

**University of Technology, Jamaica**  
**Faculty of Engineering and Computing**  
**School of Computing and Information Technology**  
**Analysis of Programming Languages (CIT4004)**  
**Semester 2 - 2023/2024**

**Draft Student Group Project (10% theory + 20% practical)**

**A Programming Language Interpreter/Compiler that offers a natural language interface and runs on a cloud platform**

**Lecturer:** David W. White (dwwhite@utech.edu.jm)

**Date Given:** Week of February 5, 2024

**Due Date:** Week of March 29, 2024

**Design a programming language that interfaces with a large language (LLM) model like OpenAI's ChatGPT or Google's Bard, etc, is deployed on a cloud platform such as Microsoft's Azure, etc, and which demonstrates at least 5 characteristics of a good programming language that you have studied in this class.**

This project requires your group to develop a new programming language based on the principles you have learned in this class. Your group will build a compiler or interpreter for your language, and your language must feature a natural language interface allowing natural language input, as well as cloud deployment enabling access to your language over the internet.

Your language must perform lexical, syntactic and semantic analysis, and must incorporate the results of the semantic analysis into the core logic of your compiler/interpreter, and allow the resulting target code to be executed. Your programming language must feature at least five (5) of the characteristics of a good programming language that you studied in class. The grammar of your language can be specified in regular CFG, BNF, EBNF, or PEG formats. You are free to use any development language of your choice, and you may also use compiler development tools such as LEX/YACC (generates C code), FLEX/BISON (generates C/C++ code), JLEX/CUP (generates Java code), Jack (generates Java code), PLY (generates Python code), or similar tools.

You may use OpenAI's ChatGPT, Google's Bard or a similar LLM to perform natural language processing for your language. You may also use Microsoft's Azure or similar platform to deploy your language on the cloud. Note that you have free student access to certain elements of the Microsoft Azure platform once you log in using your UTech credentials.

Organize yourselves into groups of no more than four (4) persons per group, and produce a project report and a project implementation, both of which will be graded separately. The Project Report will represent the Individual Assignment grade on your portal, while the Project Implementation will represent the Assignment 1 grade on your portal.

## **Required**

To complete this project successfully, you should:

- Produce a project report containing the components listed below in the grading scheme.
- Develop an executable that performs lexical, syntactic and semantic analysis on input source code written in your programming language
- Provide users with a natural language interface to your programming language
- Generate executable target code
- Allow your programming language to be accessed and/or deployed on the cloud
- Upload your project report, source code and working application code to the project assignment space on the course portal
- Make a 7 minute presentation on your project to the class in the allotted tutorial time

At a minimum, the **syntax** for your programming language must allow programmers to write source code that assigns values to identifiers, computes arithmetic expressions, performs conditional branching or conditional execution of statements depending on the value of identifiers, and displays the value of identifiers.

The **semantics** for your programming language must include appropriate interpretation of identifier declarations and assignments, arithmetic expressions, conditional execution/conditional branching, type checking, and scope rules.

- **Language Design:** Your group should define a programming language with a clear syntax and grammar. Your programming language interface should support the following features:
  - Allow basic control structures (sequence, selection, iteration)
  - Allow use of identifiers, keywords and reserve words
  - Allow for scope and binding
- **Integration with LLM:** Your chosen LLM (i.e. ChatGPT, Bard, etc.) must be integrated into your project, using the relevant APIs or libraries provided by the LLM to receive lexically, syntactically and semantically correct text inputs from the LLM
- **User-Friendly Interface:** Create a user-friendly command-line, mobile, web, graphical or Text to Speech/Speech to Text interface that allows users to write source code in your language and execute the target code easily.
- **Cloud Deployment:** Use Microsoft Azure (or similar platform) to host your compiler/interpreter and make it securely accessible to users.
- **Error Handling:** Include error handling, lexical, syntax and semantics validation for user supplied source code, and interaction with the LLM. Provide informative error messages to users.
- **Documentation:** Prepare clear documentation explaining how to use your custom designed programming language and the interface you created.

- **Examples and Tutorials:** Include sample code and tutorials to help programmers get started with writing programs in your programming language.

### **Bonus Marks**

You can provide a secure execution environment for running code, mitigating security vulnerabilities. You can also extend your programming language with additional features such as functions, complex data structures, libraries and others.

### **Grading Scheme**

#### **Project Report (50 marks)**

- Paradigm the language you developed belongs to (1/50 marks)
- Explaining whether your language is general purpose or domain specific (1/50 marks)
- Explaining whether your language is low level or high level (1/50 marks)
- Correct grammar for the language you developed (10/50 marks)
- Complete parse tree/AST for a sample program in your language (10/50 marks)
- Full list of tokens for the language you developed (5/50 marks)
- Regular expressions you used to recognize all the tokens for the language you developed (10/50 marks)
- Demonstration of scope and binding in sample code written in your programming language (5/50)
- Details on the programming language you used to develop your compiler (2/50 marks)
- Three characteristics of a good programming language (from those you studied in class) that are evident in your designed programming language, and examples of how do these characteristics affect the readability, writability and reliability of your designed programming language (5/50 marks)

#### **The Application (50 marks)**

- Receive appropriate output from the LLM for processing (5/50 marks)
- Correctly perform lexical analysis and tokenization (5/50 marks)
- Correctly perform syntax analysis and parse tree/AST generation (7/50 marks)
- Correctly perform semantic analysis (7/50 marks)
- Target code runs and produces expected result (8/50 marks)
- Appropriate user interface and feedback to user (4/50 marks)

- Adequate error handling (7/50 marks)
- Effective cloud deployment (7/50 marks)

### Important Note

Your completed project must run, you must upload your project report and application source code to the online course portal provided, and you must conduct the 7 minute presentation in your tutorial class to receive a project grade. Bring a copy of this project sheet with you to the presentation and place the names and id numbers of all your group members in the space provided. Projects which do not run will not receive a passing grade. No individual projects will be accepted. Plagiarism is considered as a very serious offense by the University and will be penalized as outlined "Academic Misconduct" section of the Student Handbook.

### **Useful Compiler Generation and LLM Resources:**

#### **Lex and YACC primer/HOWTO by Bert Hubert**

<https://tldp.org/HOWTO/Lex-YACC-HOWTO.html>

#### **Lex and Yacc: A Brisk Tutorial by Saumya K. Debray**

<https://www2.cs.arizona.edu/~debray/Teaching/CSc453/DOCS/tutorial-large.pdf>

#### **PLY (Python Lex-Yacc) by David M. Beazley**

<https://www.dabeaz.com/ply/ply.html>

#### **Write text parsers with yacc and lex by Martin Brown**

<https://developer.ibm.com/technologies/systems/tutorials/au-lexyacc/>

#### **Using JFlex and CUP to implement a compiler**

<https://www.cs.auckland.ac.nz/courses/compsci330s1c/lectures/330ChaptersPDF/Chapt1.pdf>

#### **What is ANTLR (ANother Tool for Language Recognition)? by Terence Parr**

<https://www.antlr.org/>

#### **The Design of a Full Computer Language**

<https://www.cs.auckland.ac.nz/courses/compsci330s1c/lectures/330ChaptersPDF/Chapt9.pdf>

#### **How can I access the ChatGPT API?**

<https://help.openai.com/en/articles/7039783-how-can-i-access-the-chatgpt-api>

#### **Bard API**

<https://aibard.online/category/bard-api/>

#### **Microsoft Azure**

<https://azure.microsoft.com/en-us/>