CSE 321

Homework 3

Deadline: 24.12.2020 23:55

1) Solve the following recurrence relation and give Θ relation for each of them.

a) $T(n)=27\,T(n/3)+n^2$

b) $T(n)=9\,T(n/4)+n$

c) $T(n)=2\,T(n/4)+\sqrt{n}$

d) $T(n)=2\,T(\sqrt{n})+1$

e) $T(n)=2T(n-2)$, $T(0)=1$, $T(1)=1$

f) $T(n)=4T(n/2)+n$, $T(1)=1$

g) $T(n)=2\,T(\sqrt[3]{n})+1$, $T(3)=1$;

2)

How many lines (as a function of n) does the following program print? Write a recurrence relation and solve it by backward substitution. You may assume that n is a power of 2.

```
function f(n)
        if n <= 1:
                print_line("**")
        else:
                for i=1 to n
                        f(n/2)
                end for
```

3) Let T(n) denote the worst case number of comparisons (A[0]>A[1]) made by the following function for an input array of n numbers. Give a recurrence relation for T(n). Solve the recurrence relation.

```
Algorithm Function_f (A[0..n-1])
 //Input: Array A of n numbers
```

```
//Output: A is sorted in increasing order
if n=2 and A[0]>A[1], then swap(A[0],A[1])
if n>2 then {
        Function_f (A[0..ceil(2n/3)]) .
        Function_f (A[floor(n/3)..n])
        Function_f (A[0..ceil(2n/3)])
}
```

4)

Implement the quick sort and insertion sort algorithms and count the number of swap operations to compare these two algorithms. Analyze the average-case complexity of the algorithms. Compare the operations count in your report file to decide which algorithm is better and support your analysis by using the theoretical average-case analysis of your algorithms.

5) What are the running times of each of these algorithms (in big-O notation), and which would you choose**?**

a) An algorithm that divides the problem into 5 subproblems where the size of each subproblem is one third of the original problem size, solves each subproblem recursively and then combines the solutions to the subproblems in quadratic time.

b) An algorithm that divides the problem into 2 subproblems where the size of each subproblem is half of the original problem size, solves each subproblem recursively and then combines the solutions to the subproblems in $O(n^2)$ time.

c) An algorithm that solves the problem by recursively solving the subproblem of size n-1 and then combine the solutions in linear time.