JavaCC Assignment: TianleLang

Student : Tianle Shu , A00268357

October 29, 2023

Lecturer: Dr.Thiago Braga
Lecturer: Dr.Niall Murray

1 Idea And Background

This assignment implements a compiler (parser) for a new computer language (TianleLang) done with javacc. This language is simple and syntax like Python has the following constraints[4]: Variable(lowercase permit) Assign Expression, Recognise Double type number and did arithmetic operation, Display Function (show the result and String words to the user), Implemented the simple structure for a loop that can iterate a variable and print something according to the requirement(only works positive Integer), Comment function implemented[6](Comment Symbol is '#').

2 Challenges

During the assignment, I encountered some challenges:
♦ 1. It is how to let the declare dynamic, not a fixed number of variables.
♦ 2. how to display values of variable
♦ 3. loop function implements

For the ♦ 1 and ♦ 2 points, I searched for some information on the internet, and this website[2] was very helpful in implementing the declaration and print function. Through reading, I learned that I can access variable values through the HashMap.
For the Print Function, I can get the value from HashMap through the key.
For the loop function, I got a logic idea from Python range syntax[5][1]: **range(start, stop, steps)**
start - Optional. An integer number is to start. The default is 0.
stop - Required. An integer number is to stop (not included).
steps - Optional. An integer number specifies the incrementation. Default is 1.
I use ArrayList to store the value of the loop result, and then store this List in the Hashmap(loopMap)[3] In order to ensure that the display of the result of the variable does not conflict, the key can only exist uniquely in these two HashMaps. This needs to be judged when declaring the loop statement.

```java
// store the variable's value
public HashMap < String, Double > hashMap = new HashMap < String, Double > ();
// use for loop function
public HashMap < String, ArrayList < Integer > > loopMap = new HashMap < String, ArrayList < Integer > > ();
// use Array list to store the loop values
public ArrayList < Integer > ele = new ArrayList < > ();
public static void main(String args []) throws ParseException
{
  TianleLang parser = new TianleLang(System.in);
      System.out.println("Welcom to Tianle-Language  ^_^");
      parser.Start();
}
```

```java
void Loop() :
{
  Token t;
  String s1;
  String s2;
  String key;
}
{
  < LOOP >
  t = < ID >
  {
    //check the key in the loopMap already or in the variableMap already
    //if not it will do next step otherwise throw ParseException
    if (!loopMap.containsKey(t.image) && !hashMap.containsKey(t.image))
    {
      ele = new ArrayList < > ();
          key = String.valueOf(t.image);
          loopMap.put(key, ele);
    }
    else
        //System.err.println("already defined");
        throw new ParseException(t + " is already defined");
  }
  < IN >
  Indexing(t.image, key)
  < COLON >
```

To print the values of the variable, I use HashMap[3] to store items in "key/value" pairs, the key is our variable name, value is the number that the user assigned. So I created a GetValue function that will return the value of the key in these 2 maps.

```
String GetValue(String t) :
{
  String value;
  String i = "";
}
{
  {

    // check the value of variable in the variableMap
    value = String.valueOf(hashMap.get(t));
    // if the values of variable in the loopMap
    if (loopMap.containsKey(t))
    {
      ele = loopMap.get(t);
      for (int e : ele)
      {
        i = i + String.valueOf(e) + "\n";
      }
      value = i; //Debug - System.out.println("keyiii-"+value);
     }

    }
  {
    return value;
  }
}
```

Then call the GetValue method in the print function. can get the result.

```
< PRINT >
    < OPEN_PAR >
     (
        t = < ID >
      {
        try
        {
              target = GetValue(String.valueOf(t.image));
              if(target == null || target.equals("null") ){
                  flag = true;
                  target = "Invalid Syntax";
              }
            }
            catch (Exception e)
            {
              flag = true;
              target = "Invalid Syntax";
            }
      }
    | target = PrintString()
    | target = Expression()
```

3 Limitations

The limitation of this language, it does not provide a method part.

The loop Function can only handle positive integers.Cannot handle different types of variables like Negative Numbers, Strings, etc.

4 Testing Methodology and Explanation

The Testing Methodology is a Manual Test, there are 3 different photos to show in 3 different scenarios.
The Figure1 shows the Assign Expression syntax and print string and The results of simple four arithmetic operations. If print a not assigned variable will show **Invalid Syntax** red error message.

TianleLang_Input1 — 已编辑

```
# Declare variable use 'var' or 'VAR'
# Display the result - print();
# End of the code line remember use ';'
# loop format: loop i in (startNum, endNum, steps):

# declare variables
VAR num1=5.5;
VAR num2=20;

#print not declare variable will give a err info
print(a);

# sum =25.5
var sum = num1+num2;
PRINT('~~~sum~~~');
print(sum);

# minus =14.5
var minus = num2-num1;
PRINT('~~~minus~~~');
print(minus);

# multi =110.0
var multi = num2*num1;
PRINT('~~~multiply~~~');
print(multi);

# minus =0.275
var divide = num1/num2;
PRINT('~~~divide~~~');
```

```
MacBook-Pro:~/TUS/TeamProject/Javacc_labs/src/assignment1 $quella java assignment1.TianleLang < TianleLang_Input1
Welcom to Tianle-Language  ^_^
Invalid Syntax
~~~sum~~~
25.5
~~~minus~~~
14.5
~~~multiply~~~
110.0
~~~divide~~~
0.275
```

Figure 1: To show the result of simple operations

The below's picture shows the Assign Expression syntax and print string and The results of complex four arithmetic operations and a simple loop

TianleLang_Input2

```
# welcome to Tianle-Lang, declare variable use 'var' or 'VAR'
# display the words or result - print() or PRINT();
# end of the code line remember use ';'
# loop format: loop i in(startNum, endNum, steps):

# declare variables, do Arithmetic operations
VAR num1=9+9/9;
var num2=9.9+10*3.3;
VAR num3 = 9-(9+1);

# display the result
print('~~~num1~~~');
print(num1);

print('~~~num2~~~');
print(num2);

print('~~~num3~~~');
print(num3);

# print String words
print('hello_wold');

#print calculate result
print('~~~result~~~');
print(9+0*7/2);

print('~~~loop_i_rsult~~~');
# loop function
loop i in (10):
```

```
MacBook-Pro:~/TUS/TeamProject/Javacc_labs/src/assignment1 $quella java assignment1.TianleLang < TianleLang_Input2
Welcom to Tianle-Language  ^_^
~~~num1~~~
10.0
~~~num2~~~
42.9
~~~num3~~~
-1.0
hello_wold
~~~result~~~
9.0
~~~loop_i_rsult~~~
0
1
2
3
4
5
6
7
8
9
10
```

Below's picture is the third scenario to show the 3 types of loop functions.
The LOOP Statement: **loop variable in (start,stop, steps):**
If there is only one number in the brackets, then the default is to start from zero and increment by one until that number is reached.
If there are two numbers in parentheses, then by default it starts with the first number and increments by one, and ends with the second number.
If there are three numbers in parentheses, start with the first number and increment to the third number until the second number is reached.

TianleLang_Input3

```
# welcome to Tianle-Lang, declare variable use 'var' or 'VAR'
# display the result - print(); or PRINT();
# end of the code line remember use ';'
# loop format: loop i in (startNum, endNum, steps):


# loop section
# loop statement: loop variable in (start,stop,steps):
# start - Optional. An integer number is to start. Default is 0
# stop  - Required. An integer number is to stop (not included).
# step  - Optional. An integer number specifying the incrementation. Default is 1

PRINT('~~~~~~~~~~~i~~~~~~~~~~');
loop i in (10):
print(i);

print('~~~~~~~~~~~i2~~~~~~~~~~');
loop i2 in (5,10):
print(i2);



print('~~~~~~~~~~~i3~~~~~~~~~~');
loop i3 in (10,30,5):
print(i3);
```

```
MacBook-Pro ~/TUS/TeamProject/Javacc_labs/src/assignment1 $quella java assignment1.TianleLang < TianleLang_Input3
Welcom to Tianle-Language  ^_^
~~~~~~~~~~~i~~~~~~~~~~
0
1
2
3
4
5
6
7
8
9
10

~~~~~~~~~~~i2~~~~~~~~~~
5
6
7
8
9
10

~~~~~~~~~~~i3~~~~~~~~~~
10
15
20
25
30
```

References

[1] https://github.com/JameelSi/PythonFor/blob/main/PythonFor.jj.

[2] Display and declare. https://blog.csdn.net/qq$_1$5089775/$article/details/123601486?spm = 1001.2101.3001.6650.15utm_medium = distribute.pc_relevant.none - task - blog - 2$

[3] Hashmap. https://www.w3schools.com/java/java$_hashmap.asp$.

[4] Javacc tutor. https://www.cnblogs.com/suhaha/p/11733487.html.

[5] A mathematical theory of communication. https://www.w3schools.com/python/ref$_func_range.asp$.

[6] JavaCC. https://javacc.github.io/javacc/faq.htmlquestion-4.5.