Turing Machine exercises to prepare Practice 4 and the Exams.

1. Try the tutorial on designing Turing Machines with JFLAP, which proposes a machine that recognises strings from the language L = { $a^nb^nc^n$ | n>0 }. Note that the Fast Run trace apparently does not work. To debug the designed machines try running step by step with: **Input->Step->Enter input word->Step button**.

2. **Moving Symbols**. Given the input alphabet {a, b}, design a Turing Machine that transcribes strings of type {a^nb^m | n, m>0} into strings of type {b^ma^n | n, m>0}. That is, for an initial tape with the string ##aabbb## it must end containing ##bbbaa##. Test your machine with 10 words. The machine must not handle initial words that don't follow the specifications. Use a transducer type Turing machine and end with the head placed at the beginning (left) of the string. Testing should be done via: Input->Multiple Run (Transducer).

3. **Successor**. Given the input alphabet {0, 1}, design a Turing Machine that generates the successor of a **binary number** stored on the initial tape. Given the string ##1111## it should generate #10000##. Test your machine with 10 words that include some problematic cases. Pay attention to the carry propagation.

4. **Specific Marks and Restoration**. Given the input alphabet {a, b, c}, and an initial string w ∈ $(a+b+c)^n$, n≥0 design a Turing Machine that generates the output string $w1^n$. For an input word #abcaa# it must generate #abcaa11111#. Test your machine with 10 words. This exercise is number 6 of the proposed exercises, section c), and is very similar to 6b).

5. **Binary addition**. Given a tape with two binary numbers, separated by the symbol $, build a Turing machine that calculates the sum of both. The final tape will contain only the result. Test your machine with 10 words that include some problematic cases. Note that the difficulty depends on how you organise the tape and the elements involved in the algorithm (addends, result, marks, etc.).

6. **Palindromes**. Given the input alphabet {a, b}, and an initial string w Є $(a+b)^+$ design a Turing Machine that generates a palindrome of the form ww^T. For an input word #abaaaaa# it should generate #abaaaaaaaaaaba#. Test your machine with 10 words.

7. **Predecessor**. Given the input alphabet {0, 1}, design a Turing Machine that generates the predecessor of a binary number stored on the initial tape. Given the string ##10000## it should generate ##1111##. Test your machine with 10 words that include some problematic cases. Pay attention to the carry propagation.

8. **Subtraction**. Given a tape with two **binary numbers**, separated by the symbol $, build a Turing machine that calculates the subtraction of both. The final tape will contain only the result. Test your machine with 10 words that include some problem cases. Pay attention that the difficulty depends on how you organise the tape with the elements involved in the algorithm (operands, result, marks, etc.).

9. Review your exercises 5 and 8, addition and subtraction. They should be very similar and contain recognisable machines (successor and predecessor). Do you have more than 12 states? This is not recommended. What would happen if you chose to mark with different symbols the 0/1 from both operands, e.g., the left operand with a/b and the right operand with x/y? Or if you delete the bits of the operand that is not overwritten? What would happen if you were asked to keep the original operands next to the result? Or if you write the result to the right of the operands?