CS 254: Computability and Complexity	**Anonymous submission**
Problem Set #06	*October 26, 2019*

6. if there is an enumerator, there is a decider.

Proof: An enumerator for A, called E.
We can get a decider for A.

```
1. Get some input w
2. Simulate E
    a. print out every string in A
        i. check against w if match, accept.
        ii. if |w| < |cur output|, reject
```

Validation: We know that this will be a decider, because w is finite we know the strings length can only go up to |w|. So regardless if L(A) is infinite, we will always halt or accept due to the fact that we can only create a finite amount of strings given the restriction of having length $\leq$|w|

If there's a decider, there's an enumerator.

Proof: Decider A, called M.
Build E: (must output in canonical order)

```
1. Simulate M on tape 1
2. For each string s in {canonical order}
    a. Run M on s
    b. If accept
        i. write s on output tape
        ii. enumerate
    c. else, continue
```

Validation: We can run the TM, trying every possible string and if the string is accepted we will print and enumerate the accepted string. This fully simulates the TM by outputing every string in L(A)

Therefore if there is an enumerator, there is a decider, and if there is a decider, there is an enumerator.