

## CA Assignment 20%

- **Submission Date:** The assignment source code, with all files necessary to run the program and also test data, together with the report are to be delivered to me for testing - as an upload to Brightspace in a zipped folder by the **26th of November 2025 before 5pm.**

Please name your upload folder on Brightspace using your name and student number  
e.g. IreneMurtaghB0001234.

- **Assignment Value:** This assignment is worth **20%** of the available marks for this module.

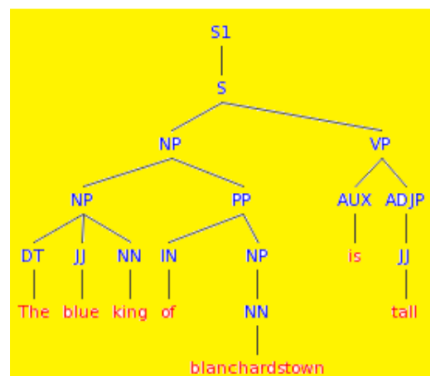
**You are required to:**

*Describe, design and implement* in software a parser for Regular Expressions using the programming language of your choice.

**INPUT:** The parser should take as input and analyse a set of regular expressions taken from the acceptable sentences below:

*The/A king/kings dislike(s)/like(s) the new cat*

**OUTPUT:** Your input sentence should be parsed, and either accepted or rejected as a valid regular expression. The **output** should be **bracketed phrasal structure** for the sentence **for as far as it is grammatical**. You should also output an appropriate syntax parse tree displaying the sentence structure similar to below but suited to your inputted regular expression.



→ [ S1 [ S [ NP [ NP [ DT [ The ] ] [ JJ [ blue ] ] [ NN [ king ] ] ] [ PP [ IN [ of ] ] [ NP [ NN [ blanchardstown ] ] ] ] ] [ VP [ AUX [ is ] ] [ ADJP [ JJ [ tall ] ] ] ] ] ]

The parser program may be implemented in Python or Java (or any language or your

choice) and **MUST** be accompanied by a lexicon file rich enough to establish a syntactic set of rules that will also accompany your system. <sup>[1]</sup><sub>SEP</sub>

The system should maintain both a file to hold all lexicon entries of the allowable vocabulary as well as a file to hold the rules that enable the syntax of a regular expression to be understood. HINT: Use text files if you wish. <sup>[1]</sup><sub>SEP</sub>

The program must output the bracketed structure of the grammar and whether or not a regular expression is acceptable as well as a parse tree of the sentence structure.

Marks will also be allocated for additional features and any additional grammatical features you may wish to add are encouraged, transitivity, subordination, etc. <sup>[1]</sup><sub>SEP</sub>

**REPORT:** The report should include:

- A discussion describing how you have implemented the various lexical categories e.g. noun, verb etc. in terms of the architecture of the lexicon.
- A discussion describing how you have implemented your parser program and its interface to the lexicon and how it uses the lexical entries.
- A description of your system architecture, providing a description of your software design stating the problems that you needed to address and how you surmounted them... for each of these, state individually the problem and then your solution.
- Screen shots illustrating the execution of your completed program at different stages of execution.

**You will receive marks for:**

1. All work attempted and submitted.
2. The quality of your solution design and you report describing this in detail.
3. The quality of your code and in-program documentation.
4. A bug-free, fully tested, working application.
5. Innovation and creativity in your design and solution.
6. A professionally written and word-processed report

**Please include in your upload:**

1. All software files necessary to allow running and testing of your software program.
2. Test data, which will be used to test your parser (RE's for inputting)
3. A professionally written and word-processed report that contains information relating to the report specification above, including your source code in an appendix section.

***Academic Honesty:*** *Academic dishonesty will be dealt with severely. At a minimum, you will receive a mark of zero.*

