Programming Language Translation

Mini Project

Total marks: 50%

Instructions:

1. This project is an individual project.
2. Those who plagiarize/copy will get zero mark.
3. An interview session will be conducted online from 29 Nov 2021 onwards. You will need to register a slot to present your work to the examiner. More on this will be announced on the MMLS.
4. If you fail to attend the interview session on the registered slot, you will be given zero mark.
5. Your working/methodology must be based on what was taught to you during the lecture/tutorial only.
6. Late submission will get zero mark.

Mini Project Due Date: 29 Nov 2021

Question:

In this project, you are going to develop a mini compiler for an animal language. You can pick an animal you want. This animal programming language will allow one to express *expressions/statements* that the animal will utter.

For example:

This is a BNF for a Sheep Language that will utter *sentences* like “ba”, “baa”, “baaaaa”, etc

```
<SHEEP_TALK> → ba <SHEEP_NOISE>
             | ba
<SHEEP_NOISE> → a <SHEEP_NOISE>
              | a
```

This compiler would only perform lexical analysis, and syntax analysis. Semantic analysis is not required. You need to be imaginative on the kind of tokens and expressions/statements the programming language can support that will mimic the animal utterance or even you can be imaginative like how cartoon animal characters “speak/utter”.

These are the tasks you have to perform.

1. Identify tokens in the animal language. And build a DFA for the tokens.

2. Design the animal language syntax (use BNF notations). Make sure the animal language has got 4 to 5 non-terminals (for example, in the Sheep language above we have 2 non-terminals - sheep_talk and

sheep_noise). Use your imagination to craft the terminals and the non-terminals required in your animal language. Please refrain from complicating your animal language with too many non-terminals (4 to 5 non-terminals is sufficient).

3. Build a parse table to perform syntax analysis (you can choose to do top-down parsing or bottom-up parsing)

4. Build a program that performs lexical and syntax analysis. You may use tools taught in the course or do from scratch. Test your program such that it accepts all valid language and give compilation errors for the invalid ones. The program must be able to show the output of a lexical analysis, and separately show the output of a syntax analysis, and finally show the combined result of lexical and syntax analysis.

When designing and building the program, it must be methodological as taught in the lectures of this course (do not use any methods not taught in the course). The methodology used must be documented and justified.

Marking Rubric.

Criteria	Very Good 8 - 10 marks	Good 6 - 8 Marks	Average 4 - 6 marks	Poor 2 - 4 marks	Very poor 0 -2 marks
Lexical analysis done methodologically (i.e DFA)	The steps on how lexical analysis done shown clearly and in detail. Presentation was clear with all details explained.	The steps on how lexical analysis done shown with some missing details. Presentation was clear with some details missing.	Only the output of lexical analysis shown with most of the steps not shown. Presentation done with many missing points.	The methodology adopted is not correct. Presentation was poor.	No evidence or strong deficiencies in the analysis. Presentation was poor.
Language design is well formed (using BNF)	language designed and the language has been made compatible for parsing with all the steps shown in detail. Presentation was clear with all details explained.	language designed and the language has been made compatible for parsing with some steps missing. Presentation was clear with some details	language designed and the language has been made compatible for parsing with no steps shown. Presentation done with many missing points.	language design but no effort made to make it compatible. Presentation was poor.	Language design don't make sense. Presentation was poor.

		missing.			
Syntax analysis done methodologically (i.e. top down parser/ bottom up parser)	The steps on how syntax analysis done shown clearly and in detail. Presentation was clear with all details explained.	The steps on how syntax analysis done shown with some missing details.	Only the output of syntax analysis shown with most of the steps not shown. Presentation done with many missing points.	The methodology adopted is not correct.Presen tation was poor.	No evidence or strong deficiencies in the analysis. Presentation was poor.
Program(i.e. the compiler) build systematically and modular	The program is systematic and modular. Presentation was clear with all details explained.	The program can be made more modular and systematic but missing. Presentation was clear with some details missing.	Only a few parts of the program is modular and systematic. Presentation done with many missing points.	The program is largely not systematic and modular. Presentation was poor.	The program is not systematic and modular. Presentation was poor.
The program runs as per expectation	All the the program output is correct. Presentation was clear with all details explained.	Some of the program output is wrong. Presentation was clear with some details missing.	A 50:50 correct:wrong ratio for the output Presentation done with many missing points.	Most of the program output is wrong Presentation was poor.	The program fails to run.Presentatio n was poor.