

Analysis of Algorithms

Problem 2-1

In the analysis of the quicksort algorithm, the term S_N represents the expected number of pushes to the stack. Find out an expression for S_N .

Problem 2-2

Consider the following algorithm. The array $a[]$ contains a random permutation of the numbers $1, \dots, N$.

```
void doSomething (int a[], int N) {  
    for (int i = 0; i < N - 1; ++i)  
        while (a[i] < a[i+1])  
            ++a[i];  
}
```

What is the expected number of executions of the instruction $++a[i]$?

Homework Assignment 2-1 (10 Points)

We already analyzed C_n , the *total* expected number of comparisons in the two innermost *while*-loops of the quicksort algorithm (see the program fragment below).

What is the expected number of executions of the single comparison $a[i] < k$?

```
[...]  
    i = l - 1; j = r ; k = a[j];  
    do{  
        do{i++;} while ( a[i] < k );  
        do{j--;} while ( k < a[j] );  
        t = a[i]; a[i] = a[j]; a[j] = t;  
    } while ( i < j );  
[...]
```

Homework Assignment 2-2 (10 Points)

Last week we analysed the running of a program that computes *amicable numbers*. Recall that two natural numbers $m \neq n$ are called *amicable*, if the sum of all proper factors of m equals n — and the other way around.

This week we will turn to the father's program. Determine the expected number of executions of the instruction $p += i$ as a function of N of the form $f(N) + O(N)$.

Which of the two programs is faster?

Son

```
1 #include <iostream>

3 int e[150000];
  int realdiv(int a) {
5   int n=0;
    for(int i=1; i+i<=a; i++)
7     if(a%i==0) n+=i;
    e[a] = n;
9   return n;
  }

11 main() {
13   for(int i=0; i<150000; i++) {
        int a = realdiv(i);
15     if(a >= i) continue;
        if(e[a]==i) std::cout << i
17         << " " << a << "\n";
    }
19 }
```

Father

```
1 #include <stdio.h>
  #define N 1000000
3 int factorsum[N];
  int main() {
5   int i;
    for(i=1; i<N; i++) {
7     int p=i;
        while(p<N) {
9         factorsum[p] += i;
            p += i;
11        }
    }

13   for(i=1; i<N; i++) {
        int a = factorsum[i]-i;
15     if(a<i && i==factorsum[a]-a)
        printf("%d %d\n", a, i);
17    }
    return 0;
19 }
```