University of Zurich[UZH]

Department of Informatics
Database Technology Group
Prof. Dr. M. Böhlen

# Informatics II
# Exercise 3

Mar 04, 2019

## Algorithmic Complexity and Correctness

**Task 1.** Algorithm `whatDoesItDo(A,n)` gets an array `A[1...n]` of $n$ integers as an input.

**Algo:** whatDoesItDo(A,n)

```
counter = 0;
val = 1;
B = A;
C[1..n] = 0;
for i = 1 to n do
    for j = 1 to n do
        if A[i] == B[j] then
            C[j] = val;
            val = val+1;
    val = 1;
for i = 1 to n do
    if C[i] == 2 then
        counter = counter + 1;
return counter;
```

a) Implement the algorithm as a C program that reads the elements of $A$ and prints the result.

b) Describe what the algorithm does.

c) Do an exact analysis of the running time and calculate the asymptotic complexity of the algorithm.

**Task 2.** Consider the selection sort algorithm as it is defined in the lecture slides. Formulate the loop invariant for the outer loop of the selection sort algorithm and prove its correctness.

University of Zurich[UZH]

Department of Informatics
Database Technology Group
Prof. Dr. M. Böhlen

# Asymptotic Complexity

**Task 3.** A quadratic algorithm with processing time $T(n) = cn^2$, where $c$ is a constant factor, spends $T(N)$ seconds for processing $N$ data items. How much time will be spent for processing $n = 5000$ data items, assuming that $N = 100$ and $T(N) = 1$ms?

**Task 4.** Calculate the asymptotic tight bound for the following functions and rank them by their order of growth (lowest first). Clearly work out the calculation steps in your solution.

$$
\begin{aligned}
f_1(n) &= \sqrt{n} + 24 \log n + 24 \\
f_2(n) &= 4^{2n} + n^n + 8n \log(9) \\
f_3(n) &= 4^{log_2 n} \\
f_4(n) &= \log(14^n) + 14\sqrt{n} \\
f_5(n) &= n^2 \log(n+1) + n \log n^2 + 0.5n \\
f_6(n) &= 7n^4 + 100n \log n + \sqrt{32} + n \\
f_7(n) &= 3n^{1.5} + 2n + 50n^{2.5} + \log n \\
f_8(n) &= \log(34(n+11)n^{7n-8}) + 24^{48} \\
f_9(n) &= (n+2)! \\
f_{10}(n) &= 2\log(6^{\log n^2}) + \log(\pi n^2) + n^3
\end{aligned}
$$

# Special Case Analysis

**Task 5.** The informatics and psychology departments of UZH have prepared their seminar schedules for the upcoming semester. On Wednesdays, only one shared seminar room will be available. During the planning, the departments must compare their seminar schedules for Wednesdays to find out if there are any conflicts.

A seminar schedule is represented as an array of time-slots. A time-slot is defined through its starting and ending time. A conflict occurs if both departments want to use the seminar room during overlapping time-slots.

The seminar schedules of the informatics and psychology departments are stored in arrays I and P. For example, if $I = [\{8.30, 9.30\}, \{10.00, 12.45\}]$ and $P = [\{9.30, 10.00\}]$ there are no conflicts since no time-slots overlap. However, if $I = [\{8.30, 9.30\}, \{10.00, 12.45\}]$ and $P = [\{9.00, 10.00\}]$, there is a conflict because the time-slots $\{8.30, 9.30\}$ and $\{9.00, 10.00\}$ overlap from 9:00 to 9:30.

a) Given the schedule arrays of the informatics (I) and psychology (P) departments, specify all the special cases that need to be considered during the detection of conflicts. For each of the cases, provide an example of the input data and determine the corresponding output.

b) Write a Pseudocode for the function `int conflictsCalculator (slot I[], int nI, slot P[], int nP)`. The `type slot` is used for the im-

plementation of a time slot as a struct including the starting and the ending time:

```
struct timeslot {
        float init;
        float end;
};
```

The `conflictsCalculator` function detects the conflicts between the informatics and psychology schedules. For each conflict, it prints the index of the two time slots involved in the form $(index_I, index_P)$. At the end of this process, it returns the total number of conflicts detected. Make sure that your function runs properly for all the special cases you provided. An example output would be:

```
I = [{ 10.00, 11.00} ,{ 11.00, 12.45} ]
P = [{ 9.00, 12.00} ]
conflicting_slots = (0,0) (1,0)
conflicts = 2
```

Also implement your function in C to verify your pseudocode. Use the following input/output template for C code.

```
Type elements of I seperated by spaces (non-number to stop):10.00 11.00 11.00 12.45
Type elements of P seperated by spaces (non-number to stop): 9.00 12.00
conflicting_slots = (0,0) (1,0)
conflicts = 2
```