6.00 Handout, Lecture 8
(Not intended to make sense outside of lecture)

```
def f(n):
    assert n >= 0
    answer = 1
    while n > 1:
        answer *= n
        n -= 1
    return answer
```

```
def fact(n):
assert n >= 0
if n <= 1:
    return n
else:
    return n*fact(n - 1)
```

```
def g(n):
    x = 0
    for i in range(n):
        for j in range(n):
            x += 1
    return x
```

```
def h(x):
    assert type(x) == int and x >= 0
    answer = 0
    s = str(x)
    for c in s:
        answer += int(c)
    return answer
```

```
def search(L, e):
    for elem in L:
        if elem == e:
            return True
        if elem > e:
            return False
    return False

def bSearch(L, e, low, high):
    global numCalls
    numCalls += 1
    if high - low < 2:
        return L[low] == e or L[high] == e
    mid = low + int((high - low)/2)
    if L[mid] == e:
        return True
    if L[mid] > e:
        return bSearch(L, e, low, mid - 1)
    else:
        return bSearch(L, e, mid + 1, high)

def search(L, e):
    return bSearch(L, e, 0, len(L) - 1)

L = range(100)
numCalls = 0
print search(L, 3)
print numCalls
…
```