PLT 4115 LRM: **JaTesté**

Andrew Grant
amg2215@columbia.edu

Jemma Losh
jal2285@columbia.edu

Jared Weiss
jbw2140@columbia.edu

Jake Weissman
jdw2159@columbia.edu

March 2, 2016

1 Introduction

The goal of JaTesté is to design a language that promotes good coding practices - mainly as it relates to testing. JaTesté will require the user to explicitly define test cases for any function that is written in order to compile and execute code. This will ensure that no code goes untested and will increase the overall quality of programmer code written in our language. The user will be required to provide some test cases for their code, and the language will also generate some important test cases for their code as well. JaTesté is mostly a functional language with a syntax quite similar to C. The details of our language usage is provided in the rest of the document.

2 Lexical Conventions

This chapter will describe how input code will be processed and how tokens will be generated.

2.1 Identifiers

Identifiers are used to name a variable, a function, or other types of data. An identifier can include all letters, digits, and the underscore character. An identifier must start with either a letter or an underscore - it cannot start with a digit. Capital letters will be treated differently from lower case letters.

2.2 Keywords

Keywords are a set of words that serve a specific purpose in our language and may not be used by the programmer for any other reason. The list of keywords the language recognizes and reserves is as follows:

```
int char float struct if else for while break continue with test using func return
```

2.3 Constants

2.3.1 Integer Constants

2.3.2 Character Constants

2.3.3 Real Number Constants

2.3.4 String Constants

2.4 Operators

2.5 Separators

2.6 White Space

3 Data Types

3.1 Primitives

3.2 Structures

3.2.1 Defining Structures

3.2.2 Initializing Structures

3.2.3 Accessing Structure Members

3.3 Arrays

3.3.1 Defining Arrays

3.3.2 Initializing Arrays

3.3.3 Accessing Array Elements

3.3.4 Multidimensional Arrays

3.3.5 Arrays of Structures

Text

4 Expressions and Operators

4.1 Expressions

4.2 Assignment Operators

4.3 Incrementing and Decrementing

4.4 Arithmetic Operators

4.5 Comparison Operators

4.6 Logical Operators

4.7 Comma Operator

4.8 Operator Precedence

4.9 Order of Evaluation

5 Statements

5.1 Expression Statements

5.2 If Statement

5.3 While Statement

5.4 For Statement

5.5 Code Blocks

5.6 Break and Continue

5.7 Return Statement

6 Functions

6.1 Function Declarations

6.2 Calling Functions

6.3 Function Parameters

6.4 Main Function

6.5 Recursive Functions

7 Program Structure and Scope

7.1 Program Structure

7.2 Scope

8 A Sample Program