**Project Part 1: Designing a BNF Grammar for a Simple Programming Language**

|  |  |  |
| --- | --- | --- |
| Course Code: BCS 222 |  | Course Name: Programming Paradigms |
| Date: 02/25/2025 |  | Time: 1 |
| Location: |  | Instructor(s): Dr. Haythem El-Messiry |
| Number of Students: |  | Number of Pages: |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Course Learning Outcomes** | **PLOs** | **Quizzes** | **Project** | **Mid term** | **Final exam** |
| **CLO1.** Identify programming paradigms of functional, logic, and object-oriented programming languages. | **1** | **X** |  | **X** |  |
| **CLO2.** Design software using constructs of functional, logic, and object-oriented programming languages. | **3** |  | **X** | **X** | **X** |
| **CLO3.** Apply concurrency, synchronization and event handling constructs to solve synchronization problems. | **7** | **X** | **X** |  | **X** |
| **CLO4.** Analyze the behavior of programs written in functional, logic, and object-oriented programming languages. | **2** |  | **X** |  |  |
| **CLO5.** Exhibit effective writing and presentation skills in a teamwork project. | **4, 6** |  | **X** |  |  |

**Students** (double-click to edit grades)

****

**Objective:**

The aim of the project is to explore the syntax of a simple programming language by defining its grammar using Backus-Naur Form (BNF). Students will design the BNF notation and implement a parser in Python to validate code snippets written in this language.

**Project Details:**

1. **Language Specification**: Define a simple programming language that includes the following constructs:
   * Variable declaration (e.g., int a;, float pi;)
   * Assignment statements (e.g., a = 5;, pi = 3.14;)
   * Basic arithmetic operations (addition and subtraction)
   * Print statements (e.g., print(a);)
   * Control flow structures (if statements):
     + Conditional statements (e.g., if (a > 0) { print(a); })
2. **Define the BNF Grammar**: Create a BNF grammar that describes the syntax of the language you designed.
3. **Implementation Requirements**:
   * Write a Python program that uses the BNF grammar to parse a set of predefined code snippets.
   * The program should output whether each snippet adheres to the defined syntax and, if not, provide a meaningful error message.
4. **Code Snippets**: Provide at least 5 code snippets illustrating valid syntax and 3 snippets illustrating common syntax errors.
5. **Documentation**: Each student must submit a report that includes:
   * An explanation of the BNF grammar they created.
   * A brief description of how their parser works, including any libraries or techniques they used.
   * Example outputs from running their parser on the provided code snippets.

**Deliverables:**

* Python code implementing the parser.
* A well-documented BNF grammar specification.
* A report detailing the grammar design and implementation findings.

**Grading Criteria:**

* Correctness of the BNF grammar (25%)
* Functionality and correctness of the parser implementation (50%)
* Clarity and thoroughness of documentation (25%)

# Rubric

|  |  |  |  |
| --- | --- | --- | --- |
| Criterion | Needs Improvement  [0-3] | Good  [4-6] | Excellent  [7-10] |
| Coding & Report  (5%) | The code does not work properly, many requirements are missing, and the report does not detail the grammar design. | The code works partially and implements some of the requirements; the report shows some testing scenarios. | The code works fully and as expected per the requirements and the report shows all testing scenarios and full detail the grammar design. |