We conjecture that the closed form of the recurrence relation is the following series:

\[ T(n) = a^{\log_b(n)}e + cn \cdot \sum_{i=0}^{\log_b(n)-1} \left( \frac{a}{b} \right)^i + d \cdot \sum_{i=0}^{\log_b(n)-1} a^i. \] (1)

We distinguish three cases:

**Case 1:** \( a \neq b, a \neq 1 \). In this case, our conjecture becomes

\[ T(n) = a^{\log_b(n)}e + cn \cdot \left( \frac{a}{b} \right)^{\log_b(n)-1} - 1 + d \cdot \frac{a^{\log_b(n)-1} - 1}{a - 1}. \] (2)

**Base step:** The conjecture is true for \( n = 1 \), because \( T(1) = a^0e + c \cdot 0 + d \cdot 0 = e \).

**Inductive Hypothesis:** We assume our conjecture to be true for \( T(n/b) \), so (we have \( \log_b(\frac{n}{b}) = \log_b(n) - 1 \))

\[ T(n/b) = a^{\log_b(n)-1}e + cn \cdot \left( \frac{a}{b} \right)^{\log_b(n)-1} - 1 + d \cdot \frac{a^{\log_b(n)-1} - 1}{a - 1}. \] (3)

**Inductive step:** We show that the conjecture holds for \( T(n) \) by using the inductive hypothesis:

\[
T(n) = aT(n/b) + cn + d
\]

\[
= a \left( a^{\log_b(n)-1}e + cn \cdot \left( \frac{a}{b} \right)^{\log_b(n)-1} - 1 + d \cdot \frac{a^{\log_b(n)-1} - 1}{a - 1} \right) + cn + d
\]

\[
= a^{\log_b(n)}e + cn \cdot \left( \frac{a}{b} \right)^{\log_b(n)-1} + cn \cdot \frac{a}{b} - 1 + d \cdot \frac{a^{\log_b(n)} - a}{a - 1} + d \cdot \frac{a - 1}{a - 1}
\]

\[
= a^{\log_b(n)}e + cn \cdot \left( \frac{a}{b} \right)^{\log_b(n)-1} + d \cdot \frac{a^{\log_b(n)} - 1}{a - 1}.
\] (4)

**Case 2:** \( a \neq b, a = 1 \). In this case, our conjecture becomes

\[ T(n) = e + cn \cdot \left( \frac{1}{b} \right)^{\log_b(n)-1} - 1 + d \log_b(n) = d \log_b(n) + cb \cdot \frac{1 - n}{1 - b} + e. \] (5)

**Base step:** The conjecture is true for \( n = 1 \), because \( T(1) = d \cdot \log_b(1) + cb \cdot \frac{1 - 1}{1 - b} + e = 0 + 0 + e \).

**Inductive Hypothesis:** We assume our conjecture to be true for \( T(n/b) \), so

\[ T(n/b) = d \log_b(n/b) + cb \cdot \frac{1 - n}{1 - b} + e. \] (6)

\[ T(n/b) = d \log_b(n/b) + cb \cdot \frac{1 - n}{1 - b} + e. \] (7)
**Inductive Hypothesis:** We show that the conjecture holds for $T(n)$ by using the inductive hypothesis:

$$T(n) = T(n/b) + cn + d$$  \hspace{1cm} (10)

**Inductive step:**

$$T(n/b) = d \log_b(n/b) + cb \cdot \frac{1 - \frac{n}{b}}{1 - b} + e + cn + d$$  \hspace{1cm} (11)

$$= d \log_b(n) + c \cdot \frac{b - n}{1 - b} + cn + e$$  \hspace{1cm} (12)

$$= d \log_b(n) + c \cdot \frac{b - n + n - bn}{1 - b} + e$$  \hspace{1cm} (13)

$$= d \log_b(n) + cb \cdot \frac{1 - n}{1 - b} + e.$$  \hspace{1cm} (14)

**Case 3:** $a = b$ $(a \neq 1)$. In this case, our conjecture becomes

$$T(n) = ne + cn \log_b(n) + d \cdot \frac{n - 1}{a - 1}.$$  \hspace{1cm} (15)

**Base step:** The conjecture is true for $n = 1$, because $T(1) = 1 \cdot e + c \cdot 1 \cdot 0 + d \cdot \frac{1 - 1}{a - 1} = e.$

**Inductive Hypothesis:** We assume our conjecture to be true for $T(n/b)$, so

$$T(n/b) = \frac{n}{b} \cdot e + c \cdot \frac{n}{b} \cdot \log_b(n/b) + d \cdot \frac{n - 1}{a - 1}.$$  \hspace{1cm} (16)

**Inductive step:** We show that the conjecture holds for $T(n)$ by using the inductive hypothesis:

$$T(n) = aT(n/b) + cn + d$$  \hspace{1cm} (17)

$$= a \left( \frac{n}{b} \cdot e + c \cdot \frac{n}{b} \cdot \log_b(n/b) + d \cdot \frac{n - 1}{a - 1} \right) + cn + d$$

$$= \frac{a}{b} \cdot ne + \frac{a}{b} \cdot c \cdot \log_b(n) - \frac{a}{b} \cdot cn + cn + d \cdot \frac{\frac{a}{b} \cdot n - a + a - 1}{a - 1}$$

$$= ne + cn \log_b(n) + d \cdot \frac{n - 1}{a - 1}.$$

**Solution 2.2 Estimating asymptotic running time.**

a) First we underestimate the running time: In the last $n/2$ iterations of the outer loop, the inner loop is executed at least $n/4$ times. Therefore we have at least $\frac{3}{2} \cdot \frac{n}{4} \in O(n^2)$ many iterations.

Now we overestimate the running time: The inner loop is never executed more than $n$ times. Therefore we have at most $O(n^2)$ many iterations in overall, and we obtain a running time of $\Theta(n^2)$.

b) First we observe that the inner loop runs exactly $\lceil \log_4(i^2) \rceil = \lceil 2 \log_4(i) \rceil$ many times. Now we can use the trick from the previous exercise. In the last $\frac{n^2}{2}$ iterations of the outer loop, we have $i \geq \frac{n^2}{2}$ and therefore $i^2 \geq \frac{n^4}{4}$. The inner loop runs at least $\lceil \log_4(\frac{n^4}{4}) \rceil = \lceil 4 \log(n) - 1 \rceil \in \Omega(\log(n))$ times, and the running time is $\Omega(n^2 \log n)$.

On the other hand, we have $i \leq n^2$, and the inner loop has at most $\lceil 2 \log_4(n^2) \rceil = \lceil 4 \log_4(n) \rceil$ many iterations. Therefore the running is bounded by $O(n^2 \log n)$, and the overall running time is $\Theta(n^2 \log n)$.

2
c) Ignoring the recursive call in step 6, the steps 1 – 8 just need linear time, i.e., the execution time is bounded by \( cn \) for an appropriate constant \( c > 0 \). The overall running time (in consideration of the recursive calls) is bounded by

\[
n + c \cdot \frac{n}{3} + c \cdot \frac{n}{3^2} + \ldots = cn \left( 1 + \frac{1}{3} + \frac{1}{3^2} + \ldots \right) \leq cn \left( 1 + \frac{1}{2} + \frac{1}{2^2} + \ldots \right) \leq 2cn, \tag{17}
\]

and is therefore lying in \( \Theta(n) \).

**Solution 2.3** Various topics.

a) \( x = 1572, y = 8687 \) or \( x = 8786, y = 7215 \)

b) Sequence: 5, 4, 3, 2, 1

c) Depending on the implementation of insertion sort, there exist two possible solutions: (15, 8), (7, 15), (7, 8), (12, 15) or (15, 8), (7, 8), (12, 7), (12, 8)

**Solution 2.4** Lower bounds / Algorithm design.

a) Let the given coins be \( M_1, \ldots, M_9 \). The following decision tree describes a strategy that determines the false coin with exactly two weighings. Each node describes which coins lie in the left and in the right scalepan. After each weighing there are three possible outcomes: the coins on the left are either lighter (\(<\)), have the same weight (\(=\)) or are heavier (\(>\)) than the coins on the right. After each step the false coin is on the side with highest weight.

![Decision Tree](attachment:decision_tree.png)

b) Let the given coins be \( M_1, \ldots, M_n \). For \( n = 1 \) we found the false coin and are finished. Otherwise, we split the coins into three groups \( G_1 := \{ M_1, \ldots, M_{n/3} \} \), \( G_2 := \{ M_{n/3+1}, \ldots, M_{2n/3} \} \) and \( G_3 := \{ M_{2n/3+1}, \ldots, M_n \} \), and we put \( G_1 \) in the left pan and \( G_2 \) in the right pan. If the coins on the left are heavier, then the false coin is in \( G_1 \) and we continue with this group. Similarly we continue with \( G_2 \) if the right side is heavier, or with \( G_3 \), if both sides are equally heavy.

It remains to show that in this way we really have only \( \log_3(n) \) weighings. We show by induction over \( m \in \mathbb{N} \) that the following statement is correct: For \( n = 3^m \), the above procedure performs exactly \( m \) weighings to find the false coin.

**Base step** \((m = 1)\): For \( m = 1 \) we have \( n = 3 \), and exactly one weighing is required to find the false coin.

**Inductive hypothesis:** Let the statement be true for \( m \), i.e. for \( n = 3^m \), the above procedure needs exactly \( m \) weighings to find the false coin.
**Inductive step** \((m \to m + 1)\): Let \(n = 3^{m+1}\). We divide the coins into three groups and continue with the heaviest group. This group contains exactly

\[
n/3 = 3^{m+1}/3 = 3^m
\]

many coins and by the inductive hypothesis, \(m\) weighings are sufficient to find the false coin. Thus the method requires a total of \(m + 1\) weighings.

We now observe that for \(m = \log_3(n)\), we have \(n = 3^m\) and therefore \(\log_3(n)\) weighings are sufficient.

c) Let the given coins be \(M_1, ..., M_n\). Every possible algorithm can be described by a decision tree like the one on the previous page. The maximum depth \(T\) of this tree is the maximum number of weighings in the worst case. Therefore we must show that the depth \(T\) of every valid decision tree is at least \(\log_3(n) - 1\).

First, we observe that each weighing possesses only three possible outcomes. Therefore any valid decision tree is a ternary tree, i.e. each node has at most three children. We use induction to show that there exist at most \(3^k\) nodes at depth \(k\).

**Base step** \((k = 0)\): At depth 0 there exists a single node, namely, the root of the tree.

**Inductive hypothesis**: We assume that the assertion is true for \(k\), i.e., there exist at most \(3^k\) nodes at depth \(k\).

**Inductive step** \((k \to k + 1)\): Consider the nodes at depth \(k + 1\). Such a node can be reached by an edge only from a node at depth \(k\). Each node has at most three children and by the inductive hypothesis, there are at most \(3^k\) nodes at depth \(k\). So there exist at most \(3 \cdot 3^k = 3^{k+1}\) nodes at depth \(k + 1\).

In some nodes of the tree the algorithm outputs a result, since the available information is sufficient to identify the false coin. Since each result \(M_1, ..., M_n\) is possible, there are \(n\) possible situations that the algorithm has to distinguish. For a correctly working algorithm we have

\[
n \leq \text{Number of nodes} \leq \sum_{k=0}^{T} 3^k = \frac{3^{T+1} - 1}{2} < 3^{T+1},
\]

and thus we get

\[
\log_3(n) < T + 1 \iff T > \log_3(n) - 1.
\]

Thus, the depth of each decision tree is at least \(\log_3(n) - 1\), and therefore every correct algorithm must perform at least these many weighings in the worst case.