

6.00 Handout, Lecture 10
(Not intended to make sense outside of lecture)

```
def bsearch(s, e, first, last):
    """assumes s is a sorted list
       returns True if e is in s and false otherwise."""
    global numCalls
    if (last - first) < 2:
        return s[first] == e or s[last] == e
    mid = int((last + first)/2)
    if s[mid] == e:
        return True
    numCalls += 1
    if s[mid] > e:
        return bsearch(s, e, first, mid - 1)
    else:
        return bsearch(s, e, mid + 1, last)

def testSearch():
    global numCalls
    for power in range(1, 20):
        n = 2**power
        print n
        numCalls = 1
        print bsearch(range(n), n+1, 0, n - 1), numCalls

def bsearch1(s, e, first, last):
    while first <= last:
        mid = int((last + first)/2)
        if s[mid] > e:
            last = mid - 1
        elif s[mid] < e:
            first = mid + 1
        else:
            return True
    return False

def selSort(L):
    """Assumes that L is a list of
       elements that can be compared
       using >.
       sorts L in ascending order"""
    for i in range(len(L) - 1):
        minIndx = i
        minVal= L[i]
        j = i + 1
        while j < len(L):
            if minVal > L[j]:
                minIndx = j
                minVal= L[j]
            j = j + 1
        temp = L[i]
        L[i] = L[minIndx]
        L[minIndx] = temp
```