Laufzeitanalyse (1) demogr.

```
for (int j = 1; j <= n; j = j * 2) {
   print(j);
}
```

Die Laufzeit in Abhängigkeit von n ist

- (1) $\Theta(\log n)$
- (2) $\Theta(n)$
- (3) $\Theta(n \log n)$
- (4) $\Theta(n^2)$
- (5) Θja, ich kann mich nicht entscheiden . . .

Auflösung: (1) $\Theta(\log n)$.

Laufzeitanalyse (2) demogr.

```
for (int i = 0; i < n; i++) {
    for (int j = 0; j < 2*i; j++) {
        print(j + i);
    }
}
```

Die Laufzeit in Abhängigkeit von n ist

- (1) $\Theta(\sqrt{n})$
- (2) $\Theta(n)$
- (3) $\Theta(n \log n)$
- (4) $\Theta(n^2)$
- (5) Zu früh ...

Auflösung: (4) $\Theta(n^2)$.

Laufzeitanalyse (3) demogr.

```
int j;
for (i = 1; i <= n; i++) {
    j = 0;
    while (j < n) {
        j = j + i*i;
    }
}
```

Die Laufzeit in Abhängigkeit von n ist

- (1) $\Theta(\log n)$
- (2) $\Theta(n)$
- (3) $\Theta(n \log n)$
- (4) $\Theta(n^2)$
- (5) jetzt reichts aber ...

Auflösung: (2) $\Theta(n)$.

Laufzeitanalyse (4) demogr.

```
int i=1;
for (int j = 1; j <= n; j = j+i) {
    i++;
    print(j);
}
```

Die Laufzeit in Abhängigkeit von n ist

- (1) $\Theta(\log n)$
- (2) $\Theta(\sqrt{n})$
- (3) $\Theta(n)$
- (4) $\Theta(n \log n)$
- (5) $\Theta(n^2)$

Auflösung: (2) $\Theta(\sqrt{n})$.

Laufzeitanalyse (5) demogr.

```
for (int i = 1; i < n; i++) {
    for (int j = 0; j < log(i*i); j++) {
        print(j + i);
    }
}
```

Die Laufzeit in Abhängigkeit von n ist

- (1) $\Theta(\log n)$
- (2) $\Theta(n)$
- (3) $\Theta(n \log n)$
- (4) $\Theta(n \log^2 n)$
- (5) $\Theta(n^2)$

Auflösung: (3) $\Theta(n \log n)$.

Laufzeitanalyse (6) demogr.

```
int j;
for (i = 1; i <= n; i++) {
    j = 2;
    while (j < n) {
        j = j * j;
    }
}
```

Die Laufzeit in Abhängigkeit von n ist

- (1) $\Theta(n \log \log n)$
- (2) $\Theta(n \log n)$
- (3) $\Theta(n \log^2 n)$
- (4) $\Theta(n \cdot \sqrt{n})$
- (5) $\Theta(n^2)$

Auflösung: (1) $\Theta(n \log \log n)$.

Rekursionsgleichungen (1) demogr.

Die Rekursion $T(1) = c$, $T(n) = a \cdot T\left(\frac{n}{b}\right) + t(n)$ sei zu lösen. n ist eine Potenz der Zahl $b > 1$ und $a \geqslant 1$, $c > 0$ gelte.

(a) Wenn $t(n) = O\left(n^{(\log_b a)-\varepsilon}\right)$ für eine Konstante $\varepsilon > 0$, dann ist $T(n) = \Theta(n^{\log_b a})$.

(b) Wenn $t(n) = \Theta(n^{\log_b a})$, dann ist $T(n) = \Theta(n^{\log_b a} \cdot \log_b n)$.

(c) Wenn $t(n) = \Omega\left(n^{(\log_b a)+\varepsilon}\right)$ für eine Konstante $\varepsilon > 0$ und $a \cdot t\left(\frac{n}{b}\right) \leqslant \alpha \cdot t(n)$ für eine Konstante $\alpha < 1$, dann $T(n) = \Theta(t(n))$.

Lösung für $T(1) = \Theta(1)$, $T(n) = 2 \cdot T(n/2) + \Theta(n)$?

- (1) $T(n) = \Theta(n^2)$
- (2) $T(n) = \Theta(n \cdot \log^2 n)$
- (3) $T(n) = \Theta(n \cdot \log n)$ $\surd$
- (4) $T(n) = \Theta(\log^2 n)$
- (5) $T(n) = \Theta(\log n)$

Rekursionsgleichungen (2) demogr.

Die Rekursion $T(1) = c$, $T(n) = a \cdot T\left(\frac{n}{b}\right) + t(n)$ sei zu lösen. n ist eine Potenz der Zahl $b > 1$ und $a \geqslant 1$, $c > 0$ gelte.

(a) Wenn $t(n) = O\left(n^{(\log_b a)-\varepsilon}\right)$ für eine Konstante $\varepsilon > 0$, dann ist $T(n) = \Theta(n^{\log_b a})$.

(b) Wenn $t(n) = \Theta(n^{\log_b a})$, dann ist $T(n) = \Theta(n^{\log_b a} \cdot \log_b n)$.

(c) Wenn $t(n) = \Omega\left(n^{(\log_b a)+\varepsilon}\right)$ für eine Konstante $\varepsilon > 0$ und $a \cdot t\left(\frac{n}{b}\right) \leqslant \alpha \cdot t(n)$ für eine Konstante $\alpha < 1$, dann $T(n) = \Theta(t(n))$.

Lösung für $T(1) = \Theta(1)$, $T(n) = 4 \cdot T(n/2) + \Theta(n \cdot \log n)$?

- (1) $T(n) = \Theta(n^4 \cdot \log n)$
- (2) $T(n) = \Theta(n^2 \cdot \log^4 n)$
- (3) $T(n) = \Theta(n^2 \cdot \log n)$
- (4) $T(n) = \Theta(n^2)$ $\surd$
- (5) $T(n) = \Theta(n \cdot \log^4 n)$

Rekursionsgleichungen (3) demogr.

Die Rekursion $T(1) = c$, $T(n) = a \cdot T\left(\frac{n}{b}\right) + t(n)$ sei zu lösen. n ist eine Potenz der Zahl $b > 1$ und $a \geqslant 1$, $c > 0$ gelte.

(a) Wenn $t(n) = O\left(n^{(\log_b a)-\varepsilon}\right)$ für eine Konstante $\varepsilon > 0$, dann ist $T(n) = \Theta(n^{\log_b a})$.

(b) Wenn $t(n) = \Theta(n^{\log_b a})$, dann ist $T(n) = \Theta(n^{\log_b a} \cdot \log_b n)$.

(c) Wenn $t(n) = \Omega\left(n^{(\log_b a)+\varepsilon}\right)$ für eine Konstante $\varepsilon > 0$ und $a \cdot t\left(\frac{n}{b}\right) \leqslant \alpha \cdot t(n)$ für eine Konstante $\alpha < 1$, dann $T(n) = \Theta(t(n))$.

Lösung für $T(1) = \Theta(1)$, $T(n) = 9 \cdot T(n/3) + n^3$?

- (1) $T(n) = \Theta(n^9 \cdot \log^3 n)$
- (2) $T(n) = \Theta(n^3 \cdot \log^2 n)$
- (3) $T(n) = \Theta(n^3)$ $\surd$
- (4) $T(n) = \Theta(n^2 \cdot \log^3 n)$
- (5) $T(n) = \Theta(n^2)$

Laufzeitanalyse (7) demogr.

```
for (int j = 1; j <= n; j = j * 2) {
    print(j);
}
```

Die Laufzeit in Abhängigkeit von n ist

- (1) $\Theta(\log n)$
- (2) $\Theta(\sqrt{n})$
- (3) $\Theta(n)$
- (4) $\Theta(n \log n)$
- (5) $\Theta(n^2)$

Auflösung: (1) $\Theta(\log n)$

Laufzeitanalyse (8) demogr.

```
int i=1;
for (int j = 1; j <= n; j = j+2*i) {
    i++;
    print(j-i*3);
}
```

Die Laufzeit in Abhängigkeit von n ist

- (1) $\Theta(\log n)$
- (2) $\Theta(\sqrt{n})$
- (3) $\Theta(n)$
- (4) $\Theta(n \log n)$
- (5) $\Theta(n^2)$

Auflösung: (2) $\Theta(\sqrt{n})$

Laufzeitanalyse (9) demogr.

```
int j;
for (i = 1; i <= n; i++) {
    j = 3;
    while (j < n) {
        j = j * j * j;
    }
}
```

Die Laufzeit in Abhängigkeit von n ist

- (1) $\Theta(n)$
- (2) $\Theta(n \log \log n)$
- (3) $\Theta(n \log n)$
- (4) $\Theta(n \log^2 n)$
- (5) $\Theta(n^2)$

Auflösung: (2) $\Theta(n \log \log n)$