BILKENT UNIVERSITY

CS315 PROGRAMMING LANGUAGES
PROJECT 2 REPORT

SPRING 2019-2020

PIRULI THE PROGRAMMING LANGUAGE

ZEYNEP KORKUNÇ STUDENT ID: 21702571
OLCAY AKMAN STUDENT ID: 21702671

SECTION: 1

APRIL 8, 2020

1. BNF Description of Piruli

```
<program> ::= <statements>
<statements> ::= <statement>;
          | <statement> ; <statements>
<statement> ::= <type> <identifier>
          | <type> <identifier> = <variable>
          | <identifier> = <variable>
          | <identifier> = <expression>
          | <if_stmt>
          |<ifelse_stmt>
          | <loop>
          | <function>
          | <function_call>
          | <read_from_file>
          | <write_on_file>
          | <expression>
          | <delete_set>
          | <output_stmt>
          | <input_stmt>
          | <add_stmt>
<read_from_file> ::= readFile(<string>)
<write_on_file> ::= <string>.writeFile(<param>)
<output_stmt> ::= piwrite(<f_c_params>)
<input_stmt> ::= piread()
<add_stmt> ::=<set> add <variable>
          | <identifier> add <variable>
<function> ::= <type> <identifier> <function_body>
          | <identifier> <function_body>
<function_body> ::= ( <params> ) { <statements> }
<function_call> ::= <identifier>(<f_c_params>)
<params> ::=
          | <type> <identifier>
          | <type> <identifier>, <params>
<f_c_params> ::=
          | <param>
          | <param> , <f_c_params>
<param> ::=
          | <variables>
          | <variable>
          | <identifier>
<type> ::= int
          | string
          | set
<identifier> ::= <chars>
          | <chars><number>
<variables> ::=  <variable>, <variables>
```

```
<variable> ::= <number>
        | <set>
        | <set> <set_op> <set>
        | <identifier> <set_op> <identifier>
        | <string>
        | <function_call>
        | <input_stmt>
<numbers> ::= <number>
        | <number>, <numbers>
<number> ::= <digits>
<digits> ::= <digit>
        | <digit><digits>
<digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<set> ::= {<set_elements>}
<set_elements> ::= <set_element>
        | <set_element> , <set_elements>
<set_element> ::=
        | <variables>
        | <identifier>
<string> ::= “”
        | “<chars>”
<chars> ::= <char>
        | <char><chars>
<char> ::= a | b | c | d | e | f | g | h | i | j | k | l | m | n
        | o | p | q | r | s | t | u | v | w | x | y | z
        | A | B | C | D | E | F | G | H | I | J | K | L | M
        | N | O | P | Q | R | S | T | U | V | W | X | Y | Z
        | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<set_op> ::= union
        | diff
        | intersect
        | cartesian
<if_stmt> ::= if ( <condition_stmts> ) { <statements> }
<ifelse_stmt> ::= <if_stmt> else{ <statements> }
 <condition_stmts> ::= <condition_stmt>
        | <condition_stmt> <logic_op> <condition_stmts>
<condition_stmt> ::= <variable> membof <set>
        | <identifier> membof <set>
        | <set> <condition_op> <set>
        | <identifier><condition_op><identifier>
        | <number> <math_logic_op> <number>
        | <identifier> <math_logic_op> <number>
        | <identifier> <math_logic_op> <identifier>
<delete_set> ::= delete <identifier>
<condition_op> ::= subset
        | superset
        | ==
        | !=
<logic_op> ::= and
        | or
```

```
<math_logic_op> ::= <
        | >
        | ==
        | !=
        | >=
        | <=
<loop> ::= <while_loop>
        | <for_loop>
<while_loop> ::= while ( <condition_stmts> ) { <statements> }
<for_loop> ::= for (int <identifier> = <number>; <condition_stmt>; <math_op><identifier>)
                                                                  { <statements> }
        | for <identifier> in <identifier> { <statements> }
<math_op> ::= +
        | -
        | *
        | /
<expression> ::= <expression> + <term>
        | <expression> - <term>
        | <term>
<term> ::= <term> * <exp>
        | <term> / <exp>
        | <exp>
<exp> ::= (<expression>)
        | <number>
```

2. Language Constructs

2.1 Integers, Strings and Sets

In Piruli, there are three main variable types: string, integer and set. Integer and string are defined similar to the Java language, and set is simply the programming language representation of a mathematical set. The Piruli language allows sets to contain any of the three variables given above . Simple mathematical operations (namely addition, division, multiplication, and subtraction) are covered for integers. Strings are given between two double quotation marks. Sets can contain any variable, separated by commas. All set elements in a set are enclosed by curly braces at the beginning and the end of a set.

2.2 Set Operations

Sets can be joined together using the "union" keyword, their difference can be found by using the "diff" keyword, and their intersection can be found by using the "intersect" keyword. Most of the mathematical set instructions are covered by Piruli: however, mathematical set properties such as the set builder notation (e.g. $A=\{x$: x is a vowel in English alphabet} [1]). Piruli also allows the user to check whether a given variable is a member of a given set, using the keyword "membof". Similarly, it is possible to check whether a given set is a subset or a superset of a second given set by using the keywords "subset" and "superset" respectively.

2.3 Statements

There are different types of statements defined in Piruli. These are: if/else statements, condition statements, assignment statements, declaration statements, functions, while loops, for loops.

2.3.1 If/Else Statements

If/else statements are defined similar to Java. An if statement can stand alone by itself without an else statement following it, or if desired, an else statement can follow it, too. If statements are executed or not by evaluating the condition given in them, which are called condition statements. If those condition statements are evaluated to be true, the statement(s) given inside the if statement is executed, otherwise not.

2.3.2 Condition Statements

Condition statements have two states: they are either true of false. The concept of a Boolean, as in Java, is not defined in Piruli. Nevertheless, conditions in Piruli are either true or false, and these values are determined by the compiler. Condition statements allow the users to

- compare integers to integers (less than, less than or equal to, equal to, not equal to, greater than, greater than or equal to),
- determine whether a set is a subset of another given set, • determine whether a set is a superset of another given set,
- determine whether two given sets are identical or not.

Condition statements in Piruli are used in if statements, for loops and while loops.

2.3.3 Assignment Statements

Assignment statements in Piruli allows variables to be assigned values. It is done by using the "=" operator. A given variable can be assigned a value by typing its identifier on the left hand side of the "=" operator, and the value to be assigned to it will be given in the right hand side of it.

2.3.4 Declaration Statements

A declaration statement in Piruli allows a variable of a given type to have a certain identifier as its name. A declaration statement can be combined with an assignment statement to form a single statement, or it can stand alone. A declaration statement alone looks like <type> <identifier>; whereas a declaration statement combined with an assignment statement looks like <type> <identifier> = <variable>;.

2.3.5 Functions

Functions in Piruli are defined implicitly. That is, functions are not similar to Python, where each function declaration begins with "def". A function begins with its return type, followed by its identifier, which is then followed by braces in which parameters to be passed to the function are given, if any. The function body is thus followed by this declaration, given in curly braces.

2.3.6 While Loops

While loops are defined similarly to Java. Followed by the "while" keyword, the user of Piruli gives a condition statement in braces, and this is followed by curly braces in which is the statement(s) to be run is given, which will be executed as long as the condition given in the while statement is true.

2.3.7 For Loops

For loops in Piruli are similar to Java as well. Followed by the "for" keyword, three different statements are given inside braces, which are then followed by curly braces in which the statement(s) to be run will be given. The three different statements given after the "for" keyword are:

- the integer declaration

- the condition statement
- incrementing/decrementing the integer declared

3. Language Evaluation

The readability, writability and reliability of a programming language depends on several properties of it. While creating Piruli, we considered these criteria and hopefully created a nice balance of all these factors while allowing the users to use/create mathematical sets in a programming language environment.

3.1 Readability

The readability of a programming language depends on several of its properties. These, along with examples of how they are applied in Piruli are given below:

- Simplicity: Piruli is simple to read, as many operations are written as keywords almost the same as in English.
- Orthogonality: there are three main types in Piruli, integer, string and set. Having a small number of variables allows orthogonality.
- Data types: the data types on Piruli are selected to be only the necessary types for performing set operations.
- Syntax design: special words in Piruli are simple to understand, they are very close to English words. The statement designs are also similar to the English language which allows it to be easily read.

3.2 Writability

A program is writable if it can be used to create programs chosen for a problem domain. [2] The writability of a programming language depends on several of its properties. These, along with examples of how they are applied in Piruli are given below:

- Simplicity and orthogonality: the small number of different constructs means that when writing/declaring new variables, there is not a ton of choices to choose from, which increases the writability of Piruli.
- Expressivity: In Piruli, some mathematical operations are simplified, to allow for easier writing in for loops. For instance, the integer declared in the for loop can be incremented by writing “+5” which increments the integer by 5 on every iteration.

3.3 Reliability

A program is reliable if it performs to its specifications under all conditions. [2] The reliability of a programming language depends on several of its properties. These, along with examples of how they are applied in Piruli are given below:

- Type Checking: Piruli allows users to define three types, and allows sets to contain integers, strings, sets, or all. Type errors can be simply detected in Piruli thanks to the small scope of variable types.
- Exception handling: Although Piruli does not allow extensive exception handling like C++ or Java, it will, with the help of the Yacc to be written, handle certain exceptions to intercept run-time errors.
- Aliasing: Piruli allows a single memory address to be accessed by zero, one or more distinct identifiers.

4. References

[1] “Discrete Mathematics - Sets.” *Tutorialspoint*, www.tutorialspoint.com/discrete_mathematics/discrete_mathematics_sets.htm.

[2] Sebesta, Robert W., et al. *Concepts of Programming Languages*. Pearson, 2016.