CGC

Lieyang Chen, Zhuoxuan Li, Tianze Huang, Fanhao Zeng

Overview

- Introduction & Background
- Development Environment
- Syntax & Usage
- Architecture
- Testing
- Demo

Introduction & background

CGC: C with Garbage Collection

- General-Purpose Programming Language, with core features extracted from C
- Simple syntax from C
- Simple Object-Oriented functionality
- Simple garbage collector

Development Environment

Syntax

Comments

```
// Single line comment



/*

Multi-line Comment

*/
```

Operators

```
+                   //add
-                   //subtract
*                   //multiply
/                   //divide
==                  //eq.
!=                  // not eq.
&&                  //and
||                  //or
!                   //not
>                   //gt.
<                   //lt.
>=                  //geq.
<=                  //leq.
```

Built-in Types

```
Int             //4115

bool            //True

float           //0.25

Char            //"h"

void

array           //
```

Syntax

Array

```
int main() {
  int[] array;
  array = new int[5];
  print(array[0]);
  print(array[4]);
  return 0;
}
```

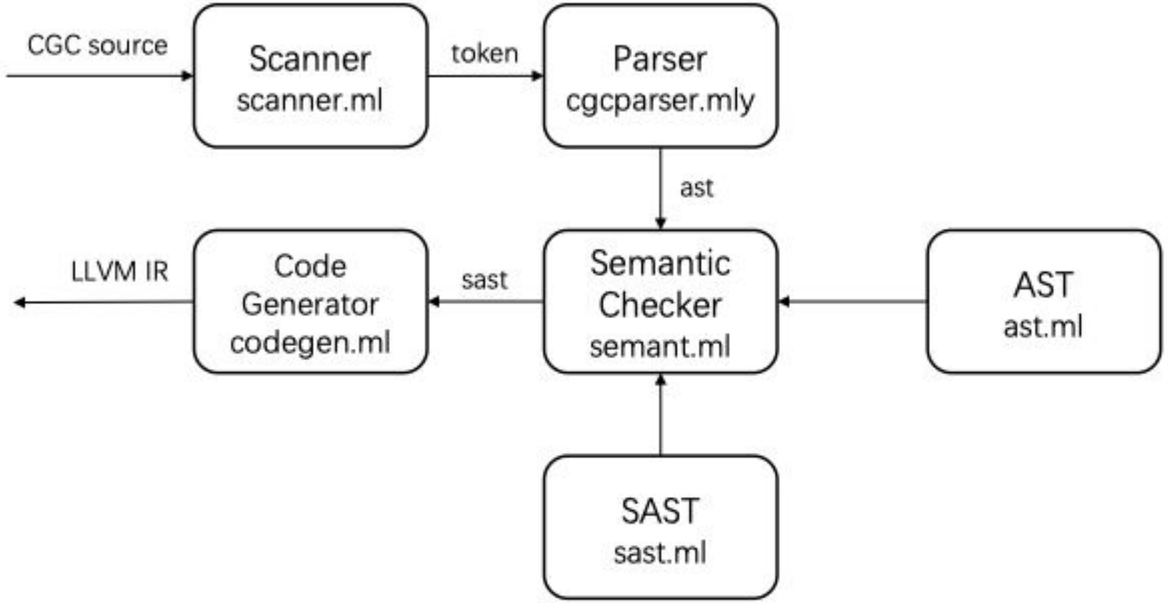
Control Flow

```
int main()
{
  while(true) {
  if (false) { break; }
  else { continue;} }
  for(i = 0; i < 1 ; ){ }
}
```

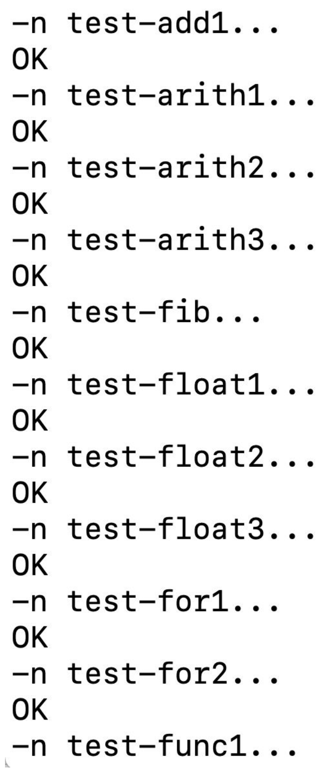
Classes

```
class Example{
  int x;
  constructor(int a){
    x=a;
   }
}
```

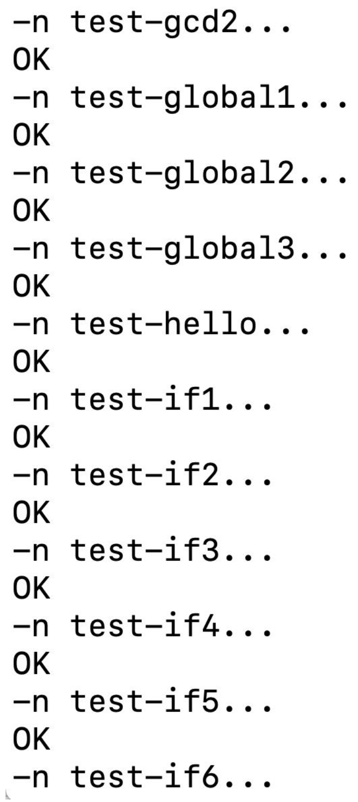
Architecture



Testing



```
-n test-add1...
OK
-n test-arith1...
OK
-n test-arith2...
OK
-n test-arith3...
OK
-n test-fib...
OK
-n test-float1...
OK
-n test-float2...
OK
-n test-float3...
OK
-n test-for1...
OK
-n test-for2...
OK
-n test-func1...
```



```
-n test-gcd2...
OK
-n test-global1...
OK
-n test-global2...
OK
-n test-global3...
OK
-n test-hello...
OK
-n test-if1...
OK
-n test-if2...
OK
-n test-if3...
OK
-n test-if4...
OK
-n test-if5...
OK
-n test-if6...
```

- Automated in testall.sh
- Compares output with test_case.out
- New test cases are added when new features are implemented

Future Work

- Further implement of GC
  - Currently, we store the pointer in a list each time we do malloc
  - All the memory on heap will be freed before the main function returns
  - Need to implement more robust GC algorithms
- Conduct more integration test
  - Need to do some testing after we update our GC algorithm

DEMO