Problem 1
Design a Turing Machine that can decide the following language (i.e. accept strings that are in the language and reject strings that are not)

\[ L = \{ 0^n1^n \mid n > 0 \} \]

Solution:

Proof Idea: We will construct a decider for the language. The decider will cross off corresponding pairs of 0s and 1s in the proper order, and reject if there's anything left on the tape after

Claim: \( L \) is decidable

Proof by construction.

Construct a decider for \( L \), called \( D_L \) as follows:

\( D_L \) on input \( x \):

i. Check that the input begins with a 0. If not, REJECT

ii. Search left for a 0

• If none found, go to step v.
• Else, cross it off

iii. Search right for a 1

• If none found, REJECT
• Else, cross it off

iv. Go to step ii

v. Check if other unmarked characters are on the tape

• If none, ACCEPT
• Else, REJECT

Case Analysis

Case 1: \( x \in L \), so \( x \) is of the form \( 0^n1^n \). So \( D_L \) will cross off 0’s and 1’s in sequence, check for other unmarked characters, find none, and ACCEPT

Case 2: \( x \notin L \), so \( x \) is not of the form \( 0^n1^n \). This means that \( x \) is of one of the following forms: it begins with a 1, contains more 0s than 1s, contains more 1s than 0s, or contains 0s following 1s.
For the first case, $D_L$ will reject immediately. For the second case, $D_L$ will reject in step iii. For the third case, $D_L$ will reject in step v. For the fourth case, $D_L$ will reject in step v. In all cases, $D_L$ rejects the input.

Since $D_L$ accepts all inputs in the language and rejects all inputs not in the language, $D_L$ is a decider for $L$, and thus $L$ is decidable.
Problem 2

Prove that it is not possible to construct a Turing Machine to decide the following language

\[ L = \{\langle M \rangle \mid M \text{ is a TM that prints a number greater than one half}\} \]

Solution:

Proof Idea: We will assume that \( L \) is decidable, and then use this assumption to show that an undecidable language can be decided, leading to a contradiction.

Claim: \( L \) is undecidable

Given: EverPrint\(_1\) is undecidable

Proof by contradiction. Assume that \( L \) is decidable, which means that it is possible to construct a decider for it, called \( D_L \). This Turing Machine will accept all inputs in \( L \) and reject otherwise.

We will now use this Turing Machine to construct a decider for EverPrint\(_1\), called \( D_E \), defined as follows

\( D_E \) on input \( M \):

i. Create a new machine \( M' \) which is a copy of \( M \) with an additional state that prints a 1, and then immediately transitions to the original start state of \( M \).

ii. Run \( D_L \) on \( M' \), DWID (do what it does, ie if \( D_L \) accepts: accept, if \( D_L \) rejects: reject)

Case Analysis:

Case 1: \( \langle M \rangle \in \text{EverPrint}_1 \). So, \( M \) prints a 1 at some point in its computation. This means that \( M' \) prints a one followed by another one at some point in its computation. This means that \( M' \in L \), and so \( D_L \) accepts, and thus \( D_E \) accepts.

Case 2: \( \langle M \rangle \notin \text{EverPrint}_1 \). So, \( M \) does not print 1 at some point in its computation. This means that \( M' \) prints a one and then does not a print another one at some point in its computation. This means that \( M' \notin L \), and so \( D_L \) rejects, and thus \( D_E \) rejects.

Since \( D_E \) accepts when \( \langle M \rangle \in \text{EverPrint}_1 \) and rejects when \( \langle M \rangle \notin \text{EverPrint}_1 \), \( D_E \) is a valid decider for \( D_E \). However, we know that EverPrint\(_1\) is undecidable, so our assumption is invalid \( \Box \).
Problem 3

Prove that it is not possible to construct a Turing Machine to decide the following language

\[ L = \{ \langle M \rangle \mid M \text{ is a TM that will not print 0’s and 1’s in combination} \} \]

Solution:

Proof Idea: We will assume that \( L \) is decidable, and then use this assumption to show that an undecidable language can be decided, leading to a contradiction.

Claim: \( L \) is undecidable

Given: EverPrint\(_0\) is undecidable

Proof by contradiction. Assume that \( L \) is decidable, which means that it is possible to construct a decider for it, called \( D_L \). This Turing Machine will accept all inputs in \( L \) and reject otherwise.

We will now use this Turing Machine to construct a decider for EverPrint\(_0\), called \( D_E \), defined as follows

\( D_E \) on input \( M \):

i. Create a new machine \( M' \) which is a copy of \( M \) with an additional state that prints a 1, and then immediately transitions to the original start state of \( M \).

ii. Run \( D_L \) on \( M' \)
   - If \( D_L \) accepts, REJECT
   - If \( D_L \) rejects, ACCEPT

Case Analysis:

Case 1: \( \langle M \rangle \in \text{EverPrint}_0 \). So, \( M \) prints a 0 at some point in its computation. This means that \( M' \) prints a one and then prints a zero at some point in its computation. So, \( M' \notin L \) since it prints both a zero and a one. So, \( D_L \) rejects, and and \( D_E \) accepts.

Case 2: \( \langle M \rangle \notin \text{EverPrint}_0 \). So, \( M \) does not print 0 at some point in its computation. This means that \( M' \) prints a one and then does not a print a zero at some point in its computation. This means that \( M' \in L \), and so \( D_L \) accepts, and thus \( D_E \) rejects.

Since \( D_E \) accepts when \( \langle M \rangle \in \text{EverPrint}_0 \) and rejects when \( \langle M \rangle \notin \text{EverPrint}_0 \), \( D_E \) is a valid decider for \( D_E \). However, we know that EverPrint\(_0\) is undecidable, so our assumption is invalid \( \Box \).