The BigDuck Programming Language

Jair Antonio Bautista Loranca

November 16, 2021

Contents

Part I

Description and Technical Documentation

Chapter 1

Project

1.1 Introduction

1.1.1 Purpose

This document describes the software developement process, technical documentation, and user manual, for the final project of the Compiler Design course. Which consists on the design and implementation of a programming language and a virtual machine.

1.1.2 Scope

The programming language developed is specified to be a compiled imperative, with support of modules and structured types. Additionaly it is required to develop a virtual machine capable to execute the output code generated by the compiler.

1.2 Software Requirements

1.2.1 Analysis

Based on the specifications and recomendations given by the teachers, the following requirements were defined as necessary for the successful development of this project.

Functional requirements

1. The programming language must aim to solve a domain specific problem.
2. The compiler must support scoped and global variables.
3. The compiler must support numeric data types.
4. The compiler must support conditional statements.
5. The compiler must support loop statements.
6. The compiler must support modules.
7. The compiler must support recursion.
8. The compiler must support structured types.
9. The compiler must report compile-time errors.
10. The compiler must generate intermidiate code.
11. The virtual machine must execute generated code.
12. The virtual machine must manage program memory.
13. The virtual machine report run-time errors.

Non-Functional requirements

1. The language grammar must be non-ambiguous.
2. The compiler shall use a scanner and parser generation tool.
3. The compiler must be efficient in time and memory.
4. The virtual machine must be efficient in time and memory.

1.2.2 Test Cases

1.3 Software Developement Process

1.3.1 Developement Process Description

The project was developed throught weekly sprints, were each sprint consisted in developing a major feature needed for the programming language compilation or execution. It must be said that despite having an suggested schedule, the reality is that the project went a little bit different from this schedule. This is because some features were prioritize to be implemented first.

1.3.2 Weekly Log

Week no.	Date	Description
0	Sep 20	Proposal Developement
1	Sep 27	Lexic and syntax analysis
2	Oct 4	Symbol table and sematic cube
3	Oct 11	Expressions compilation
4	Oct 18	Conditionals compilation
5	Oct 25	Loops compilation
6	Nov 1	Procedures compilation
7	Nov 8	Sematic analysis, memory map, and virtual machine
8	Nov 15	Structured types compilation, and application specific code

1.3.3 Git Commitments

Chapter 2

Language

2.1 General Overview

2.1.1 Language Name

The for the programming language was given as a small joke, one of the homeworks on the semester was to develop a scanner and parser for a small language called LittleDuck. Therefore BigDuck could be considered as the next step for the previous mentioned language, even though there is no similarities but the name between these languages.

Additionaly to this, I really like birds and use them as a naming scheme for my devices, thus the decision seemed natural and adecuate.

2.1.2 Main Features Description

BigDuck is language aimed for the developement of mathematical models commonly used in Machine-Learning and Data Science. Therefore this language includes integer and floating point arithmetic, vector and matrix operations, and some basic utilities for reading and writing `.csv` iles. All this to make it easier for the user to work within the Machine-Learning and Data Science fields.

2.2 Language Errors

2.2.1 Compile-Time Errors

2.2.2 Run-Time Errors

Chapter 3

Compiler

3.1 Development Environment

3.2 Lexical Analisis

3.3 Syntactical Analisis

3.4 IR Code and Semantic Analisis

3.4.1 Operation Code

3.4.2 Virtual Addresses

3.4.3 Syntax Diagrams

3.4.4 Semantic and IR Generation Actions

3.4.5 Semantic Consideration Table

3.5 Memory Management

Chapter 4

Virtual Machine

4.1 Development Environment

4.2 Memory Management

4.2.1 Architecture

4.2.2 Data Structures

4.2.3 Virtual Address Translation

Chapter 5

Execution Evidence

5.1 Test Cases

5.1.1 Test Implementation

5.1.2 IR Code Output

5.1.3 Execution Output

Chapter 6

Code Documentation

6.1 Modules Description

Part II

User Manual

Chapter 7

Quick Reference

7.1 Enviroment

7.2 Variables

7.3 Statements

7.3.1 Assignments

7.3.2 Arithmetic Expressions

7.3.3 Operator Hierarchy

7.4 Conditional Statements

7.5 Loop Statements

7.5.1 Infinite Loop

7.5.2 While Loop

7.5.3 For Loop

7.5.4 Control Flow Statements

7.6 Procedures

7.7 Tensorial Types

7.8 Built-in Procedures