C240 Computability and Complexity   Tutorial 6

Questions 1,2,5 use the machine M[w], EDIT, and the Universal Turing Machine, U; all are assumed to be standard; see sections 4.3 and 5.3 of the notes.

1. Let R be a standard Turing machine that reverses its input. So, for example, $f_R(abc) = cba$. Evaluate:
   a) $f_R(alucard)$ b) $f_{R[Lee]}(Cushing)$;   c) $f_{R[marb]}(\square)$

2. Let S be any standard Turing machine, and let v, w be any words of C.
   Check that : $f_U(f_{EDIT}(code(S)*v)*w) = f_S(v)$.

3. Uniform Halting problem (the problem of whether a Turing machine halts on every input). Show by reduction that there is no Turing machine UH such that for all standard Turing machines S,

$$f_{UH}(code(S)) = \begin{cases} y & \text{if S halts and succeeds on every input} \\ n & \text{otherwise} \end{cases}$$

[Adapt the argument for the empty input halting problem].

4. Inverse problem. Show by reduction that there is no Turing machine INV such that for all standard Turing machines S_1, S_2 and all words w of C:

$$f_{INV}(code(S_1)*code(S_2)*w) = \begin{cases} da & \text{if } f_{S_2}(f_{S_1}(w)) = w \\ nyet & \text{otherwise} \end{cases}$$

That is, on input w, S_2 inverts the effect of S_1. Note that $f_{S_2}(f_{S_1}(w))$ is undefined if $f_{S_1}(w)$ is undefined, so $f_{INV}(code(S_1)*code(S_2)*w) = nyet$ in this case.
[you might try to transform an instance code(S)*w of HP into code(S)*code(W)*w for some suitable Turing machine W.]

5. [‘Recursion Theorem’, Stephen Kleene, 1938]. The Turing machine ADIT is similar to EDIT (see 5.3), but, on input code(M)*w, it outputs code(M{w}). M{w} is a standard TM similar to M[w], but on input x, it writes w* in front of x (shifting x right to make room), then runs M. So $f_{M\{w\}}(x) = f_M(w*x)$. Let S be any standard TM and consider the TM G, below:

G is assumed standard by scratch character elimination. On input code(M)*w, G splits off code(M), duplicates it, and runs ADIT then S; G then tacks on **w** to the output and runs U. So $f_G(code(M)w) = f_U(f_S(f_{ADIT}(f_{Duplicator}(code(M))))*w)$.
a). Let K=G{code(G)}Check that for any word w of C, we have $f_K(w) = f_U(f_S(code(K))*w)$.
b) Deduce that, given any algorithm to modify TMs (ie their codes) in any way whatever, there is always some TM whose input/output function is unchanged by the modification.