```
# sorting.py

import time, random

######################################################
# swap
######################################################

def swap(a, i, j):
    (a[i], a[j]) = (a[j], a[i])

######################################################
# selectionSort
######################################################

def selectionSort(a):
    n = len(a)
    for startIndex in xrange(n):
        minIndex = startIndex
        for i in xrange(startIndex, n):
            if (a[i] < a[minIndex]):
                minIndex = i
        swap(a, startIndex, minIndex)

######################################################
# bubbleSort
######################################################

def bubbleSort(a):
    n = len(a)
    end = n
    swapped = True
    while (swapped):
        swapped = False
        for i in xrange(1, end):
            if (a[i-1] > a[i]):
                swap(a, i-1, i)
                swapped = True
        end -= 1

######################################################
# mergeSort
######################################################

def merge(a, start1, start2, end):
    index1 = start1
    index2 = start2
    length = end - start1
    aux = [None] * length
    for i in xrange(length):
        if (index1 == start2):
            aux[i] = a[index2]
            index2 += 1
        elif (index2 == end):
            aux[i] = a[index1]
            index1 += 1
```

```
        elif (a[index1] < a[index2]):
            aux[i] = a[index1]
            index1 += 1
        else:
            aux[i] = a[index2]
            index2 += 1
    for i in xrange(start1, end):
        a[i] = aux[i - start1]

def mergeSort(a):
    n = len(a)
    step = 1
    while (step < n):
        for start1 in xrange(0, n, 2*step):
            start2 = min(start1 + step, n)
            end = min(start1 + 2*step, n)
            merge(a, start1, start2, end)
        step *= 2

####################################################
# builtinSort (wrapped as a function)
####################################################

def builtinSort(a):
    a.sort()

####################################################
# testSort
####################################################

def testSort(sortFn, n):
    a = [random.randint(0,2**31) for i in xrange(n)]
    sortedA = sorted(a)
    startTime = time.time()
    sortFn(a)
    endTime = time.time()
    elapsedTime = endTime - startTime
    assert(a == sortedA)
    print "%20s n=%d  time=%6.3fs" % (sortFn.__name__, n, elapsedTime)

n = 2**12
for sortFn in [selectionSort, bubbleSort,
               mergeSort, builtinSort]:
    testSort(sortFn, n)
```