

MINISTRY OF NATIONAL EDUCATION



TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA

FACULTY OF AUTOMATION AND COMPUTER SCIENCE
COMPUTER SCIENCE DEPARTMENT

SueC - An Editor and Interpreter for Pseudocode

LICENSE THESIS

Graduate: Mihai PÎȚU
Supervisor: dr. eng. Emil Ștefan CHIFU

2019

MINISTRY OF NATIONAL EDUCATION



TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA

**FACULTY OF AUTOMATION AND COMPUTER SCIENCE
COMPUTER SCIENCE DEPARTMENT**

DEAN,
Prof. dr. eng. Liviu MICLEA

HEAD OF DEPARTMENT,
Prof. dr. eng. Rodica POTOLEA

Graduate: **Mihai PÎȚU**

SueC - An Editor and Interpreter for Pseudocode

1. **Project proposal:** *Short description of the license thesis and initial data*
2. **Project contents:** *(enumerate the main component parts) Presentation page, advisor's evaluation, title of chapter 1, title of chapter 2, ..., title of chapter n, bibliography, appendices.*
3. **Place of documentation:** *Example:* Technical University of Cluj-Napoca, Computer Science Department
4. **Consultants:**
5. **Date of issue of the proposal:** November 1, 2017
6. **Date of delivery:** February 18, 2019 *(the date when the document is submitted)*

Graduate: _____

Supervisor: _____

MINISTRY OF NATIONAL EDUCATION



TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA

FACULTY OF AUTOMATION AND COMPUTER SCIENCE
COMPUTER SCIENCE DEPARTMENT

**Declarație pe proprie răspundere privind
autenticitatea lucrării de licență**

Subsemnatul(a)

_____, legiti-
mat(ă) cu _____ seria _____ nr. _____
CNP _____, autorul lucrării _____

elaborată în vederea susținerii examenului de finalizare a studiilor de licență la Facul-
tatea de Automatică și Calculatoare, Specializarea _____
din cadrul Universității Tehnice din Cluj-Napoca, sesiunea _____ a an-
ului universitar _____, declar pe proprie răspundere, că această lucrare este
rezultatul propriei activități intelectuale, pe baza cercetărilor mele și pe baza informațiilor
obținute din surse care au fost citate, în textul lucrării și în bibliografie.

Declar, că această lucrare nu conține porțiuni plagiate, iar sursele bibliografice au
fost folosite cu respectarea legislației române și a convențiilor internaționale privind drep-
turile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în fața unei alte
comisii de examen de licență.

În cazul constatării ulterioare a unor declarații false, voi suporta sancțiunile admin-
istrative, respectiv, *anularea examenului de licență*.

Data

Nume, Prenume

Semnătura

De citit înainte (această pagină se va elimina din versiunea finală):

1. Cele trei pagini anterioare (foaie de capăt, foaie sumar, declarație) se vor lista pe foi separate (nu față-verso), fiind incluse în lucrarea listată. Foaia de sumar (a doua) necesită semnătura absolventului, respectiv a coordonatorului. Pe declarație se trece data când se predă lucrarea la secretarii de comisie.
2. Pe foaia de capăt, se va trece corect titulatura cadrului didactic îndrumător, în engleză (consultați pagina de unde ați descărcat acest document pentru lista cadrelor didactice cu titlaturile lor).
3. Documentul curent **nu** a fost creat în MS Office. E posibil să fie mici diferențe de formatare.
4. Cuprinsul începe pe pagina nouă, impară (dacă se face listare față-verso), prima pagină din capitolul *Introducere* tot așa, fiind numerotată cu 1.
5. E recomandat să vizualizați acest document și în timpul editării lucrării.
6. Fiecare capitol începe pe pagină nouă.
7. Folosiți stilurile predefinite (Headings, Figure, Table, Normal, etc.)
8. Marginile la pagini nu se modifică.
9. Respectați restul instrucțiunilor din fiecare capitol.

Contents

Chapter 1	Introduction - Project Context	11
Chapter 2	Project Objectives and Specifications	12
Chapter 3	Bibliographic research	13
3.1	Lexical Analyzer	13
3.2	Syntactic Analyzer	13
3.3	C Programming Language	13
3.4	Python Programming Language	13
Chapter 4	Analysis and Theoretical Foundation	14
4.1	Title	14
4.2	Other title	14
Chapter 5	Detailed Design and Implementation	15
Chapter 6	Testing and Validation	16
6.1	Title	16
6.2	Other title	16
Chapter 7	User's manual	17
7.1	Title	17
7.2	Other title	17
Chapter 8	Conclusions	18
8.1	Title	18
8.2	Other title	18
	Bibliography	19
	Appendix A Relevant code	20
	Appendix B Other relevant information (demonstrations, etc.)	21

Chapter 1

Introduction - Project Context

Computer science and programming is taught in schools around Romania for at least 30 years, especially in high schools, but also in secondary schools, starting with the 5th grade. Before introducing directly to a programming language, many teachers use a pseudocode language which serves as a mean of understanding programming concepts in a more universal manner, bringing it closer to the natural spoken language.

Nowadays, in Romanian schools, teachers use C or Pascal programming languages and software related to these in order to develop the students' ideas regarding computer science and programming basics and paradigms.

The purpose of this project is to create an easier way of learning programming concepts for students who are new into this domain. This will serve as a fresh renewal of software used in schools today, as older tools such as Code::Blocks and/or Free Pascal are still used in schools and programming contests.

SueC is the name of the editor which creates, edits and compiles files which represent pseudocode files. This editor will work also like any other editors, providing some error-checking mechanisms and returning the result after compiling a pseudocode file.

Chapter 2

Project Objectives and Specifications

As the title of the project suggests - "An Editor and Interpreter for Pseudocode" - this is an application which will serve as an educational tool for using the pseudocode as a programming language. The main objectives of this project are:

- Developing an user-friendly application which handles the main file handling operations and communicating with the compiler of the pseudocode source files.
- Creating an understandable programming language that resembles the pseudocode used by teachers in schools and/or universities. For a technical point of view, the pseudocode will be created like any other programming languages, having similar elements to existing ones that are used nowadays, but also with specific structural elements bringing it closer to the natural language.
- Developing a compiler for this programming language by defining a lexical and syntactic analyzer respectively. These analyzers contain the set of rules that apply to the programming language.

Chapter 3

Bibliographic research

For this project, my research done was focused on the main components and technologies included in the project

- 1. Lexical Analyzer**
- 2. Syntactic Analyzer**
- 3. C Programming Language**
- 4. Python Programming Language**

3.1 Lexical Analyzer

A lexical analyzer is an analyzer which processes the source code, being the first step for compiling. It takes the modified source code from language preprocessors

3.2 Syntactic Analyzer

3.3 C Programming Language

3.4 Python Programming Language

Chapter 4

Analysis and Theoretical Foundation

Together with the next chapter takes about 60% of the whole paper

The purpose of this chapter is to explain the operating principles of the implemented application. Here you write about your solution from a theory standpoint - i.e. you explain it and you demonstrate its theoretical properties/value, e.g.:

- used or proposed algorithms
- used protocols
- abstract models
- logic explanations/arguments concerning the chosen solution
- logic and functional structure of the application, etc.

YOU DO NOT write about implementation.

YOU DO NOT copy/paste info on technologies from various sources and others alike, which do not pertain to your project.

4.1 Title

4.2 Other title

Chapter 5

Detailed Design and Implementation

Together with the previous chapter takes about 60% of the paper.

The purpose of this chapter is to document the developed application such a way that it can be maintained and developed later. A reader should be able (from what you have written here) to identify the main functions of the application.

The chapter should contain (but not limited to):

- a general application sketch/scheme,
- a description of every component implemented, at module level,
- class diagrams, important classes and methods from key classes.

Chapter 6

Testing and Validation

About 5% of the paper

6.1 Title

6.2 Other title

Chapter 7

User's manual

In the installation description section you should detail the hardware and software resources needed for installing and running the application, and a step by step description of how your application can be deployed/installed. An administrator should be able to perform the installation/deployment based on your instructions.

In the user manual section you describe how to use the application from the point of view of a user with no inside technical information; this should be done with screen shots and a stepwise explanation of the interaction. Based on user's manual, a person should be able to use your product.

7.1 Title

7.2 Other title

Chapter 8

Conclusions

About. 5% of the whole
Here your write:

- a summary of your contributions/achievements,
- a critical analysis of the achieved results,
- a description of the possibilities of improving/further development.

8.1 Title

8.2 Other title

Bibliography

- [1] E. Bellucci, A. Lodder, and J. Zelezniak, “Integrating artificial intelligence, argumentation and game theory to develop an online dispute resolution environment.” in *16th International Conference on Tools with Artificial Intelligence*, 2004, pp. 749–754.
- [2] G. Antoniou, T. Skylogiannis, A. Bikakis, M. Doerr, and N. Bassiliades, “Dr-brokering: A semantic brokering system.” *Knowledge-Based Systems*, vol. 20, no. 1, pp. 61–72, 2007.
- [3] S. J. Russell, P. Norvig, J. F. Canny, J. M. Malik, and D. D. Edwards, *Artificial intelligence: a modern approach*. Prentice hall Englewood Cliffs, 1995, vol. 2.
- [4] W. Strunk, Jr. and E. B. White, *The Elements of Style*, 3rd ed. Macmillan, 1979.

Appendix A

Relevant code

```
/** Maps are easy to use in Scala. */
object Maps {
  val colors = Map("red" -> 0xFF0000,
                   "turquoise" -> 0x00FFFF,
                   "black" -> 0x000000,
                   "orange" -> 0xFF8040,
                   "brown" -> 0x804000)

  def main(args: Array[String]) {
    for (name <- args) println(
      colors.get(name) match {
        case Some(code) =>
          name + " has code: " + code
        case None =>
          "Unknown color: " + name
      }
    )
  }
}
```

Appendix B

Other relevant information
(demonstrations, etc.)

Appendix C

Published papers