ALGONQUIN COLLEGE

ASSIGNMENT 1.1 – NEW COMPILER SPECIFICATION

General View

Due Date: prior or on **Sep 18th 2021 (midnight)**

- **2nd Due date** (until 25th Sep) - 50% off.

Earnings: 5% of your course grade

Purpose: Define a new language (adaptation from SOFIA[1])

Submission: Individual or by teams (only 2 students from same Lab Session).

Tip 1: About teams
As mentioned in Submission standard, teams are suggested due to the possibility of peer-review.

GENERAL VIEW

- This is the first task in Compilers: *You need to choose a language and define a subset that is close from SOFIA implementation*.
- During this term, the rules to be adopted are mentioned here:
  - Each language must have a **city name** (in my case, I am using "Sofia" – the Capital of Bulgaria).
- This language is supposed to have at least what a minimal programming language can support:
  - Element 0 – Block comments and keywords

[1] **SOFIA** is the language adapted from PLATYPUS (originally created by Prof. Svillen Ranev to Compilers Course at Algonquin College). As also mentioned in this specification, the name is appropriate because it is matching with the capital city of Bulgaria – homeland of Prof. Svillen.

- Element 1 – Datatypes: Numbers (both integer and real / float point) and text (string).
- Element 2 – Commands: Assignment, selection, interaction, input (read) and output (write).
- Element 3 – Modules: Subroutines / functions.

❖ **PART I:** What you need to do is (user view):

- Propose basic definitions for this language that could be understood by a new learner, describing this language.
- Give examples of a basic Hello World in this language.

❖ **PART II:** Under the perspective of software engineering:

- Describe the main ideas about how to create this language.
- Imagine what are the challenges that you need face when you are defining a language.

Task 1 – Defining a new language – User Reference (3 marks)

Here are some ideas that you need to explore when defining a new language:

❖ Give a name and specification:

➢ In our case, the name is supposed to be given by the CITYNAME:

- For instance, if you're the name of my language is "Rio[2]", my team / identification is also "Rio".
- In the case of multiple words, I suggest using **CamelCase**. For instance, if the language is "Rio de Janeiro", use "RioDeJaneiro", etc.

➢ Create a brief introduction for this language, mentioning:

- A pseudo "genealogy" of this language. For instance, if it is Python-like, etc.
- Define an extension and specify rules for it. For instance, "**.rio**" (from Rio) or ".**rdj**" (from "RioDeJaneiro").

Note 1: The extension files

*The extension name for files is very important because all practical assignments will require to include **input files** to be tested.*

- Imagine the ADVANTAGES of using this new language (suppose that you are trying to compete with Microsoft, Google, etc.). How did you sell it?

➢ Simple example:

- Create a "**Hello World"** program in this language, considering:

[2] **Rio** (Rio de Janeiro) is a famous city from Brazil, homeland from Prof. Paulo.

- Comments.
- A “main” function.
- An output statement.

❖ Suppose that you are creating a **User Manual**. Define:

- Element 0 – Basic elements:
  - How comments are made.
  - What are the keywords from this language.
- Element 1 – How to express integer, real numbers and string datatypes.
  - Imagine the number of bytes for each one and, therefore, the range.
- Element 2 – Define the most common statements:
  - Assignment / attribution: How attribution can be made.
    - Question: Do you imagine casting operations?
  - Selection: How “if” clauses are defined.
    - Additional idea: Imagine switch case syntax.
  - Interaction: How loops can be defined.
    - Exploring this idea: Do you have different alternatives such as for-loop, while-do loop or repeat-until iterations?
  - Input: How to enter data to this language.
  - Output: How to print something.
- Element 3 – The way you can provide subroutines / functions.
  - Think about syntax for receiving parameters;
  - How to return values:

Task 2 –Architectural Aspects (2 marks)

❖ GENERAL VIEW: Now, you need to be able to identify **implementation aspects**:

- How this language should be built (using C language).
  - For instance, about datatype, if your numbers are using integers of 2-bytes, which kind of ANSI C data you should use.
- What could be your strategy to identify the elements of the language (that we will call “tokens”).
  - For instance, if you have a keyword “**println**” for outputs, what is the strategy to detect inside a piece of code.

        - Think how to be able to identify constants (that we will call "literals"). Ex: "34".
    - Since you need to have iteration / conditionals, how to define a scope for these elements.
        - For instance, in C language, you have "{" and "}" as "delimiters".
        - Additionally, how to define subroutines scope.
- CHALLENGES: Being leader of the implementation team, what are your main concerns about the implementation.
    - Think about how to be able to allocate memory, how to evaluate the datatypes, etc.
        - Give examples about difficult that you can imagine when implementing this language.
- SOLUTION: Now, as a leader, how to solve some of these problems?
    - For instance, algorithms and structures (ex: tables, linked lists) that you can use to solve them.
    - Additional ideas that you have about implementation.

Tip 1: Use your creativity.

To create a language, the balance between user-friendly and complexity is a continuous trade-off. Use practical philosophies such as: KISS ("Keep It Simple, Sir"). SQL language is a very good example of that.

Evaluation

- Please read the Assignment **Submission Standard and Marking Guide** (at "Assignments > Standards" section).
    - The submission must follow the course submission standards. You will find the Assignment Submission Standard as well as the Assignment Marking Guide (**CST8152_ASSAMG.pdf)** for the Compilers course on the Brightspace.
- **About Plagiarism**: Any Internet reference not cited or mentioned is consider a plagiarism. Avoid losing marks because of that.
- Check the **A11 Marking Guide** to check if your language is well defined.

Submission Details

- **Digital Submission: Compress** into a **zip** file with the document (DOC / DOCX / PDF) that you created specifying the language.
- Upload the zip file on Brightspace. The file must be submitted prior or on the due date as indicated in the assignment.

- ***IMPORTANT NOTE:*** The name of the file must be **Your Last Name** followed by the last three digits of your student number followed by your lab **section number**. For example: **A11_Sousa123.zip**.
  - If you are working in teams, please, include also your partner. For instance, something like: **A11_Sousa123_Cormier456_s10.zip**.
  - **Remember:** Only students from the **same section** can constitute a specific team.
- ***IMPORTANT NOTE:*** **Assignments will not be marked if there are not source files in the digital submission.** Assignments could be late, but the lateness will affect negatively your mark: see the Course Outline and the Marking Guide. All assignments must be successfully completed to receive credit for the course, even if the assignments are late.

Marking Rubric

Maximum Deduction (%)	Deduction Event
	Severe Errors:
100	Late submission (after 1 week due date)
100	Plagiarism detection (not referenced)
up to 100	**Does not comply with the assignment specifications**
up to 30	Missing language definition topics (name, genealogy, example)
up to 50	Missing user language elements (datatypes, instructions, etc)
up to 20	Missing architectural aspects
up to 10	Other minor errors
up to 10	Bonus: interesting ideas, enhancements, discretionary points.
Final Mark	**Formula:** 5*((100- ∑ penalties + bonus)/100), max score 5%**.**

File update: Sep 2nd 2021.

Good luck with A11!