COMP 3803 — Fall 2025 — Problem Set 6

Question 1: We have seen in class that the language

$$Halt = \{\langle M, w\rangle : M \text{ is a Turing machine that terminates on the input string } w\}$$

is undecidable. In the language *PrintB* that is defined below, Σ denotes the input alphabet of the Turing machine M, and Γ denotes its tape alphabet.

$$PrintB = \{\langle M, w, b\rangle : M \text{ is a Turing machine, } w \in \Sigma^*, b \in \Gamma, \text{ when running } M \text{ on input } w, M \text{ writes } b \text{ on the tape at least once}\}.$$

Prove that *PrintB* is undecidable.
Hint: Given an input $\langle M, w\rangle$ for *Halt*, modify M such that the resulting Turing machine prints a new symbol, say #, at the moment it terminates.

Question 2: Let A be an arbitrary language that is enumerable, but not decidable. Recall what it means to enumerable: There exists a Turing machine M, such that for any input string w:

- If $w \in A$, then, on input w, M terminates in the accept state.
- If $w \notin A$, then, on input w, M either terminates in the reject state or does not terminate.

Consider the following function $f : \{0,1\}^* \to \mathbb{N}$:

$$f(w) = \begin{cases} \text{the number of steps made by } M \text{ on input } w, \text{ if } M \text{ terminates on } w, \\ 0, \text{ if } M \text{ does not terminate on } w. \end{cases}$$

In this question, you will prove that the function f is not computable, i.e., there does not exist an algorithm that, for any input string $w \in \{0,1\}^*$, terminates and returns the value of $f(w)$.
(2.1) Let $g : \{0,1\}^* \to \mathbb{N}$ be an arbitrary computable function. Prove that there exists a string w in $\{0,1\}^*$ such that $f(w) > g(w)$.
Hint: As you can expect, the proof is by contradiction. Thus, you assume that the claim is not true. Define a new Turing machine N that, for any input string w in $\{0,1\}^*$, runs the Turing machine M for $g(w)$ steps and then "does something".
(2.2) Prove that the function f is not computable.

Question 3: We have seen in class that the language

$$Halt = \{\langle M, w\rangle : M \text{ is a Turing machine that terminates on the input string } w\}$$

is undecidable. Consider the language

$$Halt_\varepsilon = \{\langle M\rangle : M \text{ is a Turing machine that terminates on the input string } \varepsilon\}.$$

Professor Justin Bieber claims that the following reasoning proves that $Halt_\varepsilon$ is undecidable:

- We know that *Halt* is undecidable.
- Since $Halt_\varepsilon$ is a subproblem of *Halt*, $Halt_\varepsilon$ is also undecidable.

Is Professor Bieber's reasoning correct?

Question 4: Consider again the languages *Halt* and $Halt_\varepsilon$ from the previous question.

Prove that $Halt_\varepsilon$ is undecidable.

Hint: You are not allowed to say "Oh this follows directly from Rice's Theorem". Instead, you must give a complete proof.

Question 5: In class, we have seen that the language

$$Halt = \{\langle P, w \rangle : \; P \text{ is a Java program that terminates on the binary input string } w\}$$

is undecidable.

A Java program P is called a *Hello-World-program*, if the following is true: When given the empty string ϵ as input, P can do whatever it wants, as long as it outputs the string `Hello World` and terminates. (We do not care what P does when the input string is non-empty.)

Consider the language

$$HW = \{\langle P \rangle : \; P \text{ is a Hello-World-program}\}.$$

The questions below will lead you through a proof of the claim that the language HW is undecidable.

(5.1) Consider a fixed Java program P and a fixed binary string w.

We write a new Java program J_{Pw} which takes as input an arbitrary binary string x. On such an input x, the Java program J_{Pw} does the following:

> **Algorithm** $J_{Pw}(x)$:
> run P on the input w;
> print `Hello World`

- Argue that P terminates on input w if and only if $\langle J_{Pw} \rangle \in HW$.

(5.2) The goal is to prove that the language HW is undecidable. We will prove this by contradiction. Thus, we assume that H is a Java program that decides HW. Recall what this means:

- If P is a Hello-World-program, then H, when given $\langle P \rangle$ as input, will terminate in the accept state.
- If P is not a Hello-World-program, then H, when given $\langle P \rangle$ as input, will terminate in the reject state.

We write a new Java program H' which takes as input the binary encoding $\langle P, w \rangle$ of an arbitrary Java program P and an arbitrary binary string w. On such an input $\langle P, w \rangle$, the Java program H' does the following:

> **Algorithm** $H'(\langle P, w \rangle)$:
> construct the Java program J_{Pw} described above;
> run H on the input $\langle J_{Pw} \rangle$;
> **if** H terminates in the accept state
> **then** terminate in the accept state
> **else** terminate in the reject state
> **endif**

Argue that the following are true:

- For any input $\langle P, w \rangle$, H' terminates.
- If P terminates on input w, then H' (when given $\langle P, w \rangle$ as input), terminates in the accept state.
- If P does not terminate on input w, then H' (when given $\langle P, w \rangle$ as input), terminates in the reject state.

(5.3) Now finish the proof by arguing that the language HW is undecidable.

Question 6: Consider the two languages

$$Empty = \{\langle M \rangle : \ M \text{ is a Turing machine for which } L(M) = \emptyset\}$$

and

$$UselessState = \{\langle M, q \rangle : \ M \text{ is a Turing machine, } q \text{ is a state of } M, \text{ for every input string } w \text{, the computation of } M \text{ on input } w \text{ never visits state } q\}.$$

(6.1) Use Rice's Theorem to show that *Empty* is undecidable.

(6.2) Use **(6.1)** to show that *UselessState* is undecidable.