6.00 Handout, Lecture 7
(Not intended to make sense outside of lecture)

```
def isPalindrome(s):
    """Returns True if s is a palindrome and False otherwise"""
    if len(s) <= 1:
        return True
    else:
        return s[0] == s[-1] and isPalindrome(s[1:-1])

def isPalindrome1(s, indent):
    """Returns True if s is a palindrome and False otherwise"""
    print indent*' ', 'isPalindrome1 called with', s
    if len(s) <= 1:
        print indent*' ', 'About to return True from base case'
        return True
    else:
        ans = s[0] == s[-1] and isPalindrome1(s[1:-1], indent + indent)
        print indent*' ', 'About to return', ans
        return ans
```

```
s = 0.0
for i in range(10):
    s += 0.1
print s
print s == 1.0
print s == .99999999999999989
```

```
x = 1e300
print x
print (x*x)/x
== x*(x/x)
print x*(x/x)
print (x*x)/x
```

```
def squareRootBi(x, epsilon):
    """Assumes x a non-negative float,
               epsilon a non negative float > 0
    Return y s.t. y*y is within epsilon of x"""
    assert float(x) and float(epsilon) and epsilon > 0.0
    low = 0
    high = x
    guess = (low + high)/2.0
    ctr = 1
    while abs(guess**2 - x) > epsilon and ctr <= 100:
        #print 'low:', low, 'high:', high, 'guess:', guess
        if guess**2 < x:
            low = guess
        else:
            high = guess
        guess = (low + high)/2.0
        ctr += 1
    assert ctr <= 100, 'Iteration count exceeded'
    print 'Iteration:', ctr, 'guess:', guess
    return guess

squareRootBi(4, 0.0001)
squareRootBi(9, 0.0001)
squareRootBi(1000, 0.0001)
squareRootBi(2, 0.0001)
squareRootBi(1234567890, 0.0001)
```