**Universidad Nacional**

**Autónoma de México**

**Facultad de Ingeniería**

**And**

**The alphaX team introduces**

Logotipo

Descripción generada automáticamente

AlphaX Compiler

**Developers:**

**Flores Constantino Diego.**

**Rojas Castañeda Karen Arleth.**

**Compilers.**

**Supervisor: Ing. Norberto Jesús Ortigoza Márquez.**

**Project Charter**

**Overview**

This document is presented as part of documentation, here is where the purpose, members role and high-level requirements are portrayed. This will help the stakeholders to identify the most important parts of the project development, which are based on the following

1. Proposal and creation of innovative solutions
2. Establish and development of the test plan (including base tests and additional test suites).
3. Establish the point where the project's phase II is complete.

**Purpose**

The objective of this project is to develop a C programming language compiler. In this second delivery the compiler must include unary operators in the already developed process.

**Members Role**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Department | Role | Responsibilities |
| *Diego Flores Constantino* | Direction | Project Manager | General Management |
| *Karen Arleth Rojas Castañeda* | Version Management | System Integrator/Analyst | Integrator |
| *Diego Flores Constantino* | Planning and Architecture | System Architect | Architecture Design |
| *Karen Arleth Rojas Castañeda* | Tests | Tester | Test Plan and Test suites |
| *Diego Flores/Arleth Rojas* | Development | Developer | Develop Analysis |

**Project Details**

|  |  |
| --- | --- |
| Project Type | Course project Phase II (Unary Operators) |
| Project Name | AlphaX Compiler |
| Start Date | On the 14th of December |
| Deadline | On the 1st of January |
| Sponsor/Supervisor/Client | Norberto Ortigoza Márquez |
| Project Manager | Diego Flores Constantino |
| Signature | Compilers |

**Project high-level requirements (in detail for a proper design)**

|  |  |
| --- | --- |
| Identifier | Requirement |
| R - 1 | Compile a program written in C programming language. |
| R - 2 | The program must contain a single function called main. |
| R - 3 | The function main shall return a decimal integer number (with or without a unary operator, depending on code). |
| R - 3.1 | The returned decimal integer number could be variable between a decimal range |
| R - 4 | The scanner (Parser) should set up a complete token list collected from the C source code; furthermore, add a relational identifier to make more evident about the token’s position. (Such as the code line where it is). |
| R - 5 | Parser must be able to identify the syntax problems that might appear in code; here is where the common code typing mistakes are analyzed and where the code is (usually) rejected if necessary. |
| R – 6 | The code development of the compiler must be in Elixir programming language |
| R - 6.1 | The development technique must be done to build a matching pattern for the creation of an Abstract Syntax Tree (AST) |
| R – 7 | Assembly code generation must be created under AT&T assembly syntax; for GNU purposes |
| R - 7.1 | Assembler code must be written under 64-bits set of instructions |

**Support Resources/Documents**

* Sandler, N. (2017). Writing a C Compiler, Part 1. <https://norasandler.com/2017/11/29/Write-a-Compiler.html>
* (N. A.) (2006) AT & T Assembly Syntax. <https://csiflabs.cs.ucdavis.edu/~ssdavis/50/att-syntax.htm>