Problems Solved:

31	32	33	34	35

Name:

Matrikel-Nr.:

Problem 31. Let $M = (Q, \Gamma, \sqcup, \Sigma, \delta, q_1, F)$ be a Turing maching with $\Sigma = \{0,1\}$, $\Gamma = \{0,1,\sqcup\}$, $Q = \{q_1, \ldots, q_n\}$, $F = \{q_2\}$, and transition function δ. We denote the symbols $0, 1, \sqcup$ in this order by X_1, X_2, X_3 and the head movement direction L, R by D_1, D_2. An operation $\delta(q_i, X_j) = (q_k, X_l, D_m)$ shall be coded as $0^i10^j10^k10^l10^m$. The Turing maschine M itself shall be coded as

$$111code_1 11code_2 \ldots 11code_r 111$$

where each $code_1$ up to $code_r$ encode the operations given by δ. We denote such a code of a Turing machine M by $[M]$. Note that this encoding is different from the code $\langle M \rangle$ that is given in the lecture notes.

1. Let $w \in \Sigma^*$. Is it decidable whether w is the code of a Turing machine?
2. Is there a Turing machine U_ε that takes an arbitrary word $w \in \Sigma^*$ as input, checks whether this word is the code of a Turing machine and, if yes, simulates the behaviour of the respective Turing machine when started on the empty word?
3. Is it decidable whether there is a Turing machine U_ε as described in the previous question?
4. If the answer to the second question is "yes", is it decidable whether such a Turing machine U_ε halts on every word $w \in \Sigma^*$ that is the code of a Turing machine.
5. Optional: If the answer to the second question is "yes", is it decidable whether such a Turing machine U_ε halts on every word $w \in \Sigma^*$ that is **not** the code of a Turing machine.

Give reasonable arguments for your answers. Informal reasoning is enough, but simply stating "yes" or "no" does not count as a solution of this exercise.

Problem 32. Describe (informally) a Turing machine H that generates the the following language

$$L_h = \{\langle M \rangle w \mid M \text{ is a Turing machine and } M \text{ halts on } w\},$$

i.e., $L_h = Gen(H)$.
You do not have to give an explicit definition of such a machine, but you must clearly describe how such a machine can in principle work, i.e., use higher level constructs to describe the "algorithm" that such a machine represents.

Problem 33. Show that the Acceptance Problem is reducible to the restricted Halting problem.

Problem 34. Show that there is no Turing machine P that for an arbitrary Turing machine M decides whether for some $n > 0$ it holds $0^n \in L(M) \subseteq \{0,1\}^*$.

Problem 35. Let $M_0, M_1, M_2, \dots$ be a list of all Turing machines with alphabet $\Sigma = \{0, 1\}$ such that the function $i \mapsto \langle M_i \rangle$ is computable. Let $w_i := 10^i 10^i 1$ for all natural numbers i. Let $A := \{w_i \mid i \in \mathbb{N} \wedge w_i \in L(M_i)\}$ and $\overline{A} = \Sigma^* \setminus A$.

(a) Is $\overline{A}$ recursively enumerable? (Justify your answer.)

(b) Suppose there is an oracle X_{Delphi} that decides the Halting problem, i. e., you can give to X_{Delphi} the code $\langle M \rangle$ of a a turing machine M and a word w and X_{Delphi} returns 1 (YES) or 0 (NO) depending on whether or not M halts on w.

Show that one can construct an Oracle-Turing machine T (which is allowed by a special extension to give some word $\langle M \rangle$ (a Turing machine code) and a word w to X_{Delphi} and gets back 1 or 0 depending on whether or not M halts on w) such that $L(T) = \overline{A}$.

(c) Does it follow from (a) and (b) that X_{Delphi} is not a Turing machine? Justify your answer. Note that you are not allowed to use the fact that the Halting problem is undecidable, but you must give a proof that only follows from (a) and (b).