# CS 0011 SEC 1020

Prof Matt de Lima Barbosa

Lab TA: Katelyn Morrison

University of Pittsburgh

# Quick Review - Object Oriented Programming

- Abstract methods from client
  - Can use methods without even knowing how the method works
  - Widely used because of this
- Entails Objects
  - hold data attributes & define behaviors
    - Variables
    - Methods

# Quick Review - Object Oriented Programming

- Making a class in python:
  - `class Plant:`
- Initializer method (set all necessary attributes)
  - `def __init__(self):`
- `self` is an explicit term to refer to the `class`
  - Don't need to pass an argument in for self

# Quick Review - Object Oriented Programming

- What's up with the underscores?
  - __methodName__
    - Specific to python language
    - Avoid using this for personal methods
    - Use only for __init__ or __str__
  - __varName or __methodName
    - Tells Python to rewrite the attribute name to avoid naming conflicts

# Quick Review - Object Oriented Programming

- What's up with the underscores?
  - __varName or __methodName
    - Name Mangling
  - Example:

```
class Test:
    def __init__(self):
        self.foo = 11
        self._bar = 23
        self.__baz = 23
```

```
>>> t = Test()
>>> dir(t)
['_Test__baz', '__class__', '__delattr__', '__dict__', '__dir__',
 '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__',
 '__gt__', '__hash__', '__init__', '__le__', '__lt__', '__module__',
 '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__',
 '__setattr__', '__sizeof__', '__str__', '__subclasshook__',
 '__weakref__', '_bar', 'foo']
```

# Quick Review - Object Oriented Programming

- Initialize variables in a
  - Typically done in __init__(self)

```
def __init__(self):
    self.type = “Tree”
    self.subtype = “Conifer”
    self.name = “Blue spruce”
    self.age = 2
    self.healthy = True
```

- Create an instance of this class

```
MyPlant = Plants()
```

# Quick Review - Object Oriented Programming

- Stuck?
  - Remember to look over the instructions before lab
  - Use lab to ask questions
  - Check out Examples from 10 (download the code)

## Project 3 Instructions

# Project 3

- Model Microbial Growth
  - Create a simulation
- Project 3 Tasks:
  - Four Parts:
    - Represent Nutrients
    - Represent Microbes
    - Represent the Grid
      - PetriCell and PetriDish
    - Run the simulation

# Project 3 - Part 1

- Functions
  - *__init__(self)*
  - *getMoved(self)*
  - *setMoved(self)*
  - *clearMoved(self)*
- Create a class
  - Class called *Nutrient*

# Project 3 - Part 2

- Functions
  - *__init__(self)*
  - *hasNutrient(self)*
  - *takeNutrient(self)*
  - *consumeNutrient(self)*
- Create a class
  - Class called *Microbe*

# Project 3 - Part 3

- Functions
  - *__init__(self)*
  - *getMicrobe(self)*
  - *hasMicrobe(self)*
  - *createMicrobe(self)*
  - *getNutrient(self)*
  - *getUnmoved(self)*
  - *clearAllMoved(self)*
  - *hasNutrients(self)*
  - *placeNutrient(self, nutrient)*
  - *__str__(self)*
- Create a class
  - Class called *PetriCell*

# Project 3 - Part 4

- Functions
  - *__init__(self, x, y, concentration, microbes)*
  - *moveNutrients(self)*
  - *checkMicrobes(self)*
  - *step(self, iterations)*
  - *__str__(self)*
- Create a Class
  - Class called *PetriDish*