DALHOUSIE UNIVERSITY
Inspiring Minds

Winter 2020 (Jan6-Apr6)
Faculty of Computer Science

CSCI 4152/6509 — Natural Language Processing

Assignment 1

Due: *Tuesday, Feb 4, by midnight*
Worth: 99 marks $(99 = 22 + 30 + 22 + 15 + 10)$
Instructor: Vlado Keselj, CS bldg 432, 902.494.2893, vlado@dnlp.ca

Assignment Instructions:

The submission process for Assignment 1 is based on the `submit-nlp` command on `bluenose` as discussed in the lab, or in the equvalent way by using the course web site, where you need to follow ‘Login’ and then the ‘File Submission’ menu option.

Important: You must make sure that your course files on bluenose are not readable by other users. For example, if you keep your files in the directory `csci6509` or `csci4152` you can check its permission using the command:

```
    ls -ld csci6509
or  ls -ld csci4152
```

and the output must start with `drwx------`. If it does not, for example if it starts with `drwxr-xr-x` or similar, then the permissions should fixed using the command:

```
    chmod 700 csci6509
or  chmod 700 csci4152
```

1) (22 marks) Complete the Lab 2 as instructed. In particular, you will need to properly:

a) (4 marks) Submit the file ‘hello.pl’ as instructed.

b) (4 marks) Submit the file ‘example2.pl’ as instructed.

c) (4 marks) Submit the file ‘example5.pl’ as instructed.

d) (5 marks) Submit the file ‘task1.pl’ as instructed.

e) (5 marks) Submit the file ‘task2.pl’ as instructed.

Notice that the examples from (a) and (b) need to compile; if a syntax error got introduced to an example program by your typing mistake or by introducing incorrect characters through copying and pasting from a pdf file, so that the example program does not compile, it will not be accepted. The lab instructions state that the programs should be tested before being submitted. Unless specified differently, you should always type your code and not copy and paste it. This will give you more chance to learn the illustrated concepts.

2) (30 marks) Complete the Lab 2 as instructed. In particular, you will need to properly:

a) (5 marks) Submit the file '`matching.pl`' as instructed,

b) (5 marks) Submit the file '`matching-data.pl`' as instructed,

c) (5 marks) Submit the file '`word_counter.pl`' as instructed,

d) (5 marks) Submit the file '`replace.pl`' as instructed, and

e) (5 marks) Submit the file '`ngram-output.txt.gz`' as instructed.

f) (5 marks) Submit the file '`line-count.pl`' as instructed.

Note that any program that you submit needs to compile. Even if a complete source code is given in the lab, you need to type it instead of using cut-and-paste, and you need to make sure not to introduce any errors into the program. This follows from the lab instructions that programs must be tested before submitting.

3) (22 marks) Submit your answer to this question as a plain-text file called `a1q3.txt` using the `submit-nlp` command, or the course web site. Clearly separate your answers to parts a) and b).

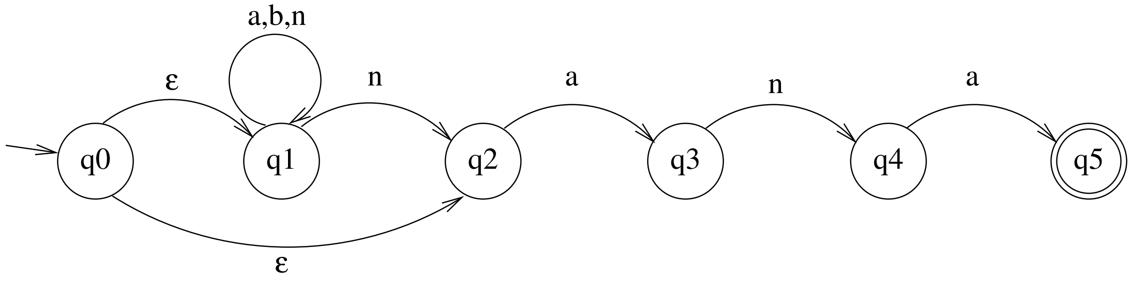
a) (7 marks) List the levels of NLP, with one-sentece description for each of them.

b) (10 marks) Give example of an ambiguous sentence and briefly explain what are two different possible interpretations of the sentence. Do not use sentences given in the lectures.

c) (5 marks) For your example in b), describe which level of NLP or levels are involved in the ambiguity.

4) (15 marks) Submit answer to this question in a plain-text file named `a1q4.txt` using the `submit-nlp` command. Clearly separate a) and b) parts in the solution.

Consider an NFA (Non-deterministic Finite Automaton) described by the following graph:



a) (5 marks) Give three examples of words accepted by this NFA. Briefly describe what language is accepted by this NFA.

b) (10 marks) Translate this NFA into a DFA using the process discussed in class. Submit your solution in plain text (as a part of file `a1q4.txt`), where the DFA is shown as the textual table in a format shown below. You **must** use the process described in class, and follow the required format.

```
State      |   a    |    b   |    n   |
-----------+--------+--------+--------+
S: q0q1    |  q0q1  |  q0q1  |  q0q1  |
-----------+--------+--------+--------+
   q0q1    |  q0q2  |  q0q2  |  q0q2  |
-----------+--------+--------+--------+
F: q0q2    |  q0q1  |  q0q1  |  q0q1  |
-----------+--------+--------+--------+
```

Explanation: Use characters minus, vertical line, and plus to draw the table. The columns correspond to input characters. The DFA states are set of NFA states shows as sequences of states in a sorted order by index (for example, use `q0q1` rather than `q1q0`). use labels `S:` and `F:` to denote start and finish states. If an NFA state is empty set, then use the word ‘`empty`’ to denote it.

5) (10 marks) Write a Perl program named `a1q5.pl` and submit it using the `submit-nlp` command.

The program reads the standard input (use the diamond operator) and at the end produces basic statistics about the input in exactly the following form:

```
number of lines: 5
shortest line length: 0
longest line length: 23
```

The final LF (or CRLF) characters are not considered to be part of the line, and you should also remove any trailing spaces from each line, before computing the lengths. For example, the above output would be obtained from the following file:

```
first line
12345678901234567890123
a trailing spaces line:

above line has spaces
```

or, if we make the white-space characters visible (␣ for space, \t for tab, and \n for new-line or LF):

```
first␣line\n
12345678901234567890123\n
a␣trailing␣spaces␣line:␣\t␣␣\n
```

```
␣␣␣\n
above␣line␣has␣spaces\n
```

Hint: The Perl function `length()` can be used to get length of a string.