LIVA
A Lite Version of Java

Shanqi Lu, Jiafei Song, Zihan Jiao, Yanan Zhang, Hyoyoon Kate Kim
Introduction

What we are looking for from LIVA

Simple
It is designed to let programmers, who are familiar with class-based languages, to feel comfortable with developing common algorithms like GCD. It is lite in the sense that it maintains some but not all features in Java.

Object-Oriented
It has a Java-like syntax and supports object-oriented paradigm and inheritance.

Portable
LIVA is a portable language and compiled down to LLVM.
GitHub
Version Control

VMware
Make Development Consistent

Ubuntu
Operating System

OS X
Operating System
Project Schedule

36 Days’ Project !!!

- **Proposal**
  - Brainstorm for proposed language design
  - July 6 - July 11

- **Hello World Milestone**
  - Milestone with main function call and print feature
  - July 12 - July 20

- **Language Reference Manual**
  - Syntax and scope determined
  - July 21 - Aug 1

- **Final Work**
  - All features and test suite completed
  - Aug 2 - Aug 11
## Features

<table>
<thead>
<tr>
<th>LIVA</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Types</td>
<td>✓</td>
</tr>
<tr>
<td>Operators</td>
<td>✓</td>
</tr>
<tr>
<td>Functions</td>
<td>✓</td>
</tr>
<tr>
<td>Classes</td>
<td>✓</td>
</tr>
<tr>
<td>Inheritance</td>
<td>✓</td>
</tr>
<tr>
<td>Arrays</td>
<td>✓</td>
</tr>
<tr>
<td>Loops</td>
<td>✓</td>
</tr>
<tr>
<td>If-Else</td>
<td>✓</td>
</tr>
<tr>
<td>Standard Library</td>
<td>✗</td>
</tr>
</tbody>
</table>
### Syntax

#### Type

```java
int a = 1;
float b = 2.2;
char c = '3';
boolean d = true;

int[] x = new int[10];
float[] y = new float[10];
y[1] = 1.0;
```

#### Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>+</code></td>
<td>PLUS</td>
</tr>
<tr>
<td><code>-</code></td>
<td>MINUS</td>
</tr>
<tr>
<td><code>*</code></td>
<td>TIMES</td>
</tr>
<tr>
<td><code>/</code></td>
<td>DIVIDE</td>
</tr>
<tr>
<td><code>%</code></td>
<td>MODULO</td>
</tr>
<tr>
<td><code>=</code></td>
<td>ASSIGN</td>
</tr>
<tr>
<td><code>==</code></td>
<td>EQ</td>
</tr>
<tr>
<td><code>!=</code></td>
<td>NEQ</td>
</tr>
<tr>
<td><code>&lt;</code></td>
<td>LT</td>
</tr>
<tr>
<td><code>&lt;=</code></td>
<td>LEQ</td>
</tr>
<tr>
<td><code>&gt;</code></td>
<td>GT</td>
</tr>
<tr>
<td><code>&gt;=</code></td>
<td>GEQ</td>
</tr>
<tr>
<td><code>&amp;</code></td>
<td>AND</td>
</tr>
<tr>
<td>`</td>
<td>`</td>
</tr>
<tr>
<td><code>!</code></td>
<td>NOT</td>
</tr>
</tbody>
</table>

#### Comments

```java
/*This is a Liva comment!*/
```
Control Flow

```
int i;
for (i=0; i<10; i=i+1){
    print(i);
}
while (i > 0){
    print(i);
    i = i - 1;
}
if (true){
    print(42);
} else {
    print(8);
}
```

Class

```
class myclass{
    int calc (int x, int y){
        int z;
        z = x + y;
        return (z);
    }
}
class subcls extends myclass{
    int b;
    constructor(int a){
        this.b = a;
    }
    int calc (int x, int y){
        int z;
        z = x - y;
        return (z);
    }
}
```

Object

```
class test {
    void main(){
        int x = 9;
        int y = 6;
        int z;
        class myclass obj = new myclass();
        z = obj.calc(x, y);
        print("z=",z);
    }
}
```
AST

PROGRAM

Classes

Fields

Constructors

Methods

Statements

Expressions
Testing

Test procedure
- Compile and run test-if1.liva
- Compare the output with test-if1.out
- If they are the same, done! Otherwise, find out the problems.

120 test files
- All passed!

Testall.sh
- Based on the test script of MicroC
- Test all files in a single command
Testing
Unit Test

Successful & Unsuccessful

Small pieces
Testing

Integration Test
Small tests integrated into larger one
Lessons Learned

- **Time Management**
  - Start the project early

- **Cooperation**
  - Teamwork and integration

- **Communication**
  - Avoid doing the same work

- **Software Tools**
  - Efficiency improvement
DEMO