Answer the questions below, paying particular attention to the logic of your arguments. Short descriptions for how a function might computed (on any machine weve talked about) are sufficient to show that it is computable.

1. Show that if A is recognizable and A reduces to $\overline{A}$, then A is decidable.

   Answer:

   - If A reduces to B and B is recognizable, then A is recognizable. *(Sipser, theorem 5.28)*
   - In this case B is $\overline{A}$, hence $\overline{A}$ is recognizable.
   - A language is decidable iff it is Turing-recognizable and co-Turing recognizable. *(Sipser, theorem 4.28)*
   - Hence A is decidable

2. Let $B = \{\langle M \rangle \mid \text{M accepts exactly one of the strings 00 and 11}\}$.

   (a) What does Rice's theorem say about B?

   Answer:

   - The property P, that M accepts exactly one of the strings 00 and 11, is non-trivial, therefore B is undecidable.

   (b) Show that halting problem reduces to B.

   Answer:

   - Show a reduction from $H_{TM} \leq_M B$, where H = $\{< M >\mid \text{M halts on } \epsilon\}$

```
def R(<M>):
    def N(x):
        if x == ''00'':
            M("")
            return Accept
        elif x == ''11'':
            return Reject
        else:
            return Reject
    return(N)
```

   - Commentary on the reduction:
     - If N is in B then N accepts 00 or 11, but not both. By construction N doesn't accept 11, so it must accept 00. The only way it can accept 00 is when $M(\epsilon)$ either accepts or rejects.
     - If N is not in B then it can't accept on the previous conditions: N must accept either both 00 and 11, or neither. By construction N rejects 11, which means it can't also reject 00. Again by construction N can't ever reject 00: it either accepts 00 or it loops.

(c) Show that halting problem reduces to the complement of B.

Answer:

- $H_{TM} \leq_M \overline{B}$ via a similar reduction as above:

```
def N(x):
    if x == ''00'':
        M("")
        return Accept
    elif x == ''11'':
        return Accept
    else:
        return Reject
return(N)
```

- Commentary on the reduction:
  - If N is in $\overline{B}$ then N accepts either both 00 and 11, or neither. By construction N accepts 11, so it must also accept 00. The only way it can accept 00 is when $M(\epsilon)$ halts (accepts or rejects). If $M(\epsilon)$ loops, then N will either loop or reject.
  - If N is not in $\overline{B}$ then then N accepts 00 or 11, but not both. By construction N accepts 11, which means it can't also accept 00, which it does.

(d) Are B or its complement recognizable?

Answer:

- H_{TM} is recognizable, $\overline{H_{TM}}$ is not.
- But since $H_{TM} \leq_M B$, we know $\overline{H_{TM}} \leq_M \overline{B}$ and since $\overline{H_{TM}}$ is unrecognizable, then $\overline{B}$ is unrecognizable.
- And since $H_{TM} \leq_M \overline{B}$, we know $\overline{H_{TM}} \leq_M B$ and since $\overline{H_{TM}}$ is unrecognizable, then B is unrecognizable.
- In summary, both B and its complement are unrecognizable.

3. A language is *co-recognizable* if its complement is recognizable. Argue why each of the following languages is or is not recognizable and why it is or is not co-recognizable.

   (a) $L1 = \{\langle M \rangle \mid \text{M enters state q27 for some input string x}\}$.[1]

   Answer:

   - We can reduce H_{TM} to L1 by defining a new machine M' that halts if M enters q27.
   - H_{TM} is not decidable, hence L1 is not decidable.
   - H_{TM} is recognizable (you can run it until it halts), but $\overline{H_{TM}}$ is not.
   - $\overline{H_{TM}}$ reduces to $\overline{L1}$. **Proof reference**
   - $\overline{H_{TM}}$ is not recognizable, therefore $\overline{L1}$ is not recognizable.
   - Consequently L1 is recognizable, but not co-recognizable.

   (b) $L2 = \{\langle M \rangle \mid \text{L(M) contains at most two strings}\}$.

   Answer:

   - Take the complement of L2: $\overline{L2} = \{\langle M \rangle \mid \text{L(M) contains at least two strings}\}$.
   - A_{TM} is recognizable.
   - Reduce A_{TM} to $\overline{L2}$ by running A_{TM} on successive w's via dovetailing until you've found at least 2 strings.
   - This demonstrates that $\overline{L2}$ is recognizable.
   - Via Rice's theorem we know that L2 is undecidable, and via Sipser, theorem 4.28
   - Via Sipser, theorem 4.28 we know that if L2 is undecidable and $\overline{L2}$ is recognizable, that L2 must be unrecognizable.
   - Consequently L2 is unrecognizable, but is co-recognizable.

   (c) $L3 = \{\langle M \rangle \mid L(M) \subseteq \Sigma^*\}$

   Answer:

   - This language accepts everything in the powerset of the alphabet which is the universal language that accepts everything, L_U.
   - From Lance's Lesson from last week L_U is recognizable.
   - Since from Rice's theorem we know that L_U is undecidable, $\overline{L_U}$ is not recognizable.
   - Consequently L3 is recognizable, but is not co-recognizable.

[1] NB: This is the dead code problem, which is not decidable

4. A computable verifier is a deterministic Turing machine V that takes two arguments: x (the input) and y (the proof). A computable verifier always halts. Show that a language L is recognizable if and only if there exists a computable verifier V such that

   (a) if $x \in L$, then there is a string y such that $V(x, y)$ accepts, and

   (b) if $x \notin L$, then $V(x, y)$ rejects for every string y.

   Answer:

   - *Theorem:* If a language L is recognizable then there exists a computable verifier V.
     - If L is recognizable then there is a machine, M, that accepts every string in L.
     - Let $y = \epsilon$ and concatenate y to every string in L.
     - M is a computable verifier for L since it recognizes every string in L in the presence of y.
   - *Theorem:* If there exists a computable verifier V, then language L is recognizable.
     - A computable verifier $V(x, y)$ is a decider for L in the presence of y: By definition it accepts when x is in L and rejects when x is not in L.
     - If a string x is in L then $V(x, y)$ will accept it.
     - L is recognizable.
   - Hence a language L is recognizable if and only if there exists a computable verifier V such that

     (a) if $x \in L$, then there is a string y such that V(x,y) accepts, and

     (b) if $x \notin L$, then V(x,y) rejects for every string y.

5. Consider the following property: $P = \{\langle M\rangle \mid$ L(M) is accepted by some Turing machine that has an odd number of states$\}$. Show that P is a trivial property.

   Answer:

   - Modify the Turing machine in question by adding an inactive state.
   - L(M) is now accepted by some Turing machine with an even number of states.
   - The same language L(M) is now accepted regardless of whether the property P applies
   - Consequently the property is trivial.

6. Bonus: Read about The Recursion Theorem in the Sipser text. One implication of the recursion theorem is that in any general purpose programming language, one can write code that outputs the code itself. Write a python program that prints its own code. Do not use any file operations.

   https://www.udacity.com/course/viewer#!/c-ud557/l-1209378918/m-2986218594

   Answer:

   I submitted the following code to the link above. The answer was accepted.

```
x='x=%s\nprint x%%`x`'
print x%`x`
```