Language structure support

This interpreter supports:

- Basic structure, integer, plus string and Boolean variables
- For loops and nested for loops
- Detection of runtime errors

Output of Sample Programs

1. `prog1.txt`

```
a=40
b=5400
```

2. `prog2.txt`

```
A=155
B=10
C=30
D=14
E=290
```

3. `prog3.txt`

```
A=17
a=19
Num=1
```

4. `prog4.txt`

```
a=31
b=55
a="helloworld"
b="world"
```

5. `prog5.txt`

```
A="Happy Birthday !"
```

6. `prog6.txt`

```
A="a b cw x y z"
```

7. `prog7.txt`

```
A="a b c47"
```

8. `prog8.txt`

```
A=TRUE
A=FALSE
A=FALSE
A="TRUE"
```

9. `prog9.txt`

```
A=1032
B=41
```

10. `prog10.txt`

```
A="ABABABABABABABABABABABABABABAB"
```

11. `prog11.txt`

```
A=19532250
B=4882810
```

12. `prog12.txt`

```
A="4444444444444444"
```

13. `prog13.txt`

```
A=1038
B=154
```

14. `prog14.txt`

```
A=50752
```

15. `prog15.txt`

```
RUNTIME ERROR: line 4
```

16. `prog16.txt`

```
RUNTIME ERROR: line 2
```

17. `prog17.txt`

```
RUNTIME ERROR: line 2
```

18. `prog18.txt`

```
RUNTIME ERROR: line 2
```

Prog19 - prog24 used unimplemented structures, so they are not listed here.

Benchmark

The L program used for benchmark is as follows,

```
FOR 400000 { A = 0 ; B = 1 ; FOR 15 { tmp = A ; tmp += B ; A = B ; B = tmp ; } }
```

This L program is executed by the interpreter written in Java.

And the equivalent C++ program,

```cpp
#include <iostream>

using namespace std;

int main() {
    int a;
    for (int i = 0; i < 400000; i++) {
        a = 0;
        int b = 1;
        for (int j = 0; j < 15; j++) {
            int tmp = a;
            tmp += b;
            a = b;
            b = tmp;
        }
    }
}
```

The compiler for the C++ program is g++ with no optimization.

For reference, we have also tested two interpreted languages, Python and JavaScript.

Python program:

```python
for i in range(0, 400000):
    a = 0
    b = 1
    for j in range(0, 15):
        tmp = a
        tmp += b
        a = b
        b = tmp
```

JavaScript program:

```
let a
for (let i = 0; i < 400000; i++) {
    a = 0;
    let b = 1;
    for (let j = 0; j < 15; j++) {
        let tmp = a;
        tmp += b;
        a = b;
        b = tmp;
    }
}
```

The execution environment for JavaScript is Node.js, which is powered by V8 JavaScript engine.

Result

Both programs are tested 5 times and the average execution times are taken.

Language	L	C++	Python	JavaScript
Execution time	33.42s	0.02s	1.16s	0.04s

As shown above, all interpreted languages are slower than unoptimized C++, among which JavaScript performs best, thanks to the powerful V8 engine. L is the slowest, the primary reasons are as follows,

1. The L interpreter is run by JVM, which interprets the `.class` file and thus introduces extra overhead.
2. The L interpreter directly operates on the source language. All production-level interpreters introduce some kind of intermediate representations for source languages, which allow faster execution.