Problem #1
a) Define the number of steps taken by a NDTM on input $x$.
b) Define the number of steps taken by a NDTM on inputs of length of $n$.

a) $\# NDTMi [x] =
case1: \min(y \in \text{\{\text{summation} \ast\}\}) \{ \# DT Mi[y,x] \}$ (this is the deterministic half of the NDTMi)

if there exists $y$ such that $DTMi[y,x]$ accepts (i.e. enters an accepting state)

case2: 1 otherwise

b) $\# NDTMi(n) = \max\{\# NDTMi[x]\}, |x| = n$

Problem #2
Define two (computational) problems $p_1$, $p_2$ to be poly-time equivalent iff it is possible to solve $p_1$ in polynomial time given an algorithm to solve $p_2$ in polynomial time ($p_1 \leq p_2$), and vice-versa ($p_2 \leq p_1$).

Are the following two problems poly-time equivalent?
If so, prove it.
If not, explain why not.

Decision:
Instance: NDTMi, $x \in \{0,1\}^*$, $m$ in unary

........m

$\text{(i.e. } 1 \ldots 1 = 1\text{.)}.$

Question: Does NDTMi accept $x$ in $m$ steps? i.e. does there exist a $y \in \{0,1\}^*$ s.t. $DTMi$ accepts $(y,x)$ in $m$ steps, if any (i.e. if such $y$ exists);
"NONE" if there is no such $y$.

1) YES! Decision \leq Optimization:
   - If optimization program returns $y$, then the Decision program returns YES
   - If optimization program returns "NONE", then the Decision program returns NO.

The running time of the Decision algorithm = running time of the Optimization algorithm + $O(1)$.

2) Optimization \leq Decision:
   - If Decision alg returns No, then the Opt. alg returns NONE.
   - If Dec. alg returns YES, then the Opt. alg must find $y$. It can do this by finding the bits of $y = Yk Y(k-1)$ ...
     $Y_0$ one at a time starting say with $Y_0$. 


If the Dec. algor rejects [NDTMi(\(Y_0 = 0\)), x,m], then we know that \(Y_0 = 1\). Else we know that it's ok to let \(Y_0 = 0\).

In general, having determined \(Y(i-1) = A(i-1),..., Y_0 = A_0\), one can determine \(Y_i\) by augmenting the NDTMi to NDTMi\(|oA(i-1)...A_0\), which overwrites the guessed \(Y_i Y(i-1)...Y_0\) with \(oA(i-1)...A_0\).

If the NDTMi\(oA(i-1)...A_0\) rejects, then we know that \(A_i = 1\).
Else ok to let \(A_0 = 0\). The size of the augmented DTMi\(A_i...A_0\) is just \(|DTMi| + O(m)\), which is poly in \(|NDTMi| + |x| + m\).

**Problem #3**

Explain what problems if any you encounter in doing the above reductions

In the case that \(m\) is given in binary instead of unary.

- Decision <= Optimization as before, but
- Optimization (NOT <=) Decision:
  The reason is that the required output \(y\) (\(y = 2^x\)) man be terribly long, length \(|y| = x\), for inputs of length \(n = |NDTMi| + |x| + \lg m\) \((m = poly(|y|) = |NDTMi| + |x| + O(|x|) = poly(|x|)\).

  The decision algorithm can take just poly\((n)\) steps, because it knows the existence of \(y\) without having to exhibit it. But the optimization algorithm must exhibit (print) \(y\).