Prof. Dr.-Ing Georg J. Schneider

Datenstrukturen und Algorithmen

Übung 2 Komplexitätsklassen

Aufgabe 1

Klassifizieren Sie die folgenden Größen mit Hilfe der O-Notation:

$1020\,n + 1$; $10^n + 1$; $n^{20} / n^{18} + 1$; 2^{100000} ; $3^3 * \log_2 n$.

Ordnen Sie die Größen in aufsteigender Reihenfolge und beschreiben Sie deren Wachstumsverhalten jeweils durch ein entsprechendes Adjektiv.

Aufgabe 2

Zur Lösung eines Problems seien drei verschiedene Algorithmen A_i , i = 1…3, vorhanden, deren Laufzeitfunktionen wie folgt bekannt sind:

- A_1: $1000\,n$
- A_2: $100\,n \log_2 n$
- A_3: 2^n

Wenn es auf möglichst geringen Zeitaufwand ankommt, welchen Algorithmus wählen Sie dann bei welcher Eingabegröße von *n*? Geben Sie dazu die entsprechenden „**Grenz"-Werte für *n*** an, also z.B. „ab *n* = …", oder „für *n* von … bis ...".
Hinweis: Wenn Sie die Gleichungen nicht mathematisch korrekt lösen können, versuchen Sie das Problem numerisch zu lösen (Tabelle).

Aufgabe 3

Welchen Wert hat die Variable *zaehler*, wenn nachfolgende Funktion durchlaufen wird? Der Zähler ist mit 0 initialisiert, `n` sei eine vorher festgelegte positive Ganzzahl. Geben Sie die Werte von 1 bis 8 an und leiten Sie daraus eine allgemeine Formel für die Laufzeit ab.
Hinweis: Unterscheiden Sie die Fälle n gerade, n ungerade!

```
def schleife(n):
    zaehler = 0
    for i in range(1, n + 1):
        for j in range(1, i + 1, 2):
            zaehler += 1
    return zaehler
```

In welcher **Komplexitätsklasse** liegt die Methode, wenn man annimmt, dass das Erhöhen des Zählers einen konstanten Zeitaufwand hat? Wie ändert sich der Zeitaufwand, wenn der Parameter n verdoppelt wird?

Beweisen Sie mit Hilfe der Definition der O-Notation ihre Behauptung bzgl. der Komplexitätsklasse.