

Exercise for Analysis of Algorithms

Exercise 1

Consider the following algorithm that computes the maximum element in an array of positive integers. We assume that all elements are pairwise different and that each permutation occurs with equal probability.

```
int maxElem(int *a, int N)
{
    int i,max;

    max = -1;           /* 1 */
    for (i=0; i<N; i++) /* 2 */
        if (a[i] > max) /* 3 */
            max = a[i]; /* 4 */
    return max;         /* 5 */
}
```

- How often are the lines 3 and 4 executed in the *worst case* (in the *best case*)?
- What is the probability that this worst case occurs?
- How often are the lines 3 and 4 executed in the *average case*?

Exercise 2

Let S_N be the expected number of pushes on the stack in the Quicksort program, where the input consists of a random permutation of N distinct keys. Analyze S_N for all values of N and M .

Exercise 3

As we discussed at the first discussion session, we would assume that C' —the number of comparisons done in quicksort during partitioning phases *on the left side* of the array — should be around half the number of comparisons, i.e., around $C/2$.

This is as all comparisons are distributed among the following two lines in the program. We split the number of comparisons into C' for the first line and C'' for the second line.

```
do {i++;} while(a[i]<k);
do {j--;} while(k<a[j]);
```

Can we motivate the recurrence relation for C'_N , which shows this?

Exercise 4

Two natural numbers $m \neq n$ are called *friendly*, if the sum of all factors of m equals n — or the other way round. A son and a father wrote the following programs, that compute friendly numbers. What is their running time?

Son

```
#include <iostream>
int e[150000];
int realdiv(int a) {
    int n=0;
    for(int i=1; i+i<=a; i++)
        if(a%i==0) n+=i;
    e[a] = n;
    return n;
}
main() {
    for(int i=0; i<150000; i++) {
        int a = realdiv(i);
        if(a >= i) continue;
        if(e[a]==i) {
            std::cout << i << " "
            << realdiv(i) << "\n";
        }
    }
}
```

Father

```
#include <stdio.h>
#define N 1000000
int factorsum[N];
int main() {
    int i;
    for(i=1; i<N; i++) {
        int p=i;
        while(p<N) {
            factorsum[p] += i;
            p += i;
        }
    }
    for(i=1; i<N; i++) {
        int a = factorsum[i]-i;
        if(a<i && i==factorsum[a]-a)
            printf("%d %d\n", a, i);
    }
    return 0;
}
```