask
  - Decorators and routes
  - MVC in the context of Flask
- SQL queries
Final project roundtable
Conceptual basics

Review Week 8 slides (http://brandon.wang/cs50)

- You must know the basics of Python to proceed!
  - Syntax, structures, loops, data types

- Most people will make a final project in Python/Flask.

- Less important to carry over skills from C,
  More important to know how Python implements things
Brief review

- Data types in Python
  - Lists
  - Tuples
  - Dictionaries

- Function definitions
  - Optional arguments
Functions

In Python, functions are first-class objects.

- Data type like everything else
- Can be overridden
- Can be passed around (although try not to)
def one():
    return 1

# overriding a function means, here, increasing its return value by 1
def override(func):
    def incr():
        return return func() + 1
    return incr

print(one())  # 1
one = override(one)
print(one())  # 2
Functions

Why does this matter?

- Python’s flexible definition of a data type is a design choice
- Can’t call a function the same name as a variable
- Same scoping of a variable applies to a function
Decorators in Python are functions that modify the behavior of other functions, typically applying some extra functionality hereto.
Decorators in Python are functions that modify the behavior of other functions, typically applying some extra functionality hereto.

- Within the context of CS50/pset 7, decorators set the “route”, require users to be logged in, etc.

- Within Python, decorators are a simple way of adding wrapper functionality to a program.
def override(func):
    def incr():
        return func() + 1
    return incr

@override
def one():
    return 1

# What happens?
print(one())
Routes

- Think of routes as pathways to functions
- We’re mapping URLs to functions
  - It's a many-to-one relationship
Routes

- Think of routes as pathways to functions
- We’re mapping URLs to functions
  - It’s a many-to-one relationship
- In Flask, routes are defined using a decorator: `@app.route()`
Routes

A really simple Flask app might look like:

```python
from flask import Flask

app = Flask(__name__)

@app.route("/")
def index():
    return "You are at the index!"
```
MVC

In pset 7, we use a loose form of MVC.

- Models and controllers go in application.py.
- Set up your views (eg. visual layouts) through Jinja templates.
What are the distinctions to know?

- Routes don’t map to URLs (many URLs to one route)
- Routes **do** map to functions (many routes to one function)
- Views/templates don’t map to routes
- Views/templates don’t map to anything
- Views/templates are just things that your functions can call
Flask views are primarily structured through Jinja.

Jinja is a Python-inspired language for making templates, ways of showing things (rendering) in the browser.
Flask views are primarily structured through Jinja.

Jinja is a Python-inspired language for making templates, ways of showing things (rendering) in the browser.

- Cool things with Jinja
  - Jinja templates cascade (i.e. extend)
  - Jinja templates interweave HTML and Python

Not too much explanation here-- read problem spec.
Databases and SQL
Databases and SQL

- Key elements of database design
  - Databases have multiple tables
  - Columns have data types
  - Tables have primary keys
  - Tables have commonalities
SQL queries are statements that you send to a database server. The server responds according to your statement.

In Flask, use `db.execute()` to run SQL queries.

SQL = “Structured query language”
Do programmers write SQL queries anymore? (Not really.) But the underlying mechanism generally continues to be SQL.
SQL obeys the **CRUD** process:

- **C**: Create
- **R**: Read
- **U**: Update
- **D**: Delete
SQL obeys the **CRUD** process:

- **C**: Create
  - `INSERT INTO`
- **R**: Read
  - `SELECT`
- **U**: Update
  - `UPDATE`
- **D**: Delete
  - `DELETE`
SQL → Base vocabulary

Insertions:
“INSERT INTO <table> (<columns>) VALUES (<values>)”

Selections:
“SELECT <columns> FROM <table>”

Updates:
“UPDATE <table> SET <column1> = <value1>, <column2> = <value2>”

Deletions:
“DELETE FROM <table>”
SQL → More vocabulary

All of these are usually paired with conditions, etc.:

- **WHERE** is nearly always included
  - Limits the scope of the query
- **JOIN** lets operations on multiple tables occur
SQL example
That’s all for today!