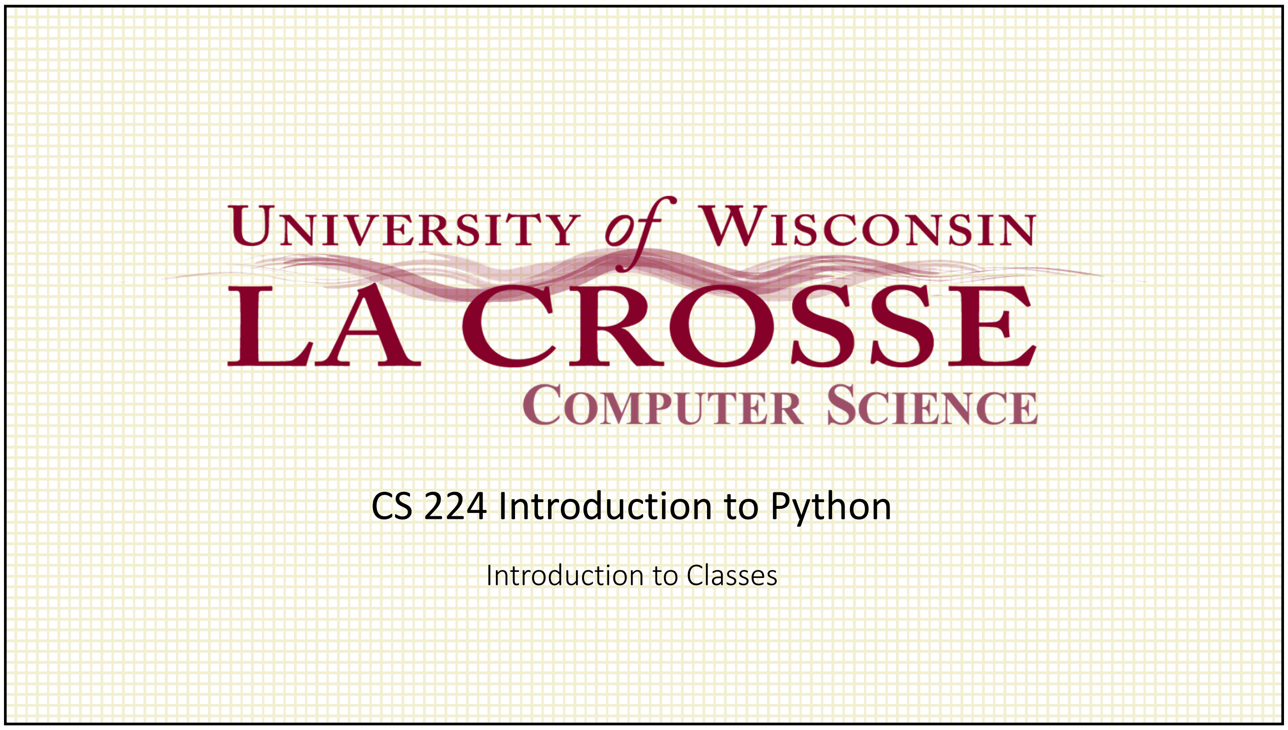
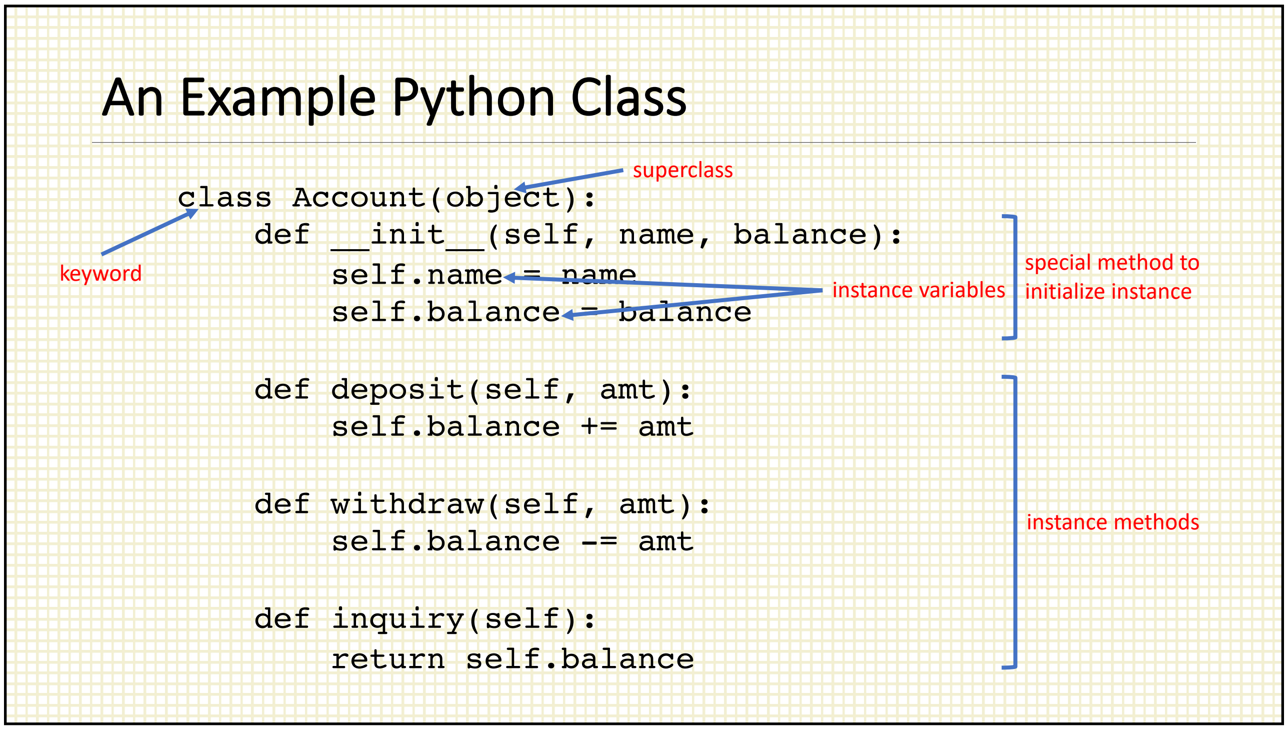


UNIVERSITY of WISCONSIN LA CROSSE COMPUTER SCIENCE

CS 224 Introduction to Python

Introduction to Classes

1



An Example Python Class

```
class Account(object):
    def __init__(self, name, balance):
        self.name = name
        self.balance = balance

    def deposit(self, amt):
        self.balance += amt

    def withdraw(self, amt):
        self.balance -= amt

    def inquiry(self):
        return self.balance
```

superclass
keyword
instance variables
special method to initialize instance
instance methods

2

The Details

- <u>object</u> is the root of the class tree
  - use it as superclass when you are not extending another class
- `__init__` implicitly defines instance variables
- self is a parameter to all instance methods
  - but you don't include it in the parameter list when you call an instance method
  - technically you can use any identifier in place of self (but don't do it!)

3

Augmenting the Example:

```
class Account(object):
    num_accounts = 0
    def __init__(self, name, balance):
        self.name = name
        self.balance = balance
        Account.num_accounts += 1
    def __del__(self):
        Account.num_accounts -= 1

    def deposit(self, amt):
        self.balance += amt

    def withdraw(self, amt):
        self.balance -= amt
```

class variable

special method to cleanup, etc.

4

More Details

- class variables are like static variables in Java
  - they belong to the class not to an instance
  - all instances share a single copy of a class variable
- __del__ is often absent
  - it is used to do things such as update class variables (as in our example), close network connections, release locks, etc.
  - calling del on an object does not necessarily invoke __del__ -- del reduces reference count

5

Creating Instances:

```
from account import Account

checking = Account('David', 50000)
savings = Account('David', 1000000)

# it's payday
checking.deposit(25000)

# Hey honey, can I buy a Ferrari 488?
if savings.inquiry() > 500000:
    print("Go shopping.  Make sure it's red.")

print('Created: {}'.format(Account.num_accounts))
```

Create two Account instances

self does not appear here

access class variable

prints 2

6

Exercise

- Exercise: create a Python class called `Car`
- This class should include:
  - Three attributes
    - `make`
    - `model`
    - `Year`
  - Class variable `num_cars` that tracks the number of cars created
  - An `__init__` method
  - Method `print_description` that prints the attributes for a `Car` instance
  - Setter methods for each of the attributes
- Write a main method that creates a couple of `Car` instances and applies methods to them

7

Exercise solution

```
class Car(object):
     num_cars = 0
     def __init__(self, make, model, year):
          self.make = make
          self.model = model
          self.year = year
          num_cars += 1
```

8

Exercise solution

```
    def print_description(self):
         print('Make: {}'.format(self.make))
         print('Model: {}'.format(self.model))
         print('Year: {}'.format(self.year))

    def set_make(self, new_make):
         self.make = new_make

    def set_model(self, new_model):
         self.model = new_model

    def set_year(self, new_year):
         self.year = new_year
```

9

Exercise solution

```
def main():
     f488 = Car('Ferrari', '488', '2019')
     jcw = Car('MINI', 'Cooper S', '2003')

     f488.print_description()
     jcw.set_model('Cooper S JCW')

     print('Number of cars = {}'.format(Car.num_cars))

if __name__ == '__main__':
     main()
```

10