HULK

Rafael A. Sanchez Martinez
Universidad de la Habana
Grupo: C-113

Figure 1: Havana University Language for Kompilers

1 Introduccion

Bienvenidos al manual de usuario de HULK, el lenguaje de programacion de la Universidad de la Habana. Este manual esta dividido en 5 secciones, cada una de ellas dedicada a una parte del interprete. En la primera seccion se explica como instalar el interprete, en la segunda seccion se explica como utilizar el interprete, en la tercera seccion se explica como utilizar el analizador lexico, en la cuarta seccion se explica como utilizar el analizador sintactico y en la quinta seccion se explica como utilizar el evaluador de expresiones.

1.1 Comienzo

- Para comenzar a utilizar el interprete, se debe descargar el codigo fuente del interprete desde el repositorio de github del proyecto. Una vez descargado el codigo fuente, se debe ejecutar el proyecto usando el comando 'dotnet run' desde una terminal abierta en la carpeta del proyecto.
- Otra opcion es ejecutar el script el cual se encargara de todo.(Ver informe previo en `https://github.com/ARKye03/mini_kompiler.git`)

1.2 Resumen

HULK (Havana University Language for Kompilers) es un lenguaje de programación didáctico, con seguridad de tipos, orientado a objetos e incremental, diseñado para el curso Introducción a los Compiladores de la carrera de Informática de la Universidad de La Habana.

Un simple 'Hola mundo' en HULK se ve así:
print('Hola mundo');

A vista de pájaro, HULK es un lenguaje de programación orientado a objetos, con herencia simple, polimorfismo y encapsulación a nivel de clase. Además, en HULK es posible definir funciones globales fuera del alcance de todas las clases. También es posible definir una única expresión global que constituya el punto de entrada al programa.

La mayoría de las construcciones sintácticas en HULK son expresiones, incluidas instrucciones condicionales y ciclos. HULK es un lenguaje de tipado estático con inferencia de tipos opcional, lo que significa que algunas (o todas) las partes de un programa se pueden anotar con tipos y el compilador verificará la coherencia de todas las operaciones.

Pero, en este caso, al ser un proyecto para estudiantes de 1er año, pues se simplifica un poco

2 Analizador Lexico

```
private object power()
{
    var left = primary();

    while (true)
    {
        var token = lexer.get_next_token();

        if (token.type != TokenType.Operator || token.value != "^")
        {
            lexer.unget_token(token);
            return left;
        }
        var right = primary();
        left = BinaryOperation(left, token, right);
    }
}
```

3 Introduction

This is the report of my compiler project. The compiler is written in C++ and

4 Introduction

This is the report of my compiler project. The compiler is written in C++ and

5 Introduction

This is the report of my compiler project. The compiler is written in C++ and