ECE-203 – Programming for Engineers

Contact

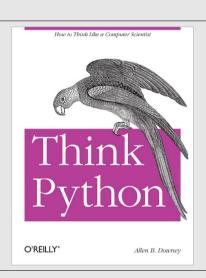
Dr. James Shackleford shack@drexel.edu Bossone 211

Office Hours: 3 - 4 pm (Tuesday)

Course Website: http://learn.dcollege.net

Textbook

Think Python
by Allen Downey
O'Reilly Press, 2015
ISBN-13: 978-1449330729
(Freely available in PDF format, check course website)



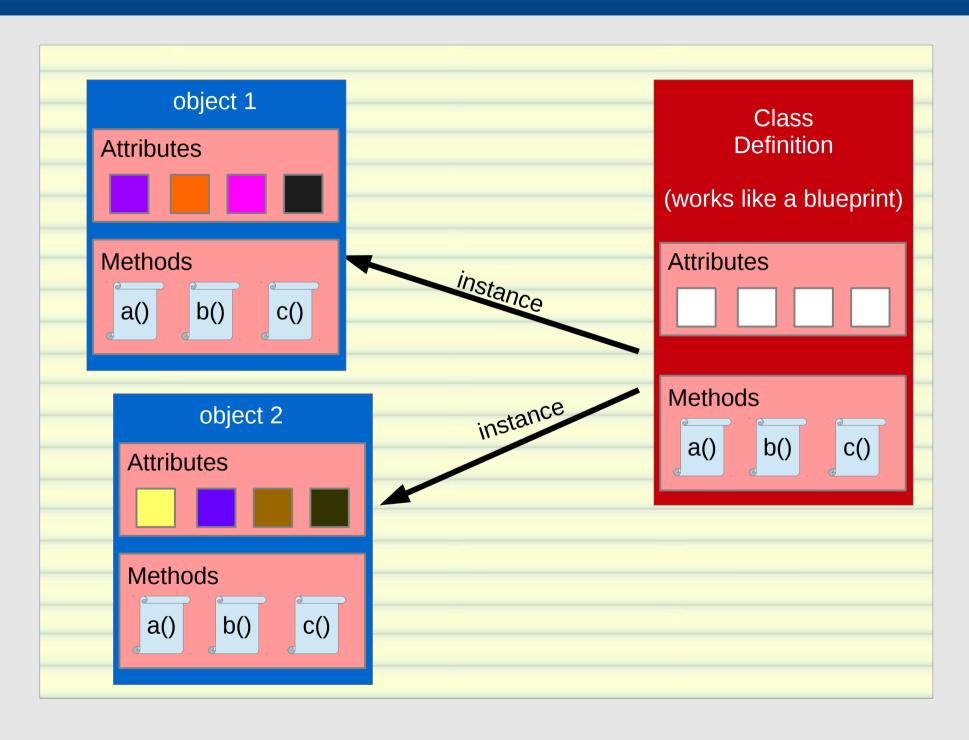
Grading

- 10% In-lab Programming Assignments
- 10% Take-Home Programming Assignments
- 35% Mid-term Exam
- 45% Final Exam

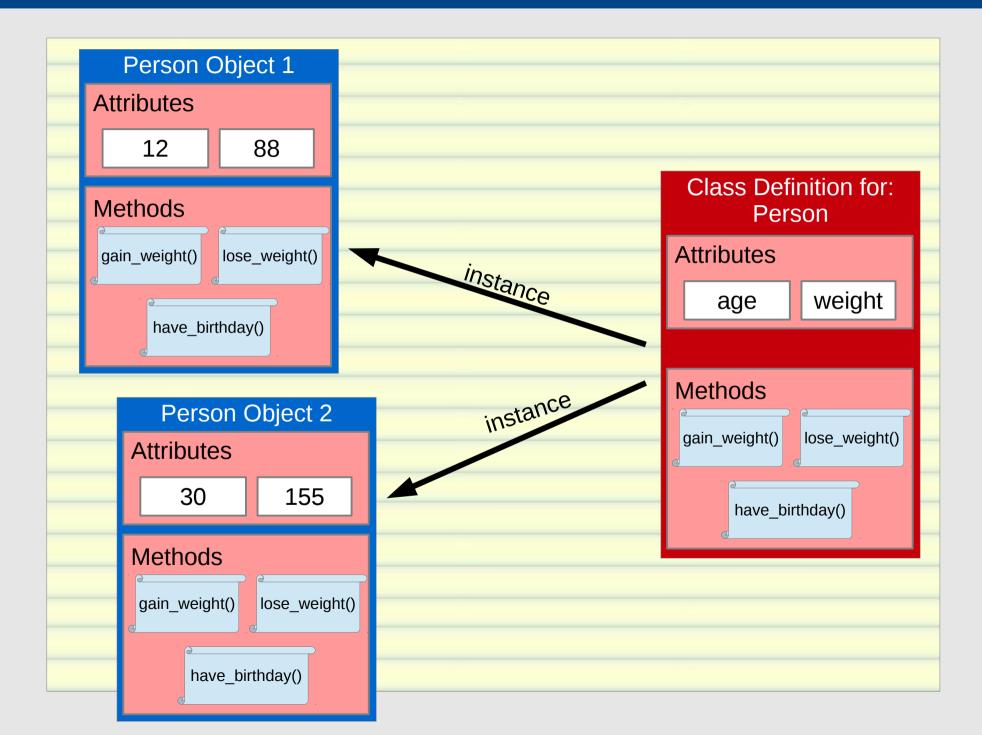
Introduction to Object Oriented Programming

Classes, Objects, Instances, Methods, and Attributes

The Big Idea



The Big Idea



```
class Person:
       This is the docstring for the Person class!!
                                                                class name
         The Person class (loosely) represent a person.
       0.00
 6
       def __init__(self, age, weight):
           self.age = age
 9
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
           """Add 'amount' weight to a Person object"""
13
14
           self.weight += amount
15
16
       def lose_weight(self, amount):
17
           """Remove 'amount' weight from a Person object"""
18
           self.weight -= amount
19
20
       def have birthday(self):
           """Increase the age of a Person object by 1"""
21
           self.age += 1
22
23
24 \text{ joe} = Person(12, 88)
   bob = Person(30, 155)
```

```
class Person:
       10.000
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       1111111
 6
       def __init__(self, age, weight):
 9
           self.age = age
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
           """Add 'amount' weight to a Person object"""
13
           self.weight += amount
14
15
16
       def lose_weight(self, amount):
           """Remove 'amount' weight from a Person object"""
17
18
           self.weight -= amount
19
20
       def have birthday(self):
           """Increase the age of a Person object by 1"""
21
           self.age += 1
22
23
24 \text{ joe} = Person(12, 88)
   bob = Person(30, 155)
```

class docstring

```
class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       \mathbf{H} \mathbf{H} \mathbf{H}
 6
       def __init__(self, age, weight):
 9
            self.age = age
            self.weight = weight
10
11
12
       def gain_weight(self, amount):
            """Add 'amount' weight to a Person object"""
13
            self.weight += amount
14
15
16
       def lose_weight(self, amount):
            """Remove 'amount' weight from a Person object"""
17
            self.weight -= amount
18
19
20
       def have_birthday(self):
            """Increase the age of a Person object by 1"""
21
22
            self.age += 1
23
24 \text{ joe} = Person(12, 88)
   bob = Person(30, 155)
```

methods

```
class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       0.00
 6
       def __init__(self, age, weight):
           self.age = age
 9
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
           """Add 'amount' weight to a Person object"""
13
14
           self.weight += amount
15
16
       def lose_weight(self, amount):
           """Remove 'amount' weight from a Person object"""
17
           self.weight -= amount
18
19
                                      some instances
20
       def have_birthday(self):
           """Increase the age of a Person object by 1"""
21
22
           self.age += 1
23
24 joe = Person(12, 88)
   bob = Person(30, 155)
```

```
class Person:
                                                          called when object
       This is the docstring for the Person class!!
                                                               is created
         The Person class (loosely) represent a persor
       0.00
 6
                                                                  ...or...
       def __init__(self, age, weight):
           self.age = age
                                                             instantiated
           self.weight = weight
10
11
       def gain_weight(self, amount):
12
           """Add 'amount' weight to a Person object"""
13
           self.weight += amount
14
15
16
       def lose_weight(self, amount):
           """Remove 'amount' weight from a Person object"""
17
           self.weight -= amount
18
19
20
       def have birthday(self):
           """Increase the age of a Person object by 1"""
21
22
           self.age += 1
23
24 \text{ joe} = Person(12, 88)
   bob = Person(30, 155)
```

```
class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       0.00
 6
       def __init__(self, age, weight):
           self.age = age
 9
           self.weight = weight
10
11
       def gain_weight(self, amount):
12
           """Add 'amount' weight to a Person object"""
13
14
           self.weight \neq= amount
15
       def lose_weight(self, amount):
16
           """Remove /amount' weight from a Person object"""
17
18
           self.weight -= amount
19
       def have_birthday(self):
20
           """Increase the age of a Person object by 1"""
21
22
           self.age += 1
23
24 joe = Person(12, 88)
   bob = Person(30, 155)
```

```
???
   class Person:
       1111111
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       0.00
 6
       def __init__(self age, weight):
           self.age = age
 9
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
           """Add 'amount' weight to a Person object"""
13
14
           self.weight += amount
15
16
       def lose_weight(self, amount):
17
           """Remove 'amount' weight from a Person object"""
           self.weight -= amount
18
19
20
       def have birthday(self):
           """Increase the age of a Person object by 1"""
21
           self.age += 1
22
23
24 \text{ joe} = Person(12, 88)
   bob = Person(30, 155)
```

```
this is the instance!
   class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       0.00
 6
       def __init__(self age, weight):
           self.age = age
 9
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
           """Add 'amount' weight to a Person object"""
13
14
           self.weight += amount
15
16
       def lose_weight(self, amount):
17
           """Remove 'amount' weight from a Person object"""
           self.weight -= amount
18
19
20
       def have birthday(self):
           """Increase the age of a Person object by 1"""
21
           self.age += 1
22
23
24 \text{ joe} = Person(12, 88)
   bob = Person(30, 155)
```

Instance/Object Attributes

```
class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       0.00
 6
                                                 these are instance
       def __init (self, age, weight):
           self.age = age
 9
                                                      attributes
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
           """Add 'amount' weight to a Person object"""
13
14
           self.weight += amount
15
16
       def lose_weight(self, amount):
           """Remove 'amount' weight from a Person object"""
17
           self.weight -= amount
18
19
20
       def have birthday(self):
           """Increase the age of a Person object by 1"""
21
22
           self.age += 1
23
24 \text{ joe} = Person(12, 88)
   bob = Person(30, 155)
```

Initializing Objects

```
class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       \mathbf{n} \mathbf{n} \mathbf{n}
 6
       def init (self, age, weight):
           self.age = age
 9
                                                               initializing them
            self.weight = weight
10
11
12
       def gain_weight(self, amount):
            """Add 'amount' weight to a Person object"""
13
14
            self.weight += amount
15
16
       def lose_weight(self, amount):
17
            """Remove 'amount' weight from a Person object"""
18
            self.weight -= amount
19
20
       def have birthday(self):
            """Increase the age of a Person object by 1"""
21
            self.age += 1
22
23
24 \text{ joe} = Person(12, 88)
   bob = Person(30, 155)
```

we are

Accessing Object Attributes

```
class Person:
       1111111
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       1111111
       def __init__(self, age, weight):
           self.age = age
10
           self.weight = weight
11
12
       def gain_weight(self, amount):
           """Add 'amount' weight to a Person object"""
13
14
           self.weight += amount
15
16
       def lose weight(self, amount):
           """Remove 'amount' weight from a Person object"""
17
           self.weight -= amount
18
19
20
       def have birthday(self):
           """Increase the age of a Person object by 1"""
21
22
           self.age += 1
23
24 \text{ joe} = Person(12, 88)
                                            accessing
   bob = Person(30, 155)
                                       attributes is easy
   print joe.age
   print bob.weight
```

```
class Person:
       1111111
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       1111111
       def __init__(self, age, weight):
           self.age = age
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
13
           """Add 'amount' weight to a Person object"""
           self.weight += amount
14
15
       def lose_weight(self, amount):
16
           """Remove 'amount' weight from a Person object"""
17
18
           self.weight -= amount
19
20
       def have birthday(self):
           """Increase the age of a Person object by 1"""
21
22
           self.age += 1
23
24 \text{ joe} = Person(12, 88)
                                                calling methods
25 bob = Person(30, 155)
                                                    is also easy
26
   joe.have_birthday()
```

```
class Person:
       1111111
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       1111111
       def __init__(self, age, weight):
           self.age = age
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
13
           """Add 'amount' weight to a Person object"""
           self.weight += amount
14
15
       def lose_weight(self, amount):
16
           """Remove 'amount' weight from a Person object"""
17
18
           self.weight -= amount
19
       def have birthday(self):
20
           """Increase the age of a Person object by 1"""
21
22
           self.age += 1
23
                                             btw. this is the same
24 \text{ joe} = Person(12, 88)
25 bob = Person(30, 155)
                                                (but uncommon)
26
27 joe.have_birthday()
  Person.have birthday(joe)
```

```
class Person:
       1111111
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       1111111
       def __init__(self, age, weight):
           self.age = age
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
13
           """Add 'amount' weight to a Person object"""
           self.weight += amount
14
15
       def lose_weight(self, amount):
16
           """Remove 'amount' weight from a Person object"""
17
18
           self.weight -= amount
19
       def have birthday(self):
20
           """Increase the age of a Person object by 1"""
21
22
           self.age += 1
23
                                                   here we are
24 joe = Person(12, 88)
                                                explicitly passing
25 \text{ bob} = Person(30, 155)
26
                                                   the instance
27 joe.have_birthday()
  Person.have birthday(joe)
```

```
class Person:
       1111111
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       1111111
       def __init__(self, age, weight):
           self.age = age
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
13
           """Add 'amount' weight to a Person object"""
           self.weight += amount
14
15
       def lose_weight(self, amount):
16
           """Remove 'amount' weight from a Person object"""
17
18
           self.weight -= amount
19
20
       def have birthday(self):
           """Increase the age of a Person object by 1"""
21
22
           self.age += 1
23
24 \text{ joe} = Person(12, 88)
                                                here, it is passed
25 bob = Person(30, 155)
                                                      implicitly
26
   joe.have_birthday()
   Person have birthday(joe)
```

```
class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
 6
       def __init__(self, age, weight):
           self.age = age
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
           """Add 'amount' weight to a Person object"""
13
           self.weight += amount
14
15
16
       def lose_weight(self, amount):
           """Remove 'amount' weight from a Person object"""
17
18
           self.weight -= amount
                                                             same stuff going on
19
       def have_birthday(self):
20
                                                               here, but with an
           """Increase the age of a Person object by 1"""
21
22
           self.age += 1
                                                                   argument
23
24 \text{ joe} = Person(12, 88)
  bob = Person(30, 155)
26
  bob.lose_weight(5)
  Person.lose weight(bob, 5)
```

```
class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       1111111
 6
       def __init__(self, age, weight):
           self.age = age
 9
           self.weight = weight
10
11
    fine.
12
                                 unt):
13
                                   to a Person object"""
14
    ...but if I want to define
15
16
                                 unt):
     more people, I need
                                 ight from a Person object"""
17
     rewrite the program
18
19
20
       def have_birthday(self):
           """Increase the age of a Person object by 1"""
21
22
           self.age = 1
23
  joe = Person(12, 88)
  bob = Person(30, 155)
26
   bob.lose_weight(5)
   Person.lose weight(bob, 5)
```

```
class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
 6
      def __init__(self, age, weight):
           self.age = age
10
           self.weight = weight
11
    Problem
12
                                unt):
13
    variables (i.e. names)
                                 to a Person object"""
14
15
16
                                unt):
    must be defined at
17
                                ight from a Person object"""
    'compile time'
18
19
20
       def have_birthday(self):
           "Increase the age of a Person object by 1"""
21
22
           self.age += 1
23
  joe = Person(12, 88)
  bob = Person(30, 155)
26
  bob.lose_weight(5)
  Person.lose weight(bob, 5)
```

```
class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       1111111
       def __init__(self, age, weight):
           self.age = age
           self.weight = weight
10
11
             Solution:
       def q
12
              use a dictionary
                                           rson object"""
13
14
15
16
       def l
17
                                           a Person object"""
18
19
20
       def have birthday(self):
           """Increase the age of a Person object by 1"""
21
           self.age += 1
22
   people = {}
   people['joe'] = Person(12, 88)
   people['bob'] = Person(30, 155)
28
   people['bob'].have_birthday()
30 print people['bob'].age
```

Whoa.

```
class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       HIIII.
       def __init__(self, age, weight):
           self.age = age
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
13
                                           son object"""
           se Solution:
14
15
16
       def lo
              'keys' can be defined
                                          a Person object"""
17
18
              at 'runtime' (!!)
19
20
       def ha
              Increase the age of a Person object by 1"""
21
22
           self.age += 1
23
   people = {}
25
   people['joe'] = Person(12, 88)
   people['bob'] = Person(30, 155)
                                                                              Whoa.
   people['bob'].have_birthday()
   print people['bob'].age
```

```
class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       1111111
       def __init__(self, age, weight):
           self.age = age
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
           """Add 'amount' wel
13
           self.weight += amol Solution:
14
15
16
       def lose weight(self,
                              the 'value' associated
           """Remove 'amount'
17
           self.weight -= amou
18
                              with each 'key' is an
19
       def have_birthday(self instance of Person
20
           """Increase the age of a Person poject by I
21
22
           self.age += 1
23
   people = {}
25
   people['joe'] = Person(12, 88)
   people['bob'] = Person(30, 155)
28
   people['bob'].have birthday()
30 print people['bob'].age
```

Whoa.

```
class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       1111111
       def __init__(self, age, weight):
           self.age = age
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
           """Add 'amount' we:
13
14
           self.weight += amou
                               Analysis:
15
16
       def lose_weight(self,
17
           """Remove 'amount'
                               So, people['bob'] is an
           self.weight -= amo
18
                               instance of Person
19
20
       def have birthday(self
           """Increase the age of a Person object by I
21
22
           self.age += 1
23
   people = {}
25
   people['joe'] = Person(12, 88)
   people['bob'] = Person(30, 155)
28
                                                                              Whoa.
   people['bob'].have_birthday()
   print people['bob'].age
```

```
class Person:
       This is the docstring for the Person class!!
         The Person class (loosely) represent a person.
       1111111
       def __init__(self, age, weight):
           self.age = age
           self.weight = weight
10
11
12
       def gain_weight(self, amount):
           """Add 'amount' wel
13
           self.weight += amod Analysis:
14
15
16
       def lose weight(self,
                              This calls the
           """Remove 'amount'
17
           self.weight -= amou
18
                              have_birthday() method
19
       def have_birthday(self for this Person instance
20
           """Increase the age of a Person object by I
21
22
           self.age += 1
23
   people = {}
25
   people['joe'] = Person(12, 88)
   people['bob'] = Person(30, 155)
28
                                                                             Whoa.
   people['bob'].have_birthday()
   print people['bob'].age
```

Now we just need a function that can:

- 1) create a Person instance
- 2) add the instance to a dictionary

```
our example import Person
   def add_person(people, p):
       people[p['name']] = Person(p['age'], p['weight'])
   def list people(people):
       for name, person in people.items():
10
           print name
           print '
11
                       age: %s' % person.age
           print ' weight: %s' % person.weight
13
14
15 if
      name == ' main ':
16
      people = {}
       person = {}
17
18
       while True:
           print '----'
19
                                         Name ")
           person['name'] = raw input("
20
           if person['name'] == '':
21
               break
23
           person['age'] = raw input("
                                        Age: ")
           person['weight'] = raw_input("Weight: ")
25
26
           add_person(people, person)
27
       list_people(people)
28
```

Solution:

This implements both requirements

what's all that other stuff?



```
our example import Person
   def add person(people, p):
       people[p['name']] = Person(p['age'], p['weight'])
   def list people(people):
       for name, person in people.items():
10
           print name
           print '
11
                       age: %s' % person.age
           print ' weight: %s' % person weight
12
13
14
      name == ' main
16
       people = {}
17
      person = {}
18
      while True:
           print '----
19
           person['name'] = raw input(" Name: ")
20
           if person['name'] == '':
21
               break
22
23
           person['age'] = raw input("
                                        Age: ")
           person['weight'] = raw input("Weight: ")
25
           add_person(people, person)
26
27
       list_people(people)
28
```

Getting User Input:

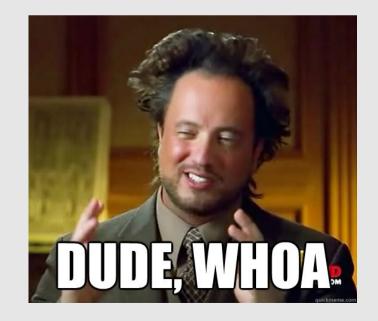
Used to store input from user about a Person



```
our example import Person
   def add person(people, p):
       people[p['name']] = Person(p['age'], p['weight'])
   def list people(people):
       for name, person in people.items():
10
           print name
           print '
                       age: %s' % person.age
11
           print ' weight: %s' % person.weight
12
13
14
     name == ' main ':
16
       people = {}
17
       person = {}
18
      while True:
           print '----'
19
           person['name'] = raw input(" Name: ")
20
           if person['name'] == '':
21
22
               break
23
           person['age'] = raw input("
                                         Age: ")
           person['weight'] = raw input("Weight: ")
25
26
           add_person(people, person)
27
       list_people(people)
28
```

Getting User Input:

Get user input until they "just hit enter" at the **Name:** prompt without providing a name.



```
our example import Person
   def add_person(people, p):
       people[p['name']] = Person(p['age'], p['weight'])
   def list people(people):
       for name, person in people.items():
10
           print name
           print '
11
                       age: %s' % person.age
           print ' weight: %s' % person.weight
12
13
14
     name == ' main ':
16
       people = {}
       person = {}
17
18
       while True:
           print '----'
19
20
           person['name'] = raw input("
                                         Name: ")
           if person['name'] == '':
21
               break
22
23
           person['age'] = raw input("
                                          Age: ")
           person['weight'] = raw input("Weight:
25
26
           add person(people, person)
27
       list_people(people)
28
```

Getting User Input:

Collect person attributes from user via keyboard



```
our example import Person
   def add person(people, p):
       people[p['name']] = Person(p['age'], p['weight'
   def list people(people):
       for name, person in people.items():
10
           print name
           print '
11
                       age: %s' % person.age
           print ' weight: %s' % person.weight
12
13
14
     name == ' main ':
16
       people = {}
       person = {}
17
18
       while True:
           print '-----
19
           person['name'] = raw input(" Name:/'
20
           if person['name'] == '':
21
22
               break
23
           person['age'] = raw input("
                                          Age: ")
           person['weight'] = raw_input "Weight: ")
25
           add_person(people, person)
26
27
       list_people(people)
28
```

Adding the Person:

Converts person dictionary into a Person instance and adds it to the...

people dictionary!



```
our example import Person
   def add person(people, p):
       people p['name'] = Person(p['age'], p['weight'])
   def list people(people):
       for name, person in people.items():
10
           print name
           print '
11
                       age: %s' % person.age
           print ' weight: %s' % person.weight
12
13
14
     name == ' main ':
16
       people = {}
       person = {}
18
      while True:
           print '-----
19
           person['name'] = raw input(" Name: ")
20
           if person['name'] == '':
21
22
               break
23
           person['age'] = raw input("
                                        Age: ")
           person['weight'] = raw input("Weight: ")
25
           add_person(people, person)
26
27
       list_people(people)
28
```

Adding the Person:

This is a **string** containing the name entered by the user

this string will be a key



```
our example import Person
   def add person(people, p):
       people[p['name']] = Person(p['age'], p['weight'])
   def list people(people):
       for name, person in people.items():
10
           print name
           print '
11
                       age: %s' % person.age
           print ' weight: %s' % person.weight
12
13
14
     name == ' main ':
16
       people = {}
       person = {}
17
18
      while True:
           print '----'
19
           person['name'] = raw input(" Name: ")
20
           if person['name'] == '':
21
22
               break
23
           person['age'] = raw input("
                                        Age: ")
           person['weight'] = raw input("Weight: ")
25
           add_person(people, person)
26
27
       list_people(people)
28
```

Adding the Person:

Create a Person instance with the **age** and **weight** values entered by the user



```
our example import Person
   def add person(people, p):
       people[p['name']] = Person(p['age'], p['weight'])
   def list people(people):
       for name, person in people.items():
10
           print name
           print '
11
                       age: %s' % person.age
           print ' weight: %s' % person.weight
12
13
14
     name == ' main ':
16
       people = {}
       person = {}
17
18
       while True:
           print '-----
19
           person['name'] = raw input(" Name: ")
20
           if person['name'] == '':
21
22
               break
23
           person['age'] = raw input("
                                        Age: ")
           person['weight'] = raw input("Weight: ")
25
           add_person(people, person)
26
27
       list_people(people)
28
```

Adding the Person:

Insert the instance into the dictionary.

We have something like:

{'bob': <Person instance>}



```
our example import Person
   def add person(people, p):
       people[p['name']] = Person(p['age'], p['weight'])
  def list people(people):
       for name, person in people.items():
           print name
           print '
                       age: %s' % person.age
           print ' weight: %s' % person.weight
  if name == ' main ':
16
       people = {}
       person = {}
17
18
      while True:
           print '-----
19
           person['name'] = raw input(" Name: ")
20
           if person['name'] == '':
21
22
              break
23
           person['age'] = raw input("
                                        Age: ")
           person['weight'] = raw input("Weight: ")
25
           add_person(people, person)
26
27
       list_people(people)
28
```

Listing People:

Simply cycles through the dictionary and prints all the People we have inserted



```
our example import Person
   def add_person(people, p):
       people[p['name']] = Person(p['age'], p['weight'])
  def list people(people):
       for name, person in people.items():
10
           print name
           print '
11
                       age: %s' % person.age
           print ' weight: %s' % person.weight
12
13
14
  if __name__ == '__main ':
15
16
       people = {}
       person = {}
17
18
       while True:
           print '----'
19
           person['name'] = raw input(" Name: ")
20
           if person['name'] == '':
21
               break
22
23
           person['age'] = raw input("
                                         Age: ")
24
           person['weight'] = raw input("Weight: ")
25
           add_person(people, person)
26
27
       list_people(people)
28
```

Wait.

There is a pattern here.



```
our example import Person
   def add_person(people, p):
       people[p['name']] = Person(p['age'], p['weight'])
  def list_people people :
       for name, person in people.items():
10
           print name
           print '
11
                       age: %s' % person.age
           print ' weight: %s' % person.weight
13
14
      name == ' main ':
16
       people = {}
       person = {}
17
       while True:
18
19
           print '----'
           person['name'] = raw input(" Name: ")
20
           if person['name'] == '':
21
               break
23
           person['age'] = raw input("
                                         Age: ")
           person['weight'] = raw input("Weight: ")
           add_person(people, person)
26
27
28
       list_people(people)
```

These functions all operate on people

maybe People should be an Object

```
class Person:
    This is the docstring for the Person class!!
     The Person class (loosely) represent a person.
    def __init__(self age, weight):
        self.age = age
        self.weight = weight
    def gain_weight(self, amount):
        """Add 'amount' weight to a Person object"""
        self.weight += amount
    def lose_weight(self, amount):
        """Remove 'amount' weight from a Person object"""
        self.weight -= amount
    def have_birthday(self):
        """Increase the age of a Person object by 1"""
        self.age += 1
joe = Person(12, 88)
bob = Person(30, 155)
joe.have birthday()
Person.have birthday(joe)
```

An example of encapsulation

```
1 from our example import Person
 3 class People:
       def init (self):
           self.people = {}
       def add(self):
           print '-----'
10
           name = raw_input(" Name: ")
           if name == '':
11
12
               return False
13
14
           age = raw input(" Age: ")
15
           weight = raw_input("Weight: ")
16
           self.people[name] = Person(age, weight)
17
18
           return True
19
       def show(self):
20
           for name, person in self.people.items():
21
22
               print name
23
               print ' age: %s' % person.age
24
               print ' weight: %s' % person.weight
25
26
   if name == '__main__':
       people = People()
28
29
30
       while people.add():
31
           pass
32
33
       people.show()
```

People as a Class:

Encapsulates a dictionary of Person instances

users of the People class

never access the dictionary

of instances directly

...they use the methods provided by the People class instead.