1. For each given recurrence relation, find the first five terms, derive the closed-form solution, and check it by substituting it back to the recurrence relation.
   (a) $a_n = a_{n-1} + n$ with $a_0 = 2$
   (b) $a_n = 2a_{n-1} + 2$ with $a_0 = 1$
   (c) $a_n = 3a_{n-1} + 2^n$ with $a_0 = 5$
   (d) $a_n = 4a_{n-1} + 5a_{n-2}$ with $a_0 = 1$, $a_1 = 17$
   (e) $a_n = 4a_{n-1} - 4a_{n-2}$ with $a_0 = 3$, $a_1 = 11$
   (f) $a_n = 2a_{n-1} + a_{n-2} - 2a_{n-3}$ with $a_{0,1,2} = 3, 2, 6$

2. Solve the following recurrences by applying the Master Theorem. For the cases where the Master theorem does not apply, use the Akra–Bazzi method. In cases where neither of these two theorems apply, explain why and solve the recurrence relation by closely examining the recursion tree. Solutions must be in the form $T(n) \in \Theta(\dots)$.
   (a) $T(n) = 2T(n/2) + n$
   (b) $T(n) = T(3n/4) + T(n/4) + n$
   (c) $T(n) = 3T(n/2) + n$
   (d) $T(n) = 2T(n/2) + n/\log n$
   (e) $T(n) = 6T(n/3) + n^2 \log n$
   (f) $T(n) = T(3n/4) + n \log n$
   (g) $T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + n$
   (h) $T(n) = T(n/2) + T(n/4) + 1$
   (i) $T(n) = T(n/2) + T(n/3) + T(n/6) + n$
   (j) $T(n) = 2T(n/3) + 2T(2n/3) + n$
   (k) $T(n) = \sqrt{2n}T(\sqrt{2n}) + \sqrt{n}$
   (l) $T(n) = \sqrt{2n}T(\sqrt{2n}) + n$

3. Consider a recurrence relation $a_n = 2a_{n-1} + 2a_{n-2}$ with $a_0 = a_1 = 2$. Solve it (*i.e.* find a closed formula) and show how it can be used to estimate the value of $\sqrt{3}$ (hint: observe $\lim_{n\to\infty} a_n/a_{n-1}$). After that, devise an algorithm for constructing a recurrence relation with integer coefficients that can be used to estimate the square root $\sqrt{k}$ of a given integer k. Implement your algorithm using any high-level language and benchmark[1] it against the standard `sqrt` function.

[1] You can, for example, use `hyperfine` for benchmarking your binaries. Make sure to setup a fair environment for comparison, *i.e.* minimize the possible delays of shell, interpreter, VM, *etc.*