Aufgabe 1. Ordnen Sie folgende Funktionen nach Dominanz, beginnend mit der asymptotisch am schwächsten wachsenden. Es genügt die Funktionen zu reihen und die Reihung erklären zu können, ein Beweis der Gültigkeit der Relationen ist nicht erforderlich.

$$\log_2(n^{10}) \quad \frac{(n+1)!}{100} \quad 0.78^{2n} \quad \frac{n^{4.5}}{n^2} \quad \log_2(2^n) \quad 3^n \quad 2^n n \quad 30n^2 \quad 100 \cdot n! \quad 4^{n/2}$$

---

1. $0.78^{2n} = (0.78^2)^n = \mathbf{0.6084^n}$
   (Geht gegen 0, wird daher von allem Dominiert)
2. $\log_2(n^10) = 10 \cdot \log_2(n) \to \mathbf{log(n)}$
   (Trivial)
3. $\log_2(2^n) = \mathbf{n}$
   (Trivial)
4. $30n^2 \to \mathbf{n^2}$
   (Trivial)
5. $\frac{n^{4.5}}{n^2} = \mathbf{n^{2.5}}$
   (Trivial)
6. $4^{n/2} = (4^{1/2})^n = \mathbf{2^n}$
   (Trivial)
7. $\mathbf{2^n \cdot n}$
   ($\lim_{n\to\infty} \frac{2^n}{2^n \cdot n} = \lim_{n\to\infty} \frac{1}{n} = 0$)
8. $\mathbf{3^n}$
   ($\frac{2^n \cdot n}{3^n} = \frac{2^n \cdot n}{(2^{\log_2(3)})^n} = \frac{2^n \cdot n}{(2^n)^{\log_2(3)}} = \frac{n}{(2^n)^{(\log_2(3)-1)}} = \frac{n}{(2^{(\log_2(3)-1)})^n} = \frac{n}{1.5^n}$; $\lim_{n\to\infty} \frac{n}{1.5^n} = 0$)
9. $100 \cdot n! \to \mathbf{n!}$
   (Trivial)
10. $\frac{(n+1)!}{100} = \frac{n+1}{100} \cdot n! \to \mathbf{(n+1)!}$
    ($\lim_{n\to\infty} \frac{100 \cdot n!}{(n+1)/(100) \cdot n!} = \lim_{n\to\infty} \frac{10000}{n+1} = 0$)

Aufgabe 2. Gegeben sind die folgende Funktionen:

$$f(n) = 30n^2(3^n + 2^n + 1^n)$$

$$g_1(n) = 3^n + n^2$$

$$g_2(n) = \begin{cases} 4^n & \text{falls } n \text{ durch 7 teilbar ist} \\ 2^n & \text{sonst} \end{cases}$$

$$g_3(n) = \begin{cases} 4^n + 300n^2 + 400n & \text{falls } n < 10^{30} \\ 20 \cdot 3^n \cdot n^2 + 20 \cdot 3^n \cdot n + n^2 & \text{sonst} \end{cases}$$

Kreuzen Sie in der folgenden Tabelle die zutreffenden Felder an und begründen Sie Ihre Antworten:

$f(n)$ ist in	$\Theta(.)$	$O(.)$	$\Omega(.)$	keines
$g_1(n)$			X	
$g_2(n)$				X
$g_3(n)$	X	X	X	

Hinweis: Setzten Sie statt dem Punkt die entsprechend Funktion (g_1, g_2 bzw. g_3) ein. Beispielsweise ist die Zelle links oben als „$f(n)$ ist in $\Theta(g_1(n))$“ zu lesen.

$f(n)$ wächst schneller als (dominiert) $g_1(n)$.

$f(n)$ wächst langsamer als 4^n, aber scheller als 2^n. Für $f(n) = O(g_2(n))$ müsste $f(n) \leq g_2(n)$ für **alle** $n > n_0$ gelten. Das selbe gilt für $f(n) = \Omega(g_2(n))$ mit $f(n) \geq g_2(n)$.

Der *falls* $n < 10^{30}$ Teil ist irrelevant. Wegen der Additivität und der Irrelevanz von Konstanten kann sowohl $30n^2(3^n + 2^n + 1^n)$ als auch $20 \cdot 3^n \cdot n^2 + 20 \cdot 3n \cdot n + n^2$ auf $3^n \cdot n^2$ reduziert werden.

Aufgabe 3. Beweisen oder widerlegen Sie die folgenden Aussagen:

(a) $\max(f(n), g(n)) = \Theta(f(n) + g(n))$.

(b) $\min(f(n), g(n)) = \Theta(f(n) + g(n))$.

(c) Wenn $f_1(n) = O(g_1(n))$ und $f_2(n) = O(g_2(n))$, dann $f_1(n) \cdot f_2(n) = O(g_1(n) \cdot g_2(n))$.

Anmerkung: Wir nehmen an, dass alle Funktionen die nicht-negativen ganzen Zahlen als Definitions- und Wertebereich haben.

$f(n) + g(n) = \Omega(max(f(n), g(n)))$ [Additivität]
$f(n) + g(n) = O(max(f(n), g(n)))$ [O kann beliebig groß gewählt werden]
$f(n) + g(n) = \Theta(max(f(n), g(n)))$
Daher ist $f(n) + g(n)$ in der selben Äquivalenzklasse wie $max(f(n), g(n))$ und die Aussage damit wahr.
`Musterlösung: 1/2 (f+g) <= max(f,g) <= 1 (f+g)`

$f(n) = n; g(n) = n^2$
$min(f(n), g(n)) = f(n)$ [für alle $n > 1$]
$\Theta(f(n) + g(n)) = \Theta(n + n^2)$
$n + n^2 = O(n^2) \wedge n + n^2 = \Omega(n^2) \Leftrightarrow n + n^2 = \Theta(n^2)$ [Additivität]
$f(n) = \Theta(n + n^2) \wedge n + n^2 = \Theta(n^2) \Rightarrow f(n) = \Theta(n^2)$ [Transitivität]
Widerspruch!

$f_1(n) = O(g_1(n))$, es existiert eine Konstante c_1 und n_{01}, sodass für alle $n \geq n_{01}$ gilt: $f_1(n) \leq c \cdot g_1(n)$
$f_12n) = O(g_2(n))$, es existiert eine Konstante c_2 und n_{02}, sodass für alle $n \geq n_{02}$ gilt: $f_2(n) \leq c \cdot g_2(n)$
Ziel: Zeigen, dass es ein c und n_0 gibt, sodass $f_1(n) \cdot f_2(n) \leq c \cdot g_1 \cdot g_2$
Wenn $b = max(c_1, c_2)$ und $n_0 = max(n_{01}, n_{02})$ gilt $f_1(n) \leq b \cdot g_1(n)$ und $f_2(n) \leq b \cdot g_2(n)$ für alle $n \geq n_0$
Daher gilt auch $f_1(n) \cdot f_2(n) \leq (b \cdot g_1) \cdot (b \cdot g_2) \Leftrightarrow f_1(n) \cdot f_2(n) \leq b^2 \cdot g_1 \cdot g_2$ (Begründung ist trivial). Wenn also $c = max(c_1, c_2)^2$ gewählt wird ist die Aussage wahr. q.e.d.

Aufgabe 4. Bestimmen Sie die Worst-Case und Best-Case Laufzeiten der unten angegebenen Algorithmen in Abhängigkeit von n in Θ-Notation.

(a)
```
s ← 0
for i = 0, ..., n − 1
    A[i] = 2 · A[i]
    if ⌊A[i]/3⌋ < 0 then
        s ← s + A[i]
return s
```

(b)
```
m ← 100
for i = 0, ..., n − 1
    if A[i] = m then
        return i
return −1
```

(c)
```
s ← 0
i ← 0
while i ≤ n − 1
    i ← i + 2
    for j = i, ..., n − 1
        s ← s + j
    for k = 0, ... i − 1
        s ← s + k
return s
```

(d)
```
a ← n
for i = 0 ... n − 1
    d ← n
    while d > 0
        d ← ⌊d/2⌋
        a ← ⌈d/3⌉
return a
```

(a) Best-Case: $\Theta(n)$
Worst-Case: $\Theta(n)$

(b) Best-Case: $\Theta(1)$
Worst-Case: $\Theta(n)$

(c) Best-Case: $\Theta(n^2)$
Worst-Case: $\Theta(n^2)$

(d) Best-Case: $\Theta(n \cdot \log(n))$
Worst-Case: $\Theta(n \cdot \log(n))$

Aufgabe 5. Bestimmen Sie die Laufzeiten **und** die Werte der Variablen a und b nach der Ausführung der unten angegebenen Algorithmen in Abhängigkeit von n in Θ-Notation. Verwenden Sie hierfür möglichst einfache Terme.

(a)

$a \leftarrow 1$
$b \leftarrow 1$
for $c \leftarrow 1, \ldots, \lfloor \log_4 n \rfloor$
    $a \leftarrow a + a$
    $b \leftarrow b + 1$
$i \leftarrow 4^{4b}$
while $i > 1$
    $i \leftarrow \lfloor \frac{i}{2} \rfloor$

(b)

$a \leftarrow 1$
for $i = 0, \ldots, n-1$
    $a \leftarrow a + a$
    for $j = i, \ldots, n-1$
        if $(j \mod 2) = 0$ **then**
            $a \leftarrow a + j$
    for $j = i+1, \ldots, n$
        if $(j \mod 2) = 1$ **then**
            $a \leftarrow a - j$
$p \leftarrow \sqrt{a}$
$b \leftarrow 0$
while $p > 1$
    $p \leftarrow \lfloor p/2 \rfloor$
    $b \leftarrow b + p$

(a) Laufzeit: $\Theta(\log_4(n) + \log_2(256 \cdot n^4)) = \Theta(\log(n))$
$a(n) = 2^{log_4(n)}$
$b(n) = 1 + log_4(n)$

(b) Laufzeit: $\Theta(n^2 + log_2(\sqrt{2^n})) = \Theta(n^2 + n) = \Theta(n^2)$
$a(n) = \Theta(2^n)$
$b(n) = \Theta((\sqrt{2})^n * \frac{1-(\frac{1}{2})^{\log_2(\sqrt{2}^n)+1}}{1-\frac{1}{2}} = \Theta((\sqrt{2})^n * n)$