Question 1. [6 points]
Give a recursive definition for the set of strings of digits 0, 1, 2, 3, ...9 that cannot start with the digit 0 nor with the digit 1.

Solution
We call this set of strings POSITIVE01

Rule 1: 2,3,4,5,6,7,8,9 are in POSITIVE01
Rule 2: If w is in POSITIVE01, w0, w1, w2, w3, w4, w5, w6, w7, w8, w9 are also words in POSITIVE01

Question 2. [6 points]
Show whether or not the following two regular expressions define the same language:

\[(a+b)^*ba(a+b)^* + ab^* \quad \text{and} \quad (a+b)(a+b)^*\]

Solution
\((a+b)^*ba(a+b)^* + ab^*\) corresponds to all words with “ba”. A word without a “ba” should be a sequence as a’s (may be empty) followed by a sequence of b’s (may be empty) but not all of these are generated by \(ab^*\). For instance the word \(aabb\) is not generated by the expression \((a+b)^*ba(a+b)^* + ab^*\).
However \((a+b)(a+b)^*\) corresponds to all non empty words on \{a, b\}. Therefore these two expressions don’t define the same language.

Question 3. [8 points] YOU DON’T HAVE TO EXPLAIN YOUR ANSWERS.
We consider the following transition graph T.
i) [4 points] Describe the language L accepted by T.
Solution
All words that consist of a sequence of ab’s (may be empty) followed by a “b” and a sequence of a’s (may be empty), as well as all words that start and end with an “a” and without any double letter.

ii) [4 points] Using the description found in (i), give a regular expression corresponding to the language L.
Solution
(ab)*ba* + a(ba)*

Question 4. [10 points]
Let $\Sigma = \{a, b\}$ and let L be the language of all words on $\Sigma^*$ starting with ab. For instance, the word abbbabb is in L, however aabbaab is not in L.

(i) [4 points] Construct a finite automaton for the language L.
Solution

(ii) [2 points] Give a regular expression corresponding to the language L.
Solution
ab(a+b)*

(iii) [4 points] Describe (in English phrases) the language L corresponding to the regular expression
$E = (b + \Lambda) (ab)^* (a + \Lambda)$
Solution
All words without two consecutive a’s nor two consecutive b’s.
**Question 5. [6 points]** Using the method seen in class (Proof of Kleene’s theorem), give a regular expression for the language accepted by the following transition graph:

![Transition Graph]

**Solution**
We start by creating a unique final state, then we delete the non final and initial states one by one.
The regular expression is:

\[ [ab(a+)]^* \ (ab)(a^+ \Lambda) + [ab(a+)]^* \ b(a)^*b \]
Question 5. [4 points]
Let L be a regular language on $\Sigma = \{a, b\}$. Give an algorithm that will transform a transition graph for the language L into a new transition graph for the language complement of L. [EXPLAIN YOUR ANSWER]

[The complement L’ of a language L is $L’ = \Sigma^* - L$]

Solution:

*Algorithm:*
First you should transform the transition graph into a finite automaton (in two steps: from Transition graph to a regular expression and then from the regular expression to a finite automaton. Both steps are a part of the proof of Kleen’s theorem).
Then, we change in the finite automaton
   - final states into non final states and
   - non final states into final states.