PYTHON TIPS

CS16: Introduction to Data Structures & Algorithms

Don’t name an instance variable the same as a method’s name!

```
def __init__(self):
  self.size = 3

def size(self):
  return self.size
```

```
def __init__(self):
  self._size = 3

def size(self):
  return self._size
```

Don’t mess with signatures!

If we give you a Python stencil with a method **capacity(self)...**

```
def getMyAwesomeCapacity(self):
  return self._capacity
```

```
def capacity(self):
  return self._capacity
```

Don't leave "pass" lying around.

In Python "pass" is just a filler statement – it does absolutely nothing.

```
def array_max(input):
  pass
  max = input[0]
  for i in input:
    if i > max:
      max = I
    else:
      pass
    pass
  return max
  pass
```

```
def array_max(input):
  max = input[0]
  for i in input:
    if i > max:
      max = I
  return max
```

Range is [inclusive, exclusive)

```
def print_list(list):
  for i in range(0, len(list)-1):
    print list[i]
```

```
def print_list(list):
  for i in range(0, len(list)):
    print list[i]
```

Don’t use indices when you don’t have to

```
def print_list(list):
  for i in range(0, len(list)):
    print list[i]
```

```
def print_list(list):
  for element in list:
    print element
```

Returning booleans

```
def is_empty(self):
  if self._size == 0:
    return True
  else:
    return False
```

```
def is_empty(self):
  return self._size == 0
```

Don’t use tabs ever!

```
def method(self):
(TAB)(SPACE)(SPACE)
```

```
def method(self):
(SPACE)(SPACE)(SPACE)(SPACE)
```

Follow specification exactly!

```
# Returns the height of the tree
# and throws an exception if empty.
```

```
def height(self):
  if not self.is_empty():
    return self.height
  else:
    print “Invalid height!”
```

```
def height(self):
  if not self.is_empty():
    return self.height
  else:
    raise InvalidHeightException()
```