Solution to Problem 17

Problem. Use the impossibility of zero-checker (that we proved in class) to prove that no algorithm is possible that, given a program $p$ that always halts, checks whether this program always computes $8n^2 + 5$.

Solution. We will prove that if such a checker exists, then we can construct a zero-checker – and we already know that zero-checkers are not possible. Indeed, let us assume that we have an algorithm $\text{checker}(p)$ that, given a program $p$ that always halts, checked whether $\forall n \ (p(n) = 8n^2 + 5)$. Suppose that we have a program $q$ that always halts and we want to check whether this program $q$ always returns 0. To check this, we form the following auxiliary program that always returns $q(n) + 8n^2 + 5$:

\[
\text{public static int aux(int n)}
\{ \text{return } q(n) + 8 \ast n \ast n + 5; \}
\]

The value $q(n) + 8n^2 + 5$ is always equal to $8n^2 + 5$ if and only if the value $q(n)$ is always equal to 0.

Thus, the algorithm $\text{checker}(q(n) + 8n^2 + 5)$ that applies $\text{checker}$ to the above auxiliary program is a zero-checker. However, we have proven that zero-checkers do not exist. This contradiction shows that our assumption – that the desired checkers are possible – leads to a contradiction. Thus, such checkers are not possible. The theorem is proven.