

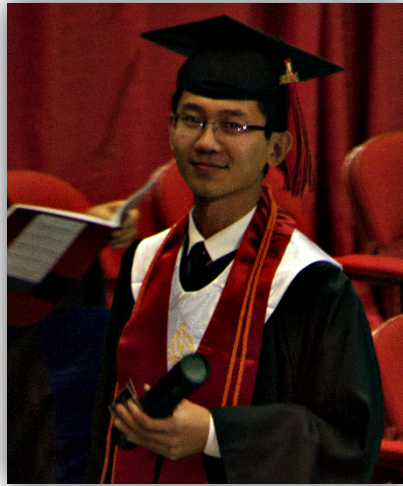
GEO: Shapes that Matter

THE GEO TEAM

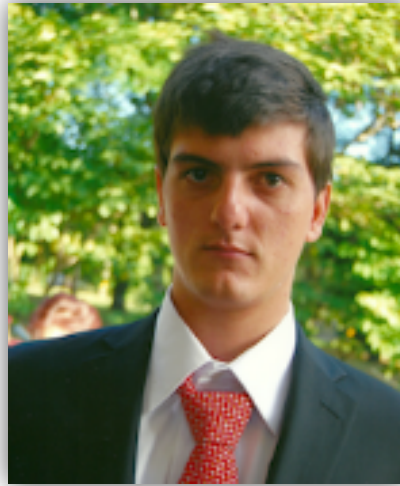
The GEO Team



Ruibo Li
Project Manager



Yi Guo
Language Guru



Nicolas Mesa
System Architect



Di Ruan
System Integrator



Yin Zhao
QA Engineer

Introduction

Simple

High-Level

Graphics Friendly

Imperative

Statically Typed

Hello World

```
> printf("Hello World")
```

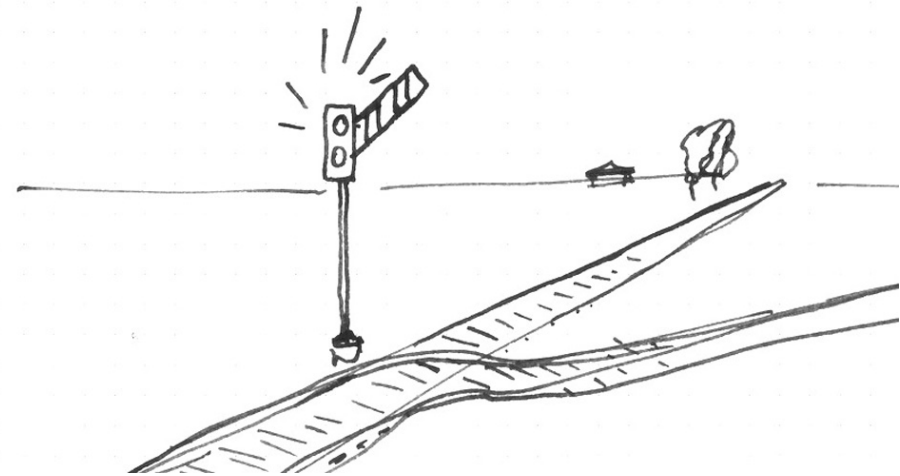
```
> printf("Hello World")
```

```
> printf("Hello World")
```



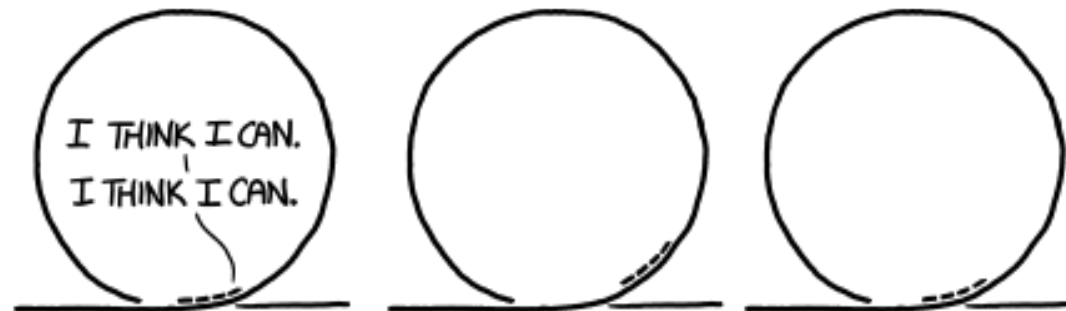
if (you like GEO)

```
> string you := "person"  
> if (you = "guy")  
    printl("You like GEO!")  
ef (you = "girl")  
    printl("You still like GEO")  
el  
    printl("Everyone likes GEO")  
end
```



But, for how long?

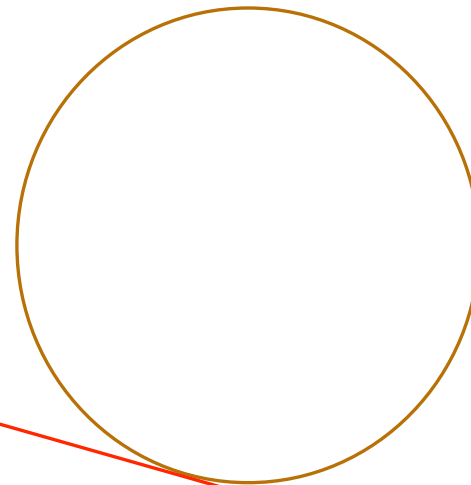
```
> int age := 21
> while (age < 120)
  printl("GEO, I'm lovin' it!")
  age := age + 1
end
```



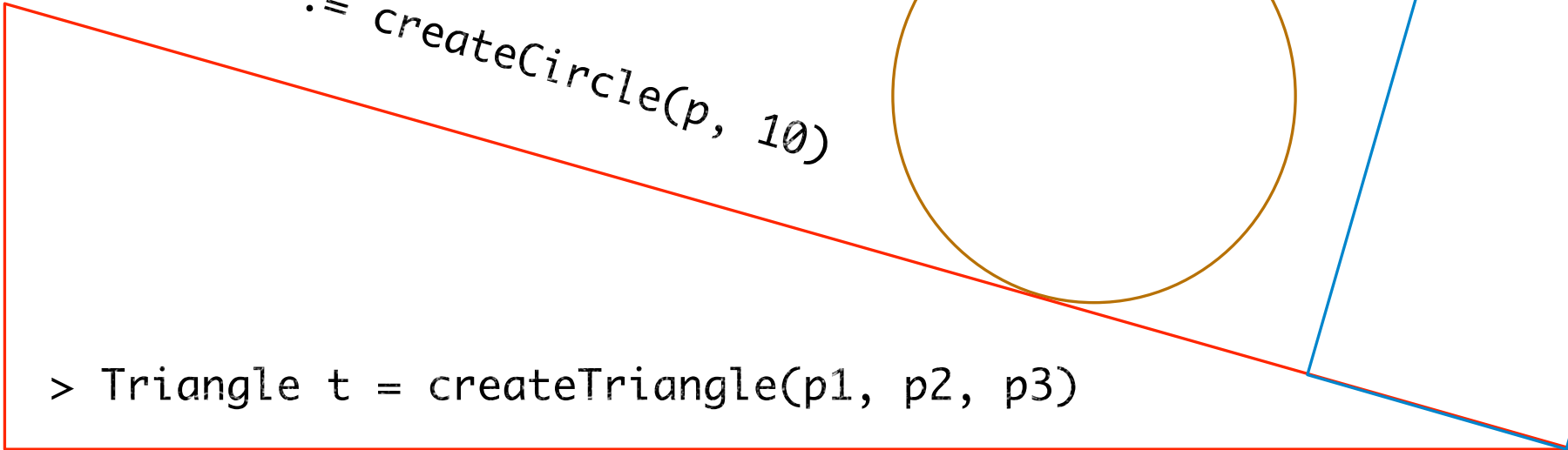
Give me a shape

> Rectangle r := createRectangle(p1, p2)

> Circle c := createCircle(p, 10)

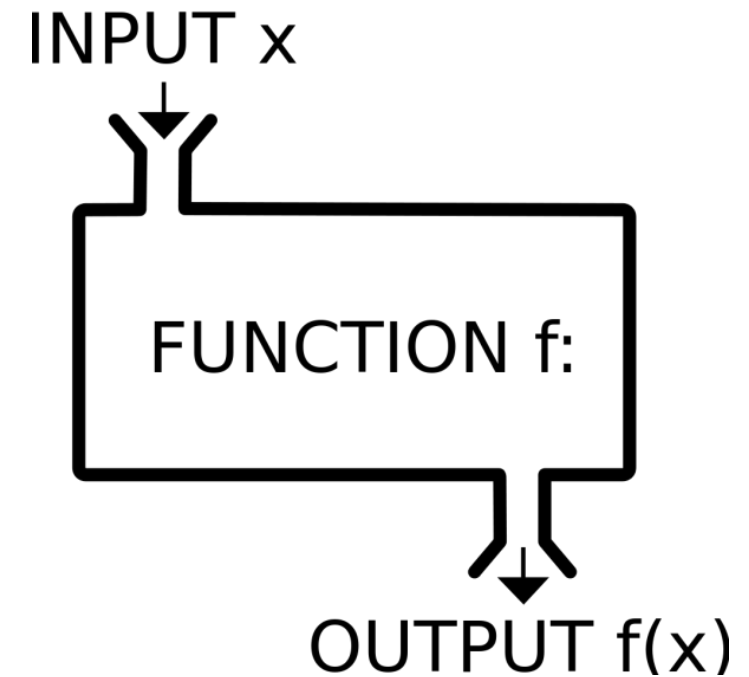


> Triangle t = createTriangle(p1, p2, p3)



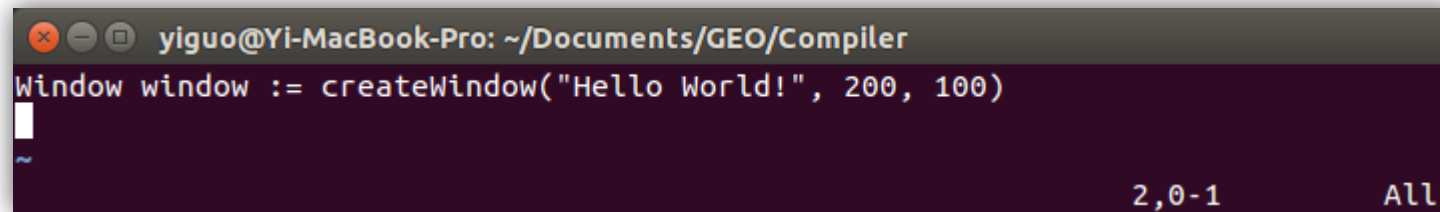
It's MAGIC!

```
> int fib(int a) := b
  if (a = 1 || a = 2)
    b := 1
  el
    b := fib(a - 2) + fib(a - 1)
  end
end
```



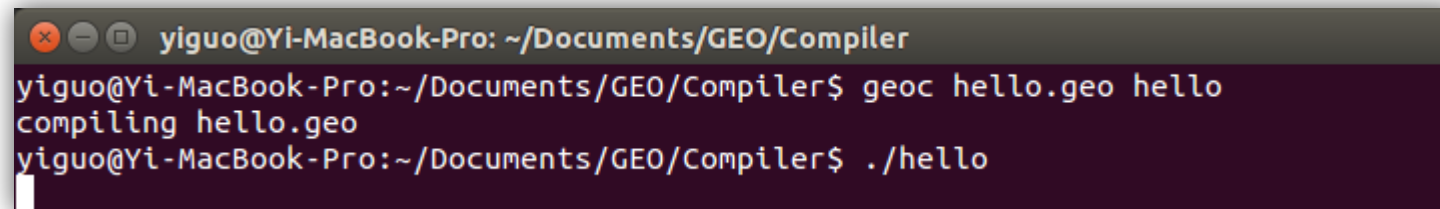
Hello World, Again!

1. Create a .geo file

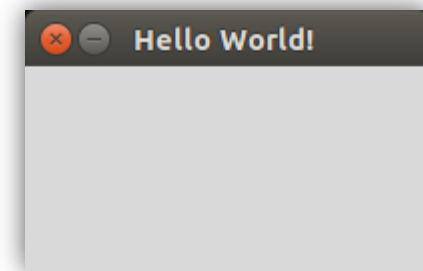


```
yiguo@Yi-MacBook-Pro: ~/Documents/GEO/Compiler
Window window := createWindow("Hello World!", 200, 100)
~
2,0-1 All
```

2. Issue the command



```
yiguo@Yi-MacBook-Pro: ~/Documents/GEO/Compiler
yiguo@Yi-MacBook-Pro:~/Documents/GEO/Compiler$ geoc hello.geo hello
compiling hello.geo
yiguo@Yi-MacBook-Pro:~/Documents/GEO/Compiler$ ./hello
```

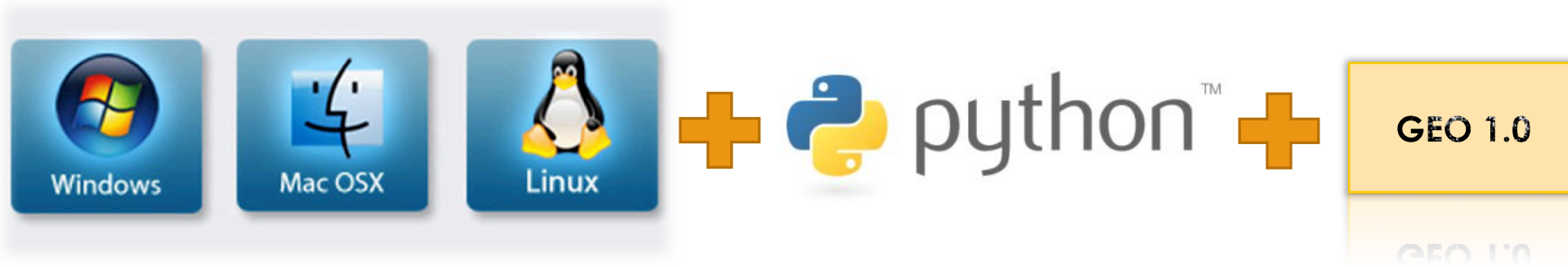


Environment

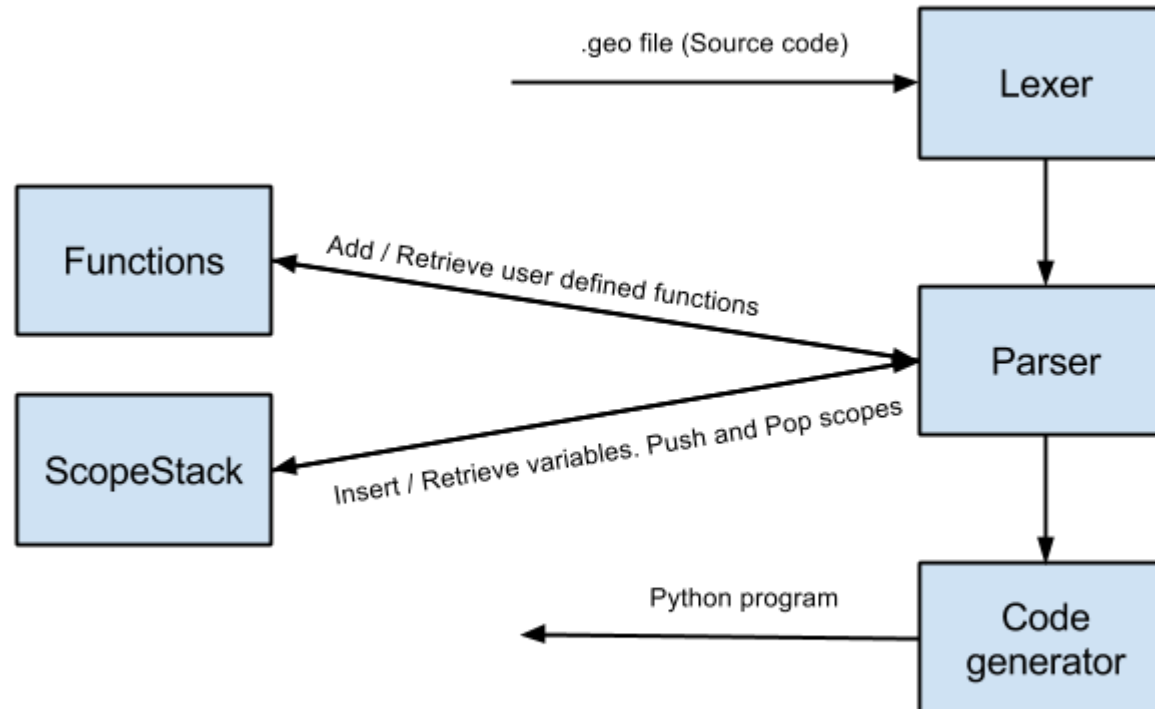
➤ Development Environment



➤ Runtime Environment



Compiler Architecture



Scope Stack

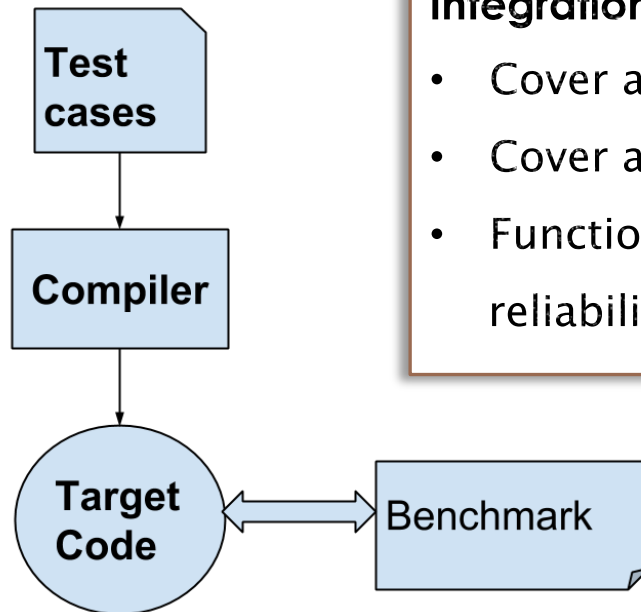
```
> int a := 0
  int b := 10
  while (a < 5)
    int b := a
    a := a + 1
  end
  print1(str(b))
  a := 0
  while (a < 5)
    b := a
    a := a + 1
  end
  print1(str(b))
```

```
> a_1 = 0
  b_1 = 10
  while (a_1 < 5):
    b_2 = a_1
    a_1 = a_1 + 1
  print1(str(b_1))
  a_1 = 0
  while (a_1 < 5):
    b_1 = a_1
    a_1 = a_1 + 1
  print1(str(b_1))
```

Test

Unit Test:

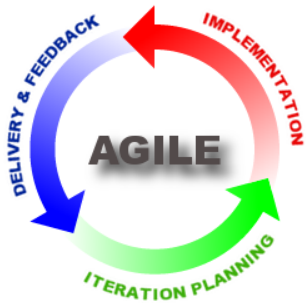
- Test all the built-in functions
- Isolate smallest testable parts
- Return FALSE if any built-in functions malfunction.



Integration Test:

- Cover all the grammar
- Cover all the basic workflow
- Functionality, performance, & reliability

Project Management



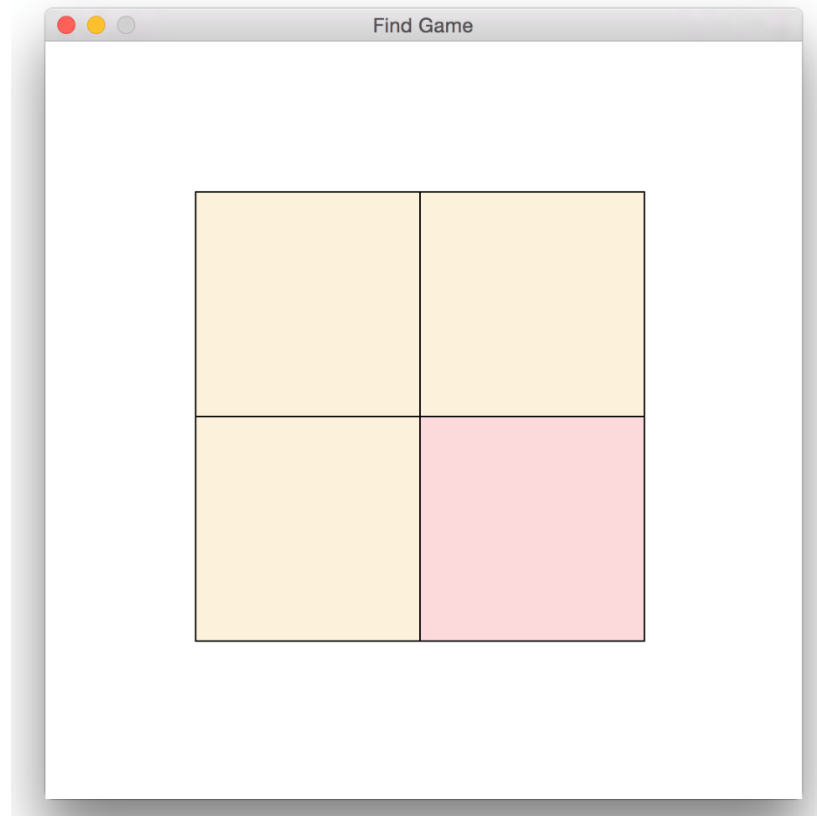
What have we learned?



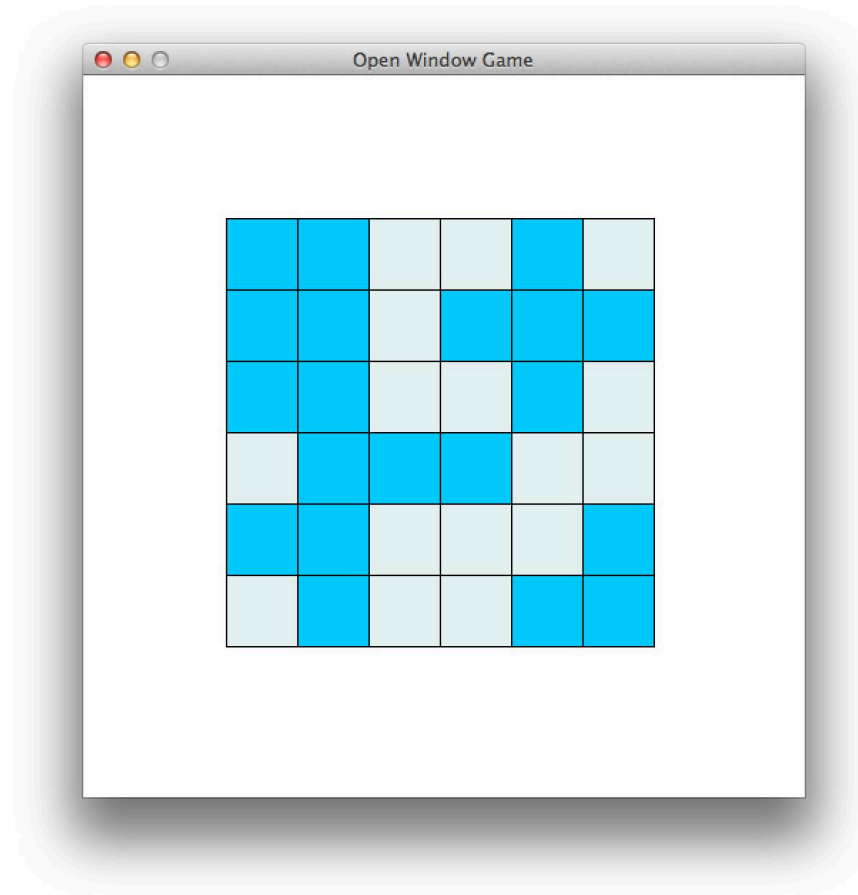
```
1 #include <ncurses.h>
2 int m[256]
3 ,b ;;; ;;; WINDOW*w; char*l="" "\176qxl"
4 xm "x" "c" "j" "v" "u"
5 ]= "Z" "ptiftb" "qdc" "ou" "dq!$cinmf"/**
6 int u, int v{
7 1] l=2,m[u][v-1] & 487W[v-1 ] & 15]]):0:0;u?m[
8 u- 1][ v]8 48?
9 15] ):0:0;v< 255 7m[ u][v+1]l=0,m[u][v+1]& 48
10 ):0 :0; u < 255 7m[ u+
11 4,m[u+1][ v]&487W+1][v]&15]]):0:0;W[ v]& 15] ]
12 *q ?cu (q+ 1)& 17q [
13 q[0 ]-- :1; }d( int u, int/**/v, int/**/s
14 Y=y -v, X=x -u; int S,s ;Y<
15 s=- 1:( s=1);X<0?X=-X,S --1 :(S= 1); Y<=
16 int f=y -(X >>1 );;
17 f>= 0?v=s,f=-X:0;u +=S ;f+= Y;m[u][v]l=32;mvw
18 ][ v]8 64? 60: 46)
19 v]&16){c(u,v);; ;;; return;}} }else{int f=
20 (v l-y )(f >=0 ?u +=S,
21 v +=S :f=X;m[u][v]l= 32;mvwdich(w,v,u,m[S
```



Find Difference



Open Window



Tic Tac Toe

