This homework is not for submission. However, we are planning to ask a few (true/false, multiple-choice, or short-answer) questions about undecidability on the final exam, so we still strongly recommend treating these questions as regular homework. Solutions will be released next Monday.

1. Let $\langle M \rangle$ denote the encoding of a Turing machine $M$ (or if you prefer, the Python source code for the executable code $M$). Recall that $w^R$ denotes the reversal of string $w$. Prove that the following language is undecidable.

   $$\text{SelfRevAccept} := \{ \langle M \rangle \mid M \text{ accepts the string } \langle M \rangle^R \}$$

   Note that Rice's theorem does not apply to this language.

2. Let $M$ be a Turing machine, let $w$ be a string, and let $s$ be an integer. We say that $M$ accepts $w$ in space $s$ if, given $w$ as input, $M$ accesses at most the first $s$ cells on its tape and eventually accepts. (If you prefer to think in terms of programs instead of Turing machines, “space” is how much memory your program needs to run correctly.)

   Prove that the following language is undecidable:

   $$\text{SomeSquareSpace} = \{ \langle M \rangle \mid M \text{ accepts at least one string } w \text{ in space } |w|^2 \}$$

   Note that Rice’s theorem does not apply to this language.

   [Hint: The only thing you actually need to know about Turing machines for this problem is that they consume a resource called “space”.

3. Prove that the following language is undecidable:

   $$\text{Picky} = \{ \langle M \rangle \mid M \text{ accepts at least one input string and } M \text{ rejects at least one input string} \}$$

   Note that Rice’s theorem does not apply to this language.
Solved Problem

4. Consider the language $\text{SometimesHalt} = \{ \langle M \rangle | M \text{ halts on at least one input string} \}$. Note that $\langle M \rangle \in \text{SometimesHalt}$ does not imply that $M$ accepts any strings; it is enough that $M$ halts on (and possibly rejects) some string.

(a) Prove that $\text{SometimesHalt}$ is undecidable.

Solution (Rice): Let $\mathcal{L}$ be the family of all non-empty languages. Let $N$ be any Turing machine that never halts, so $\text{HALT}(N) = \emptyset \not\in \mathcal{L}$. Let $Y$ be any Turing machine that always halts, so $\text{HALT}(Y) = \Sigma^* \in \mathcal{L}$. Rice’s Halting Theorem immediately implies that $\text{SometimesHalt} = \text{HALTIN}(\mathcal{L})$ is undecidable.

Solution (closure): Let $\text{Encodings}$ be the language of all Turing machine encodings (for some fixed universal Turing machine); this language is decidable. We immediately have $\text{Encodings} = \text{NeverHalt} \cup \text{SometimesHalt}$, or equivalently, $\text{NeverHalt} = \text{Encodings} \setminus \text{SometimesHalt}$.

The lectures notes include a proof that $\text{NeverHalt}$ is undecidable. On the other hand, the existence of a universal Turing machine implies that $\text{Encodings}$ is decidable. So Corollary 3(d) in the undecidability notes implies that $\text{SometimesHalt}$ is undecidable.

Solution (reduction from $\text{HALT}$): We can reduce the standard halting problem to $\text{SometimesHalt}$ as follows:

$\text{DecideHALT}(\langle M, w \rangle)$:

Encode the following Turing machine $M'$:

$M'(x)$:

(0) run $M$ on input $w$

return $\text{DecideSometimesHalt}(\langle M' \rangle)$

We prove this reduction correct as follows:

$\Rightarrow$ Suppose $M$ halts on input $w$.
Then $M'$ halts on every input string $x$.
So $\text{DecideSometimesHalt}$ must accept the encoding $\langle M' \rangle$.
We conclude that $\text{DecideHALT}$ correctly accepts the encoding $\langle M, w \rangle$.

$\Leftarrow$ Suppose $M$ does not halt on input $w$.
Then $M'$ diverges on every input string $x$.
So $\text{DecideSometimesHalt}$ must reject the encoding $\langle M' \rangle$.
We conclude that $\text{DecideHALT}$ correctly rejects the encoding $\langle M, w \rangle$. ■