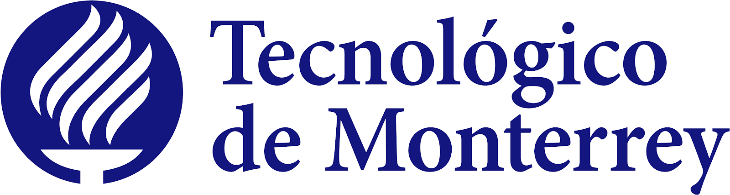
**Instituto Tecnológico y de Estudios Superiores**

**de Monterrey, Campus Monterrey**

****

**Diseño de Compiladores**

**Profesores**

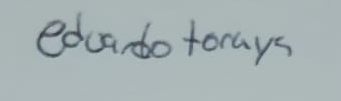
**Elda Quiroga González**

**Héctor Ceballos Cancino**

***Documentación Final***

***Forever Alone***

**Eduardo Alejandro Toraya Solís A00819785**



**Fecha de entrega: 2 de Junio del 2020.**

**Índice**

**a) DESCRIPCIÓN DEL PROYECTO**

**a.1) Propósito, Objetivos y Alcance del Proyecto.**

**El Propósito de este proyecto es desarrollar un compilador en Python utilizando como herramienta lexx and yacc para parsear de tal forma que cumpla con los requerimientos funcionales y de diseño propuestos por el documento de ForeverAlone.**

**a.2) Análisis de Requerimientos y Casos de Uso generales.**

**Requerimientos:**

* **Debe manejar tipos de dato int, float y carácter.**
* **Debe contar con una función principal y funciones adicionales opcionales.**
* **Debe poder declarar variables, obtener variables del usuario y mostrar variables al usuario.**
* **Debe manejar arreglos de una dimensión en sus operaciones.**
* **Debe contar con operaciones aritméticas lineales.**
* **Debe contar con llamadas a funciones.**
* **Debe contar con manejo de estatutos cíclicos y condicionales.**
* **Maneja expresiones como +, -, \*, /, &, |, <, <= , == , <>, >=, y > para floats e integers y para caracteres sólo las lógicas.**
* **Las funciones podrán ser de tipo void(no regresan valor) o de tipos carácter, float o integer que regresarán valor de su mismo tipo.**
* **Permite recursividad**

**Casos de uso generales:**

* **Definir/Ejecutar programa.**
* **Declarar variables individuales o de arreglo.**
* **Llamar variables individuales o de arreglo**
* **Iniciar/terminar estatutos no secuenciales (while, for, if).**
* **Llamar a funciones para su ejecución.**
* **Asignar valores.**
* **Mostar/obtener valores para usuario**
* **Escribir comentarios.**
* **Escribir strings.**
* **Iniciar o detener recursión.**

**a.3) Descripción de los principales test cases.**

**T.C.1: En el testCase del file MVTestOperaciones, se prueba la consistencia de las operaciones de suma, resta, multiplicación, operadores de comparación, asignación, lectura de valor de usuario y escritura.**

**T.C.2: MVTestCiclos, se prueba la consistencia de operaciones no lineales. Se maneja el while, el if/else, el for y estos mismos anidados. Mientras se realizan operaciones aritméticas, escritura y asignación.**

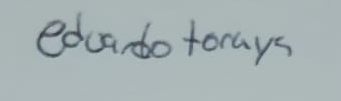
**T.C.3: MVTestFunciones, se prueba la consistencia y el correcto funcionamiento de la memoria en funciones void, con retorno y recursivas, así como llamadas recursivas y con llamada interna para confirmar el no colisionamiento de espacio de dirección. Este test case incluye la producción de un factorial y de la serie de Fibonacci hasta un valor dado por el usuario.**

**T.C.4: MVTestArrays, se prueba la consistencia de asignación de memoria de los arreglos, en el cual se prueba la asignación, lectura y operaciones aritméticas correctas de los arreglos. Se revisa el funcionamiento de arreglos anidados ejemplo: A[A[3]] y se prueba un sorteo de un arreglo definido en el mismo código combinando con características de ciclos.**

**a.4) Descripcion del PROCESO.**

**El proceso para llevar a cabo este compilador, fue el desarrollo por partes de acuerdo a los avances del mismo. Se comenzó con escoger una propuesta de proyecto de las ofrecidas por los profesores, en este caso se desarrolló el lexer, de ahí el código para operaciones lineales, seguido de operaciones no lineales, seguido de funciones y finalmente se desarrollaron los arreglos en el parser. Posteriormente, se desarrollo una máquina virtual que pudiese leer los cuadruplos generados en un texto objeto por el parser y los interpreta en Python.**

**Reflexión principal, Se tiene un mejor entendimiento del proceso interno que se puede llevar para poder traducir el código de un programador en instrucciones realizables por una computadora.**

**Firma:** 

**b) DESCRIPCIÓN DEL LENGUAJE:**

**b.1) Nombre del Lenguaje: ForeverAlone**

**b.2) Descripción genérica de las principales características del lenguaje.**

**Este lenguaje utiliza palabras en español para la sus palabras reservadadas de operaciones. Comienza definiendo el nombre del programa seguido de su declaración de variables globales, posteriormente es opcional crear las funciones las cuales deberán ser declaradas con la palabra clave de la misma, seguida de su tipo y de los parámetros que tendrá la misma, similar a la instanciación de funciones en c++. Posteriormente se cuenta con un bloque Principal() de código y debe colocarse hasta el final del código para correcta ejecución, con las funciones auxiliares arriba del mismo.**