(1) On the course web-page you can download a simple Python implementation of a shift-reduce recognizer. You can run it either in the interactive mode by executing

```
python sr.py "the input sentence"
```

from the command line.

a) You will realize that the recognizer is fairly fast for grammatical sentences such as “the cat on the roof of the house near the library of the university sleeps” but it is very slow for ungrammatical inputs like the incomplete sentence “the cat on the roof of the house near the library of the university”. Try to explain why this is the case.

b) Try to improve the algorithm by taking advantage of the fact that the input grammar has a special form: rules either have exactly one terminal symbol (word) on the righ-hand side, or an arbitrary number of non-terminal symbols.

Hint: If you apply the algorithm to ungrammatical inputs like “the cat … of the university” you will see that a large number of “useless” configurations – configurations that cannot lead to an accepting configuaration if the grammar has the form sketched above – are generated by the recognizer. How can the creation of such configuration be blocked?

Note: You don‘t need to implement anything. Answers for a) and b) can be given in plain English. (Implementations are nevertheless welcome)

(2) Implement the CYK algorithm. Use the grammar from slide 32 and the following sentences as test cases:

a) The boy shot an elephant in his pajamas

b) The cat on the roof of the house near the library of the university chases a mouse

c) The cat on the roof of the house near the library of the university

For b) and c) you have to extend the grammar with rules like “N → cat,” “V → chases” and so on. The output should be as follows (or similar):

```
1, 2, DET
2, 3, N
1, 3, NP
...
```

(3) On the course web-page you can download a relative large grammar. This grammar is not in Chomsky normal form, so the CYK recognizer cannot be applied to it directly.

a) Convert the grammar into Chomsky normal form (see slide 39)

b) Implement at least two of the methods on slides 40/41 and compare runtimes.