

Universität Leipzig Institut für Informatik Automatische Sprachverarbeitung	Algorithmen und Datenstrukturen WS 2017/18 – Serie 1		
Prof. Dr. Uwe Quasthoff	Ausgabe am 17.10.2017	Abgabe am 24.10.2017	Seite 1/2

Algorithmen und Datenstrukturen – Serie 1

1 (10 Punkte) Komplexitätsklassen

Gegeben sind die folgenden 10 Funktionen, bezeichnet als T_a, T_b, \dots, T_j :

$$\begin{aligned}
 T_a : n &\mapsto 100 + 10n^2 + \frac{n^5}{n-1} & T_b : n &\mapsto 4^{\frac{n}{2}+1} + n \\
 T_c : n &\mapsto n^2\sqrt{2n^2} & T_d : n &\mapsto 4\sqrt{e^{2n} + 3} \\
 T_e : n &\mapsto \log(2n^n) + \sqrt{n} & T_f : n &\mapsto \frac{5n}{2n+2^3} \\
 T_g : n &\mapsto \frac{4n^2}{\log(n^4+2)} & T_h : n &\mapsto 4n + \sin(2\pi n^3) \\
 T_i : n &\mapsto 1000n^{10} + 2^n + e^n & T_j : n &\mapsto 7n^2/\log n + 5n \log n + 20\sqrt{n^3}
 \end{aligned}$$

Berechnen Sie für jede Funktion T_x ($x = a, \dots, j$) die **jeweils kleinstmögliche** Komplexitätsklasse $O(f)$, so dass gilt $T_x \in O(f)$, für eine Funktion f von n .

Die möglichen Komplexitätsklassen sehen so aus: $O(1)$, $O(n^2)$, $O(n^{3/2})$, $O(n^4 \log n)$, $O(n^2/\log n)$ oder $O(3^n)$, wobei sie gegebenenfalls andere Konstanten verwenden müssen. Drücken sie die Klassen so weit vereinfacht wie möglich aus [je 1 Punkt]. Ordnen Sie die Funktionen T_x nach aufsteigender Komplexität, d.h. es stehe T_x vor T_y , genau dann wenn $T_x \in O(T_y)$.

2 (6 Punkte) Mastertheorem

Gegeben seien die folgenden Rekursionsgleichungen jeweils für eine Funktion T :

- $T(n) = 3n^2(n/\log_2(8)) + 16T(n/2)$
- $T(n) = nT(n-1)$
- $T(n) = 16T(n/2) + n^4 + n^2$

Bestimmen Sie jeweils eine exakte Schranke für das Verhalten von T . Finden Sie also eine Funktion f , sodass $T \in \Theta(f)$. In den Fällen, in denen das Master-Theorem anwendbar ist, beschreiben sie dessen Anwendung zum Finden von f ; geben sie a , b und k an. [je 2 Punkte]

3 (4 Punkte) Münzwiegen

Ein König besitzt einen Schatz vergoldeter Münzen. In dieser grossen Menge von identischen, gleich schweren Münzen befindet sich eine besonders schwere Münze aus purem Gold, die von den anderen Münzen nur durch ihr Gewicht zu unterscheiden ist.

Um die begehrte Stellung der Hofinformatikerin/des Hofinformatikers zu erlangen, müssen sie die Goldmünze mit möglichst wenigen Wiegevorgängen identifizieren. (Der Aufwand zum Abzählen oder Aufteilen von Münzmengen wird vernachlässigt.)

Mit der –als einzige Möglichkeit der Gewichtsbestimmung– zur Verfügung gestellten Waage lassen sich je zwei beliebig grosse Mengen von Münzen exakt vergleichen. Das

Universität Leipzig Institut für Informatik Automatische Sprachverarbeitung	Algorithmen und Datenstrukturen WS 2017/18 – Serie 1		
Prof. Dr. Uwe Quasthoff	Ausgabe am 17.10.2017	Abgabe am 24.10.2017	Seite 2/2

heisst, es lässt sich in je einem Wiegevorgang feststellen, ob zwei Mengen von Münzen gleich schwer sind bzw. welche von beiden schwerer ist. Beschreiben sie genau wie sie vorgehen, um mit möglichst wenigen Wiegevorgängen auszukommen. [4 Punkte]

Hinweise: 1) Es sind deutlich weniger Schritte nötig als linear zur Anzahl N der Münzen (nämlich nur $O(\log(N))$). 2) Mengen lassen sich auch in mehr als zwei Teile zerlegen.

4 (5 Punkte) Laufzeit- und Speicherplatzkomplexität

Gegeben sind zwei Funktionen als Pseudocode:

```

a) int a(int[] N, int[] M) {
    int A[M.length+1]; int B[M.length+1];
    for (int j=0; j<=M.length; j++) { A[j]=0; }
    for (int i=1; i<=N.length; i++) {
        B[0]=0;
        for (int j=1; j<=M.length; j++) {
            int x = A[i-1];
            if (N[i] == M[j]) { x=x+s; } else { x=x+r; }
            B[i]=max(0,max(x,max(B[i-1]+g,A[i]+g)));
        }
        A=B;
    }
    for (int j=1; j<=M.length; j++) { A[0]=max(A[0],A[j]); }
    return A[0];
}

b) int b(int n, int x) {
    int erg = x;
    if (n>1) {
        for (int i = 1; i <= n; i++) {
            erg = erg + (-1)^i * i;
        }
        erg = erg + b(n/3,erg/2);
        erg = erg + b(n/3,erg/4);
        erg = erg + b(n/3,erg/8);
    }
    return erg;
}

```

Geben Sie für die Funktionsaufrufe $a(N, M)$ und $b(n, 0)$ jeweils die exakte Schranke der Zeitkomplexität und Speicherplatzkomplexität in Abhängigkeit von n mittels Θ -Notation an. **Begründen sie Ihre Antwort!** Für (a) nehmen sie dabei an das beide Listen N und M die gleiche Länge n haben. Beachten sie, dass hier s , r und g Konstanten sind. Bei (b) beginnen sie damit, eine Rekursionsgleichung für den Zeitbedarf aufzustellen. [a) 2 Punkte, b) 3 Punkte]