Lab 09: Runtime Analysis

AUTHOR
Christian Osendorfer

Runtimes

```
def intersect(L1, L2):!
    tmp = []
    for e1 in L1:
        for e2 in L2:
            if e1 == e2:
                tmp.append(e)
    res = []
    for e in tmp:
        if not(e in res):
            res.append(e)
    return res
```

What is this function doing? What is the runtime of each of the two main blocks? What is the overall runtime?

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

What is this function doing? What is the runtime? Provide an equivalent algorithm of runtime $O(1)$.

Implement in Python

As usual, do not use any additional `import`ed modules that would do the main task. Implement

- Search for sorted and non-sorted lists.
- Bubble sort.
- Selection sort.
- Insertion sort
- Merge sort.

Test on randomly generated lists with sizes 100, 1000, 1000000, 10000000 elements, properly `time` these runs (that is, do many iterations to see an average behaviour) and provide a printout of these timing results for all tested algorithms on the console in a tabular way.