Beihang University Beijing, Nov. 8th, 2019
School of General Engineering Sebastian Wandelt, Pengfei Yi, Junyu Zhao

Computer Science and Programming Lab Class 8

Task 1. Time complexity analysis - Master Method *(15 minutes)*

(1) The recursive code for Pascal's triangle is shown as follows. First, estimate the runtime complexity by performing experiments with different values of n. For each n, measure the runtime using the *time.time()* function. Then, please compute the computational complexity and use $O(f(n))$ to denote the result.

```
def pascal(n,line_i):
    print (line_i)
    if len(line_i)<n:
        line_ii=[1]
        for i in range(len(line_i)-1):
            line_ii.append(line_i[i]+line_i[i+1])
        line_ii.append(1)
        pascal(n,line_ii)
pascal(8,[1])
```

(2) Assume that a divide-and-conquer algorithm has a time complexity of $T(n)$. Please calculate the time complexity using Big-O notation for algorithms with following recurrence relations. The Master method introduced in lecture might be helpful for the calculation.

a) $T(n) = 2T(n/2) + 3n$
b) $T(n) = 2T(n/2) + 12.4n^2$
c) $T(n) = 3T(n/4) + 3.87n$
d) $T(n) = 4T(n/2) + O(n^2)$

Task 2. Integer Multiplication - Naive way *(15 minutes)*

Implement the naive multiplication algorithm introduced in the lecture which has quadratic time complexity. Test the runtime of the function for several integer pairs of different lengths to see if the runtime raises as expected.

Task 3. Integer Multiplication - Karatsuba's algorithm *(25 minutes)*

Implement Karatsuba's algorithm, a faster algoritm for integer multiplication, which is also introduced in the lecture. Similar with last task, test the runtime of the algorithm for several integer pairs of different lengths to see the runtime and compare it with the naive algorithm.

Task 4. k-selection - naive *(15 minutes)*

Implement a function to select the k-th smallest element in an unsorted list using the naive idea introduced in the lecture (to select the k^{th} element in the sorted list) and test the runtime of such function of lists with different lengths.

Task 5. k-selection - random *(25 minutes)*

Implement another function to select the k-th smallest element in an unsorted list using the random selection strategy introduced in the lecture. If you forget about the concept, go back to the slides for a short review, but do not directly copy the code or the psuedocode from the lecture slides.

With the analysis from the lecture, the time complexity of this method is in $O(n^2)$. Is this method faster or the naive method faster? Use lists with different lengths to test the runtime of both methods and compare them to see if the result is as expected.