EqEq

MATH PROBLEM HELP

Call 1-800-[(10x)(13i)^2]-sin(xy)/2.362x]

Jonathan Zacsh
Language Guru

Nam Nhat Hoang
System Architect

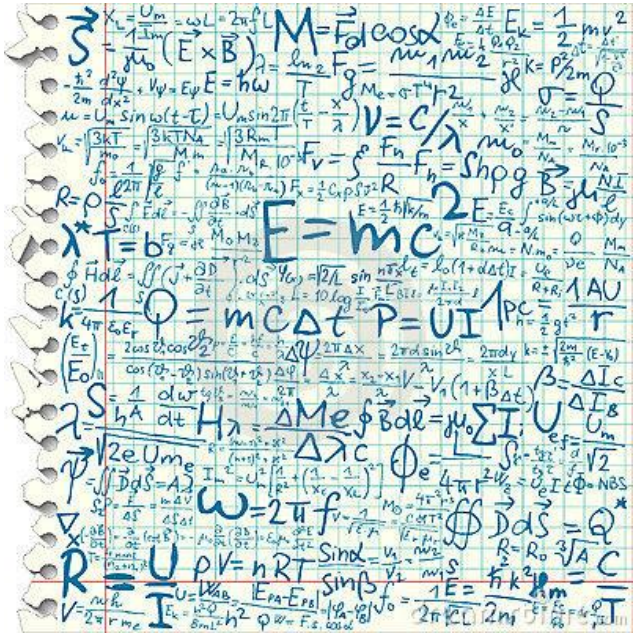
Tianci Zhong
Manager

Ruicong Xie
Manager

Lanting He
Tester

The Goal

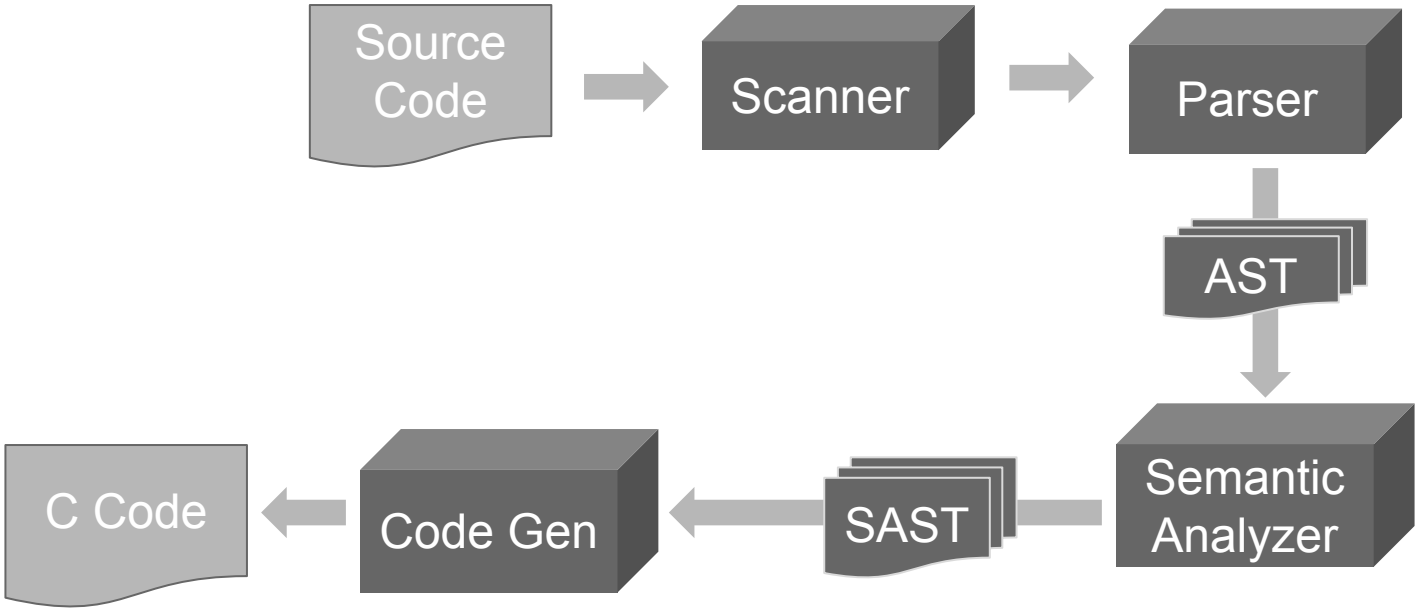
- Simple representation of math equations
- Easy usage of variables
- Solve equations in an easy program like solve them on a white board



Language Features

1. Easy equation declaration without specifying parameters
2. Implicit equation dependencies - can use variables before they’re defined
3. Context allows user to easily define complex equations to use later
4. range() allows users to test results for multiple values
5. Return expression implicitly defined

Compiler Architecture



Scanner

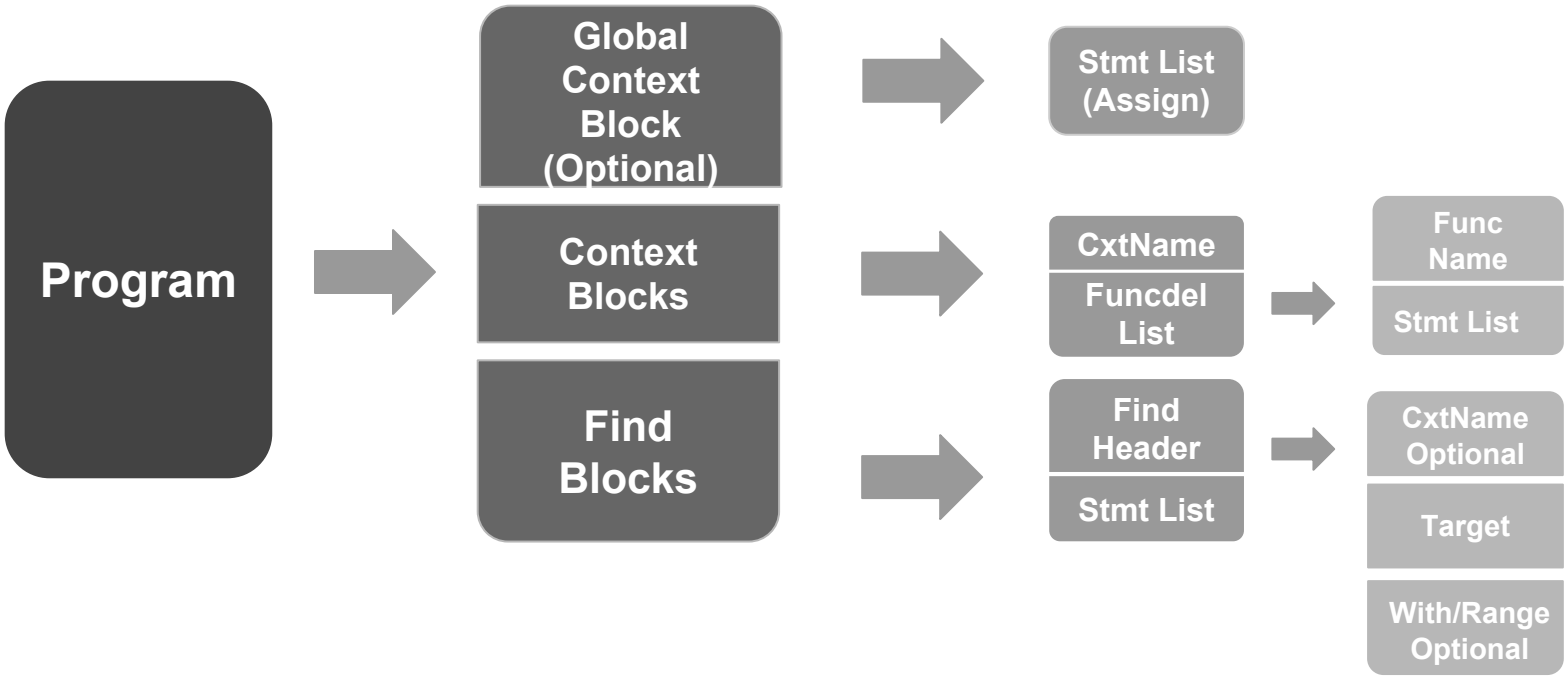
Reverved keyword:

```
"if", "else", "while", "find", "break", "continue", "with", "in", "range"
```

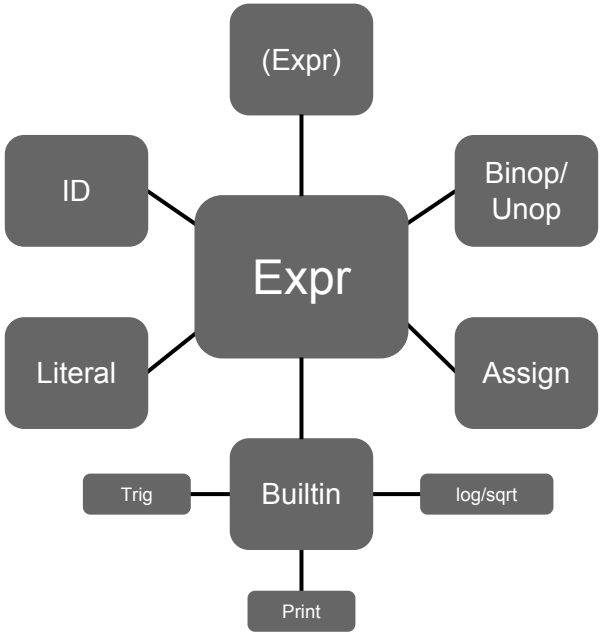
Illegal variable name in EqEq:

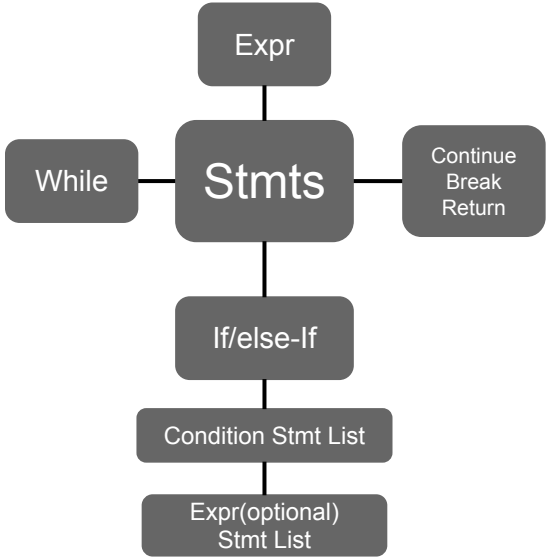
```
("int" | "double" | "char" | "float" | "const" | "void" | "short" | "struct"
| "long" | "return" | "static" | "switch" | "case" | "default" | "for" |
"do" | "goto" | "auto" | "signed" | "extern" | "register" | "enum" |
"sizeof" | "typedef" | "union" | "volatile" | "Global" )
```

Parser


Program
Global Context Block (Optional)
Context Blocks
Find Blocks
Stmt List (Assign)
CxtName
Funcdel List
Func Name
Stmt List
Find Header
Stmt List
CxtName Optional
Target
With/Range Optional

AST


(Expr)
ID
Binop/
Unop
Expr
Literal
Assign
Trig
Builtin
log/sqrt
Print


Expr
While
Stmts
Continue
Break
Return
If/else-If
Condition Stmt List
Expr(optional)
Stmt List

Exceptions

- The exception of wrong usage of reserved keyword, e.g: break
- The exception of wrong use of duplicate context block
- The exception of illegal usage of mathematical equation, e.g: cos(3,4,5)
- The exception of lower case Context name
- The exception of bad syntax of if / if-elseif /if-else
- The exception of undeclared variables and undeclared context
- The exception of wrong use of build-in function.
  e.g: `print("%0.0f ", a)-print("%0.0f ", a); range(3, 5, "abc")`
- `The exception of illegal return`
- `The exception of illegal find block declaration`
- `The exception of cyclic dependency`

Relation

```
SomeCtx = {
  a = { b + 40; }
  b = { 2 ^ c + 5; }
  c = { 5; }
  f = { 1/3; } /* never used */
}

SomeCtx:find a {
  d = 100; /* never used */
  print("a = %.0f (when b = %.0f)\n", a, b);

  b = 2;
  /* now: a depends on b which is independent */
  print("a = %.0f (when b = %.0f)\n", a, b);
}
```

```
SomeCtx : {
    deps:
        a: [b]
        b: [c]

    indeps:
        c : ```
            5.;

        ```
        f : ```
            1. / 3.;

        ```
```

```
finds:
    "find_SomeCtx_0": {
        [0]: {
            deps:
                a: [b]
                b: [c]

            indeps:
                c : ```
                    5.;

                ```
                f : ```
                    1. / 3.;

                ```

        }
```

```
]: {
  deps:
      a: [b]

  indeps:
      b : ```
          2.;

      ```
      c : ```
          5.;

      ```
      d : ```
          100.;

      ```
      f : ```
          1. / 3.;

      ```
```

Example Program

print42.eq

```
SomeContext = {
  a = { 42;/* return 42 back to a*/
      }
}

SomeContext: find a {
  print("%.0f\n", a);
}
```

Output C File

```
#include <stdio.h>

double SomeContext_a_0 (){
  return (double) (42.);
}

void find_SomeContext_0 ( ) {
  double a;
  a = SomeContext_a_0 ();
  printf("%.0f\n", (double) (a));
}

void find_SomeContext_0_range(){
  find_SomeContext_0();
}

int main() {
  find_SomeContext_0_range ();
  return 0;
}
```

Example Program

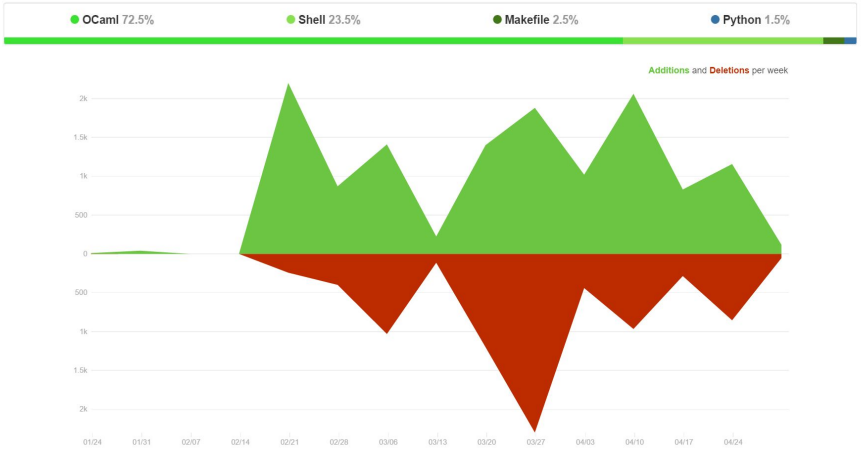
iterative-gcd.eq

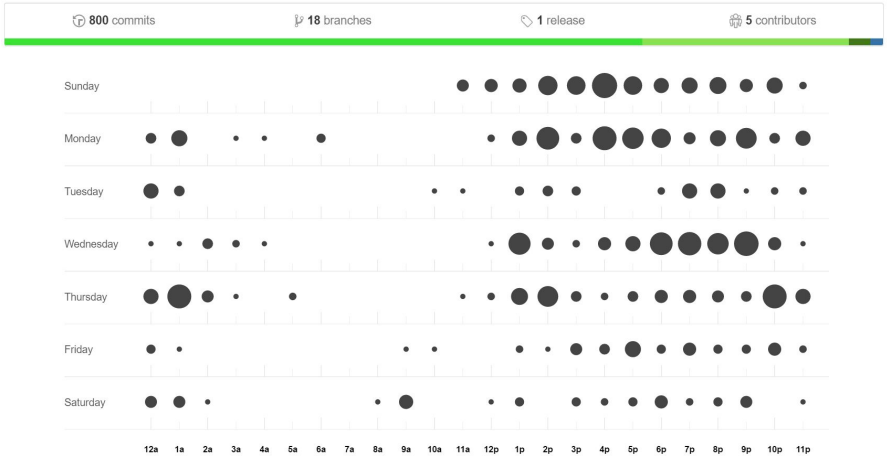
```
Gcd = {/* Context block */
  gcd = {
    a = 10;
    b = 20;
    while (a != b) {
      if (a > b) {a = a - b;}
      else {b = b - a;}
    }
    a; /* returns a back to gcd */
  } /* gcd is a multiline equation */
}

Gcd:find gcd { /* Find block */
  print("gcd between 10 and 20 is %0.f\n", gcd);
}
```

Output C File

```
#include <stdio.h>

double Gcd_gcd_0 (){
  double a=10.;
  double b=20.;

  while (a != b){
    if (a > b){ a = a - b;}
    else { b = b - a;}
  }
  return (double) (a);
}
void find_Gcd_0( ){
  double gcd;
  double b;
  double a;
  gcd = Gcd_gcd_0();
  //-----gen_finddecl_stmt-----
  printf("gcd between 10 and 20 is %0.f\n", (double) (gcd));
}

void find_Gcd_0_range(){
  find_Gcd_0();
}
int main() {
  find_Gcd_0_range ();
  return 0;
}
```

How We Work





Issues ✓ 23 Closed Pull requests ✓ 61 Closed

slack

Status build passing

Travis CI

GitHub

Google docs

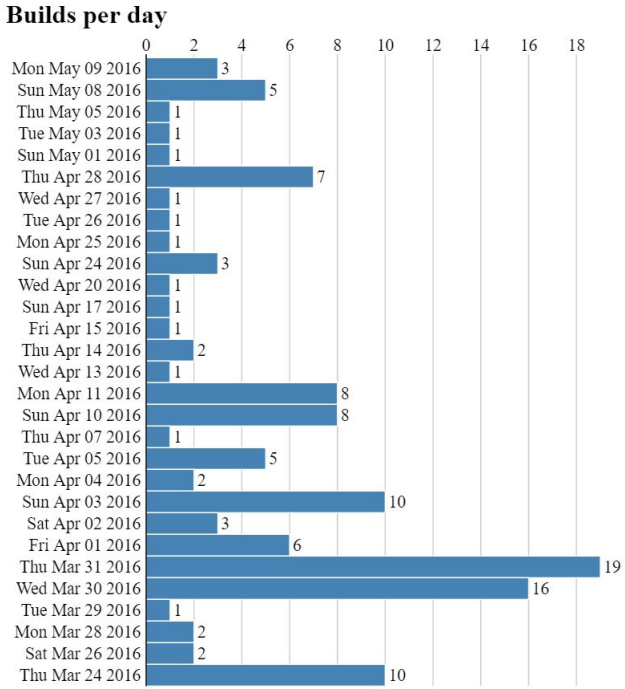
Source Code Statistics

File	Lines	Role
scanner.mll	78	Token rules
parser.mly	178	Context-free grammar
ast.ml	145	Abstract syntax tree & pretty printer
semant.ml	443	Semantic checking
relation.ml	294	Evaluate variable dependency
sast.ml	144	Semantically checked AST
codegen.ml	385	C code generation
eqeq.ml	58	Top level
Total	**1725**	

Test Files

Type	Files	Total Lines
Working.eq	52	804
Working outputs	52	213
Failling.eq	74	694
Error Messages	74	73
Total	**252**	**1784**

Test and Debug



File	Line	Function
`e2e-tests.sh`	367	Runner/Reporter
`eq-to-obj`	18	C compiler wrapper
`lint.sh`	60	
`debugtokens.ml`	51	Eqeq tokenizer
`debug_frontend.py`	31	FE debug tool
`.travis.yml`	13	Travis CI build
`Makefile`	75	
Total	**615**	

Demo

Lesson Learned

Ruicong Xie: Appreciation for the seamless development workflow we had and importance of communication when collaborating for large scale development.

Tianci Zhong: Appreciate the diversity background of the teammates and learn a lot from them.

Jonathan: Make no assumptions about what people do/don't know. People don't like to speak up and say so when something is confusing (which can cause slowdown). ie: getting ∀ on the same page with **even** the basics (eg: the VCS).

Lanting: Learned a lot from great teammates, it is great for people to work on the things they are good at.

Nam: Meeting is very important for everyone to get on the same page and learn from others (even when not everyone can come to the meeting).

Q&A

Thank you