Exercise 1

Task 1. Given the alphabet $\Sigma = \{0, 1\}$ and special symbol #, construct the following machines:

1. A Turing machine which decides the language consisting of palindromes over Σ.
2. A Turing machine that, once started, erases all the 1's from the head backwards, until it finds another #.

Task 2. In the definition of the class **P** one can replace the Turing machine model by more practical models of computation. Consider the following definition of a simple *RAM* machine model:

There exists a memory formed by cells, each storing a natural number m_i and indexed by a natural number $i \geq 0$. A program is a sequence of instructions L_l, numbered on lines $l \geq 1$. The instruction on each line can be:

1. SET (i, a), which assigns $m_i \leftarrow a$, where a is constant.
2. MOV (i, j), which assigns $m_i \leftarrow m_j$.
3. SUM (i, j), which assigns $m_i \leftarrow m_i + m_j$.
4. SUB (i, j), which assigns $m_i \leftarrow \max(0, m_i - m_j)$.
5. IfZ (i, l), which transfers control to line L_l if $m_i = 0$, where l is a constant.

In all instructions, i (similarly j) can be a single number (representing a fixed cell m_i), or also in the form $*i$, for a constant i, where i is now the address of the cell of interest (m_i).
Control begins at line L_1, and after executing L_l, it moves to line L_{l+1}, except possibly in the case of IfZ. Input and output are stored in memory at agreed-upon positions. A non-accessed cell contains the value zero. Execution terminates upon reaching the first non-existent line.

Describe how a Turing Machine *TM* could simulate the *RAM* machine from above. Hint: In the lecture we defined a single-tape TM, but one could use any number of tapes because a single-tape TM can simulate a k-tape TM. Therefore, feel free using more than one tape in your description if that helps.

Task 3. In the lecture we saw the probability amplification (Slide 13) as follows:

1. Run the machine M k times with independently chosen random strings $y_1, ..., y_k \in \{0,1\}^{r(n)}$
2. At the end the input x is accepted if $\langle x, y_i \rangle \in L(M)$ for at least $k/2$ many $i \in [1,k]$

Show that the error probability of this algorithm is $2^{-\Theta(k)}$.
Hint: Use the Chernoff bound.

Task 4.

1. Show that If L_1 and L_2 belong to **NP** then also $L_1 \cup L_2$ and $L_1 \cap L_2$ belong to **NP**.
2. Suppose that $L_1 \leq L_2$ and $\mathbf{P} \neq \mathbf{NP}$. Answer and briefly justify
   - (a) If L_1 is in **P**, L_2 is in **P**?
   - (b) If L_2 is in **P**, L_1 is in **P**?
   - (c) if L_1 is in **NP**-Complete, is L_2 **NP**-Complete?
   - (d) if L_2 is in **NP**-Complete, is L_1 **NP**-Complete?
   - (e) if $L_2 \leq L_1$, are L_1 and L_2 **NP**-Complete?