

**University of Technology, Jamaica
Faculty of Engineering and Computing
School of Computing and Information Technology**

Analysis of Programming Languages (CIT4004)

Semester 2 – 2024/2025

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

**A Programming Language Interpreter/Compiler that offers
a natural language interface to book tickets for events, etc**

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

Date Given: Week of February 17, 2025

Due Date: Week of March 31, 2025

Design a programming language that allows the user to perform bookings of just about any tickets. For example a user could book tickets to a concert, or rides on a train or bus, or airline tickets, a tickets to football match. The user should be able to use specific words to reserve the ticket bookings, confirm the ticket reservations, pay for the tickets, and cancel ticket reservations. The user should also be able to see a list of events or other resources which offer tickets, how many tickets are available and when, and what the user has reserved, confirmed, paid for, or canceled over time. You are required to connect to a large language (LLM) model like Microsoft Copilot, OpenAI's ChatGPT or Google's Gemini, etc., to provide realistic information on the resources that have reservable tickets, but any reservation, and other booking details etc, will be simulated and stored locally or in the cloud. The language should demonstrate at least 2 characteristics of a good programming language that you have studied in this class.

This project requires your group to develop a new programming language that facilitates the booking of tickets using a pseudo-natural language interface. Your design must be based on the principles you have learned in this class. Your group will build a compiler or interpreter for your language.

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 two (2) 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. Special marks will be awarded for projects deployed on the

cloud, use voice (speech-to-text and text-to-speech), contain innovative features, and that mimic real-world operation as close as possible.

You must use Microsoft's Copilot, OpenAI's ChatGPT, Google's Gemini or a similar LLM-based Artificial Intelligence platform to provide realistic real-time ticket and booking information for your language. For example, the LLM should provide a listing of all activities, events, transportation, etc for which a ticket can be reserved in your area. If the user indicates a tickets for a concert, sporting match, transportation vehicle, etc. is required, then the LLM should provide real-time information what dates and times the resource is available, how many tickets are available, the cost of the tickets, and reservation, payment and cancellation policies.

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 five (5) 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 an interface to your programming language
- Generate executable target code
- Allow your programming language to be accessed locally 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 10-minute online presentation on your project to the class in the allotted tutorial time, involving all members of the project team and with cameras on

At a minimum, the **syntax** for your programming language must allow programmers to write source code that lists resources that have tickets available, book tickets via reservation, confirm and pay for those reservations, and cancel reservations. It should also allow a list of bookings, confirmed reservations, payments, and cancellations to be seen.

The **semantics** for your programming language must include appropriate translation of user requests to list resources that have tickets available, book tickets via reservation, confirm and pay for those reservations, and cancel reservations. It should also allow correct translation of user requests to list bookings, confirmed reservations, payments, and cancellations .

- **Language Design:** Your group should define a programming language with a clear syntax and grammar, and allow for scope and binding.
- **Integration with LLM:** Your chosen LLM (i.e., Copilot, ChatGPT, Gemini, etc.) must be integrated into your project, using the relevant APIs or libraries provided by the LLM to provide additional information about the steps being performed by the use program being executed
- **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.

Example Interaction

Welcome to APL Booking Project Language (APBL Version 1.0)

List Knutsford Express schedule.

Book Knutsford Express from Montego Bay to Kingston on February 17, 2025 at 8:30 AM for Joy Reynolds.

Confirm reservation for Knutsford Express for Joy Reynolds.

Pay reservation for Knutsford Express for Joy Reynolds.

Bonus Marks

Bonus marks will be awarded for groups that implement innovative features. For example, an AI could prevent one person from consistently buying all the tickets for say a concert or a football match, with the ostensible purpose of illegally reselling them at a much higher price (price gauging).

Grading Scheme

Project Report (50 marks – 10%)

- 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)
- Two 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 – 20%)

- Integration with LLM to explain steps being executed (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 10-minute online presentation in your tutorial class to receive a project grade. Place the names and id numbers of all your group members in the project documentation and code. 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.

In the project report and during the group presentation, each group member must state which substantial portion of the project that group member worked on (this cant be just the documentation alone), and must explain that section. Group members must be present in-person in the tutorial class during the presentation to receive a grade, and no video recordings of the presenters will be accepted.

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/>

Syntax: a language agnostic parser generator by Dmitry Soshnikov

<https://dmitrysoshnikov.medium.com/syntax-language-agnostic-parser-generator-bd24468d7cfc>

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>

Google Deepmind and Google Gemini

<https://deepmind.google/technologies/gemini/>

Microsoft Azure

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