**Problem 1 (Enter the Machine)**

Many problems concerning properties of models of computations turn out to be decidable. We can break this up into two sorts:

1. *Language properties* that concern the strings that a machine accepts.
2. *Behavioral properties* that concern how a machine operates.

In this problem, we will discuss properties of the second sort. Because these properties are not tied to the acceptance/rejection behavior of the machine, undecidablity proofs of these properties all follow a strict formula.

Let's consider such a problem from the textbook:

> (Sipser 5.13) A useless state in a Turing machine is one that is never entered on any input string. Consider the problem of determining whether a Turing machine has any useless states. Formulate this problem as a language and show it is undecidable.

**Exercise (Languages)** Do what Sipser says! First formulate this problem as a language USELESS. Importantly, identify what an assumed decider $D$ for USELESS takes as input.

Now let's show that USELESS is undecidable by a mapping reduction from $A_{\mathsf{TM}}$. To do so, we'll follow the strategy suggested in the handout that accompanied today's reading. First let's establish what our mapping function should look like and how it should behave.

**Exercise (Mappings)** The heart of the mapping reduction is a mapping from inputs to $A_{\mathsf{TM}}$ to USELESS. What does the mapping function $f$ take as input and produces as output?

Once we know the types of the mapping reduction, we now need to reason about the correctness condition linking acceptance between $A_{\mathsf{TM}}$ and USELESS.

**Exercise (The Condition)** Let $\langle M, w\rangle$ be inputs to a decider for $A_{\mathsf{TM}}$ and $\langle M'\rangle$ the input to a decider for USELESS. Give the correctness condition for $f$ based on these inputs.

We have now established our correctness condition for our construction. Based on this condition, we need to figure out a way to have our machine $M'$ *conditionally* have the property of USELESS: $M'$ has a useless state. Recall that a state is useless if no input string causes $M'$ to enter that state. Because the transitions of a Turing machine are dependent on the positions of the tape head and contents of the tape at that position, we need to know how a Turing machine executes in order to determine if it contains a useless state. This is ultimately why $A_{\mathsf{TM}}$ can be reduced to USELESS.

For the purposes of our construction, however, we want to conditionally make $M'$ have a useless state in such a way that it is painfully obvious that this is the case. This will simplify our reasoning when we go to make sure that $M'$ obeys the correct condition outlined above. Two ways that we might go about this are:

1. Give $M'$ a state $u$ that has no physical transition to it from any other state in $M'$. It is obvious that $u$ is useless because there is no way to reach it in this situation.

2. Give $M'$ a state $u$ that is connected from some other state in the Turing machine, but make that transition involve reading a tape symbol that provably never appears on the tape, *e.g.*, by ensuring that it is a alphabet symbol and it is not mentioned in the read position of any transition of the Turing Machine.

The first option does not not work for our purposes because the transitions are fixed when $M'$ is constructed; we can't condition the existence of a transition based on whether $M$ accepts $w$. However, the second option is controllable at runtime—we can condition the appearance of the symbol that will allow us to transition to our useless state on whether $M$ accepts $w$.

Putting all of this together, we have the following skeleton for the construction of our mapping function and the Turing machine $M'$ it produces:

$M' =$ "On input $x$:
1. Run $M$ on $w$.
    - $M$ accepts: *($M'$ should have a useless state)*
    - $M$ rejects: *($M'$ should not have a useless state)*"

Note that also in the case where $M$ loops on $w$, we want to have the same behavior as the rejection case.

**Exercise (The Construction)** Complete the construction of $M'$ as specified above.

Finally, we need to verify that $M'$ does the right thing. Rather than proving our correctness condition instead (which is a biconditional), I find it easier to reason about the three cases that might occur as a result of $M$'s execution of $w$ (convince yourself these three cases cover the two cases of the biconditional you derived above):

- $M$ accept $w$.
- $M$ rejects $w$.
- $M$ loops on $w$.

**Exercise (The Proof of Correctness)** Prove that $M'$ has the appropriate behavior for the three cases outlined above with respect to your correctness condition for the mapping function.

**Problem 2 (Rice's Theorem)**

In this problem, we'll walk through and complete a proof of Rice's Theorem which was introduced in the reading.

Let $P$ be a decidable language whose strings are turing machine descriptions that satisfy an arbitrary predicate $f$, *i.e.*,

$$P = \{\langle M \rangle \mid f(M)\}$$

where $f$ is a proposition ranging over Turing machine descriptions For example, the following are propositions over a Turing machine, call it $M$:

- $L(M)$ is finite.
- $M$ writes a '$' on its tape for some input.

Rice's Theorem demands two properties are true about $P$.

1. $P$ is non-trivial: $|P| > 0$ and $P \neq \Sigma^*$.
2. $P$ is a language property: if $L(M_1) = L(M_2)$ then $\langle M_1 \rangle \in P \Leftrightarrow \langle M_2 \rangle \in P$.

Let's first check that we understand what we mean by language property.

**Exercise (Language Property Example)** Let:

$$P = \{\langle M \rangle \mid M \text{ writes a '\$' on its tape for some input.}\}.$$

Spend 2–3 minutes individually and write down why $P$ does not fulfill the conditions of a language property. Come back as a group and come up with a final explanation why $P$ is not a language property.

Now let's proceed with the core of the proof of Rice's theorem. First we'll complete the core reduction under a set of assumptions about our property $P$. We'll then discuss after the fact why these assumptions are sound given the constraints of Rice's theorem. Assume the following:

- There exists a TM $T$ such that $\langle T \rangle \in P$.
- Define the TM that rejects all strings be $T_\emptyset$. Assume that $\langle T_\emptyset \rangle \notin P$.

**Exercise (Core Reduction)** As a group, use $T$ to construct a mapping reduction from $A_{\mathsf{TM}}$ to $P$. Keep in mind the following insights while constructing your mapping reduction:

- Your mapping construction, when given an input to $A_{\mathsf{TM}}$ of the form $\langle M, w \rangle$ should produce a TM $\langle M' \rangle$ with the property that $M$ accepts $w$ iff $\langle M' \rangle \in P$.

- Note that we have asserted that (a) a machine $T$ exists that is in $P$ and (b) the machine that rejects all strings $T_\emptyset$ is not in $P$. Use these as the targets for your mapping function as described in "Undecidable Proof Strategies" handout.

Once you have a candidate reduction, check your answer with the course staff.

With the reduction completed, we need to show where the two constraints came from. Let's briefly talk about the first constraint.

**Exercise (The First Constraint)** Individually, spend a few minutes and explain why the constraints of Rice's Theorem on $P$ imply that there exists a TM $T$ such that $\langle T \rangle \in P$. Come back as a group and agree on a final explanation.

The second constraint is really an assumption "without loss of generality". In other words, even if $T_\emptyset \in P$, we can still complete the proof with a little bit inconsequential fix-up to our proof. Note that we can't simply work with $T_\emptyset$ directly in this case because our mapping reduction will fail in a style similar to $E_{TM}$. We instead need to do something else.

**Exercise (The Final Fix-Up)** It turns out that the fix-up is to consider instead the complement of $P$:

$$\overline{P} = \{ \langle M \rangle \mid f(M) \text{ is not provable} \}.$$

The reduction above then holds, but shows that $A_{\mathsf{TM}} \leq_m \overline{P}$ rather than $P$. As a group, determine how a decider for $\overline{P}$ could be used to create a decider for $P$, allowing us to conclude that $A_{\mathsf{TM}} \leq \overline{P}$ as well. Check your reasoning for this and the previous part with the course staff when you are done.