Dice

“Java, but worse”

David Watkins	Khaled Atef	Emily Chen	Phil Schiffrin
Project Manager	Testing Wizard	Language Guru	System Hacker

1

Introduction

Dice in a nutshell

LET’S REVIEW SOME CONCEPTS

LLVM

Our backend compiled to LL IR code. This allows for cross-platform code without recompiling the original source

Object-Oriented

The ability to define objects, define their methods, and define private/public scope

Multi-Level Inheritance

A Person object can have as many children as it wants, but only one parent

Arrays

Single dimensional arrays as well as arrays defined with expressions placed in the elements

Useful Standard Library

File, String, and Integer classes defined as closely to the Java Library as possible

Useful Error Messages

Every single error that the compiler encounters will print a rich error message for the user

2

Project Management Timeline

Dice is a big language

5921 Lines of Code

Our compiler was a huge undertaking

109 Issues Closed

526 reduce/reduce conflicts resolved

47 Custom Emojis

Total success!

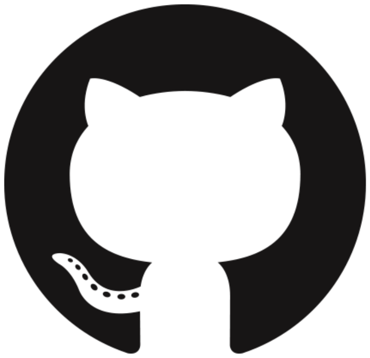
3rd Party Software Solutions

Slack

The bread and butter for group communications.

Github

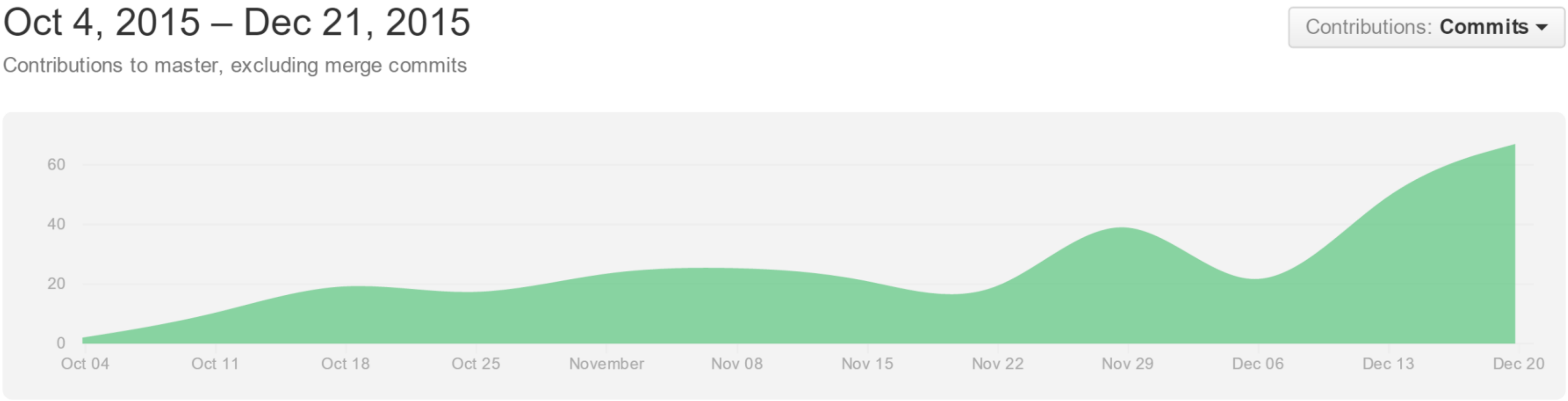
The best solution for our version control needs. Branches were key.



Ubuntu

Utilizing an Ubuntu VM stack gave us much needed consistency.

Git History



Oct 4, 2015 – Dec 21, 2015

Contributions to master, excluding merge commits

Contributions: **Commits**

3

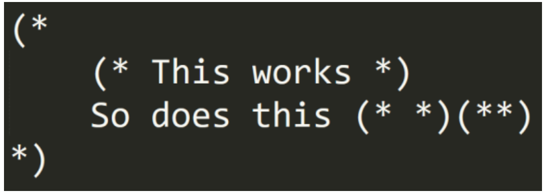
So how do you speak it?

Dice is a BIG language

Syntax

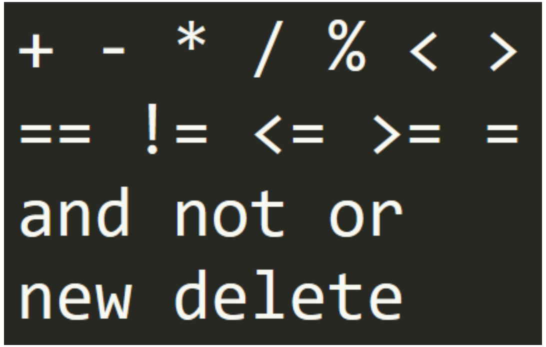
Basics

Comments



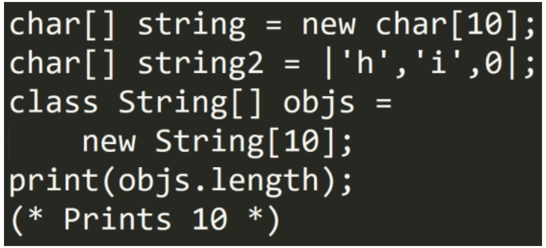
```
(*
    (* This works *)
    So does this (* *)(**)
*)
```

Operators



```
+ - * / % < >
== != <= >= =
and not or
new delete
```

Arrays

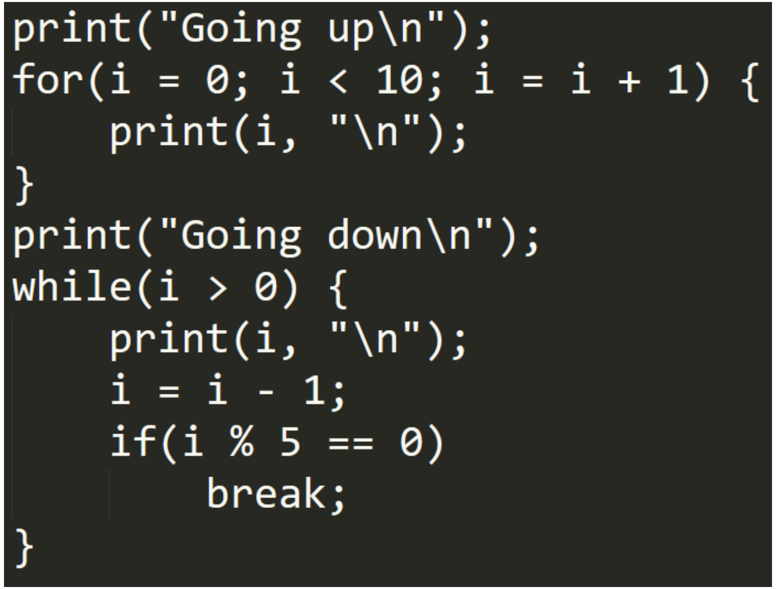


```
char[] string = new char[10];
char[] string2 = |'h','i',0|;
class String[] objs =
    new String[10];
print(objs.length);
(* Prints 10 *)
```

Syntax

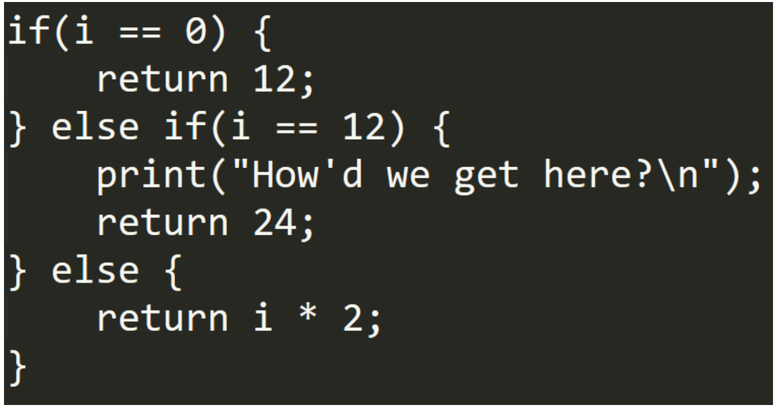
Statements

Loops



```
print("Going up\n");
for(i = 0; i < 10; i = i + 1) {
    print(i, "\n");
}
print("Going down\n");
while(i > 0) {
    print(i, "\n");
    i = i - 1;
    if(i % 5 == 0)
        break;
}
```

if-else/else-if

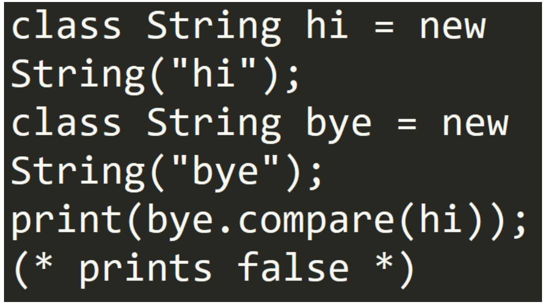


```
if(i == 0) {
    return 12;
} else if(i == 12) {
    print("How'd we get here?\n");
    return 24;
} else {
    return i * 2;
}
```

Syntax

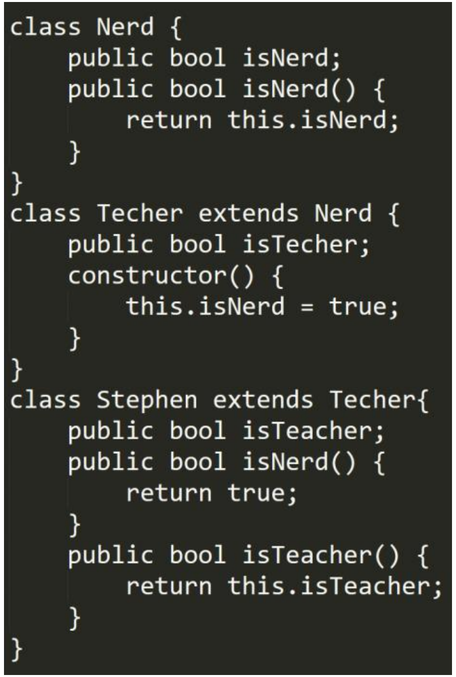
Objects

Objects



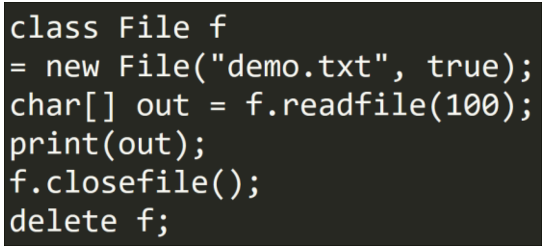
```
class String hi = new
String("hi");
class String bye = new
String("bye");
print(bye.compare(hi));
(* prints false *)
```

Inheritance



```
class Nerd {
    public bool isNerd;
    public bool isNerd() {
        return this.isNerd;
    }
}
class Techer extends Nerd {
    public bool isTecher;
    constructor() {
        this.isNerd = true;
    }
}
class Stephen extends Techer{
    public bool isTeacher;
    public bool isNerd() {
        return true;
    }
    public bool isTeacher() {
        return this.isTeacher;
    }
}
```

File IO



```
class File f
= new File("demo.txt", true);
char[] out = f.readfile(100);
print(out);
f.closefile();
delete f;
```

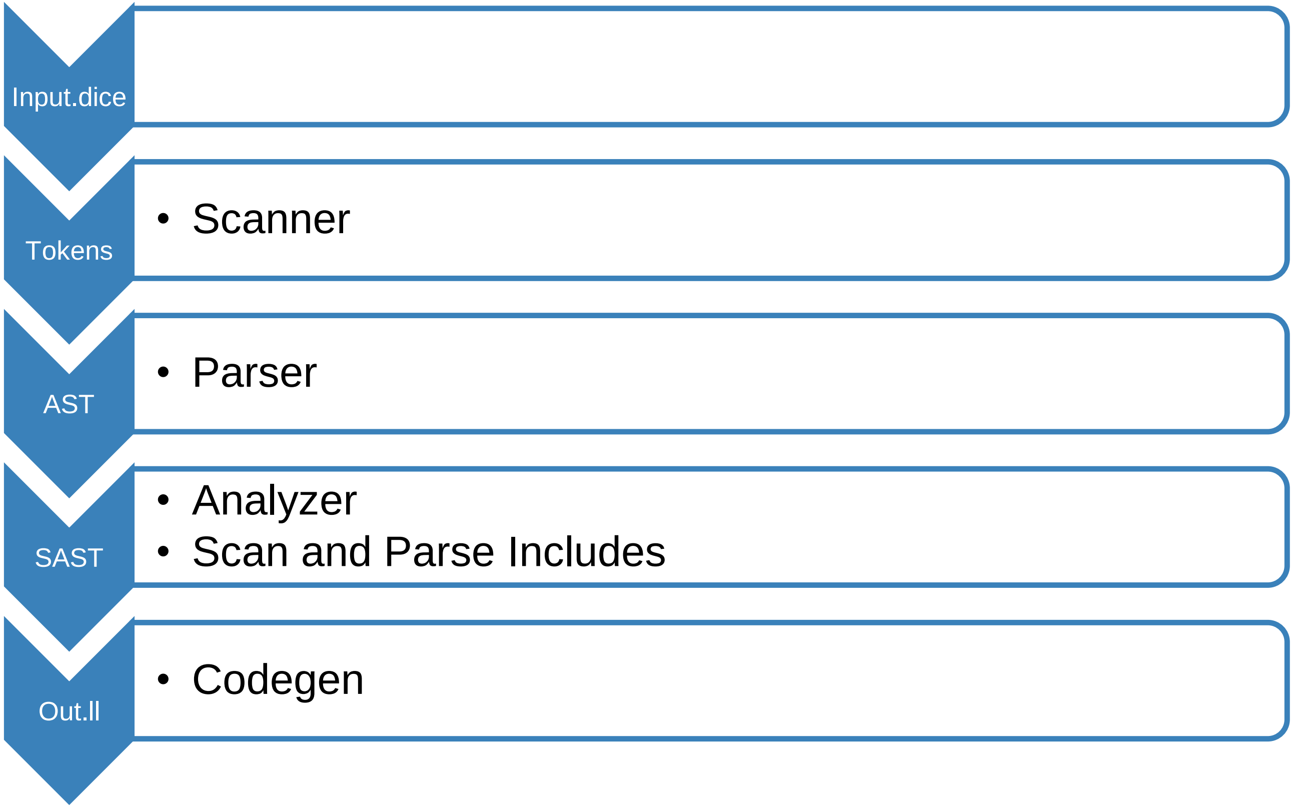
4

How it’s made

Not featured on Discovery Channel

Implementation

Design



Input.dice

Tokens

- Scanner

AST

- Parser

SAST

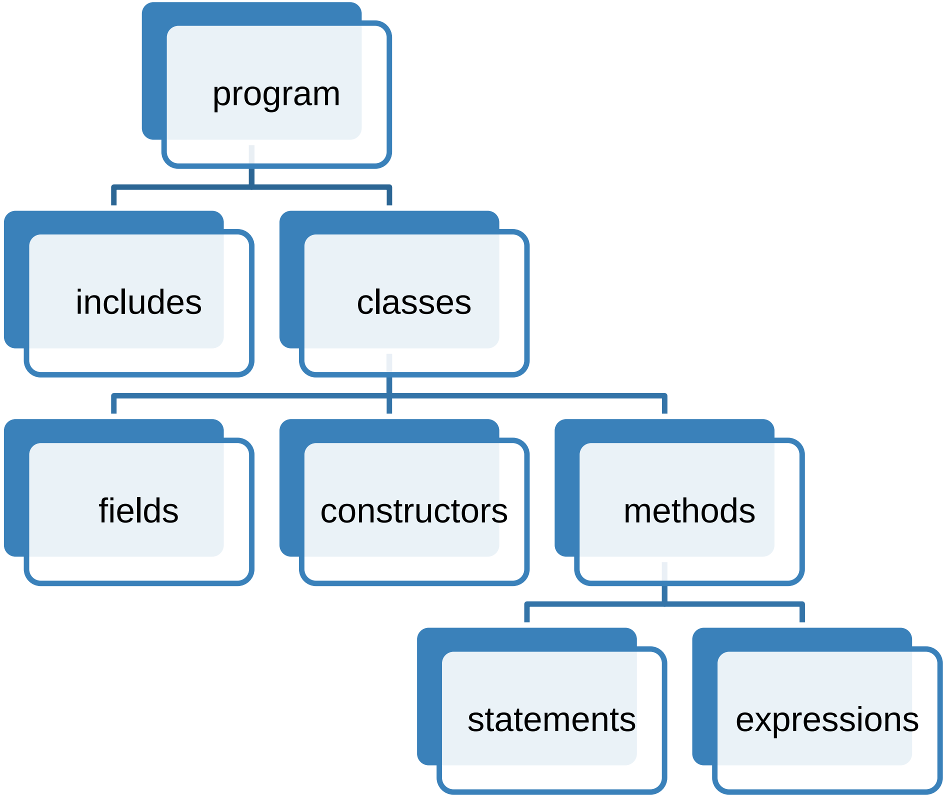
- Analyzer
- Scan and Parse Includes

Out.ll

- Codegen

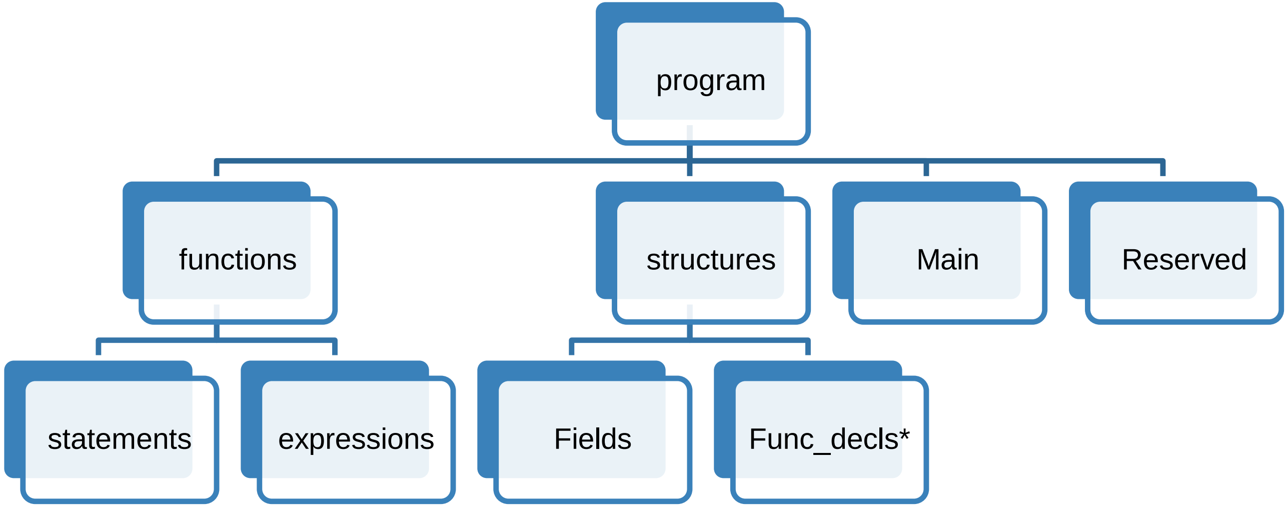
Implementation

AST



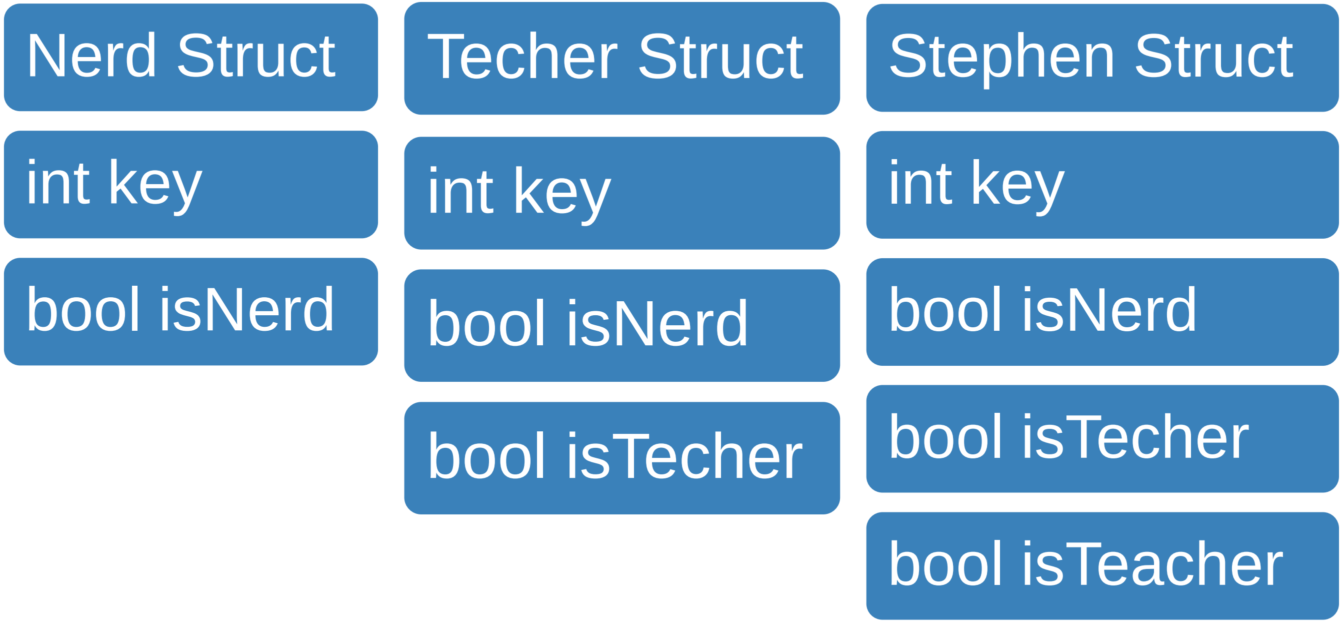
Implementation

SAST



Implementation

Structs with Inheritance



Nerd Struct	Techer Struct	Stephen Struct
int key	int key	int key
bool isNerd	bool isNerd	bool isNerd
	bool isTecher	bool isTecher
		bool isTeacher

Implementation

Virtual Function Table

Class Indexes→	Nerd	Techer	Stephen
Function Indexes→	isNerd:Nerd	isNerd:Nerd	isNerd:Stephen
			isTeacher:Stephen

- Lookup the void* using function index and class index
- Casts function pointer to correct function pointer
- Makes function calls with original parameters

Implementation

Standard Library

String Class

Includes methods for comparison of strings and obtaining the length

Integer Class

Get the string representation of integers

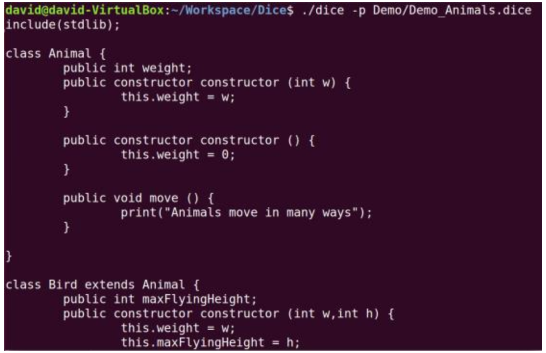
File Class

Open, close, read, and write to a file

Implementation

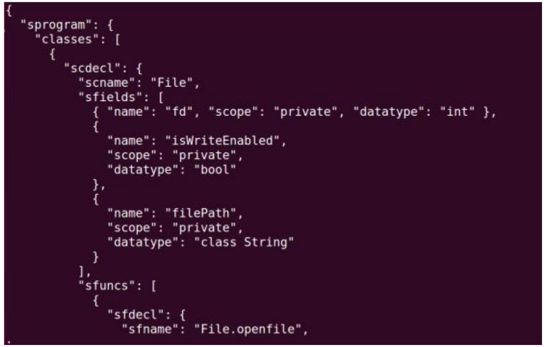
Compiler Flags

Pretty Printing



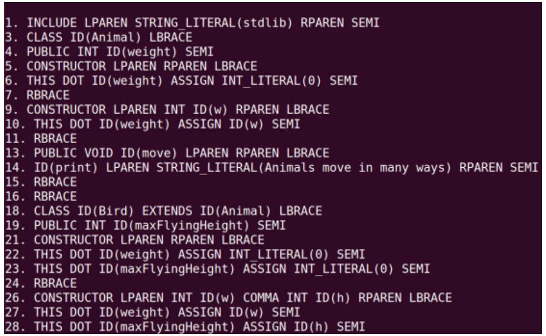
```
david@david-VirtualBox:~/Workspace/Dice$ ./dice -p Demo/Demo_Animals.dice
include(stdlib);

class Animal {
        public int weight;
        public constructor constructor (int w) {
                this.weight = w;
        }

        public constructor constructor () {
                this.weight = 0;
        }

        public void move () {
                print("Animals move in many ways");
        }

}

class Bird extends Animal {
        public int maxFlyingHeight;
        public constructor constructor (int w,int h) {
                this.weight = w;
                this.maxFlyingHeight = h;
```

SAST/AST in JSON



```
{
  "sprogram": {
    "classes": [
      {
        "scdecl": {
          "scname": "File",
          "sfields": [
            { "name": "fd", "scope": "private", "datatype": "int" },
            {
              "name": "isWriteEnabled",
              "scope": "private",
              "datatype": "bool"
            },
            {
              "name": "filePath",
              "scope": "private",
              "datatype": "class String"
            }
          ],
          "sfuncs": [
            {
              "sfdecl": {
                "sfname": "File.openfile",
```

Tokens



```
1. INCLUDE LPAREN STRING_LITERAL(stdlib) RPAREN SEMI
3. CLASS ID(Animal) LBRACE
4. PUBLIC INT ID(weight) SEMI
5. CONSTRUCTOR LPAREN RPAREN LBRACE
6. THIS DOT ID(weight) ASSIGN INT_LITERAL(0) SEMI
7. RBRACE
9. CONSTRUCTOR LPAREN INT ID(w) RPAREN LBRACE
10. THIS DOT ID(weight) ASSIGN ID(w) SEMI
11. RBRACE
13. PUBLIC VOID ID(move) LPAREN RPAREN LBRACE
14. ID(print) LPAREN STRING_LITERAL(Animals move in many ways) RPAREN SEMI
15. RBRACE
16. RBRACE
18. CLASS ID(Bird) EXTENDS ID(Animal) LBRACE
19. PUBLIC INT ID(maxFlyingHeight) SEMI
21. CONSTRUCTOR LPAREN RPAREN LBRACE
22. THIS DOT ID(weight) ASSIGN INT_LITERAL(0) SEMI
23. THIS DOT ID(maxFlyingHeight) ASSIGN INT_LITERAL(0) SEMI
24. RBRACE
26. CONSTRUCTOR LPAREN INT ID(w) COMMA INT ID(h) RPAREN LBRACE
27. THIS DOT ID(weight) ASSIGN ID(w) SEMI
28. THIS DOT ID(maxFlyingHeight) ASSIGN ID(h) SEMI
```

5

TDD

Test driven development saves lives

Testing

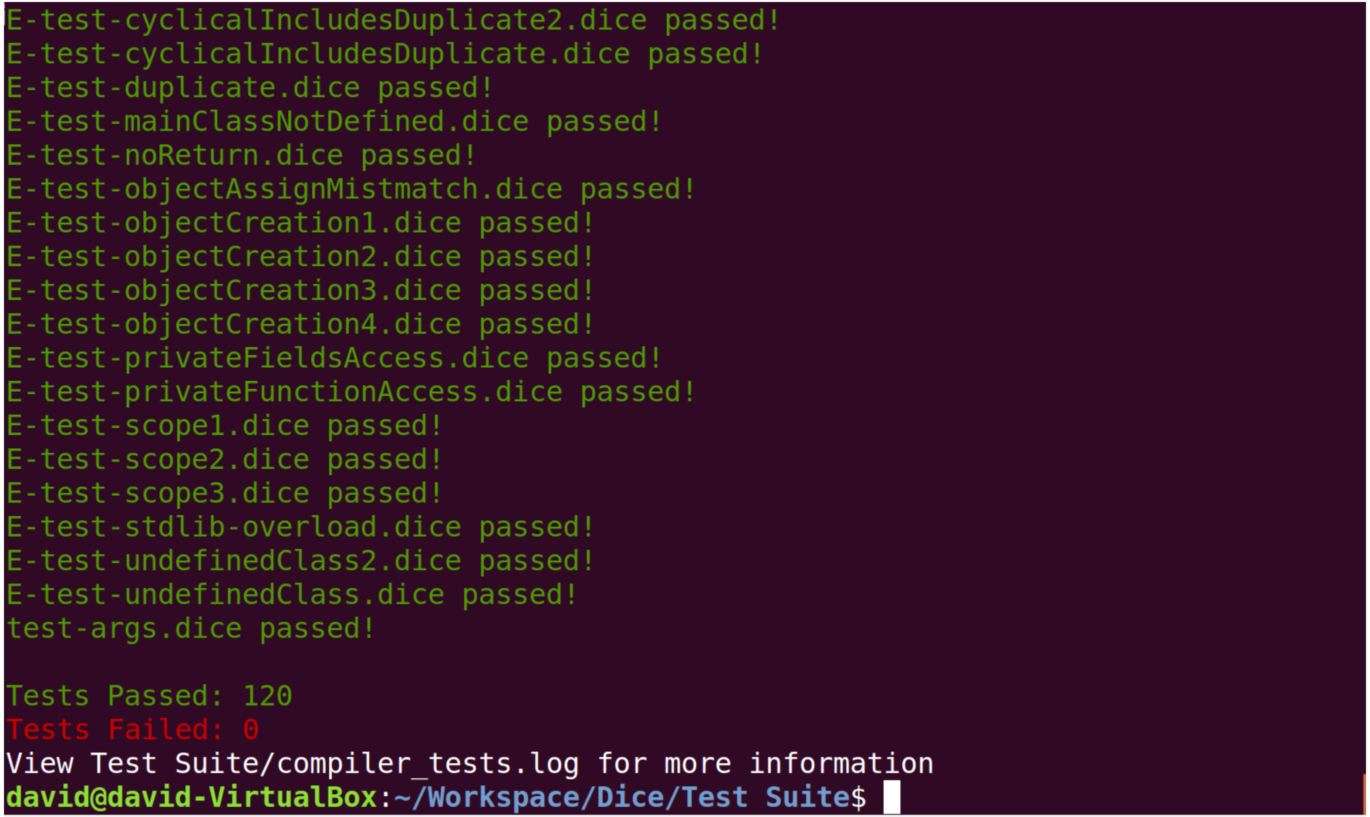
Regression Suite

TDD
From day 1 we made our regression suite. We started with MicroC, then added a test for each bug/feature we could think of

Testing Invalid Code
Making sure our compiler is not only correct, but also finds user errors

Testing

Regression Suite



```
E-test-cyclicalIncludesDuplicate2.dice passed!
E-test-cyclicalIncludesDuplicate.dice passed!
E-test-duplicate.dice passed!
E-test-mainClassNotDefined.dice passed!
E-test-noReturn.dice passed!
E-test-objectAssignMistmatch.dice passed!
E-test-objectCreation1.dice passed!
E-test-objectCreation2.dice passed!
E-test-objectCreation3.dice passed!
E-test-objectCreation4.dice passed!
E-test-privateFieldsAccess.dice passed!
E-test-privateFunctionAccess.dice passed!
E-test-scope1.dice passed!
E-test-scope2.dice passed!
E-test-scope3.dice passed!
E-test-stdlib-overload.dice passed!
E-test-undefinedClass2.dice passed!
E-test-undefinedClass.dice passed!
test-args.dice passed!

Tests Passed: 120
Tests Failed: 0
View Test Suite/compiler_tests.log for more information
david@david-VirtualBox:~/Workspace/Dice/Test Suite$
```

Time to demo!

Hope you like zoos with small dogs