Let \( G \) be a (two-player deterministic) mean-payoff game. The weights are integers and label edges. We let \( W \) denote the largest weight in absolute value. The mean-payoff objective is that the infimum limit of the weights seen along a path is non-negative (\( \geq 0 \)).

**Question 1:** Prove that if Eve has a winning strategy, then she has a strategy that ensures that at all times, the total sum remains always larger than or equal to \( -nW \). Is the value \( nW \) optimal?

**Question 2:** We construct a deterministic automaton \( A \). The alphabet is the set of weights of \( G \). The set of states is \([-nW, nW]\) plus an extra sink rejecting state \( \perp \) and a sink accepting state \( \top \). The automaton starts from the state \( 0 \) and stores the total sum of the weights, restricted to \([-nW, nW]\). The automaton rejects (goes to \( \perp \)) if the total sum goes below \(-nW\). If the total sum goes above \( nW \), it stays in \( \top \). Important! **All** states except for \( \perp \) are accepting.

Construct a safety game over the synchronised product of \( G \) and \( A \) such that Eve has a winning strategy in the mean-payoff game \( G \) if and only if Eve has a winning strategy in the safety game \( G \times A \).

**Question 3:** Construct an algorithm for solving mean-payoff games based on the construction above. Analyse its (time) complexity.

**Question 4:** Construct an algorithm for solving CoBüchi games based on the construction above. Analyse its (time) complexity.

**Question 5:** Let \( k \) be the number of different weights in the mean-payoff game \( G \). In the similar way as above, construct an algorithm for solving mean-payoff games whose complexity is \( O(mn^k) \).

**Question 6:** In the constructions above, can we avoid constructing explicitly the automaton \( A \) to have a better space complexity for solving mean-payoff games?