There is a definition of the $O$ notation that is different from the one given at the lecture. Namely, for a function $g : \mathbb{N} \to \mathbb{R}^+$, let
\begin{equation}
O(g) := \{ f : \mathbb{N} \to \mathbb{R}^+ \mid \exists c \in \mathbb{R}^+, n_0 \in \mathbb{N} \forall n \geq n_0 : f(n) \leq cg(n) \}.
\end{equation}

Analogously, we say that a function $f$ grows asymptotically at least as much as $g$, if $f \in \Omega(g)$ with
\begin{equation}
\Omega(g) := \{ f : \mathbb{N} \to \mathbb{R}^+ \mid \exists c \in \mathbb{R}^+, n_0 \in \mathbb{N} \forall n \geq n_0 : f(n) \geq cg(n) \}.
\end{equation}

A function $f$ grows asymptotically like $g$ when $f \in O(g)$ and $f \in \Omega(g)$. We denote this by $f = \Theta(g)$.

For these exercises, you can choose to use the definition given at the lecture, or use the above definition.

**Exercise 1.1** The Set $\Theta(g)$.

Give a definition of the set $\Theta(g)$ as compactly as possible (i.e., with the fewest possible parameters and quantifiers), analogously to the above definitions for the sets $O(g)$ and $\Omega(g)$.

**Exercise 1.2** Proofs about $O$ Notation.

Prove or disprove the following statements, where $f, g : \mathbb{N} \to \mathbb{R}^+$.

- a) $f \in O(g)$ if and only if $g \in \Omega(f)$.
- b) If $f \in O(g)$, then $f(n) \leq g(n)$ for every $n \in \mathbb{N}$.
- c) If $f(n) \leq g(n)$ for every $n \in \mathbb{N}$, then $f \in O(g)$.
- d) There exist different functions $f$ and $g$ such that $f \in \Omega(g)$ and $g \in \Omega(f)$.
- e) $\log_a(n) \in \Theta(\log_b(n))$ for all constants $a, b \in \mathbb{N} \setminus \{1\}$.
- f) Let $f_1, f_2 \in O(g)$ and $f(n) := f_1(n) + f_2(n)$. Then, $f \in O(g)$.
- g) Let $f_1, f_2 \in O(g)$ and $f(n) := f_1(n) \cdot f_2(n)$. Then, $f \in O(g)$.
- h) $n^d \in O(b^n)$ for fixed values $d > 0$ and $b > 1$.

**Exercise 1.3** Asymptotic Growth of Functions.

Sort the following functions from left to right such that: if function $f$ is on the left of $g$, then $f \in O(g)$.

Example: the functions $n^3, n^7, n^9$ are already in the right order since $n^3 \in O(n^7)$ and $n^7 \in O(n^9)$.

$$
\log(n^{11}), \sqrt{2n}, n!, n^n, 15^7, \sqrt{n}, \frac{2^n}{n^2}, \log(n!), \left(\frac{n}{2}\right), \frac{1}{n}, \log^3(n)
$$

Please turn over.
Exercise 1.4  Programming Exercise.

In this exercise, we want to evaluate a recurrence relation of the form

\[ R_n = \begin{cases} 
  A & \text{if } n = 0 \\
  B & \text{if } n = 1 \\
  C \cdot R_{n-1} + D \cdot R_{n-2} & \text{otherwise}
\end{cases} \]

i.e. we want to compute \( R_i \) for a given \( i \in \mathbb{N} \). For example, if \( A = 0, \ B = 1, \ C = 1 \) and \( D = 1 \), then \( R_n \) produces the well-known Fibonacci numbers 0, 1, 1, 2, 3, 5, 8, 13, \ldots

**Input**  The first line of the input contains only the number \( t \) of test instances. After that, we have exactly one line for each test instance containing the numbers \( i, A, B, C, D \) (in exactly this order, separated by spaces). While \( 0 \leq i \leq 50 \) is a natural number, \( A \) and \( B \) are integers from the interval \([−10^3, 10^3]\), and \( C \) and \( D \) are either 1 or \(-1\).

**Output**  For every test instance, we want to output a single line containing only the value \( R_i \).

**Example**

**Input:**

```
2
20 0 1 1 1
22 5 10 1 -1
```

**Output:**

```
6765
-10
```

**Notes**

1) The values \( R_i \) can be very large. You should use the data type `long` instead of `int`.

2) To read an input from the console you can import the class `java.util.Scanner` and use the following code fragment:

```java
Scanner in = new Scanner(System.in);
int value1 = in.nextInt();
int value2 = in.nextInt();
```

**Hand-in:** until Wednesday, 25th February 2015.