GridLok

Final Report

Laura Hu {lh2718}
Alice Hwang {ah2813}
Bryan Yu {by2181}
Julian Edwards {jse2122}

Introduction

The purpose of GridLok is to more easily facilitate the creation of tile based games, such as Tic-Tac-Toe, Minesweeper, or even Chess, along with their user interface. Our language implements various unique for-loops to make it easier to code for different parts of the board. Additionally, GridLok makes use of the SDL C library to render the images for the game user interface.

Computer games are widely popular for an eclectic range of demographics in today's world. The inspiration for our language stems from our early days as coders; we wanted to create a language that is simple to use, fun, and captivating for those new to coding. As an entry-level language GridLok provides a simple syntax, straightforward functionality, and a goal-oriented purpose to create fun gaming programs. The added bonus is, of course, the painless implementation of a graphical user interface that provides GridLok programmers with the immediate satisfaction of being able to see and play their game!

We hope you enjoy programming with GridLok!

- Julian, Bryan, Alice, Laura

Tutorial

Let's begin the fun! This short tutorial will guide you step-by-step on how to quickly get started programming with GridLok.

Installation

In order to use GridLok, it is necessary to install SDL and SDL_image. Instructions and installation details can be found at the SDL website, https://www.libsdl.org/index.php. You will want to download the most recent version of SDL, which is currently 2.0.4. Once you have SDL and SDL_image set up on your computer you can go ahead and download the source code for GridLok on our github, https://github.com/hwangks/gridlok. Here you will find all the necessary code to run GridLok programs including a folder called src which contains all the necessary files for the compilation, if you're into that.

Running Your Code

Once everything is downloaded you will see a script called gridlok.sh which will compile your code and create an executable:

```
./gridlok.sh [.gl file] [optional executable name]
```

If no executable name given, the script file defaults test as the executable name.

What's Included In The Box

GridLok comes with the typical basics of programming languages as well as some higher-level functionality that proves practical when coding grid based-games. The language has simple syntax and requires braces to define the end of a line of code and the scope. Here are some examples with the simple data types and functions:

```
def int i {10} /* creates variable i and assigns it to 10 */
set i {5} /* changes i to 5 */
def str myStr {"hello world"}
set myStr {"bye world"}
print {"GridLok rules!"} /* one of the many useful built-in functions! */
onClick {changeType{"x"}} /* ex of cool functions you'll learn about! */
```

The built-in functions will prove useful to you when coding games and incorporating your very own GUI!

Basic Program Structure

As I am sure you are anxious to get started, it is important to understand the basic structure of every GridLok program. Every file must start with the keyword game followed by braces. This is the entire scope of the program and equivalent to main in Java or C. Furthermore, a program cannot run without setting up a board, players, turn order of the players, and pieces. These are the more complicated data types, but you don't need to worry about the magic that goes on behind them just yet! Lastly, you must include setup which sets up the initial board such as which pieces, if any, go on the board and where they go. Here is a basic skeleton of what a GridLok program should look like:

```
game {
	/* these are always necessary for your GridLok program to run! */
	board {}

	players {}

	turnOrder {}

	/* can be multiple pieces */
	piece {}

	setup {}
}
```

board

The board keyword must contain the dimensions of the board and the file path to the image that you would like to use as the board. There is one and only board per program and the image must be a .png file.

```
board {
	dimensions {3,3}
	image {"imageName.png"}
}
```

players

The players keyword must contain all the names of the players the game will have, the names are arbitrary and there may be as many players as your game requires. Names must be strings and there must be at least one player.

```
players {“player1”, “player2”, “player3”}
```

turnOrder

The turnOrder keyword must contain the order in which the players will execute their turns. Players may appear more than once or not at all, but all players in turnOrder must have already been defined in players.

```
turnOrder {“player2”, “player1”, “player3”}
```

piece

The piece keyword must contain the name of the piece, a file path for the image of the piece, onTurn function, and the onClick function. Piece names must be unique and strings. The onTurn function allows you to add any events that may need to occur, such as shifting pieces, when it is a players turn and may be empty. The onClick function allows you to determine the actions that occur when a player clicks the piece; this function makes user interaction so easy!

```
piece {
	name {“piece1”}
	image {“imageName.png”}
	onTurn {}
	onClick {}
}
```

setup

The setup keyword fills the board with the initial conditions of the game.

```
setup {
	/* cool for loop that iterates through every space in the board! */
	for all sp in board {place {“piece1”, sp.x, sp.y}}
}
```

Some Last Remarks

Okay, okay this is the last tid-bit before you can actually get coding, I promise! The end conditions win, lose, and draw, are naturally going to be part of the games you create. Keywords winCondition, loseCondition, and drawCondition are reserved for just this. When present in your program, they are checked at the beginning of every turn.

Winning, Losing, Drawing

When creating a win, lose, or draw condition just put the necessary conditions for

winning, losing, or drawing the game. You must return a boolean, true or false, to indicate whether the condition has been met or not. You may use none to all of the conditions in your program, whatever makes sense for your game.

```
winCondition {
	/* checks if the first and last spaces in the board are the same */
	def bool win {board[0][0] == board[3][3]}
	/* returns true if they are, false if not */
	return {win}
}

loseCondition {
	for all sp in board {
		/* checks if adj spaces on the same row have the same piece */
		if(board[sp.x][sp.y] == board[sp.x][sp.y + 1]) {
			return {true}
		}
	}
	return {false}
}

drawCondition {
	/* this is from a tic-tac-toe game that we created! */
	def bool d {true}
	for all sp in board {
		if(sp.type == “placeHolder”) {set d {false})
	}
	if(d) {print {“It’s a draw...”}}
	return {d}
}
```

Now you can get started! For a more comprehensive explanation of our language please see the GridLok Reference Language Manual, located in the next section. Here you will find an exhaustive explanation of syntax, keywords, functions, and other details. Take a peek, we promise we won’t tell anyone you read the manual.

Language Reference Manual

---

1 Overview

1.1 Goals

The purpose of Gridlok is to more easily facilitate the creation of grid-based games, such as Tic-Tac-Toe, Minesweeper, or even Chess. Rather than having excessively complicated for-loops to check the values in a column or to check certain grid coordinates, GridLok has straightforward, specialized for-loops and functions for accessing the board. Furthermore, GridLok offers a simple approach to creating a graphical user interface with many built-in, intuitive functions.

2 Lexical Elements

2.1 Identifiers

Identifiers are used to name variables and functions. They are case sensitive. Only letters and digits can be used in identifiers. The first character of an identifier cannot be a digit and must be a lowercase character.

2.2 Keywords

Keywords are identifiers that are reserved for the programming language.

The following are a list of keywords used by GridLok:

game	board	dimensions	image
players	turnOrder	piece	name
onClick	return	def	set
onTurn	if	else	row
for	setup	all	in
col	surrounding	AND	OR
winCondition	loseCondition	drawCondition	bool
int	str	x	y
turn			

2.3 Constants

Built in constants that also may not be used for identifier names include:

```
true          false         empty
```

2.4 Operators

Operators are tokens used to perform actions, such as addition or subtraction, on operands. More will be discussed in section 4.

2.5 Separators

Separators separate tokens. White space is a separator, but not a token. Other separators are themselves single character tokens.

Separators in our language include:

```
{}()[],
```

2.6 White Space

White space is represented by a set of characters: the space character, the tab character, and the newline character. Whitespace is ignored, except when it is used as a separator for tokens or within a string constant.

```
piece{name{“my piece”}}
```

and

```
piece{
        name{“my piece”}
}
```

are equivalent.

2.7 Comments

GridLok only supports multiple lined comments. Comments must begin with /* and end with */.

```
/* This is a comment. */
```

3 Data Types

3.1 Primitive Data Types

The primitive data types in GridLok reflect the simplicity and purpose of the language. There are only three primitive data types that are more than sufficient for coding grid-based games.

3.1.1 Integers

The int data type will represent integer values between −2,147,483,648 to 2,147,483,647. It consists of a sequence of digits. Our language does not support decimal values.

3.1.2 Strings

Any text will be of type string. For simplicity, there will be no character data type, just one letter strings. A string is a sequence of zero or more characters and digits enclosed within double quotes.

3.3.3 Booleans

Booleans can either be true or false, and are manipulated by the !, AND, and OR operators.

3.2 Non-Primitive Data Types

The non-primitive data types, like their primitive counterparts, maintain both the simplicity and purpose of GridLok.

3.2.1 board

The board data type is used for creating a board for the game with specific dimensions. Exactly one board is required and allowed to be declared at the beginning of each program.

3.2.1.1 Declaring board

All that is required for declaring a board is the dimensions which is how many rows by columns the user wants and the pathfile for the image for the board.

```
board {
        dimensions {int numRows, int numCols}
        image {str pathfile}
}
```

Example:
Declaring a board for a tic-tac-toe game.

```
board {
        dimensions {3,3}
        image {“ttt.png”}
}
```

3.2.1.2 Accessing board

Spaces in a board can be accessed and return the name of the piece, if any, occupying the space. If the space is empty the keyword empty is returned.

```
board[i][j]
```

Where i is the row and j is the column.

3.2.2 piece

The piece data type is used for pieces that are placed in a board during a game.

3.2.2.1 Declaring a piece

Pieces are declared with the name of the piece as a string, the pathfile for the image as a string, and the onClick specification. Pieces may not be named “empty” as this is reserved.

```
piece {
        name {str piece_name}
        image {str pathfile}
        onTurn {
                /* optional triggers for specific events */
                /* can be left blank */
        }
        onClick {
                /* events that must occur when piece is clicked go here */
                /* can be left blank */
        }
}
```

Example:
Declaring a piece called “mines” for a Minesweeper game.

```
piece {
        name {“mines”}
        image {“mine.png”}
        onTurn{}
        onClick {
```

```
                visible {1} /* sets piece to visible on board */
                endTurn {}
        }
}
```

3.2.3 players

The players data type is essentially a list of strings that represent each player in the game. It is required for every GridLok program and there must be at least one player defined. There may be as many players as the game requires.

```
players {str player1, str player2, …, str playerN}
```

Example:
Declaring players for a tic-tac-toe game.

```
players  {“x”, “o”}
```

3.2.3 turnOrder

The turnOrder data type is essentially a list of strings that represent the order in which the players take turns. Players may appear more than once or not at all, but all players in turnOrder must have already been defined in players.

```
turnOrder {str player1, str player2, …, str player}
```

Example:
Declaring turnOrder for a tic-tac-toe game.

```
turnOrder {“o”, “x”}
```

3.2.4 Implicit piece Type

3.2.4.1 pieceID.x

Returns the x coordinate, the row, of the current space as an integer.

3.2.4.2 pieceID.y

Returns the y coordinate, the column, of the current space as an integer.

3.2.4.3 pieceID.type

Returns a string with the name of the piece currently occupying the space. If there is no piece it returns the reserved constant empty.

4 Expressions and Operators

4.1 Expressions

An expression consists of at least one operand and zero or more operators. Operands are typed objects such as constants, variables, and function calls that return values.

Examples include:

```
2 + 2
rows + cols
rows + 1
win OR lose
```

4.2 Assignment Operators

Assignment operators store values in variables. In order to assign a value to a variable, you must declare a variable using the def keyword and specify the variable name and value. The general form is as follows:

```
def type variableName {variableValue}
```

The data type of the variable must be specified and may not be changed unless the variableName is redefined. If passing in another variable as the value, GridLok will copy the value, not the reference.

To change the value of a defined variable use the set keyword:

```
set variableName {variableValue}
```

The value must be of the same data type that the variable was defined as.

4.3 Arithmetic, Comparison, and Logical Operators

The table below shows the arithmetic, comparison, and logical operators supported by the GridLok language, listed in order of precedence from highest to lowest.

Operator	Category	Description	Associativity

!	Logical	Logical Not	Right-to-left
* / %	Arithmetic	Multiplication, integer division, modulo	Left-to-right
+ -	Arithmetic	Addition, subtraction	
< <= > >=	Comparison	Less than, less than or equal to, greater than, greater than or equal to	
== !=	Comparison	Equal, not equal	
AND	Logical	Logical And	
OR	Logical	Logical Or	

5 Statements

5.1 if Statement

The if statement is used to execute part of a code based on the truth value of an expression.

```
if(test) {
        [then-statement]
}
else {
        [else-statement]
}
```

5.2 For-Loops

GridLok offers for-loops specialized for iterating over a board, simplifying grid-based game creation.

5.2.1 Enhanced For-loops

```
for pieceID surrounding (int i, int j)
```

Loops through all the spaces directly adjacent to coordinates (i,j).

```
for pieceID in row(int rowNum)
```

Loops through all spaces in the row corresponding to rowNum.

```
for pieceID in col(int colNum)
```

Loops through all spaces in the column corresponding to colNum.

```
for (int i, int j) surrounding (int w, int z)
```

Loops through all coordinates (i,j) directly adjacent to coordinates (w,z).

`for (int i, int j) in board`

Loops through all the locations in the board.

`for all pieceID in board`

Loops through every space in the board.

5.2.2 Looping Over Integers

`for variableName(int min, int max)`

Loops through min (inclusive), max (exclusive).

`for variableName(int min, int max), variableName2(int min2, int max2)`

A nested for loop with the second loop being the nested loop.

5.3 Setup

The setup keyword is reversed for initializing the board with what pieces, if any, go where. There must be one and only one setup condition in a GridLok program.

```
nCondition {
	[conditions]
	return {bool a}
}
```

5.4.2 Lose Condition

Defines the conditions for losing the game.

```
loseCondition {
	[conditions]
	return {bool a}
}
```

5.4.3 Draw Condition

Defines the conditions for a tie.

```
drawCondition {
	[condition]
	return {bool a}
}
```

6 Functions

6.1 Calling Functions

Functions are called using their names with the parameters between curly braces.

```
functionName {functionParameters}
```

6.2 Built-in Functions

```
click {int i, int j}
```
Clicks on a given coordinate (i,j) on the board. If a piece is present, then the its onClick method will be called.

```
removePiece {int i, int j}
```
Calling this removes this piece at the specified coordinates (i,j) from the board. If the given space is empty, removePiece does nothing.

```
changeType {str pieceName}
```
A function specific to piece that changes the piece type to the name specified.

```
visible {bool a}
```
A function specific to piece that reveals or hides a piece on the board. All pieces placed on the board are initially visible.

```
setVisibility {int k, int i, int j}
```
Sets the piece, if any, at coordinates (i,j) invisible if k is 0 and visible if k is any other integer.

```
place {str pieceName, int i, int j}
```
Places the given piece on the given i and j coordinates on the board.

```
endTurn {}
```
Ends turn so that the program can go onto the next player's turn after checking win/loss/draw condition.

```
rand {int i, int j}
```
Returns a random integer between i (inclusive) and j (exclusive).

```
print {str i}
```
This function prints i to console. It accepts only string literals.

7 Event Handling

7.1 Events

Event handling in GridLok refers to the execution of a function based on global booleans that are triggered upon a certain predefined condition. For example, if it

is a certain player's turn an event may be triggered. The turn keyword returns the name of the player whose turn it is.

Must be inside piece definition:

```
onTurn {
	/* triggers and events */
	/* can be left blank */
}
```

Example:

```
onTurn {
	if(turn == "player2") {
		for (i,j) in board {
			setVisibility {0, i, j}
		}
	}
}
```

8 Scope

8.1 Scope

Scope refers to what parts of the program can "see" a declared object. A declared object can be visible only within a particular statement or function. If unspecified, a pre-defined variable or keyword has scope anywhere in a GridLok program.
A declaration is not visible to declarations that came before it; for example:

```
def int i {5}
def int j {i + 10}
```

will work, but:

```
def i {y + 10}
def j {5}
```

will not.

Project Plan

Process

Specification

Specifications for GridLok originated from the initial Language Proposal and were later refined in the Language Reference Manual. With additional discussions with Professor Edwards and Julie, we were able to finalize the concrete syntax for our language and start working on the compiler.

Development

Development for Gridlok was initially divided up between each member of the group for the roles of Manager, Tester, System Architect, and Language Guru. We met every week on Saturdays to work on the language and Mondays after class to check up with each other. As we began work however, we realized that each of us had a stronger interest in different roles so we naturally took over different roles while also assisting with other parts of the language.

The development of the language stemmed from Professor Edwards' MicroC code which we used as a template to extrapolate our own language from. We learned how the MicroC code worked and progressed starting from the parser to the codegen.

Testing

The testing script (/gridlok/run_tests.sh) was created at the end of the process to check both fail conditions and to test different language functionalities, especially the language specific for-loops.

Programming Style Guide

OCaml

- camelCase
- 2 space indentation
- Comments to preface a block of code
- Pattern matching
- Newlines between function definitions

- Explicitly write out types for function arguments
- If and then statements should be in the same line

Version Control (Git)

- Branch from master when implementing new features
- Commit often after finishing substantial parts of a feature
- Rebase from master before merge
- When ready to merge, issue a pull request and our supreme leader Alice merges it into the repository's master branch
- Create new branch to start new features

Bash

- Files are named with snake_case
- One line per statement
- One space between each token

Project Timeline

Jan 20 - Create group (Alice, Bryan, Julia, Laura)
Feb 10 - Submit Language Proposal
Mar 7 - Submit Language Reference Manual
Apr 6 - Submit functioning Hello World for GridLok
May 10 - Present GridLok to Professor Edwards
May 12 - Submit Final Report

Roles and Responsibilities

Initial Roles:

- Manager - Alice
- Language Guru - Julian
- Systems Architect - Bryan
- Tester - Laura

Actual Roles:

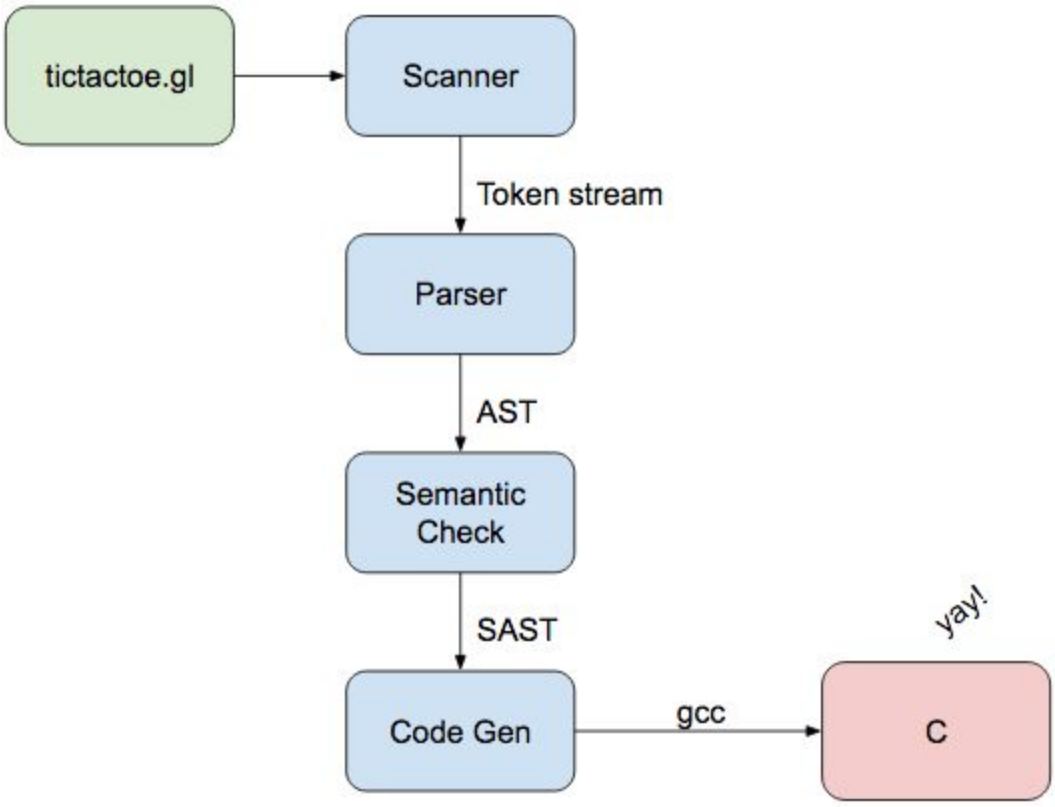
- Manager - Alice, Bryan
- Language Guru - Julian, Laura
- Systems Architect - Julian, Laura
- Tester - Alice, Bryan

Development Environment

- Languages: OCaml, C, SDL (image rendering library)
- Programming Editor: Sublime, vim
- Version Control: Git, Github
- Documentation: Google Drive

Architectural Design

The system architecture for the GridLok compiler can be broken down into the following parts: Scanner, Parser, Semantic Checker, and Code Generator.



Scanner (Laura, Julian)

The Scanner takes in a GridLok program and parses the code into tokens for identifiers, keywords, operators, comments, and values. It uniquely checks for GridLok specific keywords.

Parser (Laura, Julian)

The Parser takes in the token stream from the Scanner and created an Abstract Syntax Tree (AST) as specified by the GridLok Context-Free Grammar. The interesting part about the GridLok Parser is that it includes pattern-matching for the

different variations of win, lose, and draw conditions, and initializes all of the required parts of each declaration.

Semantic Checker (Laura, Julian)

The Semantic Checker walks through the AST and semantically checks each node for its type, converting the AST into an SAST.

Code Generator (Laura)

The Code Generator first generates all of the static code for C and SDL that will be required to render the image using SDL and setting up the program to work properly. It then traverses the SAST and generates C code by pattern-matching on the GridLok code and outputting the corresponding C code.

Test Plan

Test Files

Success Tests

We made success tests for all the functionalities of our language.

Fail Tests

We wrote fail tests for the errors the semantic check returns, as well as simple parsing errors that result in incorrectly written code.

Test Script

run_tests.sh

run_tests.sh is our script in the main directory which runs all the tests that are located in the tests/ directory. When run, the script does the following:

- Checks the tests/ directory and takes all files starting with test_ or fail_ and ending with .gl
- For test_ files:
  - Makes executable, then runs it and redirects stdout to a .out file of the same filename in a new tests/output/ directory
- For fail_ files:
  - Same as test_ files, but redirects stderr to the .out file
- Checks diff between the .out file in the tests/ directory and the .out file in the tests/output/ directory
- If there is no difference, deletes .diff file and returns OK message
- If there is a difference, keeps .diff file and returns FAILED message
- All test messages are logged in test.log file

Lessons Learned

Alice Hwang

1. Learn how to use Git properly: Every member should be working in separate branches, which should then be reviewed by one person and merged into the master branch. Get ready to resolve a lot of merge conflicts.
2. Start early: Commit to meeting frequently to make sure work gets done throughout the semester, not just in the last few weeks.
3. Meet with the TA and Professor Edwards: Meet with them often to make sure your project is going in the right direction. They are very helpful in telling you if something is a good idea or not really worth implementing.

Bryan Yu

As my first experience working with a team, as well as using a functional programming language, this project was definitely a hurdle that I had to scramble to keep up with, but in the end, it forced me to learn a lot. The entire project in itself was a completely new challenge. Everything was mostly new to us, from functional programming language to using SDL to even managing time as a group. I think the part that hurt us the most however was not having firm project deadlines from the beginning and enforcing them. As a result, we let everything get pushed to the end. In the end, we got the project done and we're very happy with it. If I were to do it again, I would start earlier, but definitely choose the same group again.

Advice: For future groups, as much as it is said, I would still definitely recommend starting early. The class moves pretty fast and covers so many topics so keeping up with OCaml throughout the course is important, so that you can still both understand what happens in class and so that you can work efficiently on the project.

Julian Edwards

This project was my first experience working with a group long-term. Most of the group projects in my classes thus far have lasted no more than a few weeks. I can

truly say it was an awesome experience and all of us got along and had fun. We are very proud of the outcome of the project and plan to continue working on and improving GridLok in the future. Nevertheless, the process of creating a language was not without its obstacles. Since it was our first time coding in a functional language and writing our compiler, we found it difficult to allocate time appropriately. Furthermore, learning OCaml was no easy feat but worthwhile; I have found functional programming to be very enjoyable.

For future groups:
It's better to be safe than sorry when allocating time and planning, give yourself double the amount of time you think you need! Keep organized with your group, learn GitHub well (it's an invaluable asset for any computer scientist), and have fun!

Laura Hu

Since this was the first long term project I've ever worked on, I learned a lot about the importance of group communication as well as clear planning. Because we weren't sure what components were needed in the beginning, we weren't really able to set concrete goals for our meeting plans ahead of time. It definitely felt like a time crunch in the end since everything came together after we finished the majority of the class. For a larger project like this, we probably should've done a better job of defining the architectural structure from the very beginning by looking at the material ahead of time, as well as establishing better methods of communication. Working with a larger code base was difficult as well. For example, I learned the importance of commenting my code thoroughly so that others can edit and add to the code more easily.

For future groups, I would strongly recommend not only meeting regularly, but also setting concrete and clearly defined goals to meet for each week. Make sure that all the group members are familiar with the architecture that's agreed on before starting the bulk of the coding for maximum efficiency, and that your group members can get along even during times of high stress, like finals week!

Appendix

Source Code

/src

ast.ml

```
(* Author: Laura *)
(*AST*)

(*operators*)

type op = Add | Sub | Mult | Div | Equal | Neq | Less | Leq | Greater | Geq |
                             And | Or | Mod | StrEqual

type uop = Not

let string_of_op = function
        Add -> "+"
        | Sub -> "-"
        | Mult -> "*"
        | Div -> "/"
        | Equal -> "="
        | Neq -> "!="
        | Less -> "<"
        | Leq -> "<="
        | Greater -> ">"
        | Geq -> ">="
        | And -> "AND"
        | Or -> "OR"
        | Mod -> "%"
        | StrEqual -> "=="


let string_of_uop= function
        | Not -> "!"


(*data types*)
```

```
type typ       =       Bool | Int | Str | Piece | Void

let string_of_typ = function
       Bool -> "boolean"
       | Int -> "int"
       | Str -> "string"
       | Piece -> ""
       | Void -> ""

(* expressions are things that evaluate to a value or an action *)

type expr =
       Id of string
       | Parenth of expr
       | BoolLiteral of bool
       | IntLiteral of int
       | StringLiteral of string
       | Binop of expr * op * expr
       | Unop of uop * expr
       | BoardAccess of expr*expr
       | Access of string * string (* p.name *)
       | Rand of expr * expr
       | Empty

(*statements: things that you can do with expressions, things that dictate control flow.*)
type stmt =
   Expr of expr
  | Assign of string * expr (* a {5}*)
  | VDecl of typ * string * expr
  | Call of string * expr list

  | Return of expr
  | If of expr * stmt list * stmt list
  (* for varName surrounding (int x, int y) *)
  | ForSurrounding of string * expr * expr * stmt list
  (*for varName in row(int rowNum)*)
  | ForRow of string * expr * stmt list
  (* for varName in col(int colNum) *)
  | ForCol of string * expr * stmt list
  (* for (int i, int j) surrounding (int x, int y) *)
  | ForSurroundingCoords of string * string * expr * expr * stmt list
  (* for varName(int min, int max) *)
  | ForMinMax of string * expr * expr * stmt list
  (* for varName(int min, int max), varName2(int min2, int max2) *)
  | ForNested of string * expr * expr * string * expr * expr * stmt list
  (* for (int x, int y) in board *)
  | ForBoard of string * string * stmt list
  | ForAll of string * stmt list
```

```
(*board definition*)
type board = {
       x: expr;
       y: expr;
       bg: expr;
}

(*a list of all players*)
type players = string list

(*a list that dictates the order in which players play*)
type turnOrder = string list

(*piece definition*)
type piece = {
       name: string;
       img: string;
       onTurn: stmt list;
       onClick: stmt list;
}

(*instructions for the setup of the board, rendered before any turns are completed*)
type setup = stmt list

(*conditions that end the game, specified as functions that return booleans. if not specified,
we assume the functions indefinitely return 0.*)
type conditions = {
       win: stmt list;
       lose: stmt list;
       draw: stmt list;
       w: bool; l: bool; d: bool;
}

(*a program must comprised of all these parts*)
type program = board * players * turnOrder * piece list * setup * conditions
```

codegen.ml

```
(* Author: Laura *)
open Printf
open Ast
open Sast

let file = "test.c"

let string_of_op = function
  Add -> "+"
  | Sub -> "-"
  | Mult -> "*"
  | Div -> "/"
  | Mod -> "%"
```

```
  | StrEqual -> failwith("Wrong way to get stringeql!")
  | Equal -> "=="
  | Neq -> "!="
  | Less -> "<"
  | Leq -> "<="
  | Greater -> ">"
  | Geq ->  ">="
  | And -> "&&"
  | Or -> "||"

let string_of_uop = function
  | Not -> "!"


let string_of_typ = function
    Int -> "int"
  | Bool -> "int"
  | Str -> "char *"
  | Piece | Void -> failwith("Semantic check failed to eliminate the tpying of piece and
voids. ")

let rec string_of_expr = function
  | Sast.Parenth(e) ->
    let _,e = e in "(" ^ string_of_expr e ^ ")"
  | Sast.Id (i) -> if (i="turn" || i="x"||i="y") then i
      else if i="height" then "HEIGHT"
      else if i="width" then "WIDTH"
      else i^"UDV"
  | Sast.BoolLiteral(b) -> if b=true then string_of_int 1 else string_of_int 0
  | Sast.IntLiteral(i) -> string_of_int i
  | Sast.StringLiteral(s) -> s
  | Sast.Binop(e1,o,e2) -> let _,e1=e1 and _,e2=e2 and t,_=e1 in
    if o=Ast.StrEqual then ("strcmp(" ^ string_of_expr e1 ^ "," ^ string_of_expr e2 ^ ")==0")
    else if t=Ast.Str then ("strcmp(" ^ string_of_expr e1 ^ "," ^ string_of_expr e2 ^ ")!=0")
    else string_of_expr e1 ^ string_of_op o ^ string_of_expr e2

  | Sast.Unop (o, e) -> let _,e=e in
    string_of_uop o ^ string_of_expr e
  | Sast.Access(v,f) ->
        if f="visible" then "v[xx][yy]"
        else if f="type" then "b[xx][yy]"
        else if f="x" then "xx"
        else if f="y" then "yy"
        else failwith ("semantic checker failed, invalid access field")
  | Sast.BoardAccess(x,y) ->
    let _,x = x and _,y=y in
    "b["^string_of_expr x ^ "][" ^ string_of_expr y ^ "]"
  | Sast.Rand(min,max) ->
      let _,min=min and _,max=max in
      "rand()%("^string_of_expr max^"+1-"^string_of_expr min^")+"^string_of_expr min
```

```
  | Sast.Empty -> "\"empty\""



let rec string_of_stmt_list = function
  []      -> ""
  | hd :: tl  ->
    let rec string_of_stmt = function
      | Sast.Expr(expr) -> let _,expr = expr in string_of_expr expr
      | Sast.Return (expr) -> let _,expr = expr in "return " ^ string_of_expr expr ^ ";"
      | Sast.If (e, s1, s2) -> let _,e = e in
         "if(" ^ string_of_expr e ^ ") {" ^ string_of_stmt_list (List.rev s1) ^ "}" ^ (if
s2=[] then "" else "else{" ^ string_of_stmt_list (List.rev s2) ^ "}")
      | Call(f, args)->
                if f = "print" then
                    let _,a0 = List.nth args 0 in
                    "printf(\"%s\\n\"," ^ string_of_expr a0 ^ ");"
                else if f = "place" then
                    let _,a0 = List.nth args 0 and _,a1=List.nth args 1 and _,a2= List.nth
args 2 in
                    "addPiece(" ^ string_of_expr a0 ^ "," ^ string_of_expr a1 ^ "," ^
string_of_expr a2
                      ^ ");"
                else if f = "removePiece" then
                    let _,a0 = List.nth args 0 and _,a1=List.nth args 1 in
                      "removePiece(" ^ string_of_expr a0 ^ "," ^ string_of_expr a1 ^ ");"
                else if f="remove" then "removePiece(x,y);"
                else if f="visible" then
                  let _,a0 = List.nth args 0 in
                    "v[x][y]=" ^ string_of_expr a0 ^ ";"
                else if f="setVisibility" then
                  let _,a0 = List.nth args 0 and _,a1 = List.nth args 1 and _,a2 = List.nth
args 2 in
                    "v[" ^ string_of_expr a1 ^"]["^string_of_expr a2 ^"]=" ^ string_of_expr a0
^ ";"

                else if f="click" then
                  let _,a0 = List.nth args 0 and _,a1=List.nth args 1 in
                  "clickSim(" ^ string_of_expr a0 ^ "," ^ string_of_expr a1 ^ ");"
                else if f="changeType" then
                  let _,a0 = List.nth args 0 in
                ( "removePiece(x,y); addPiece(" ^ string_of_expr a0 ^ ",x,y);")
                else if f="endTurn" then ("turnChange = 1;")
                else if f="break" then ("isBroken=1; break;")


                else failwith("Undefined function called! Semantic checker failed")
      | Assign (s, e) ->
        let _,e = e in
```

```
      s ^ "UDV=" ^ string_of_expr e ^ ";"
    | VDecl (id, e) ->
      let _,e=e and t,_=e in
      string_of_typ t ^ " " ^ id ^ "UDV = " ^ string_of_expr e ^ ";"
    | ForSurrounding(variable, r, c, s) ->
        let _,r=r and _,c=c in let r= string_of_expr r and c = string_of_expr c in
        " {int isBroken=0; int atemp, btemp;
          for(atemp=max(" ^ r ^ "-1,0);atemp<min(WIDTH," ^ r ^ "+2);atemp++){
            for(btemp=max(0," ^ c ^ "-1); btemp<min(HEIGHT," ^ c ^ "+2);btemp++){
              if(isBroken) break;
              int xx = atemp; int yy = btemp;
                if(!(atemp==" ^ r ^ "&& btemp==" ^ c ^")){ "^
                string_of_stmt_list (List.rev s)
          ^ "}
            if(isBroken) break;
          }
        }}"
    | ForRow(variable, r, s) ->
        let _,r=r in let r = string_of_expr r in
        "{int isBroken=0; int ctemp;
        for (ctemp=0;ctemp<WIDTH;ctemp++){ if(isBroken) break;
        int xx=ctemp; int yy=" ^ r ^ "; " ^
        string_of_stmt_list (List.rev s)^ "}}"

    | ForCol(variable, r, s) ->
        let _,r=r in let r = string_of_expr r in
        "{int isBroken=0; int dtemp;
        for (dtemp=0;dtemp<HEIGHT;dtemp++){
        if(isBroken) break;
        int yy=dtemp; int xx=" ^ r ^ "; " ^
        string_of_stmt_list (List.rev s)^ "}}"
    | ForSurroundingCoords (varX, varY, r, c, s) ->
        let _,r=r and _,c=c in let r= string_of_expr r and c = string_of_expr c in
        " {int isBroken=0; int etemp, ftemp;
          for(etemp=max(" ^ r ^ "-1,0);etemp<min(WIDTH," ^ r ^ "+2);etemp++){
            for(ftemp=max(0," ^ c ^ "-1); ftemp<min(HEIGHT," ^ c ^ "+2);ftemp++){
            if(isBroken) break;
                if(!(etemp==" ^ r ^ "&& ftemp==" ^ c ^")){
                  int "^varX ^ "UDV=etemp; int "^ varY ^"UDV=ftemp; " ^
                string_of_stmt_list (List.rev s)
          ^ "}
            if(isBroken) break;
            }
        }}"
    | ForMinMax (variable,min,max, s) ->
      let _,min=min and _,max=max in let min=string_of_expr min and max = string_of_expr max
in
        "{int isBroken=0; int gtemp;
        for (gtemp=" ^ min ^ ";gtemp<=" ^ max ^ ";gtemp++){ if(isBroken) break;
        int " ^ variable ^ "UDV=gtemp;" ^
```

```
          string_of_stmt_list (List.rev s)^ "}}"
      | ForNested (v1, min1, max1, v2, min2, max2, s) ->
        let _,min1=min1 and _,max1=max1 and _,min2=min2 and _,max2=max2 in
        let min1=string_of_expr min1 and max1 = string_of_expr max1 and min2=string_of_expr
min2 and max2 = string_of_expr max2 in
          "{int isBroken=0; int htemp, itemp;
          for (htemp=" ^ min1 ^ ";htemp<=" ^ max1 ^ ";htemp++){" ^
              "for (itemp=" ^ min2 ^ ";itemp<=" ^ max2 ^ ";itemp++){ if(isBroken) break;
              int " ^
           v1 ^ "UDV=htemp; int " ^  v2 ^ "UDV=itemp;" ^
          string_of_stmt_list (List.rev s)^"}if(isBroken) break;}}"
      | ForBoard(varX, varY,s) ->
        "{int isBroken=0; int jtemp, ktemp;
        for (jtemp=0; jtemp<WIDTH;jtemp++){" ^
              "for (ktemp=0;ktemp<HEIGHT;ktemp++){if(isBroken) break;
               int xx=jtemp; int yy=ktemp; int " ^
         varX ^ "UDV=jtemp; int " ^  varY ^ "UDV=ktemp;" ^
          string_of_stmt_list (List.rev s)^"}if(isBroken) break;}}"
      | ForAll (var, s) ->
          "{ int isBroken=0; int ltemp, mtemp;
          for (ltemp=0; ltemp<WIDTH;ltemp++){" ^
              "for (mtemp=0;mtemp<HEIGHT;mtemp++){ if(isBroken) break;
              int xx=ltemp; int yy=mtemp;" ^
          string_of_stmt_list (List.rev s)^"}if(isBroken) break;}}"

    in (string_of_stmt hd) ^ (string_of_stmt_list tl)

let string_of_board (board) =
  let _,x =board.x and _,y=board.y and _,bg=board.img in
  let x=string_of_expr x and y=string_of_expr y and bg = string_of_expr bg
  in
  "const int WIDTH=" ^ x ^ "; const int HEIGHT=" ^ y ^ "; const char* backgroundImg=" ^ bg ^
"; char * b [" ^ x ^ "][" ^ y ^ "]; int v[" ^x ^ "][" ^ y ^ "];"

let rec string_of_players = function
  [] -> ""
  | hd::[] -> hd
  | hd::tl ->
      let string_of_player = hd ^ ","
    in string_of_player ^ string_of_players tl


let playerHeader (players) = "char * players [" ^ (string_of_int (List.length players)) ^
"]={" ^ (string_of_players players) ^ "};"
let turnOrderHeader (turnOrder) = "char* turnOrder [" ^ (string_of_int (List.length
turnOrder)) ^ "]={" ^ (string_of_players turnOrder) ^ "};"


let rec piecesHeader = function
  [] -> ""
```

```
  |hd::tl ->
      let pieceHeader (piece) =
        let name = piece.name and img = piece.img in
        let name = String.sub name 1 ((String.length name)-2) in
      "const char * " ^ name ^ "IMAGE = " ^ img^ ";" ^
      "SDL_Texture * " ^ name ^ "Texture = NULL; void " ^
      name ^ "OnClick(int x, int y); void " ^ name ^ "OnTurn(); void " ^ name
      ^ "OnClick(int x, int y){" ^ (string_of_stmt_list (List.rev piece.onClick_typed)) ^ "}"
^ "void "^ name ^ "onTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) { if(strcmp(b[x][y],\"" ^ name ^
"\")==0){" ^ string_of_stmt_list (List.rev piece.onTurn_typed)
            ^ "}}}}"
      in pieceHeader hd ^ piecesHeader tl
let rec piecesInit = function
  [] -> ""
  |hd::tl ->
      let pieceInit (piece) =
        let name = piece.name  in
        let name = String.sub name 1 ((String.length name)-2) in
      name ^ "Texture = loadTexture(" ^ name ^ "IMAGE" ^ ");"
      ^ "if(" ^ name ^ "Texture == NULL ){ printf( \"Failed to load texture image!\\n\"
);success = 0;}"
      in pieceInit hd ^ piecesInit tl
let rec piecesClose = function
  [] -> ""
  |hd::tl ->
      let pieceClose (piece) =
        let name = piece.name in
        let name = String.sub name 1 ((String.length name)-2) in
      "SDL_DestroyTexture(" ^ name ^ "Texture);" ^ name ^ "Texture=NULL;"
    in pieceClose hd ^ piecesClose tl
let rec piecesToTexture = function
  [] -> "return NULL;"
  |hd::tl ->
      let pieceToTexture (piece) =
          let name = piece.name in
          let name = String.sub name 1 ((String.length name)-2) in
          "if(strcmp(piecename, \"" ^ name ^ "\")==0) return " ^ name ^ "Texture; \n else "
    in pieceToTexture hd ^ piecesToTexture tl

let rec callOnTurn = function
  [] -> ""
  |hd::tl ->
      let helper (piece) =
        let name = piece.name  in
        let name = String.sub name 1 ((String.length name)-2) in
        name^"onTurn();"
      in callOnTurn tl ^ helper hd

let textureMatchFunction (pieces) = "SDL_Texture * getTexture(char* piecename){" ^
```

```
piecesToTexture pieces ^ "}"


let rec piecesToEvent = function
  [] -> "return;"
  | hd::tl ->
    let pieceToEvent (piece) =
      let name = piece.name in
      let name = String.sub name 1 ((String.length name)-2) in
    "if (strcmp (b[x][y], \"" ^ name ^ "\")==0){" ^ name ^ "OnClick(x,y);} else "
  in pieceToEvent hd ^ piecesToEvent tl
let handleEventFunction (pieces) = "void clickSim(int x, int y) {" ^ piecesToEvent pieces ^
"}"


let rec string_of_setup (setup) = string_of_stmt_list setup

let endFunction (sList, str) = "int " ^ str ^ "(){int atemp, btemp,ctemp,
dtemp,etemp,ftemp,gtemp,htemp,itemp,jtemp,ktemp,ltemp,mtemp,isBroken; " ^ (if sList=[] then
"return 0;" else string_of_stmt_list (List.rev sList)) ^ "}"

let string_of_conditions (conditions) = endFunction(conditions.win_typed, "win") ^
endFunction(conditions.lose_typed, "lose") ^ endFunction(conditions.draw_typed, "draw")

(*prints program. currently does not consider anything besides statments. must accomodate our
structures*)

let header (board, players, turnOrder, pieces) = "#include <SDL2/SDL.h>
#include <SDL2/SDL_image.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
const int SCREEN_WIDTH = 600; const int SCREEN_HEIGHT = 600; const int SCREEN_BPP = 32; const
int DISPLAY_HEIGHT = 000;
int init();
int loadMedia();
void closeSDL();
SDL_Texture * getTexture (char * piecename);
void renderPiece(char * piecename, int x, int y);
int addPiece(char * piecename, int x, int y);
int removePiece(int x, int y);
SDL_Texture * loadTexture (char * filename);
void handleEvent( SDL_Event* e );
int xCord(int x);
int yCord(int y);
void clickSim(int x, int y);
int win();
int lose();
int draw();
int min();
```

```
int max();
SDL_Surface * backgroundTex = NULL;
SDL_Window* gWindow = NULL;
SDL_Surface* gRenderer = NULL;
int turnChange = 0;
char * turn;" ^(string_of_board board) ^ (piecesHeader pieces) ^ (playerHeader players) ^
(turnOrderHeader turnOrder)

let staticFunctions = "
void handleEvent( SDL_Event* e )
{
    //Get mouse position
    int x, y;
    SDL_GetMouseState( &x, &y );
    x=xCord(x); y=yCord(y);
   clickSim(x,y);
}
void renderPiece(char * piecename, int x, int y){
  SDL_Rect myRect = { x * SCREEN_WIDTH/WIDTH, y * SCREEN_HEIGHT/HEIGHT, SCREEN_WIDTH/WIDTH,
SCREEN_HEIGHT/HEIGHT};
  SDL_RenderCopy( gRenderer, getTexture(piecename), NULL,  &(myRect) );
}
int addPiece(char * piecename, int x, int y){
  if(strcmp(b[x][y],\"empty\")!=0){
      return 0;
  }
  b[x][y] = piecename;
  v[x][y] = 1;
  return 1;
}
int removePiece (int x, int y){
  if (strcmp(b[x][y],\"empty\")==0){
    return 0;
  }
  b[x][y] = \"empty\";
  v[x][y] = 0;
  return 1;
}
SDL_Texture* loadTexture( char * filename )
{
  SDL_Texture* newTexture = NULL;
  SDL_Surface* loadedSurface = IMG_Load( filename );
  if( loadedSurface == NULL )
  {
    printf( \"Unable to load image %s! SDL_image Error: %s\\n\", filename, IMG_GetError() );
  }
  else
  {
        newTexture = SDL_CreateTextureFromSurface( gRenderer, loadedSurface );
    if( newTexture == NULL )
```

```
    {
      printf( \"Unable to create texture from %s! SDL Error: %s\\n\", filename, SDL_GetError()
);
    }
    SDL_FreeSurface( loadedSurface );
  }
  return newTexture;
}


int init()
{
  int success = 1;
  turn = turnOrder[0];

  int i,j;
  for(i = 0; i< sizeof(b) / sizeof((b)[0]) ; i++){
    for(j=0; j < sizeof(b[i])/sizeof(b[i][0]); j++){
      b[i][j]=\"empty\";
      v[i][j] = 0;
    }
  }

  if( SDL_Init( SDL_INIT_VIDEO ) < 0 )
  {
    printf( \"SDL could not initialize! SDL Error: %s\\n\", SDL_GetError() );
    success = 0;
  }
  else
  {
    if( !SDL_SetHint( SDL_HINT_RENDER_SCALE_QUALITY, \"1\" ) )
    {
      printf( \"Warning: Linear texture filtering not enabled!\" );
    }
    gWindow = SDL_CreateWindow( \"Game\", SDL_WINDOWPOS_UNDEFINED, SDL_WINDOWPOS_UNDEFINED,
SCREEN_WIDTH, SCREEN_HEIGHT + DISPLAY_HEIGHT, SDL_WINDOW_SHOWN );
    if( gWindow == NULL )
    {
      printf( \"Window could not be created! SDL Error: %s\\n\", SDL_GetError() );
      success = 0;
    }
    else
    {
      gRenderer = SDL_CreateRenderer( gWindow, -1, SDL_RENDERER_ACCELERATED );
      if( gRenderer == NULL )
      {
        printf( \"Renderer could not be created! SDL Error: %s\\n\", SDL_GetError() );
        success = 0;
      }
      else
```

```
      {
        SDL_SetRenderDrawColor( gRenderer, 0xFF, 0xFF, 0xFF, 0xFF );
        int imgFlags = IMG_INIT_PNG;
        if( !( IMG_Init( imgFlags ) & imgFlags ) )
        {
          printf( \"SDL_image could not initialize! SDL_image Error: %s\\n\", IMG_GetError()
);
          success = 0;
        }
      }
    }
  }

  return success;
}

int xCord(int x){
  int i;
  for(i=0;i<WIDTH;i++){
    if(x>i && x<(i+1)*SCREEN_WIDTH/WIDTH){
      return i;
    }
  }

  return -1;
}

int yCord(int y){
  int i;
  for(i =0; i<HEIGHT; i++){
    if(y>i && y<(i+1)*SCREEN_HEIGHT/HEIGHT){
      return i;
    }
  }
  return -1;
}

int min(int a, int b){
  if (a<b)
    return a;
  return b;
}

int max(int a, int b){
  if (a>b)
    return a;
  return b;
}

"
```

```
let loadAndCloseFunctions (pieces)= "int loadMedia(){int success = 1;backgroundTex =
loadTexture( backgroundImg );if( backgroundTex == NULL ){printf( \"Failed to load texture
image!\\n\" );success = 0;
}" ^ piecesInit pieces^"return success;}" ^ "void closeSDL(){SDL_DestroyTexture( backgroundTex
);backgroundTex = NULL;" ^ piecesClose pieces ^ "SDL_DestroyRenderer( gRenderer );
SDL_DestroyWindow( gWindow );
gWindow = NULL;gRenderer = NULL; IMG_Quit();SDL_Quit();}"


let mainFunction (pieces, setup)= "int main( int argc, char* args[] )
{
int atemp, btemp,ctemp, dtemp,etemp,ftemp,gtemp,htemp,itemp,jtemp,ktemp,ltemp,mtemp,
isBroken;
  if( !init() )
  {
    printf( \"Failed to initialize!\\n\" );
  }
  else
  {   if( !loadMedia() )
    {
      printf( \"Failed to load media!\\n\" );
    }
    else
    {
      int quit = 0;" ^ string_of_setup setup ^ " SDL_Event e;
      int turnCounter = 0;


  while( !quit && !win() && !lose() && !draw())
      {
        while( SDL_PollEvent( &e ) != 0 )
        {
          if( e.type == SDL_QUIT )
          {
            quit = 1;
          }
          else if( e.type == SDL_MOUSEBUTTONDOWN /*|| e->type == SDL_MOUSEBUTTONUP*/ ){
              handleEvent(&e);
          }
        }
        if(turnChange){
          turnChange = 0;
          turnCounter = (turnCounter + 1) % ( sizeof(turnOrder) / sizeof(turnOrder[0]));
          turn = turnOrder[turnCounter];"
            ^ callOnTurn pieces^
        "}
        SDL_SetRenderDrawColor( gRenderer, 0xFF, 0xFF, 0xFF, 0xFF );
        SDL_RenderClear( gRenderer );
```

```
      SDL_Rect board;
      board.x = 0; board.y = 0;
      board.w = SCREEN_WIDTH; board.h=SCREEN_HEIGHT;
      SDL_RenderSetViewport(gRenderer, &board);

      SDL_RenderCopy( gRenderer, backgroundTex, NULL, NULL );
      int i, j;
      for(i=0;i<=HEIGHT;i++)
      {
        //y location at i*screenheight/height
        int y = i * SCREEN_HEIGHT/HEIGHT;
        if (y==SCREEN_HEIGHT)
          y-=1;
        SDL_SetRenderDrawColor( gRenderer, 0x00, 0x00, 0x00, 0xFF );
        SDL_RenderDrawLine( gRenderer, 0, y, SCREEN_WIDTH, y );

      }
      for(i=0;i<=WIDTH;i++)
      {

        int x = i * SCREEN_WIDTH/WIDTH;
        if (x==SCREEN_WIDTH)
          x-=1;
        SDL_SetRenderDrawColor( gRenderer, 0x00, 0x00, 0x00, 0xFF );
        SDL_RenderDrawLine( gRenderer, x, 0, x, SCREEN_HEIGHT );
      }
      for(i = 0; i< sizeof(b) / sizeof((b)[0]) ; i++){
        for(j=0; j < sizeof(b[i])/sizeof(b[i][0]); j++){
          if(strcmp(b[i][j],\"empty\")!=0 && v[i][j]==1){
            renderPiece(b[i][j],i,j);
          }
        }
      }
      SDL_Rect messageDisplay;
      messageDisplay.x=SCREEN_HEIGHT; messageDisplay.y = 0;
      messageDisplay.w = SCREEN_WIDTH; messageDisplay.h = DISPLAY_HEIGHT;
      SDL_RenderSetViewport (gRenderer, &messageDisplay);



      SDL_RenderPresent( gRenderer );



    }
  }
}
closeSDL();

return 0;
```

```
}"


let printprog (board, players, turnOrder, pieces, setup, conditions) =
  let oc = open_out file in
      fprintf oc "%s" (header (board,players, turnOrder, pieces));
            fprintf oc "%s" (textureMatchFunction pieces);
            fprintf oc "%s" staticFunctions;
            fprintf oc "%s" (loadAndCloseFunctions pieces);
            fprintf oc "%s" (handleEventFunction pieces);
            fprintf oc "%s" (string_of_conditions conditions);
            fprintf oc "%s" (mainFunction (pieces,(List.rev setup)));

  close_out oc;
```

gridlok.ml

```
(* Authors: Laura, Julian *)
open Printf


let _ =
    let lexbuf = Lexing.from_channel stdin in
     let ast = Parser.program Scanner.token lexbuf in
         let prog = Semant.semcheck ast in
                    Codegen.printprog prog;
```

parser.mly

```
/* Author: Laura */
%{
open Ast
%}
%token START BOARDDEF DIM IMG PLAYERS TURNORDER PIECE NAME ONTURN ONCLICK SETUP WIN LOSE DRAW
IN SURROUNDING ROW COL ALL EMPTY RAND
%token LPAREN RPAREN LBRACE RBRACE LBRACK RBRACK COMMA
%token PLUS MINUS TIMES DIVIDE MOD ASSIGN NOT DECLARE
%token EQ NEQ LT LEQ GT GEQ TRUE FALSE AND OR STREQ PERIOD
%token RETURN IF ELSE FOR WHILE INT BOOL STR

%token <string> ID
%token <string> STRING_LITERAL
%token <int> INT_LITERAL
%token <bool> BOOL_LITERAL
%token EOF

%nonassoc NOELSE
%nonassoc ELSE
%right ASSIGN DECLARE
%left OR
%left AND
```

```
%left EQ NEQ STREQ
%left LT GT LEQ GEQ
%left PLUS MINUS
%left TIMES DIVIDE MOD PERIOD
%right NOT

%start program
%type <Ast.program> program


%%


program:
  START LBRACE decl RBRACE EOF { $3 }


decl:
  board_decl players_decl turnOrder_decl piece_list setup conditions {$1, $2, $3, $4, $5, $6}

setup:
  SETUP LBRACE stmt_list RBRACE { $3 }

conditions:
  /*nothing*/ {{win=[]; lose=[]; draw=[]; w=false; l=false; d=false;}}
  | winCon  {{win=$1; lose=[]; draw=[]; w=true; l=false;d=false;}}
  | loseCon {{win=[]; lose=$1; draw=[];w=false;l=true;d=false;}}
  | drawCon {{win=[]; lose=[]; draw=$1;w=false;l=false;d=true;}}
  | winCon loseCon  {{win=$1; lose=$2; draw=[];w=true;l=true;d=false;}}
  | winCon drawCon  {{win=$1; lose=[]; draw=$2;w=true;l=false;d=true;}}
  | loseCon winCon  {{win=$2; lose=$1; draw=[];w=true;l=true;d=false;}}
  | loseCon drawCon {{win=[]; lose=$1; draw=$2; w=false;l=true;d=true;}}
  | drawCon winCon  {{win=$2; lose=[]; draw=$1;w=true;l=false;d=true;}}
  | drawCon loseCon {{win=[]; lose=$2; draw=$1;w=false;l=true;d=true;}}
  | winCon loseCon drawCon  {{win=$1; lose=$2; draw=$3;w=false;l=false;d=true;}}
  | winCon drawCon loseCon  {{win=$1; lose=$3; draw=$2;w=true;l=true;d=true;}}
  | loseCon winCon drawCon  {{win=$2; lose=$1; draw=$3;w=true;l=true;d=true;}}
  | loseCon drawCon winCon  {{win=$3; lose=$1; draw=$2;w=true;l=true;d=true;}}
  | drawCon loseCon winCon  {{win=$3; lose=$2; draw=$1;w=true;l=true;d=true;}}
  | drawCon winCon loseCon  {{win=$2; lose=$3; draw=$1;w=true;l=true;d=true;}}


winCon:
  WIN LBRACE stmt_list RBRACE { $3 }

loseCon:
  LOSE LBRACE stmt_list RBRACE { $3 }

drawCon:
  DRAW LBRACE stmt_list RBRACE { $3 }
```

```
board_decl:
  BOARDDEF LBRACE DIM LBRACE expr COMMA expr RBRACE IMG LBRACE expr RBRACE RBRACE
  {
    {
      x = $5;
      y = $7;
      bg = $11;
    }
  }


players_decl:
  PLAYERS LBRACE string_list RBRACE {List.rev $3}

turnOrder_decl:
  TURNORDER LBRACE string_list RBRACE {List.rev $3}

piece_list:
  /*nothing */ {[]}
  | piece_list piece_decl {$2 :: $1}

piece_decl:
  PIECE LBRACE NAME LBRACE STRING_LITERAL RBRACE IMG LBRACE STRING_LITERAL RBRACE ONTURN
LBRACE stmt_list RBRACE ONCLICK LBRACE stmt_list RBRACE RBRACE
  {
    {
      name = $5;
      img = $9;
      onTurn = $13;
      onClick = $17;
    }

  }


string_list:
  /* nothing */ {[]}
  | STRING_LITERAL {$1::[]}
  | string_list COMMA STRING_LITERAL  {$3 :: $1}

stmt_list:
    /* nothing */  { [] }
  | stmt_list stmt { $2 :: $1 }

stmt:
  | expr { Expr $1 }
  | RETURN LBRACE expr RBRACE { Return $3 }
  | IF LPAREN expr RPAREN LBRACE stmt_list RBRACE %prec NOELSE { If($3, $6, []) }
  | IF LPAREN expr RPAREN LBRACE stmt_list RBRACE ELSE LBRACE stmt_list RBRACE    { If($3, $6,
```

```
$10) }
  | FOR ID IN ROW LPAREN expr RPAREN LBRACE stmt_list RBRACE {ForRow($2,$6,$9)}
  | FOR ID IN COL LPAREN expr RPAREN LBRACE stmt_list RBRACE {ForCol($2,$6,$9)}
  | FOR ID SURROUNDING LPAREN expr COMMA expr RPAREN LBRACE stmt_list RBRACE
{ForSurrounding($2, $5, $7, $10)}
  | FOR LPAREN ID COMMA ID RPAREN SURROUNDING LPAREN expr COMMA expr RPAREN LBRACE stmt_list
RBRACE {ForSurroundingCoords($3, $5, $9, $11, $14)}
  | FOR ID LPAREN expr COMMA expr RPAREN LBRACE stmt_list RBRACE {ForMinMax($2,$4,$6,$9)}
  | FOR LPAREN ID COMMA ID RPAREN IN BOARDDEF LBRACE stmt_list RBRACE {ForBoard($3, $5,$10)}
  | FOR ID LPAREN expr COMMA expr RPAREN COMMA ID LPAREN expr COMMA expr RPAREN LBRACE
stmt_list RBRACE {ForNested($2, $4, $6,$9,$11,$13,$16)}
  | FOR ALL ID IN BOARDDEF LBRACE stmt_list RBRACE {ForAll($3,$7)}
  | ASSIGN ID LBRACE expr RBRACE { Assign($2, $4) }
  | ID LBRACE actuals_opt RBRACE { Call($1, $3) }
  | DECLARE typ ID LBRACE expr RBRACE {VDecl($2, $3, $5)}

expr:
    STRING_LITERAL              { StringLiteral($1) }
  | BOOL_LITERAL                { BoolLiteral ($1) }
  | INT_LITERAL                 { IntLiteral ($1)}
  | ID                          { Id($1) }
  | EMPTY                       {Empty}
  | LPAREN expr RPAREN       { Parenth($2) }
  | BOARDDEF LBRACK expr RBRACK LBRACK expr RBRACK {BoardAccess($3,$6)}
  | expr PLUS   expr { Binop($1, Add,   $3) }
  | expr MINUS  expr { Binop($1, Sub,   $3) }
  | expr TIMES  expr { Binop($1, Mult,  $3) }
  | expr DIVIDE expr { Binop($1, Div,   $3) }
  | expr MOD    expr { Binop($1, Mod,   $3) }
  | expr EQ     expr { Binop($1, Equal, $3) }
  | expr STREQ  expr { Binop($1, StrEqual,$3)}
  | expr NEQ    expr { Binop($1, Neq,   $3) }
  | expr LT     expr { Binop($1, Less,  $3) }
  | expr LEQ    expr { Binop($1, Leq,   $3) }
  | expr GT     expr { Binop($1, Greater, $3) }
  | expr GEQ    expr { Binop($1, Geq,   $3) }
  | expr AND    expr { Binop($1, And,   $3) }
  | expr OR     expr { Binop($1, Or,    $3) }
  | NOT         expr { Unop (Not, $2)}
  | ID PERIOD ID {Access($1,$3)}
  | RAND LBRACE expr COMMA expr RBRACE {Rand($3,$5)}

typ:
  INT {Int}
  | BOOL {Bool}
  | STR  {Str}


actuals_opt:
    /* nothing */ { [] }
```

```
  | actuals_list  { List.rev $1 }

actuals_list:
    expr                    { [$1] }
  | actuals_list COMMA expr { $3 :: $1 }
```

sast.ml

```
(* Author: Laura *)
open Ast


type var_ref = string
type var_new = string
type func_name = string
type obj_field = string

type expr_detail =
       | Parenth of expr_with_type
       | Id of var_ref
       | BoolLiteral of bool
       | IntLiteral of int
       | StringLiteral of string
       | Binop of expr_with_type * Ast.op * expr_with_type
       | Unop of Ast.uop * expr_with_type
       | Access of var_ref * obj_field
       | BoardAccess of expr_with_type * expr_with_type
       | Rand of expr_with_type * expr_with_type
       | Empty
       and expr_with_type = Ast.typ * expr_detail



(*statements: things that you can do with expressions, things that dictate control flow.*)
type stmt_detail =
   Expr of expr_with_type
  | Return of expr_with_type
  | If of expr_with_type * stmt_detail list * stmt_detail list
  (* for varName surrounding (int x, int y) *)
  | ForSurrounding of var_new * expr_with_type * expr_with_type * stmt_detail list
  (*for varName in row(int rowNum)*)
  | ForRow of var_new * expr_with_type * stmt_detail list
  (* for varName in col(int colNum) *)
  | ForCol of var_new * expr_with_type * stmt_detail list
  (* for (int i, int j) surrounding (int x, int y) *)
  | ForSurroundingCoords of var_new * var_new * expr_with_type * expr_with_type * stmt_detail
list
  (* for varName(int min, int max) *)
  | ForMinMax of var_new * expr_with_type * expr_with_type * stmt_detail list
  (* for varName(int min, int max), varName2(int min2, int max2) *)
  | ForNested of var_new * expr_with_type * expr_with_type * var_new * expr_with_type *
```

```
expr_with_type * stmt_detail list
  (* for (int x, int y) in board *)
  | ForBoard of var_new * var_new * stmt_detail list
  | ForAll of var_new * stmt_detail list
  | Assign of var_ref * expr_with_type
  | VDecl of var_new * expr_with_type
  | Call of func_name * expr_with_type list

type board_typed = {
       x: expr_with_type;
       y: expr_with_type;
       img: expr_with_type;
}

(*piece definition*)
type piece_typed = {
       name: string;
       img: string;
       onTurn_typed: stmt_detail list;
       onClick_typed: stmt_detail list;
}

(*instructions for the setup of the board, rendered before any turns are completed*)
type setup_typed = stmt_detail list

(*conditions that end the game, specified as functions that return booleans. if not specified,
we assume the functions indefinitely return 0.*)
type conditions_typed = {
       win_typed: stmt_detail list;
       lose_typed: stmt_detail list;
       draw_typed: stmt_detail list;
}
```

scanner.mll

```
(* Author: Laura *)
{ open Parser }

rule token = parse
  [' ' '\t' '\r' '\n'] { token lexbuf } (* Whitespace *)
  | "/*"     { comment lexbuf }
| "game"{ START }
| "def"{ DECLARE }
| "set" { ASSIGN }
| "board"      {BOARDDEF}
| "dimensions" { DIM }
| "image"      { IMG }
| "players" { PLAYERS }
| "turnOrder" {TURNORDER}
| "piece"      { PIECE }
| "setup"      { SETUP }
```

```
| "winCondition" { WIN }
| "loseCondition" { LOSE }
| "drawCondition"     { DRAW }
| "name"       { NAME }
| "onTurn"     { ONTURN }
| "onClick"    { ONCLICK }
| "rand"    { RAND }
| "if"         { IF }
| "else"       { ELSE }
| "for"        { FOR }
| "in"         { IN }
| "all"        { ALL }
| "surrounding" {SURROUNDING}
| "row"        {ROW}
| "col"        {COL}
| "return"     { RETURN }
| "int"        {INT}
| "str"        {STR}
| "bool"       {BOOL}
| '{'  { LBRACE }
| '}'  { RBRACE }
| ','  { COMMA }
| '('  { LPAREN }
| ')'  { RPAREN }
| '['  { LBRACK }
| ']'  { RBRACK }
| '+'  { PLUS }
| '-'  { MINUS }
| '*'  { TIMES }
| '/'  { DIVIDE }
| '%'  { MOD }
| "empty" {EMPTY}
| "==" { STREQ }
| "="  { EQ }
| "!=" { NEQ }
| '<'  { LT }
| "<=" { LEQ }
| '>'  { GT }
| ">=" { GEQ }
| "AND"{ AND }
| "OR" { OR }
| "!"  { NOT }
| "."  { PERIOD }
| "true" | "false" as lxm { BOOL_LITERAL(bool_of_string lxm) }
| ['0'-'9']+ as lxm { INT_LITERAL(int_of_string lxm) }
| '"'('\\'_|[^'"'])*'"' as lxm { STRING_LITERAL(lxm) }
| ['a'-'z' 'A'-'Z']['a'-'z' 'A'-'Z' '0'-'9' '_']* as lxm { ID(lxm) }
| eof  { EOF }
| _ as char { raise (Failure("illegal character " ^ Char.escaped char)) }
```

```
and comment = parse
  "*/" { token lexbuf }
| _    { comment lexbuf }
```

semant.ml

```
(* Authors: Laura, Julian *)
open Sast
open Ast


(*environment*)
type symbol_table = {
    parent: symbol_table option;
    mutable variables: (Ast.typ * string) list;
    mutable declFuncs: (Ast.typ * string * (Ast.typ list)) list; (*list of usable functions:
type, name, list of arguement types*)
}

type environment = {
    scope: symbol_table;
    func: Ast.typ; (*the scope's return type*)
    mutable returned: bool; (*does the scope have a return statement*)
    mutable pieceNames: string list;
    inLoop: bool; (*boolean that tells if you are in a loop. determines if break can be used*)

}

(*rewrite find variables
vdecl: find variable, if already found, raise error. otherwise, add to table*)

(*code for scope checking taken from edward's slides, can be changed*)
let rec var_local (scope: symbol_table) name =
    List.exists(fun (_,s) -> s = name) scope.variables

let rec find_variable (scope: symbol_table) name =
    try List.find (fun (_, s) -> s = name) scope.variables
    with Not_found ->
        match scope.parent with
        | Some(parent) -> find_variable parent name
        | None -> raise Not_found

let rec find_function (scope: symbol_table) name =
    try List.find (fun (_, s, _) -> s = name) scope.declFuncs
    with Not_found ->
        match scope.parent with
        | Some(parent) -> find_function parent name
        | None -> raise Not_found

(**)
let rec expr env = function
```

```
    | Ast.BoolLiteral(b)    -> Ast.Bool, Sast.BoolLiteral(b)
    | Ast.IntLiteral (i)    -> Ast.Int, Sast.IntLiteral(i)
    | Ast.StringLiteral(s)  -> Ast.Str, Sast.StringLiteral(s)
    | Ast.Binop (e1,o,e2)   ->
        let e1 = expr env e1 and e2 = expr env e2 in  (*evaluates left and right hand
expressions*)
        let t1, _ = e1 and t2, _ = e2 in (*Obtain types of left and right expressions*)
        let opTypeFail expectedType =
            failwith (Ast.string_of_op o ^ " is only for defined for " ^ expectedType)
        in (match o with
            | Ast.Add | Ast.Sub | Ast.Mult | Ast.Div | Ast.Mod ->
                if t1=t2 && t1=Ast.Int then Ast.Int, Sast.Binop(e1,o,e2)
                else opTypeFail "int"
            | Ast.Less | Ast.Greater | Ast.Leq | Ast.Geq ->
                if t1=t2 && t1=Ast.Int then Ast.Bool, Sast.Binop(e1,o,e2)
                else opTypeFail "int"
            | Equal | Neq ->
                if t1=t2 && (t1=Ast.Int || t1=Ast.Bool || t1=Ast.Piece || t1=Ast.Str) then
Ast.Bool, Sast.Binop(e1,o,e2)
                else opTypeFail "int and int, or bool and bool, or piece and piece, or string
and string"
            | And | Or ->
                if t1=t2 && t1=Ast.Bool then Ast.Bool, Sast.Binop(e1,o,e2)
                else opTypeFail "bool"
            | StrEqual ->
                if t1=t2 && t1=Ast.Str then Ast.Bool, Sast.Binop(e1,o,e2)
                else opTypeFail "string"
        )
    | Ast.Unop (o,e) ->
        let e = expr env e in (*evaluate e*)
        let t,_ = e in(*find type of expression*)
            if t=Ast.Int then t, Sast.Unop(o,e) else failwith("- only defined for int; " ^
Ast.string_of_typ t ^ " found.")
    | Ast.Parenth (e) ->
        let e = expr env e in
        let t, _= e in
            t, Sast.Parenth(e)
    | Ast.Id(i) ->
        let var =
            try find_variable env.scope i
            with Not_found -> failwith("Uninitialised variable " ^ i ^ " referenced.")
        in let t,_ = var in
        if t=Ast.Piece then failwith ("Piece types may not be explicitly called. Do you mean
.type, .x, or .y?")
        else t, Sast.Id(i)
    | Ast.BoardAccess(x,y) ->
        let varX = expr env x and varY=expr env y in
        let tX,_=varX and tY,_=varY in
        if tX!=Ast.Int || tY!=Ast.Int then failwith ("Int types expected for board access.
Other types found.")
```

```
        else Ast.Str, Sast.BoardAccess(varX,varY)
    | Ast.Access(v,f) ->
        let var =
            try find_variable env.scope v
            with Not_found -> failwith("Uninitialised variable " ^ v ^ " referenced.")
        in let t,_ = var in
        if t!=Ast.Piece then failwith (v ^ " is not of type Piece. Invalid command.")
        else if f="visible" then Ast.Bool, Sast.Access(v,f)
        else if f="type" then Ast.Str, Sast.Access(v,f)
        else if f="x" then Ast.Int, Sast.Access(v,f)
        else if f="y" then Ast.Int, Sast.Access(v,f)
        else failwith ("Invalid record name access.")
    | Ast.Empty -> Ast.Str, Sast.Empty
    | Ast.Rand(min, max) ->
        let min= expr env min and max = expr env max in
        let tmin,_=min and tmax,_=max in
        if tmin=tmax && tmin=Ast.Int then Ast.Int, Sast.Rand(min,max)
        else failwith ("Parameters of random function must evaluate to integer values")

let rec typed_stmt_list env=  function
    []->[]
    |hd::tl ->
        let rec stmt env= function
        | Ast.VDecl (t,v,e) ->
            if t=Ast.Piece then failwith ("Variable with piece type cannot be declared.")
            else if v="x" || v="y" || v="turn" then failwith ("Attempt to assign to reserved
words x, y, or turn.")
            else if var_local env.scope v then failwith (v ^ " already declared in scope")
            else let e=expr env e in let t1,_=e in
            if t1!=t then failwith (Ast.string_of_typ t ^ " declared, " ^ Ast.string_of_typ t1
^ " initialised in expression")
            else env.scope.variables<-(t,v)::env.scope.variables; Sast.VDecl(v, e)
        | Ast.Assign(v,e) ->
        let var =
            if v="x" || v="y" || v="turn" then failwith ("Attempt to assign to reserved words
x, y, or turn.")
            else
            try find_variable env.scope v
            with Not_found -> failwith("Uninitialised variable " ^ v ^ " referenced.")
        in let e = expr env e and t,_ = var
        in let t1,_ = e
        in
        if t=t1 then Sast.Assign(v, e)
        else failwith ("Attempt to assign " ^ Ast.string_of_typ t1 ^ " to variable declared as
type " ^ string_of_typ t)
        | Ast.Call (fname, argsList) ->
            if fname = "break" && env.inLoop then Sast.Call(fname,[])
            else
            let f =
            try find_function env.scope fname
```

```
            with Not_found -> failwith ("Call to invalid/undeclared function.")
        in
        let  _,_,args=f in
            let numArgs = (List.length args) in
            if (List.length argsList)!=numArgs then failwith (string_of_int numArgs ^ "
arguments expected, " ^ string_of_int (List.length argsList) ^ " found.")
            else
                let rec checkList a ae = match a, ae with
                | [],[] -> []
                | _::_, [] | [],_::_-> failwith (":(")
                | hd::tl, hd1::tl1 -> (*hd has given expressions, hd1 is the type it should
evaluate to*)
                    let helper hd hd1 =
                        let e = expr env hd in let temptype,_=e in
                        if temptype!=hd1 then failwith ("Unexpected argument type.")
                        else e
                    in (helper hd hd1)::(checkList tl tl1)
                in Sast.Call(fname,checkList argsList args)
        | Ast.Expr(e) -> Sast.Expr(expr env e)
        | Ast.If(e, s1, s2) ->
            let e = expr env e in
            let t, _ = e in
            if t=Ast.Bool then
                let scope1 = {env.scope with parent=Some(env.scope); variables=[];} and
scope2={env.scope with parent=Some(env.scope); variables=[];} in
                let env1 = {env with scope=scope1} and env2 = {env with scope=scope2} in
                Sast.If(e, typed_stmt_list env1 s1, typed_stmt_list env2 s2)
            else failwith ("If predicate must be a boolean")
        | Ast.ForRow(v,e,s) ->
            let e = expr env e in let t,_=e in
                if t!=Ast.Int then failwith ("Conditional expression must evaluate to an int
value")
                else if var_local env.scope v then failwith (v ^ " already declared in local
scope")
                else
                    let scope' = {env.scope with parent=Some(env.scope);
variables=[(Ast.Piece,v);]} in
                    let env' = {env with scope=scope'; inLoop=true} in
                    Sast.ForRow(v, e, typed_stmt_list env' s)
        | Ast.ForCol(v,e,s) ->
            let e = expr env e in let t,_=e in
                if t!=Ast.Int then failwith ("For expression must evaluate to an int value")
                else if var_local env.scope v then failwith (v ^ " already declared in local
scope")
                else
                    let scope' = {env.scope with parent=Some(env.scope);
variables=[(Ast.Piece,v)]} in
                    let env' = {env with scope=scope'; inLoop=true} in
                    Sast.ForCol(v, e, typed_stmt_list env' s)
        | Ast.ForAll (v, s) ->
```

```
            if var_local env.scope v then failwith (v ^ " already declared in local scope")
            else
                let scope' = {env.scope with parent=Some(env.scope);
variables=[(Ast.Piece,v)]} in
                let env' = {env with scope=scope'; inLoop=true} in
                Sast.ForAll(v,typed_stmt_list env' s)
        | Ast.ForSurrounding(v, e1, e2,s) ->
            let e1=expr env e1 and e2=expr env e2 in
            let t1,_=e1 and t2,_=e2 in
            if t1!=Ast.Int || t2!=Ast.Int then failwith("For loop: expressions must evaluate
to int values")
            else if var_local env.scope v then failwith (v ^ " already declared in local
scope")
            else
                let scope' = {env.scope with parent=Some(env.scope);
variables=[(Ast.Piece,v)]} in
                let env' = {env with scope=scope'; inLoop=true} in
                Sast.ForSurrounding(v, e1, e2, typed_stmt_list env' s)
        | Ast.ForMinMax(v,e1,e2,s) ->
            let e1=expr env e1 and e2=expr env e2 in
            let t1,_=e1 and t2,_=e2 in
            if t1!=Ast.Int || t2!=Ast.Int then failwith("For loop: expressions must evaluate
to int values")
            else if var_local env.scope v then failwith (v ^ " already declared in local
scope")
            else
                let scope' = {env.scope with parent=Some(env.scope); variables=[(Ast.Int,v)]}
in
                let env' = {env with scope=scope'; inLoop=true} in
                Sast.ForMinMax(v, e1, e2, typed_stmt_list env' s)
        | Ast.ForNested(v1,e11,e12,v2,e21,e22,s) ->
            let e11=expr env e11 and e12=expr env e12 and e21=expr env e21 and e22=expr env
e22 in
            let t11,_=e11 and t12,_=e12 and t21,_=e21 and t22,_=e22 in
            if t11!=Ast.Int || t12!=Ast.Int || t21!=Ast.Int || t22!=Ast.Int then failwith("For
loop: expressions must evaluate to int values")
            else if var_local env.scope v1 then failwith (v1 ^ " already declared in local
scope")
            else if var_local env.scope v2 then failwith (v2 ^ " already declared in local
scope")
            else
                let scope' = {env.scope with parent=Some(env.scope);
variables=[(Ast.Int,v1);(Ast.Int,v2)]} in
                let env' = {env with scope=scope'; inLoop=true} in
                Sast.ForNested(v1, e11, e12, v2, e21, e22, typed_stmt_list env' s)
        | Ast.ForSurroundingCoords(v1,v2,e1,e2,s) ->
            let e1=expr env e1 and e2=expr env e2 in
            let t1,_=e1 and t2,_=e2 in
            if t1!=Ast.Int || t2!=Ast.Int then failwith("For loop: expressions must evaluate
to int values")
```

```
            else if var_local env.scope v1 then failwith (v1 ^ " already declared in local
scope")
            else if var_local env.scope v2 then failwith (v2 ^ " already declared in local
scope")
            else
                let scope' = {env.scope with parent=Some(env.scope);
variables=[(Ast.Int,v1);(Ast.Int,v2)]} in
                let env' = {env with scope=scope'; inLoop=true} in
                Sast.ForSurroundingCoords(v1, v2, e1, e2, typed_stmt_list env' s)
        | Ast.ForBoard(v1,v2,s) ->
            if var_local env.scope v1 then failwith (v1 ^ " already declared in local scope")
            else if var_local env.scope v2 then failwith (v2 ^ " already declared in local
scope")
            else
                let scope' = {env.scope with parent=Some(env.scope);
variables=[(Ast.Int,v1);(Ast.Int,v2)]} in
                let env' = {env with scope=scope'; inLoop=true} in
                Sast.ForBoard(v1, v2, typed_stmt_list env' s)
        | Ast.Return(e) ->
            let e = expr env e in let t,_= e in
            if env.func=Ast.Void then failwith ("Return statement not allowed in void
functions.")
            else if t=env.func then (env.returned<-true; Sast.Return(e))
            else failwith("Return type does not match declared function return type.")
        in stmt env hd::typed_stmt_list env tl

let semcheck (board, players, turnOrder, pieces, setup, conditions) =
    (* check players *)
    let checkPlayers =
        (* check for at least one player *)
        (* check for duplicate player names, raise exception if found *)
        let reportError except1 except2 lst =
            if List.length lst < 1 then raise (Failure except1);
            let rec helper = function
                  n1 :: n2 :: _ when n1 = n2 -> raise (Failure (except2 n1))
                | _ :: t -> helper t
                | [] -> ()
            in helper(List.sort compare lst)
        in
        reportError "no players" (fun n -> "duplicate player name " ^ n)
        players
    in
    checkPlayers;

    (* check turnOrder *)
    let checkTurnOrder =
        (* check for at least one player in turnOrder *)
        (* check for any undefined players in turnOrder *)
        let reportError except1 except2 lst1 lst2 =
            if List.length lst2 < 1 then raise (Failure except1);
```

```
            let rec helper lst p = match lst with
                  [] -> 0
                | hd :: tl when hd = p -> 1
                | _ :: tl -> helper tl p
            in
            let rec check = function
                  [] -> 0
                | hd :: tl -> helper lst1 hd + check tl
            in
            if check lst2 < List.length lst2 then raise (Failure except2)
        in
        reportError "no players in turnOrder"
        "undefined players in turnOrder" players turnOrder
    in
    checkTurnOrder;


    let env = {scope = { variables=[(Ast.Int, "width");(Ast.Int, "height");]; parent=None;
                declFuncs=
[(Ast.Void,"setVisibility",[Ast.Int;Ast.Int;Ast.Int]);(Ast.Int,"rand",[Ast.Int;Ast.Int]);(Ast.
Void,"print",[Ast.Str]);(Ast.Void, "place",[Ast.Str;Ast.Int;Ast.Int]);
        (Ast.Void,"removePiece",[Ast.Int;Ast.Int]); (Ast.Void, "click",
[Ast.Int;Ast.Int]);(Ast.Int,"random",[Ast.Int;Ast.Int]) ]; };
        func=Ast.Void;returned=false; pieceNames=[]; inLoop=false;} in (*global environment
includes print, place, removePiece, click functions. return type is void. no variables, no
parent*)

    let checkBoard env board =
    let x = expr env board.x and y = expr env board.y and bg=expr env board.bg in
    let x_type,_ = x and y_type,_=y and bg_type,_=bg in
    if x_type!=Ast.Int || y_type!=Ast.Int then failwith ("Board Dimensions must be an
integer")
    else if bg_type!=Ast.Str then failwith ("Board image must be a string")
    else {x=x; y=y; img=bg;} in

    let rec checkPieces env=function
    []->[]
    | hd::tl ->
        let onTurnScope = {parent=Some(env.scope); variables=[(Ast.Str,
"turn");(Ast.Int,"x");(Ast.Int,"y")]; declFuncs=[(Ast.Void, "remove",[]);(Ast.Void,
"visible",[Ast.Bool]);
        (Ast.Void,"endTurn",[]);(Ast.Void, "changeType", [Ast.Str])]}
        and onClickScope = {parent=Some(env.scope); variables=[(Ast.Str,
"turn");(Ast.Int,"x");(Ast.Int,"y")]; declFuncs=[(Ast.Void, "remove",[]);(Ast.Void,
"visible",[Ast.Bool]);
        (Ast.Void,"endTurn",[]);(Ast.Void, "changeType", [Ast.Str])]}
        in
        let onTurnEnv = {env with scope=onTurnScope} and onClickEnv = {env with
scope=onClickScope} in
        let checkPiece env piece =
```

```
            if (List.exists (fun (s) -> s = piece.name) env.pieceNames) then failwith
("Duplicate piece names found")
            else env.pieceNames<- piece.name::env.pieceNames;
{name=piece.name;img=piece.img;onTurn_typed=(typed_stmt_list onTurnEnv piece.onTurn);
onClick_typed=(typed_stmt_list onClickEnv piece.onClick);}
        in checkPiece env hd::checkPieces env tl
    in

    let winScope = {env.scope with parent=Some(env.scope); variables=[(Ast.Str, "turn");]; }
and loseScope = {env.scope with parent=Some(env.scope); variables=[]; } and
        drawScope = {env.scope with parent=Some(env.scope); variables=[(Ast.Str, "turn");]; }
in
    let winEnv = {env with scope=winScope;func=Ast.Bool; returned=false} and loseEnv = {env
with scope=loseScope;func=Ast.Bool;returned=false} and drawEnv={env with scope=drawScope;
func=Ast.Bool;returned=false} in


    let scopeSetup = {env.scope with parent=Some(env.scope); variables=[]; } in
    let envSetup = {env with scope=scopeSetup} in

    let board = checkBoard env board in
    let setup = typed_stmt_list envSetup setup in
    let pieces = checkPieces env pieces in


    let l=conditions.l and w=conditions.w and d=conditions.d in
    let conditions = {
        win_typed = typed_stmt_list winEnv conditions.win;
        lose_typed = typed_stmt_list loseEnv conditions.lose;
        draw_typed = typed_stmt_list drawEnv conditions.draw;
    } in


(*check players and turnorder need to be added*)

    if (winEnv.returned)=false && w then failwith("Win condition requires a return
statement.")
    else if (loseEnv.returned)=false && l then failwith("Lose condition requires a return
statement.")
    else if (drawEnv.returned)=false && d then failwith("Draw condition requires a return
statement.")
    else
        board, players, turnOrder, pieces, setup, conditions;
```

Makefile

```
# Author: Laura
default: ast sast scanner parser codegen semant gridlokker
       ocamlc -o gridlok ast.cmo sast.cmo scanner.cmo parser.cmo semant.cmo codegen.cmo
gridlok.cmo
```

```
gridlokker:
	ocamlc -c gridlok.ml

scanner: parser
	ocamllex scanner.mll; ocamlc -o scanner scanner.ml

parser: ast
	ocamlyacc parser.mly; ocamlc -c parser.mli; ocamlc -c parser.ml

codegen: ast
	ocamlc -c codegen.ml

semant: sast
	ocamlc -c semant.ml

sast: ast
	ocamlc -c sast.ml

ast:
	ocamlc -c ast.ml

.PHONY: clean

clean:
	rm -f scanner.ml parser.ml parser.mli *.cmo *.cmi gridlok scanner
```

/tests

Just a few tests have been added to show the general structure.

fail_place_argtype.gl

```
/* Authors: Alice, Bryan */
game{
    board{
        dimensions {3,3}
        image {"images/hi.png"}
    }
    players{"x"}
    turnOrder{"x"}

    piece {
        name{"x"}
        image{"images/x.png"}
        onTurn{}
        onClick{}
    }
    setup{
        place{1,true,5}
```

```
    }
}
```

fail_place_argtype.out

```
Fatal error: exception Failure("Unexpected argument type.")
```

test_wincond.gl

```
/* Authors: Alice, Bryan */
game{
    board{
        dimensions {3,3}
        image {"images/hi.png"}
    }
    players{"x"}
    turnOrder{"x"}

    piece {
        name{"x"}
        image{"images/x.png"}
        onTurn{}
        onClick{}
    }
    setup{
        place{"x",1,1}
    }
    winCondition{
        print{"win"}
        return{true}
    }

}
```

test_wincond.out

```
win
```

./

gridlok.sh

```
#!/bin/bash
# Authors: Laura, Alice

set -e

if [[ $# -eq 1 ]]
then
    cd src
```

```
    make > /dev/null
    cd ../
    src/gridlok < $1

    gcc test.c -w -lSDL2 -lSDL2_image -o test

    rm test.c

elif [[ $# -eq 2 ]]
then
    cd src
    make > /dev/null
    cd ../
    src/gridlok < $1

    gcc test.c -w -lSDL2 -lSDL2_image -o $2

    rm test.c

else
    echo "Usage: ./gridlok.sh [.gl file] [executable name(optional)]"
fi
```

run_tests.sh

```
#!/bin/sh
# Author: Alice

GRIDLOK="./gridlok.sh"

ulimit -t 30

logfile=test.log
if [ -f $logfile ]
then
    rm -f $logfile
fi
if [ -d tests/output ]
then
    rm -rf tests/output
fi
mkdir tests/output

if [ $# -ge 1 ]
then
    files=$@
else
    files=$(ls tests/test_*.gl tests/fail_*.gl)
fi

for file in $files
```

```
do
    case $file in
        *test_*)
            filename=$(echo $file | sed 's/.gl//' | sed 's/tests\///')
            $GRIDLOK $file
            ./test > tests/output/${filename}.out
            if [ -f tests/${filename}.out ]
            then
                diff -b tests/${filename}.out tests/output/${filename}.out >
tests/output/${filename}.diff
                if [ -s tests/output/${filename}.diff ]
                then
                    echo "${filename}: FAILED" | tee -a $logfile
                else
                    rm tests/output/${filename}.diff
                    echo "${filename}: OK" | tee -a $logfile
                fi
            else
                echo "${filename}: FAILED - tests/${filename}.out does not exist" | tee -a
$logfile
            fi
            ;;
        *fail_*)
            filename=$(echo $file | sed 's/.gl//' | sed 's/tests\///')
            $GRIDLOK $file 2> tests/output/${filename}.out
            if [ -f tests/${filename}.out ]
            then
                diff -b tests/${filename}.out tests/output/${filename}.out >
tests/output/${filename}.diff
                if [ -s tests/output/${filename}.diff ]
                then
                    echo "${filename}: FAILED" | tee -a $logfile
                else
                    rm tests/output/${filename}.diff
                    echo "${filename}: OK" | tee -a $logfile
                fi
            else
                echo "${filename}: FAILED - tests/${filename}.out does not exist" | tee -a
$logfile
            fi
            ;;
        *)
            echo "unknown file type $file"
            exit 1
            ;;
    esac
done

rm tests/output/*.out
rm test
```

Sample code

Gridlok files

minesweeper.gl

```
/* Author: Laura */
game{
  board{
    dimensions {8,8}
    image {"images/green_board.png"}
  }
  players {"a"}
  turnOrder {"a"}
  piece {
    name {"mine"}
    image {"images/m.png"}
    onTurn{}
    onClick{
        visible{true}
    }
  }
  piece {
    name {"zero"}
    image {"images/transparent.png"}
    onTurn{}
    onClick{
      visible{true}
      changeType{"zerob"}
      for sp surrounding (x,y){
        if(sp.type!="mine"){click{sp.x,sp.y}}
      }
    }
  }
  piece {
    name {"zerob"}
    image {"images/dtrans.png"}
    onTurn{}
    onClick{
      }
  }
  piece {
    name {"one"}
    image {"images/1.png"}
    onTurn{}
    onClick{visible{true}}
  }
  piece {
    name {"two"}
```

```
  image {"images/2.png"}
  onTurn{}
  onClick{visible{true}}
}
piece {
  name {"three"}
  image {"images/3.png"}
  onTurn{}
  onClick{visible{true}}
}
piece {
  name {"four"}
  image {"images/4.png"}
  onTurn{}
  onClick{visible{true}}
}
piece {
  name {"five"}
  image {"images/5.png"}
  onTurn{}
  onClick{visible{true}}
}
piece {
  name {"six"}
  image {"images/6.png"}
  onTurn{}
  onClick{visible{true}}
}
piece {
  name {"seven"}
  image {"images/7.png"}
  onTurn{}
  onClick{visible{true}}
}
piece {
  name {"eight"}
  image {"images/8.png"}
  onTurn{}
  onClick{visible{true}}
}
setup{
  for i(0,9){
    def int rx{ rand{0,width-1} }
    def int ry{ rand{0,height-1} }
    if(board[rx][ry]==empty){
      place{"mine",rx,ry}
      setVisibility{0,rx,ry}
    }
    else{
      set i {i-1}
```

```
      }
    }

    for (i,j) in board{
      def int count{0}
      if(board[i][j]==empty){
          for p surrounding (i,j){
            if(p.type=="mine"){
              set count {count + 1}
            }
          }
        if(count=0){place{"zero",i,j}
          setVisibility{0,i,j}
        }
        if(count=1){place{"one",i,j}setVisibility{0,i,j}}
        if(count=2){place{"two",i,j}setVisibility{0,i,j}}
        if(count=3){place{"three",i,j}setVisibility{0,i,j}}
        if(count=4){place{"four",i,j}setVisibility{0,i,j}}
        if(count=5){place{"five",i,j}setVisibility{0,i,j}}
        if(count=6){place{"six",i,j}setVisibility{0,i,j}}
        if(count=7){place{"seven",i,j}setVisibility{0,i,j}}
        if(count=8){place{"eight",i,j}setVisibility{0,i,j}}
      }
    }
  }

  loseCondition{
    def bool flag{false}
    for all sp in board{
      if(sp.type=="mine" AND sp.visible){
        print{"dead"}
        return{true}
      }
    }
    return{false}
  }
}
```

tictactoe.gl

```
game{
  board{
    dimensions {3,3}
    image {"images/green_board.png"}
  }
  players {"black", "white"}
  turnOrder {"black", "white"}

  piece {
    name {"placeHolder"}
    image {"images/transparent.png"}
```

```
    onTurn{}
    onClick{
        if(turn=="black"){
         changeType{"blackPiece"}
       }
       else{
         changeType{"whitePiece"}
       }
       endTurn{}
    }
  }

  piece {
    name {"blackPiece"}
    image {"images/blackPiece.png"}
    onTurn{}
    onClick{}
  }
  piece {
    name {"whitePiece"}
    image {"images/whitePiece.png"}
    onTurn{}
    onClick{}
  }

  setup{
    for all sp in board{
      place{"placeHolder",sp.x,sp.y}
    }
  }

  winCondition{
    def bool a{
    board[0][0]!="placeHolder" AND ((board[0][0]==board[0][1] AND board[0][1]==board[0][2]) OR
(board[0][0]==board[1][0] AND board[1][0]==board[2][0]) OR
    (board[0][0]==board[1][1] AND board[1][1]==board[2][2]))}

    def bool b{
    board[1][1]!="placeHolder" AND ((board[1][1]==board[1][0] AND board[1][1]==board[1][2]) OR
(board[0][1]==board[1][1] AND board[0][1]==board[2][1]) OR
    (board[2][0]==board[1][1] AND board[1][1]==board[0][2]))
    }

    def bool c{board[2][2]!="placeHolder" AND ((board[2][2]==board[2][1] AND
board[2][2]==board[2][0]) OR (board[2][2]==board[1][2] AND board[2][2]==board[0][2])   )}

    if(a OR b OR c){
        if(turn == "black") {
            print{"White wins!"}
        }
```

```
        else {
            print{"Black wins!"}
        }
    }

    return {a OR b OR c}
  }


  drawCondition{
    def bool d{true}
    for all sp in board{
      if(sp.type=="placeHolder"){ set d {false}}
    }
    if(d){
        print{"It's a draw..."}
    }
    return {d}
  }
}
```

Generated C files

test.c (minesweeper)

```
#include <SDL2/SDL.h>
#include <SDL2/SDL_image.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
const int SCREEN_WIDTH = 600; const int SCREEN_HEIGHT = 600; const int SCREEN_BPP = 32; const
int DISPLAY_HEIGHT = 000;
int init();
int loadMedia();
void closeSDL();
SDL_Texture * getTexture (char * piecename);
void renderPiece(char * piecename, int x, int y);
int addPiece(char * piecename, int x, int y);
int removePiece(int x, int y);
SDL_Texture * loadTexture (char * filename);
void handleEvent( SDL_Event* e );
int xCord(int x);
int yCord(int y);
void clickSim(int x, int y);
int win();
int lose();
int draw();
int min();
int max();
SDL_Surface * backgroundTex = NULL;
SDL_Window* gWindow = NULL;
```

```
SDL_Surface* gRenderer = NULL;
int turnChange = 0;
char * turn;const int WIDTH=8; const int HEIGHT=8; const char*
backgroundImg="images/green_board.png"; char * b [8][8]; int v[8][8];const char * eightIMAGE =
"images/8.png";SDL_Texture * eightTexture = NULL; void eightOnClick(int x, int y); void
eightOnTurn(); void eightOnClick(int x, int y){v[x][y]=1;}void eightonTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"eight")==0){}}}}const char * sevenIMAGE = "images/7.png";SDL_Texture *
sevenTexture = NULL; void sevenOnClick(int x, int y); void sevenOnTurn(); void
sevenOnClick(int x, int y){v[x][y]=1;}void sevenonTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"seven")==0){}}}}const char * sixIMAGE = "images/6.png";SDL_Texture *
sixTexture = NULL; void sixOnClick(int x, int y); void sixOnTurn(); void sixOnClick(int x, int
y){v[x][y]=1;}void sixonTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"six")==0){}}}}const char * fiveIMAGE = "images/5.png";SDL_Texture *
fiveTexture = NULL; void fiveOnClick(int x, int y); void fiveOnTurn(); void fiveOnClick(int x,
int y){v[x][y]=1;}void fiveonTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"five")==0){}}}}const char * fourIMAGE = "images/4.png";SDL_Texture *
fourTexture = NULL; void fourOnClick(int x, int y); void fourOnTurn(); void fourOnClick(int x,
int y){v[x][y]=1;}void fouronTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"four")==0){}}}}const char * threeIMAGE = "images/3.png";SDL_Texture *
threeTexture = NULL; void threeOnClick(int x, int y); void threeOnTurn(); void
threeOnClick(int x, int y){v[x][y]=1;}void threeonTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"three")==0){}}}}const char * twoIMAGE = "images/2.png";SDL_Texture *
twoTexture = NULL; void twoOnClick(int x, int y); void twoOnTurn(); void twoOnClick(int x, int
y){v[x][y]=1;}void twoonTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"two")==0){}}}}const char * oneIMAGE = "images/1.png";SDL_Texture *
oneTexture = NULL; void oneOnClick(int x, int y); void oneOnTurn(); void oneOnClick(int x, int
y){v[x][y]=1;}void oneonTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"one")==0){}}}}const char * zerobIMAGE = "images/dtrans.png";SDL_Texture *
zerobTexture = NULL; void zerobOnClick(int x, int y); void zerobOnTurn(); void
zerobOnClick(int x, int y){}void zerobonTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"zerob")==0){}}}}const char * zeroIMAGE =
"images/transparent.png";SDL_Texture * zeroTexture = NULL; void zeroOnClick(int x, int y);
void zeroOnTurn(); void zeroOnClick(int x, int y){v[x][y]=1;removePiece(x,y);
addPiece("zerob",x,y); {int isBroken=0; int atemp, btemp;
            for(atemp=max(x-1,0);atemp<min(WIDTH,x+2);atemp++){
              for(btemp=max(0,y-1); btemp<min(HEIGHT,y+2);btemp++){
                if(isBroken) break;
                int xx = atemp; int yy = btemp;
                  if(!(atemp==x&& btemp==y)){ if(strcmp(b[xx][yy],"mine")!=0)
{clickSim(xx,yy);}}
                if(isBroken) break;
```

```
            }
          }}}void zeroonTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"zero")==0){}}}}const char * mineIMAGE = "images/m.png";SDL_Texture *
mineTexture = NULL; void mineOnClick(int x, int y); void mineOnTurn(); void mineOnClick(int x,
int y){v[x][y]=1;}void mineonTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"mine")==0){}}}}char * players [1]={"a"};char* turnOrder
[1]={"a"};SDL_Texture * getTexture(char* piecename){if(strcmp(piecename, "eight")==0) return
eightTexture;
 else if(strcmp(piecename, "seven")==0) return sevenTexture;
 else if(strcmp(piecename, "six")==0) return sixTexture;
 else if(strcmp(piecename, "five")==0) return fiveTexture;
 else if(strcmp(piecename, "four")==0) return fourTexture;
 else if(strcmp(piecename, "three")==0) return threeTexture;
 else if(strcmp(piecename, "two")==0) return twoTexture;
 else if(strcmp(piecename, "one")==0) return oneTexture;
 else if(strcmp(piecename, "zerob")==0) return zerobTexture;
 else if(strcmp(piecename, "zero")==0) return zeroTexture;
 else if(strcmp(piecename, "mine")==0) return mineTexture;
 else return NULL;}
void handleEvent( SDL_Event* e )
{
    //Get mouse position
    int x, y;
    SDL_GetMouseState( &x, &y );
    x=xCord(x); y=yCord(y);
   clickSim(x,y);
}
void renderPiece(char * piecename, int x, int y){
  SDL_Rect myRect = { x * SCREEN_WIDTH/WIDTH, y * SCREEN_HEIGHT/HEIGHT, SCREEN_WIDTH/WIDTH,
SCREEN_HEIGHT/HEIGHT};
  SDL_RenderCopy( gRenderer, getTexture(piecename), NULL,  &(myRect) );
}
int addPiece(char * piecename, int x, int y){
  if(strcmp(b[x][y],"empty")!=0){
      return 0;
  }
  b[x][y] = piecename;
  v[x][y] = 1;
  return 1;
}
int removePiece (int x, int y){
  if (strcmp(b[x][y],"empty")==0){
    return 0;
  }
  b[x][y] = "empty";
  v[x][y] = 0;
  return 1;
}
```

```
SDL_Texture* loadTexture( char * filename )
{
  SDL_Texture* newTexture = NULL;
  SDL_Surface* loadedSurface = IMG_Load( filename );
  if( loadedSurface == NULL )
  {
    printf( "Unable to load image %s! SDL_image Error: %s\n", filename, IMG_GetError() );
  }
  else
  {
        newTexture = SDL_CreateTextureFromSurface( gRenderer, loadedSurface );
    if( newTexture == NULL )
    {
      printf( "Unable to create texture from %s! SDL Error: %s\n", filename, SDL_GetError() );
    }
    SDL_FreeSurface( loadedSurface );
  }
  return newTexture;
}


int init()
{
  int success = 1;
  turn = turnOrder[0];

  int i,j;
  for(i = 0; i< sizeof(b) / sizeof((b)[0]) ; i++){
    for(j=0; j < sizeof(b[i])/sizeof(b[i][0]); j++){
      b[i][j]="empty";
      v[i][j] = 0;
    }
  }

  if( SDL_Init( SDL_INIT_VIDEO ) < 0 )
  {
    printf( "SDL could not initialize! SDL Error: %s\n", SDL_GetError() );
    success = 0;
  }
  else
  {
    if( !SDL_SetHint( SDL_HINT_RENDER_SCALE_QUALITY, "1" ) )
    {
      printf( "Warning: Linear texture filtering not enabled!" );
     }
    gWindow = SDL_CreateWindow( "Game", SDL_WINDOWPOS_UNDEFINED, SDL_WINDOWPOS_UNDEFINED,
SCREEN_WIDTH, SCREEN_HEIGHT + DISPLAY_HEIGHT, SDL_WINDOW_SHOWN );
    if( gWindow == NULL )
    {
      printf( "Window could not be created! SDL Error: %s\n", SDL_GetError() );
```

```
      success = 0;
    }
    else
    {
      gRenderer = SDL_CreateRenderer( gWindow, -1, SDL_RENDERER_ACCELERATED );
      if( gRenderer == NULL )
      {
        printf( "Renderer could not be created! SDL Error: %s\n", SDL_GetError() );
        success = 0;
      }
      else
      {
        SDL_SetRenderDrawColor( gRenderer, 0xFF, 0xFF, 0xFF, 0xFF );
        int imgFlags = IMG_INIT_PNG;
        if( !( IMG_Init( imgFlags ) & imgFlags ) )
        {
          printf( "SDL_image could not initialize! SDL_image Error: %s\n", IMG_GetError() );
          success = 0;
        }
      }
    }
  }

  return success;
}

int xCord(int x){
  int i;
  for(i=0;i<WIDTH;i++){
    if(x>i && x<(i+1)*SCREEN_WIDTH/WIDTH){
      return i;
    }
  }

  return -1;
}

int yCord(int y){
  int i;
  for(i =0; i<HEIGHT; i++){
    if(y>i && y<(i+1)*SCREEN_HEIGHT/HEIGHT){
      return i;
    }
  }
  return -1;
}

int min(int a, int b){
  if (a<b)
    return a;
```

```
  return b;
}

int max(int a, int b){
  if (a>b)
    return a;
  return b;
}

int loadMedia(){int success = 1;backgroundTex = loadTexture( backgroundImg );if( backgroundTex
== NULL ){printf( "Failed to load texture image!\n" );success = 0;
}eightTexture = loadTexture(eightIMAGE);if(eightTexture == NULL ){ printf( "Failed to load
texture image!\n" );success = 0;}sevenTexture = loadTexture(sevenIMAGE);if(sevenTexture ==
NULL ){ printf( "Failed to load texture image!\n" );success = 0;}sixTexture =
loadTexture(sixIMAGE);if(sixTexture == NULL ){ printf( "Failed to load texture image!\n"
);success = 0;}fiveTexture = loadTexture(fiveIMAGE);if(fiveTexture == NULL ){ printf( "Failed
to load texture image!\n" );success = 0;}fourTexture = loadTexture(fourIMAGE);if(fourTexture
== NULL ){ printf( "Failed to load texture image!\n" );success = 0;}threeTexture =
loadTexture(threeIMAGE);if(threeTexture == NULL ){ printf( "Failed to load texture image!\n"
);success = 0;}twoTexture = loadTexture(twoIMAGE);if(twoTexture == NULL ){ printf( "Failed to
load texture image!\n" );success = 0;}oneTexture = loadTexture(oneIMAGE);if(oneTexture == NULL
){ printf( "Failed to load texture image!\n" );success = 0;}zerobTexture =
loadTexture(zerobIMAGE);if(zerobTexture == NULL ){ printf( "Failed to load texture image!\n"
);success = 0;}zeroTexture = loadTexture(zeroIMAGE);if(zeroTexture == NULL ){ printf( "Failed
to load texture image!\n" );success = 0;}mineTexture = loadTexture(mineIMAGE);if(mineTexture
== NULL ){ printf( "Failed to load texture image!\n" );success = 0;}return success;}void
closeSDL(){SDL_DestroyTexture( backgroundTex );backgroundTex =
NULL;SDL_DestroyTexture(eightTexture);eightTexture=NULL;SDL_DestroyTexture(sevenTexture);seven
Texture=NULL;SDL_DestroyTexture(sixTexture);sixTexture=NULL;SDL_DestroyTexture(fiveTexture);fi
veTexture=NULL;SDL_DestroyTexture(fourTexture);fourTexture=NULL;SDL_DestroyTexture(threeTextur
e);threeTexture=NULL;SDL_DestroyTexture(twoTexture);twoTexture=NULL;SDL_DestroyTexture(oneText
ure);oneTexture=NULL;SDL_DestroyTexture(zerobTexture);zerobTexture=NULL;SDL_DestroyTexture(zer
oTexture);zeroTexture=NULL;SDL_DestroyTexture(mineTexture);mineTexture=NULL;SDL_DestroyRendere
r( gRenderer ); SDL_DestroyWindow( gWindow );
gWindow = NULL;gRenderer = NULL; IMG_Quit();SDL_Quit();}void clickSim(int x, int y) {if
(strcmp (b[x][y], "eight")==0){eightOnClick(x,y);} else if (strcmp (b[x][y],
"seven")==0){sevenOnClick(x,y);} else if (strcmp (b[x][y], "six")==0){sixOnClick(x,y);} else
if (strcmp (b[x][y], "five")==0){fiveOnClick(x,y);} else if (strcmp (b[x][y],
"four")==0){fourOnClick(x,y);} else if (strcmp (b[x][y], "three")==0){threeOnClick(x,y);} else
if (strcmp (b[x][y], "two")==0){twoOnClick(x,y);} else if (strcmp (b[x][y],
"one")==0){oneOnClick(x,y);} else if (strcmp (b[x][y], "zerob")==0){zerobOnClick(x,y);} else
if (strcmp (b[x][y], "zero")==0){zeroOnClick(x,y);} else if (strcmp (b[x][y],
"mine")==0){mineOnClick(x,y);} else return;}int win(){int atemp, btemp,ctemp,
dtemp,etemp,ftemp,gtemp,htemp,itemp,jtemp,ktemp,ltemp,mtemp,isBroken; return 0;}int lose(){int
atemp, btemp,ctemp, dtemp,etemp,ftemp,gtemp,htemp,itemp,jtemp,ktemp,ltemp,mtemp,isBroken; int
flagUDV = 0;{ int isBroken=0; int ltemp, mtemp;
          for (ltemp=0; ltemp<WIDTH;ltemp++){for (mtemp=0;mtemp<HEIGHT;mtemp++){ if(isBroken)
break;
              int xx=ltemp; int yy=mtemp;if(strcmp(b[xx][yy],"mine")==0&&v[xx][yy])
{printf("%s\n","dead");return 1;}}if(isBroken) break;}}return 0;}int draw(){int atemp,
```

```
btemp,ctemp, dtemp,etemp,ftemp,gtemp,htemp,itemp,jtemp,ktemp,ltemp,mtemp,isBroken; return
0;}int main( int argc, char* args[] )
{
int atemp, btemp,ctemp, dtemp,etemp,ftemp,gtemp,htemp,itemp,jtemp,ktemp,ltemp,mtemp,
isBroken;
  if( !init() )
  {
    printf( "Failed to initialize!\n" );
  }
  else
  {   if( !loadMedia() )
    {
      printf( "Failed to load media!\n" );
    }
    else
    {
      int quit = 0;{int isBroken=0; int gtemp;
          for (gtemp=0;gtemp<=9;gtemp++){ if(isBroken) break;
          int iUDV=gtemp;int rxUDV = rand()%(WIDTH-1+1-0)+0;int ryUDV =
rand()%(HEIGHT-1+1-0)+0;if(strcmp(b[rxUDV][ryUDV],"empty")==0)
{addPiece("mine",rxUDV,ryUDV);v[rxUDV][ryUDV]=0;}else{iUDV=iUDV-1;}}}{int isBroken=0; int
jtemp, ktemp;
        for (jtemp=0; jtemp<WIDTH;jtemp++){for (ktemp=0;ktemp<HEIGHT;ktemp++){if(isBroken)
break;
              int xx=jtemp; int yy=ktemp; int iUDV=jtemp; int jUDV=ktemp;int countUDV =
0;if(strcmp(b[iUDV][jUDV],"empty")==0) { {int isBroken=0; int atemp, btemp;
            for(atemp=max(iUDV-1,0);atemp<min(WIDTH,iUDV+2);atemp++){
              for(btemp=max(0,jUDV-1); btemp<min(HEIGHT,jUDV+2);btemp++){
                if(isBroken) break;
                int xx = atemp; int yy = btemp;
                  if(!(atemp==iUDV&& btemp==jUDV)){ if(strcmp(b[xx][yy],"mine")==0)
{countUDV=countUDV+1;}}
                if(isBroken) break;
              }
          }}if(countUDV==0) {addPiece("zero",iUDV,jUDV);v[iUDV][jUDV]=0;}if(countUDV==1)
{addPiece("one",iUDV,jUDV);v[iUDV][jUDV]=0;}if(countUDV==2)
{addPiece("two",iUDV,jUDV);v[iUDV][jUDV]=0;}if(countUDV==3)
{addPiece("three",iUDV,jUDV);v[iUDV][jUDV]=0;}if(countUDV==4)
{addPiece("four",iUDV,jUDV);v[iUDV][jUDV]=0;}if(countUDV==5)
{addPiece("five",iUDV,jUDV);v[iUDV][jUDV]=0;}if(countUDV==6)
{addPiece("six",iUDV,jUDV);v[iUDV][jUDV]=0;}if(countUDV==7)
{addPiece("seven",iUDV,jUDV);v[iUDV][jUDV]=0;}if(countUDV==8)
{addPiece("eight",iUDV,jUDV);v[iUDV][jUDV]=0;}}}if(isBroken) break;}} SDL_Event e;
      int turnCounter = 0;


  while( !quit && !win() && !lose() && !draw())
      {
        while( SDL_PollEvent( &e ) != 0 )
        {
```

```
            if( e.type == SDL_QUIT )
            {
              quit = 1;
            }
            else if( e.type == SDL_MOUSEBUTTONDOWN /*|| e->type == SDL_MOUSEBUTTONUP*/ ){
                handleEvent(&e);
            }
          }
          if(turnChange){
            turnChange = 0;
            turnCounter = (turnCounter + 1) % ( sizeof(turnOrder) / sizeof(turnOrder[0]));
            turn =
turnOrder[turnCounter];mineonTurn();zeroonTurn();zerobonTurn();oneonTurn();twoonTurn();threeon
Turn();fouronTurn();fiveonTurn();sixonTurn();sevenonTurn();eightonTurn();}
          SDL_SetRenderDrawColor( gRenderer, 0xFF, 0xFF, 0xFF, 0xFF );
          SDL_RenderClear( gRenderer );

          SDL_Rect board;
          board.x = 0; board.y = 0;
          board.w = SCREEN_WIDTH; board.h=SCREEN_HEIGHT;
          SDL_RenderSetViewport(gRenderer, &board);

          SDL_RenderCopy( gRenderer, backgroundTex, NULL, NULL );
          int i, j;
          for(i=0;i<=HEIGHT;i++)
          {
            //y location at i*screenheight/height
            int y = i * SCREEN_HEIGHT/HEIGHT;
            if (y==SCREEN_HEIGHT)
              y-=1;
            SDL_SetRenderDrawColor( gRenderer, 0x00, 0x00, 0x00, 0xFF );
            SDL_RenderDrawLine( gRenderer, 0, y, SCREEN_WIDTH, y );

          }
          for(i=0;i<=WIDTH;i++)
          {

            int x = i * SCREEN_WIDTH/WIDTH;
            if (x==SCREEN_WIDTH)
              x-=1;
            SDL_SetRenderDrawColor( gRenderer, 0x00, 0x00, 0x00, 0xFF );
            SDL_RenderDrawLine( gRenderer, x, 0, x, SCREEN_HEIGHT );
          }
          for(i = 0; i< sizeof(b) / sizeof((b)[0]) ; i++){
            for(j=0; j < sizeof(b[i])/sizeof(b[i][0]); j++){
              if(strcmp(b[i][j],"empty")!=0 && v[i][j]==1){
                renderPiece(b[i][j],i,j);
              }
            }
          }
```

```
        SDL_Rect messageDisplay;
        messageDisplay.x=SCREEN_HEIGHT; messageDisplay.y = 0;
        messageDisplay.w = SCREEN_WIDTH; messageDisplay.h = DISPLAY_HEIGHT;
        SDL_RenderSetViewport (gRenderer, &messageDisplay);



        SDL_RenderPresent( gRenderer );



      }
    }
  }
  closeSDL();

  return 0;
}
```

test.c (tictactoe)

```
#include <SDL2/SDL.h>
#include <SDL2/SDL_image.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
const int SCREEN_WIDTH = 600; const int SCREEN_HEIGHT = 600; const int SCREEN_BPP = 32; const
int DISPLAY_HEIGHT = 000;
int init();
int loadMedia();
void closeSDL();
SDL_Texture * getTexture (char * piecename);
void renderPiece(char * piecename, int x, int y);
int addPiece(char * piecename, int x, int y);
int removePiece(int x, int y);
SDL_Texture * loadTexture (char * filename);
void handleEvent( SDL_Event* e );
int xCord(int x);
int yCord(int y);
void clickSim(int x, int y);
int win();
int lose();
int draw();
int min();
int max();
SDL_Surface * backgroundTex = NULL;
SDL_Window* gWindow = NULL;
SDL_Surface* gRenderer = NULL;
int turnChange = 0;
char * turn;const int WIDTH=3; const int HEIGHT=3; const char*
backgroundImg="images/green_board.png"; char * b [3][3]; int v[3][3];const char *
```

```
whitePieceIMAGE = "images/whitePiece.png";SDL_Texture * whitePieceTexture = NULL; void
whitePieceOnClick(int x, int y); void whitePieceOnTurn(); void whitePieceOnClick(int x, int
y){}void whitePieceonTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"whitePiece")==0){}}}}const char * blackPieceIMAGE =
"images/blackPiece.png";SDL_Texture * blackPieceTexture = NULL; void blackPieceOnClick(int x,
int y); void blackPieceOnTurn(); void blackPieceOnClick(int x, int y){}void
blackPieceonTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"blackPiece")==0){}}}}const char * placeHolderIMAGE =
"images/transparent.png";SDL_Texture * placeHolderTexture = NULL; void placeHolderOnClick(int
x, int y); void placeHolderOnTurn(); void placeHolderOnClick(int x, int
y){if(strcmp(turn,"black")==0) {removePiece(x,y);
addPiece("blackPiece",x,y);}else{removePiece(x,y); addPiece("whitePiece",x,y);}turnChange =
1;}void placeHolderonTurn(){
        int x,y; for(x=0;x<WIDTH;x++){ for (y=0;y<HEIGHT;y++) {
if(strcmp(b[x][y],"placeHolder")==0){}}}}char * players [2]={"black","white"};char* turnOrder
[2]={"black","white"};SDL_Texture * getTexture(char* piecename){if(strcmp(piecename,
"whitePiece")==0) return whitePieceTexture;
 else if(strcmp(piecename, "blackPiece")==0) return blackPieceTexture;
 else if(strcmp(piecename, "placeHolder")==0) return placeHolderTexture;
 else return NULL;}
void handleEvent( SDL_Event* e )
{
    //Get mouse position
    int x, y;
    SDL_GetMouseState( &x, &y );
    x=xCord(x); y=yCord(y);
   clickSim(x,y);
}
void renderPiece(char * piecename, int x, int y){
  SDL_Rect myRect = { x * SCREEN_WIDTH/WIDTH, y * SCREEN_HEIGHT/HEIGHT, SCREEN_WIDTH/WIDTH,
SCREEN_HEIGHT/HEIGHT};
  SDL_RenderCopy( gRenderer, getTexture(piecename), NULL,  &(myRect) );
}
int addPiece(char * piecename, int x, int y){
  if(strcmp(b[x][y],"empty")!=0){
      return 0;
  }
  b[x][y] = piecename;
  v[x][y] = 1;
  return 1;
}
int removePiece (int x, int y){
  if (strcmp(b[x][y],"empty")==0){
    return 0;
  }
  b[x][y] = "empty";
  v[x][y] = 0;
  return 1;
```

```
}
SDL_Texture* loadTexture( char * filename )
{
  SDL_Texture* newTexture = NULL;
  SDL_Surface* loadedSurface = IMG_Load( filename );
  if( loadedSurface == NULL )
  {
    printf( "Unable to load image %s! SDL_image Error: %s\n", filename, IMG_GetError() );
  }
  else
  {
        newTexture = SDL_CreateTextureFromSurface( gRenderer, loadedSurface );
    if( newTexture == NULL )
    {
      printf( "Unable to create texture from %s! SDL Error: %s\n", filename, SDL_GetError() );
    }
    SDL_FreeSurface( loadedSurface );
  }
  return newTexture;
}


int init()
{
  int success = 1;
  turn = turnOrder[0];

  int i,j;
  for(i = 0; i< sizeof(b) / sizeof((b)[0]) ; i++){
    for(j=0; j < sizeof(b[i])/sizeof(b[i][0]); j++){
      b[i][j]="empty";
      v[i][j] = 0;
    }
  }

  if( SDL_Init( SDL_INIT_VIDEO ) < 0 )
  {
    printf( "SDL could not initialize! SDL Error: %s\n", SDL_GetError() );
    success = 0;
  }
  else
  {
    if( !SDL_SetHint( SDL_HINT_RENDER_SCALE_QUALITY, "1" ) )
    {
      printf( "Warning: Linear texture filtering not enabled!" );
    }
    gWindow = SDL_CreateWindow( "Game", SDL_WINDOWPOS_UNDEFINED, SDL_WINDOWPOS_UNDEFINED,
SCREEN_WIDTH, SCREEN_HEIGHT + DISPLAY_HEIGHT, SDL_WINDOW_SHOWN );
    if( gWindow == NULL )
    {
```

```
      printf( "Window could not be created! SDL Error: %s\n", SDL_GetError() );
      success = 0;
    }
    else
    {
      gRenderer = SDL_CreateRenderer( gWindow, -1, SDL_RENDERER_ACCELERATED );
      if( gRenderer == NULL )
      {
        printf( "Renderer could not be created! SDL Error: %s\n", SDL_GetError() );
        success = 0;
      }
      else
      {
        SDL_SetRenderDrawColor( gRenderer, 0xFF, 0xFF, 0xFF, 0xFF );
        int imgFlags = IMG_INIT_PNG;
        if( !( IMG_Init( imgFlags ) & imgFlags ) )
        {
          printf( "SDL_image could not initialize! SDL_image Error: %s\n", IMG_GetError() );
          success = 0;
        }
      }
    }
  }

  return success;
}

int xCord(int x){
  int i;
  for(i=0;i<WIDTH;i++){
    if(x>i && x<(i+1)*SCREEN_WIDTH/WIDTH){
      return i;
    }
  }

  return -1;
}

int yCord(int y){
  int i;
  for(i =0; i<HEIGHT; i++){
    if(y>i && y<(i+1)*SCREEN_HEIGHT/HEIGHT){
      return i;
    }
  }
  return -1;
}

int min(int a, int b){
  if (a<b)
```

```
    return a;
  return b;
}

int max(int a, int b){
  if (a>b)
    return a;
  return b;
}

int loadMedia(){int success = 1;backgroundTex = loadTexture( backgroundImg );if( backgroundTex
== NULL ){printf( "Failed to load texture image!\n" );success = 0;
}whitePieceTexture = loadTexture(whitePieceIMAGE);if(whitePieceTexture == NULL ){ printf(
"Failed to load texture image!\n" );success = 0;}blackPieceTexture =
loadTexture(blackPieceIMAGE);if(blackPieceTexture == NULL ){ printf( "Failed to load texture
image!\n" );success = 0;}placeHolderTexture =
loadTexture(placeHolderIMAGE);if(placeHolderTexture == NULL ){ printf( "Failed to load texture
image!\n" );success = 0;}return success;}void closeSDL(){SDL_DestroyTexture( backgroundTex
);backgroundTex =
NULL;SDL_DestroyTexture(whitePieceTexture);whitePieceTexture=NULL;SDL_DestroyTexture(blackPiec
eTexture);blackPieceTexture=NULL;SDL_DestroyTexture(placeHolderTexture);placeHolderTexture=NUL
L;SDL_DestroyRenderer( gRenderer ); SDL_DestroyWindow( gWindow );
gWindow = NULL;gRenderer = NULL; IMG_Quit();SDL_Quit();}void clickSim(int x, int y) {if
(strcmp (b[x][y], "whitePiece")==0){whitePieceOnClick(x,y);} else if (strcmp (b[x][y],
"blackPiece")==0){blackPieceOnClick(x,y);} else if (strcmp (b[x][y],
"placeHolder")==0){placeHolderOnClick(x,y);} else return;}int win(){int atemp, btemp,ctemp,
dtemp,etemp,ftemp,gtemp,htemp,itemp,jtemp,ktemp,ltemp,mtemp,isBroken; int aUDV =
strcmp(b[0][0],"placeHolder")!=0&&((strcmp(b[0][0],b[0][1])==0&&strcmp(b[0][1],b[0][2])==0)||(
strcmp(b[0][0],b[1][0])==0&&strcmp(b[1][0],b[2][0])==0)||(strcmp(b[0][0],b[1][1])==0&&strcmp(b
[1][1],b[2][2])==0));int bUDV =
strcmp(b[1][1],"placeHolder")!=0&&((strcmp(b[1][1],b[1][0])==0&&strcmp(b[1][1],b[1][2])==0)||(
strcmp(b[0][1],b[1][1])==0&&strcmp(b[0][1],b[2][1])==0)||(strcmp(b[2][0],b[1][1])==0&&strcmp(b
[1][1],b[0][2])==0));int cUDV =
strcmp(b[2][2],"placeHolder")!=0&&((strcmp(b[2][2],b[2][1])==0&&strcmp(b[2][2],b[2][0])==0)||(
strcmp(b[2][2],b[1][2])==0&&strcmp(b[2][2],b[0][2])==0));if(aUDV||bUDV||cUDV)
{if(strcmp(turn,"black")==0) {printf("%s\n","White wins!");}else{printf("%s\n","Black
wins!");}}return aUDV||bUDV||cUDV;}int lose(){int atemp, btemp,ctemp,
dtemp,etemp,ftemp,gtemp,htemp,itemp,jtemp,ktemp,ltemp,mtemp,isBroken; return 0;}int draw(){int
atemp, btemp,ctemp, dtemp,etemp,ftemp,gtemp,htemp,itemp,jtemp,ktemp,ltemp,mtemp,isBroken; int
dUDV = 1;{ int isBroken=0; int ltemp, mtemp;
          for (ltemp=0; ltemp<WIDTH;ltemp++){for (mtemp=0;mtemp<HEIGHT;mtemp++){ if(isBroken)
break;
               int xx=ltemp; int yy=mtemp;if(strcmp(b[xx][yy],"placeHolder")==0)
{dUDV=0;}}if(isBroken) break;}}if(dUDV) {printf("%s\n","It's a draw...");}return dUDV;}int
main( int argc, char* args[] )
{
int atemp, btemp,ctemp, dtemp,etemp,ftemp,gtemp,htemp,itemp,jtemp,ktemp,ltemp,mtemp,
isBroken;
  if( !init() )
  {
```

```
    printf( "Failed to initialize!\n" );
  }
  else
  {   if( !loadMedia() )
    {
      printf( "Failed to load media!\n" );
    }
    else
    {
      int quit = 0;{ int isBroken=0; int ltemp, mtemp;
          for (ltemp=0; ltemp<WIDTH;ltemp++){for (mtemp=0;mtemp<HEIGHT;mtemp++){ if(isBroken)
break;
              int xx=ltemp; int yy=mtemp;addPiece("placeHolder",xx,yy);}if(isBroken) break;}}
SDL_Event e;
      int turnCounter = 0;


  while( !quit && !win() && !lose() && !draw())
      {
        while( SDL_PollEvent( &e ) != 0 )
        {
          if( e.type == SDL_QUIT )
          {
            quit = 1;
          }
          else if( e.type == SDL_MOUSEBUTTONDOWN /*|| e->type == SDL_MOUSEBUTTONUP*/ ){
              handleEvent(&e);
          }
        }
        if(turnChange){
          turnChange = 0;
          turnCounter = (turnCounter + 1) % ( sizeof(turnOrder) / sizeof(turnOrder[0]));
          turn =
turnOrder[turnCounter];placeHolderonTurn();blackPieceonTurn();whitePieceonTurn();}
        SDL_SetRenderDrawColor( gRenderer, 0xFF, 0xFF, 0xFF, 0xFF );
        SDL_RenderClear( gRenderer );

        SDL_Rect board;
        board.x = 0; board.y = 0;
        board.w = SCREEN_WIDTH; board.h=SCREEN_HEIGHT;
        SDL_RenderSetViewport(gRenderer, &board);

        SDL_RenderCopy( gRenderer, backgroundTex, NULL, NULL );
        int i, j;
        for(i=0;i<=HEIGHT;i++)
        {
          //y location at i*screenheight/height
          int y = i * SCREEN_HEIGHT/HEIGHT;
          if (y==SCREEN_HEIGHT)
            y-=1;
```

```
          SDL_SetRenderDrawColor( gRenderer, 0x00, 0x00, 0x00, 0xFF );
          SDL_RenderDrawLine( gRenderer, 0, y, SCREEN_WIDTH, y );

        }
        for(i=0;i<=WIDTH;i++)
        {

          int x = i * SCREEN_WIDTH/WIDTH;
          if (x==SCREEN_WIDTH)
            x-=1;
          SDL_SetRenderDrawColor( gRenderer, 0x00, 0x00, 0x00, 0xFF );
          SDL_RenderDrawLine( gRenderer, x, 0, x, SCREEN_HEIGHT );
        }
        for(i = 0; i< sizeof(b) / sizeof((b)[0]) ; i++){
          for(j=0; j < sizeof(b[i])/sizeof(b[i][0]); j++){
            if(strcmp(b[i][j],"empty")!=0 && v[i][j]==1){
              renderPiece(b[i][j],i,j);
            }
          }
        }
        SDL_Rect messageDisplay;
        messageDisplay.x=SCREEN_HEIGHT; messageDisplay.y = 0;
        messageDisplay.w = SCREEN_WIDTH; messageDisplay.h = DISPLAY_HEIGHT;
        SDL_RenderSetViewport (gRenderer, &messageDisplay);



        SDL_RenderPresent( gRenderer );



      }
    }
  }
  closeSDL();

  return 0;
}
```

Git Log

```
commit d718ecd0e961b9b5eea8c516b5e71ec044832c04
Author: hwangks <ah2813@columbia.edu>
Date:   Wed May 11 17:38:51 2016 -0400

    change got reverted in codegen, added it back

commit 53a181d2ccf9b7d02f9e6548f0ae386c52cbc248
```

```
Author: hwangks <ah2813@columbia.edu>
Date:   Wed May 11 17:31:31 2016 -0400

    retrieved deleted images that are used in testing. oops

commit 97fd0b6f976d746d0cbb2deb4ec6b92f7e4e15ce
Author: hwangks <ah2813@columbia.edu>
Date:   Wed May 11 17:24:26 2016 -0400

    authors on tests

commit 587a1164b4bf8925775512a1591d68a1ac1b8a0c
Author: hwangks <ah2813@columbia.edu>
Date:   Wed May 11 17:22:04 2016 -0400

    authors

commit ee3c74a3cbf5bb6fcb18f02bc374eecc92aa441c
Author: hwangks <ah2813@columbia.edu>
Date:   Wed May 11 16:10:56 2016 -0400

    cleaning up repo

commit 27ecf3fdbd1e4c1c1a313875fe085a8c87837426
Author: Laura Hu <lh2718@columbia.edu>
Date:   Wed May 11 01:41:32 2016 -0400

    Updated codegen

commit d5ce38b5b9e9fcc7260221e322938d365dfe5f2a
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 15:41:44 2016 -0400

    edit readme

commit 416ee17408a38c3174dbc4e5c7d815bc42abd6f9
Author: Laura Hu <lh2718@columbia.edu>
Date:   Tue May 10 15:24:37 2016 -0400

    Update scanner.mll

commit b328b69655083c0e80f0e2c78a2549dba1beae65
Merge: f8bd547 4df7a82
```

```
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 14:32:36 2016 -0400

    merge

commit f8bd547efafe1999a71b924d195b8dd6c1e265d5
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 14:32:19 2016 -0400

    took out optimise

commit 4df7a82c7744ad8eac5bb4769419ab2e7bc8f2e6
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Tue May 10 14:24:33 2016 -0400

    added test for for coord loop.

commit 897e214833f7602fdf77edba084d1de009c2c59a
Merge: e72bd89 d709393
Author: Laura Hu <lh2718@columbia.edu>
Date:   Tue May 10 14:19:23 2016 -0400

    Merge pull request #11 from hwangks/final

    Final

commit d70939355f4ab3d36f86b21fab6ff799721f30e2
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Tue May 10 14:15:28 2016 -0400

    tried to fix for surrounding?

commit e72bd89fc0e7d8be62b812b99679d3b2c3ba4053
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 12:18:41 2016 -0400

    kept myscript.gl on computer, but not in remote

commit e03010ff8c99310038de302ce654e2c975657bae
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 12:16:50 2016 -0400

    renamed test script, deleted othello for now
```

```
commit d51ea8cacd6bdb9a128b5b8dd03ad795a047529b
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Tue May 10 12:15:03 2016 -0400

    added a dark transparent image for minesweeper clarity

commit c566b40be0fd87b1b0c0e78203079fe11359dba9
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 12:09:24 2016 -0400

    lose message

commit 254f365c7bdfc27d72f2e7fd807e4cc177fdcd82
Merge: fa8cc96 cece096
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 12:03:25 2016 -0400

    minesweeper

commit fa8cc96d9dcb5fb1914ee1055738ca8f6496ebbe
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 02:22:48 2016 -0400

    fixed merge

commit 419417bc5f6a2762b54982548fd56536cc77e223
Merge: b92cd32 21a66d7
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 02:20:36 2016 -0400

    todo

commit 21a66d7cf8503fbad28f4746ef0733eb9951b9f8
Merge: c38b3ad 1e6c804
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Tue May 10 02:19:09 2016 -0400

    Merge branch 'julian'

    merger

commit c38b3ade7087c49aeea5361e5b7bcaf81441e3d1
```

```
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Tue May 10 02:17:17 2016 -0400

    fixed some tests

commit b92cd32fb42e1b7b5c6d73929297d540e5d072f2
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 02:12:29 2016 -0400

    todo list

commit e805a40e866a18a80f82e1f5cf540b619087d5e7
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 01:54:36 2016 -0400

    remove todo list

commit 1e6c804b803d662176bead94c118cca62f289d0f
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 01:54:02 2016 -0400

    delete output

commit acc5fafb02320664ee5b1d6aad0cf7f784853a24
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Tue May 10 01:52:04 2016 -0400

    added tests

commit e42d82e4133ee75e2f202d0e8265ba775098481e
Author: Bryan Yu <by2181@columbia.edu>
Date:   Tue May 10 00:35:23 2016 -0400

    Fixed variable naming

commit 33308dde63146dbc18ca2507bed7ff4d6f4b9a7f
Merge: 05f0aa3 5bd6869
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 00:25:07 2016 -0400

    merge

commit 05f0aa3200d4e4a5f8f38ae546dcb2e4033f9038
```

```
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 00:24:40 2016 -0400

    fixed test.sh

commit 5bd6869f507163ae37d729fc94c5c49dbe563df2
Author: Bryan Yu <by2181@columbia.edu>
Date:   Tue May 10 00:24:37 2016 -0400

    Finished operators and initialization tests with out files

commit aee5d6ba70045c2ae3e56e10555f87551ab195d5
Merge: 052eb34 c79a828
Author: Bryan Yu <by2181@columbia.edu>
Date:   Tue May 10 00:12:17 2016 -0400

    Merge

commit c79a828cff576680aa0b22247f60c971184c10a2
Author: hwangks <ah2813@columbia.edu>
Date:   Tue May 10 00:11:41 2016 -0400

    edited test script

commit 052eb34a3713cf65940b4e6913c520aa4f489817
Author: Bryan Yu <by2181@columbia.edu>
Date:   Tue May 10 00:07:27 2016 -0400

    Initial testing for operators and variable initializations

commit a211c4abf269bb80431e6acecf5fcbd263d71f0c
Author: hwangks <ah2813@columbia.edu>
Date:   Mon May 9 23:56:00 2016 -0400

    .

commit f7dd5d3053ebf7b3d47aab1025544d7db48d17ae
Author: hwangks <ah2813@columbia.edu>
Date:   Mon May 9 23:55:08 2016 -0400

    list of tests

commit d0d432cb6b5b923ab9cd0d9cb33425474dc1ab44
```

```
Author: hwangks <ah2813@columbia.edu>
Date:   Mon May 9 23:54:19 2016 -0400

    list of tests

commit 217fc962908a4f053beae7062a594e859057b15f
Merge: cc9431c e3fe041
Author: hwangks <ah2813@columbia.edu>
Date:   Mon May 9 23:44:42 2016 -0400

    merge

commit cc9431c7e15af7147ea814f88bbc55160960e2ed
Author: hwangks <ah2813@columbia.edu>
Date:   Mon May 9 23:44:25 2016 -0400

    edited images

commit e3fe0416be401db94985c3a245b2b4194ea681f1
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Mon May 9 23:37:34 2016 -0400

    Added winning print statements

commit 8d38219b31f08e1f01e600c524b258f75227704f
Author: hwangks <ah2813@columbia.edu>
Date:   Mon May 9 23:29:38 2016 -0400

    fixed png file that caused errors

commit 0b90012b46491b2c240fec62d0d46e35980048fe
Author: hwangks <ah2813@columbia.edu>
Date:   Mon May 9 23:22:44 2016 -0400

    new prints

commit ab41fb89c36ca4187bca6f97760e5f9c796b1065
Author: hwangks <ah2813@columbia.edu>
Date:   Mon May 9 23:21:19 2016 -0400

    tictactoe prints

commit c10a1c55f1f0473c0680d5e3b1553e1aaf43ba2a
```

```
Author: hwangks <ah2813@columbia.edu>
Date:   Mon May 9 23:01:22 2016 -0400

    new readme

commit 096e3ae1dcb696efe95d2c3d3e45861514d44d35
Author: hwangks <ah2813@columbia.edu>
Date:   Mon May 9 22:50:03 2016 -0400

    fixed test outputs

commit 5f28ded7295722870280794a77de0b5646c2c840
Author: hwangks <ah2813@columbia.edu>
Date:   Mon May 9 22:29:07 2016 -0400

    didn't resolve conflict earlier

commit 9e51705df4710d5951f98e86103ccc3c20327f15
Author: hwangks <ah2813@columbia.edu>
Date:   Mon May 9 22:26:20 2016 -0400

    script

commit b421c39511da47a37f9e4e38cd0aeb027cfca48e
Merge: ade217c 2b5d78c
Author: hwangks <ah2813@columbia.edu>
Date:   Mon May 9 22:24:07 2016 -0400

    merge final into master

commit cece096683c890c9a9c2f49d9831adff6616d0bf
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Mon May 9 19:35:33 2016 -0400

    bug fixes and minesweeper demo complete

commit 2b5d78cdffcb1998648c93254c40786d14eb0f05
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Mon May 9 19:04:40 2016 -0400

    more bug fixes for for loops

commit f093b85a25b9c6b95ceacf4f8eb509af214aae72
```

```
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Mon May 9 16:55:15 2016 -0400

    a working model of the language. fixed looping bugs as well as reversed
statement bugs. added break statement in all loops. fixed piece access methods.
removed global variables, added ttt sample

commit d0438ca0d6d4cb50b406c66ced2964e4b1e56214
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Mon May 9 08:32:26 2016 -0400

    finished a working demo, no global variabls/User defined functions, no onturn
(to be implemented in a few hours)

commit ade217c643adcd98fdcdb1ce77e9802b5a2befd9
Author: hwangks <ah2813@columbia.edu>
Date:   Sun May 8 19:42:45 2016 -0400

    win/lose/draw condition tests

commit cf04e764746616af01e5cbfe407ff6e2d0eaab85
Merge: a2742b7 877695d
Author: hwangks <ah2813@columbia.edu>
Date:   Sun May 8 18:11:42 2016 -0400

    outputs for new for tests

commit a2742b7e76791d1617fb43312bdc9e73655fb682
Author: hwangks <ah2813@columbia.edu>
Date:   Sun May 8 18:06:37 2016 -0400

    skeleton for tictactoe

commit 877695d762b2b8ae8c3bdeed3b8abafa90101173
Merge: 77b3755 fb5a2c0
Author: Bryan Yu <by2181@columbia.edu>
Date:   Sun May 8 16:25:37 2016 -0400

    Merge

commit 77b3755974cd9c92c46c9e7bbe01aca5bba53bae
Author: Bryan Yu <by2181@columbia.edu>
Date:   Sun May 8 16:24:55 2016 -0400
```

```
    Fail tests for for-all

commit 06ce31ba6d7a845608aaa2dd7d9c601d211cbd33
Author: Bryan Yu <by2181@columbia.edu>
Date:   Sun May 8 16:20:00 2016 -0400

    Fail tests for for-loops on entire board

commit fb5a2c0ca72aa36abb49d9886df5843feca33143
Author: hwangks <ah2813@columbia.edu>
Date:   Sun May 8 16:15:05 2016 -0400

    for tests outputs

commit c453449edbb07a6681534d4979a43d0342154d7e
Merge: ec73386 fc614f3
Author: hwangks <ah2813@columbia.edu>
Date:   Sun May 8 16:02:18 2016 -0400

    merge

commit fc614f3ed78e96eaea0ddd38594ba2c4a2cb12ee
Author: Bryan Yu <by2181@columbia.edu>
Date:   Sun May 8 15:41:46 2016 -0400

    Fail tests for for-loops coordinates surrounging a space

commit bff51d95a02f801a59917c9c80edf190eb72f1e5
Author: Bryan Yu <by2181@columbia.edu>
Date:   Sun May 8 15:35:56 2016 -0400

    Fail tests for for-loops surround a space

commit f5b8fc1df517af022a95fe3289a1201750db9fc9
Author: Bryan Yu <by2181@columbia.edu>
Date:   Sun May 8 15:35:39 2016 -0400

    Fail tests for for-loops over rows

commit a016f67c7527ab57f1504845bee1cd915932e7d9
Author: Bryan Yu <by2181@columbia.edu>
Date:   Sun May 8 15:35:15 2016 -0400
```

```
    Fail tests for for-loops over a range

commit 79d9ce303efc95347b57e0e9dd517c81df2d0ef9
Author: Bryan Yu <by2181@columbia.edu>
Date:   Sun May 8 15:34:42 2016 -0400

    Fail tests for for-loops over columns

commit ec7338693b75c946fbe6c1bd57646bdca98012f4
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 23:32:20 2016 -0400

    small edit to test script

commit 51b81fa6aa981a31c1ac318add42e81a74daf83e
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 23:30:23 2016 -0400

    am i done with tests

commit 47e81e02f27f8841295796ed12cd2433bb567d90
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 22:14:51 2016 -0400

    even more tests

commit 8c0a943f56ee0630d77436fea61ed1ff1ff4a659
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 21:39:41 2016 -0400

    more tests

commit 22136feddd7121d72a25fbc74f18ca0484223877
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 21:09:44 2016 -0400

    gitignore microc test files

commit ff61b2756feef085a56ce86e1a992b351423d2c8
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 21:08:12 2016 -0400
```

```
    you forgot the last brace

commit 77b653c2236eca3adaf83f923ad7d32246d5cd19
Merge: 9edb349 71f7c21
Author: Bryan Yu <by2181@columbia.edu>
Date:   Sat May 7 21:05:14 2016 -0400

    merge

commit 9edb349f6c9793596fd6bfc717d5980982c27b4b
Author: Bryan Yu <by2181@columbia.edu>
Date:   Sat May 7 20:57:49 2016 -0400

    Updated set up for othello.gl

commit 71f7c211bcc7936bb05af82f8294ec87fe8db6b1
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 20:50:34 2016 -0400

    edited test.sh and added a gitignore

commit c71ab64c22532cf3151cee922dce792b8ad41cac
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 20:32:36 2016 -0400

    test outputs

commit ab83b79208af328c5c29c76debc4e9a4f52b0dbe
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 20:20:16 2016 -0400

    delete microc test files

commit 345f032d376ee514d25e167b3beddc74b76c649a
Merge: c519fe4 2ee14b2
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 20:18:29 2016 -0400

    semant error merge

commit 2ee14b200c8ad622ba416a649ae00e9131464dff
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Sat May 7 20:10:18 2016 -0400
```

```
    changed myscript to test conditions bug

commit 422b74893d31f8e9289b1d03567dc957e195ee7a
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Sat May 7 20:09:23 2016 -0400

    fixed conditions bug

commit a451815d1841b4f265d052f6f5b975b19756e2de
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Sat May 7 19:54:08 2016 -0400

    started on optimiser

commit c519fe485914c9335d73592f62cea92b8b926e34
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 19:48:43 2016 -0400

    semantic return erro

commit 4e712cd389245f2a9349565382f7a841cf69834d
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 19:40:22 2016 -0400

    semant version was wrong

commit 387e306ba49a1f852f535d604ba464c0d6824477
Merge: 5d831d6 ef4afb7
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 19:33:36 2016 -0400

    merge bryan-othello

commit ef4afb74c36f321aecdc4df0ce08041fc0290ac9
Author: Bryan Yu <by2181@columbia.edu>
Date:   Sat May 7 19:31:35 2016 -0400

    othello

commit 5d831d64e247248ba82528d16ce2f85d0fa8ce14
Merge: 7853f9a e2f69ca
Author: hwangks <ah2813@columbia.edu>
```

```
Date:   Sat May 7 19:28:21 2016 -0400

    merge julian semant

commit 19d80073307c6c40082fc0c04d6b80279031a1fa
Author: Bryan Yu <by2181@columbia.edu>
Date:   Sat May 7 19:20:17 2016 -0400

    Skeleton for Othello game

commit 45f376b4a7d42e7ce68c01923bfdcfe1ceb4ff0b
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 19:03:27 2016 -0400

    typo

commit 44a85dd26f053dcc99c99346bbb2a372a79af5bd
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 19:01:33 2016 -0400

    julian semant merge

commit e2f69ca18a31f16d15f512e982901bd7d196dc5b
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Sat May 7 18:59:35 2016 -0400

    fixed check functions

commit 7853f9aaca71878868f23951e498a40a48a0d4d7
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 18:15:25 2016 -0400

    edited script

commit dbf4dbd6bc9d119c6559c9c51ce94b2936b3747f
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 18:03:01 2016 -0400

    merge src from codegen branch

commit 1b35a8f876a409f964bf5f43a427d01da59bfd7d
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 17:58:05 2016 -0400
```

```
    edited testfiles for image folder

commit 04e901871b9c545b77945b036aeea6713b1585cf
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 17:47:28 2016 -0400

    merge codegen semant.ml

commit e7e5d572d9d490e885b1a716afb36182cd663f25
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Sat May 7 17:24:05 2016 -0400

    fixed a bug where it allowed statments such as set x{def int y {5}}

commit dd29e61c917020903ba15d72fcf8ac3e0b5029fe
Merge: 5a4b2e7 9c0dc57
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 17:22:21 2016 -0400

    images folder

commit 5a4b2e71e9ca10e8a544acae2c070d5511d52d40
Merge: 3d1a316 6775f2a
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 16:55:51 2016 -0400

    merge julian

commit 6775f2aa84d30ef45957698c7be4ca745ab2ce11
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Sat May 7 16:33:19 2016 -0400

    added two functions

commit 9c0dc5712c12a7cc4c5ad1505bdcc37602159c4c
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 16:07:57 2016 -0400

    might make separate folder for images

commit 97bd3aea9445bfb74080cfe562baae746bf422cd
Author: lauwrahoo <lh2718@columbia.edu>
```

```
Date:   Sat May 7 15:56:39 2016 -0400

    created codegen branch

commit 3d1a31657b14791bd6046a3302e7ff18309d7c67
Merge: 4985767 ca62ed4
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 15:45:31 2016 -0400

    merge laura's branch

commit ca62ed450e2f53d4bff913e4d07d7de60663ef12
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Sat May 7 15:42:53 2016 -0400

    changed players and turnorder input to string only for the parser for clarity
reasons

commit 4985767d0017ba653ae57ec4480f39a519e6bd7f
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 14:44:54 2016 -0400

    todo

commit 7d9530b32b2eb90511e3247eee08bb32397a77cd
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 14:42:31 2016 -0400

    alice todo

commit eb9fa966a48906b754b07ecabec231c38e63934f
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 14:30:14 2016 -0400

    tests

commit 3b8862632545e0a16e08285fa8337c54764512ed
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 13:30:43 2016 -0400

    gridlok.ml compile error

commit ef5bca2ea42840da4fd375df2c8b9d4099b203b8
```

```
Merge: 7c052c0 09cc3cf
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 12:56:26 2016 -0400

    merged laura-semcheck

commit 7c052c0b2a5a037aef6c0437bb79f8a1440f2b58
Author: Bryan Yu <by2181@columbia.edu>
Date:   Sat May 7 12:37:22 2016 -0400

    Started testing for for_surrounding. Still need to verify for-loop syntax

commit d69c913e484ed66a1109fc6ae5ae22add7275994
Author: hwangks <ah2813@columbia.edu>
Date:   Sat May 7 01:46:56 2016 -0400

    new simpler test script

commit 09cc3cf4f49e79364e73b4d760e033228886ac85
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Sat May 7 00:52:06 2016 -0400

    semantic checking for conditionals complete

commit 8469e2e5c6ada1e6a542567c8634d9b3a481bc05
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Sat May 7 00:11:43 2016 -0400

    finished pattern matching for semant checking. still need to check conditional
statments and turnOrder/player lists

commit 09825d03c6f05e8dbcaa1710473bb1bfd2d3738f
Author: hwangks <ah2813@columbia.edu>
Date:   Fri May 6 23:58:35 2016 -0400

    making new test script

commit f38bb138907e4d7e535a48f9f9ddd39919441219
Author: hwangks <ah2813@columbia.edu>
Date:   Fri May 6 23:44:07 2016 -0400

    small edit
```

```
commit ba57556ce0733de0366a6ab5dc03afc665341acd
Author: hwangks <ah2813@columbia.edu>
Date:   Fri May 6 23:42:30 2016 -0400

    changed gridlok.sh to allow just one argument

commit 7dad3a29b4cdfb0c3ddaa37a76ec170595a2fd84
Merge: 88d019e df2df34
Author: hwangks <ah2813@columbia.edu>
Date:   Fri May 6 22:56:19 2016 -0400

    merge

commit 88d019eb01d01008aac198f32f71ab31db7eb06e
Author: hwangks <ah2813@columbia.edu>
Date:   Fri May 6 22:55:45 2016 -0400

    new gridlok.sh to make executable

commit df2df341924317117e749efcd3984b38af1063dd
Author: Bryan Yu <by2181@columbia.edu>
Date:   Fri May 6 21:41:43 2016 -0400

    Changed template.tbag to template.gl. Initializes the board for testing

commit 80ca902b27be54f9cf8541c629990ccac95c2617
Author: Bryan Yu <by2181@columbia.edu>
Date:   Fri May 6 21:40:22 2016 -0400

    Created template.gl for tests. Please cp before creating more tests

commit 00bffbd1436f93955c2e5d90bb5bbd0b1c5c5d33
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Fri May 6 19:53:55 2016 -0400

    wrote semantic error checking for function calls: checks if function is
declared, has right num and  type of arguments

commit 18dd8023d39aabbc68969165b122f76e3737bb6b
Author: Bryan Yu <by2181@columbia.edu>
Date:   Fri May 6 19:25:39 2016 -0400

    Added print reminder for Laura in README
```

```
commit b6a3713944a67977e1178c327278f1cb4bbcd4a2
Author: hwangks <ah2813@columbia.edu>
Date:   Fri May 6 18:42:45 2016 -0400

    removed semantic.ml, added in code to bottom of semant.ml

commit 8fc513d55854607058ecf555ff82efd26e9cf928
Author: hwangks <ah2813@columbia.edu>
Date:   Fri May 6 18:35:29 2016 -0400

    test script

commit 167e35f6e414cfc1b35dcea7544d21fe00900984
Merge: fc3426a c4a3729
Author: hwangks <ah2813@columbia.edu>
Date:   Fri May 6 16:30:09 2016 -0400

    merged laura-semcheck

commit c4a3729758d9b5faeef5d62ac235a343bd3eb044
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Fri May 6 03:25:27 2016 -0400

    checked global variables and setup

commit 66359c88b7415876f06e7d2d1ff4902f6a7f7c64
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Thu May 5 17:32:39 2016 -0400

    finished semantic check for FOR loops

commit fc3426a6baa1689683f5cc4978cc820e22aad7ba
Author: Bryan Yu <by2181@columbia.edu>
Date:   Thu May 5 15:49:11 2016 -0400

    Editted fail_board.gl to match new fail_board.out file

commit 4d55837397e08a3c67461a00c050609d9b4d18b1
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Thu May 5 15:08:38 2016 -0400

    fixed issue with vdecl
```

```
commit bb56b0654ed4fbb5b5a9f83e1d5d55747e9f9520
Author: hwangks <ah2813@columbia.edu>
Date:   Thu May 5 15:04:54 2016 -0400

    some tests using test game file

commit dabdea1a5339c7c04165a25c86a1f400a44d7d09
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Thu May 5 14:14:14 2016 -0400

    added semantic checking for if statements

commit 0a6b4104ff7677ab6e8227369af55d4b9635953f
Author: hwangks <ah2813@columbia.edu>
Date:   Thu May 5 13:42:46 2016 -0400

    commented out parts in gridlok.ml causing errors

commit 0cc970d4b384f74acde810e7d8dec3a921f2e7fa
Merge: 1105e40 a3c6955
Author: hwangks <ah2813@columbia.edu>
Date:   Thu May 5 13:32:28 2016 -0400

    merged julian's pull request changes

commit a3c6955435e19576b1a2d631dba4aa25fdddb870
Merge: f913d6b 1105e40
Author: hwangks <ah2813@columbia.edu>
Date:   Thu May 5 13:31:05 2016 -0400

    julian's pull request required merges

commit b7b4e007f259ce5be29e00e63abecde8130e04db
Merge: 951f3e7 1105e40
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Thu May 5 13:00:44 2016 -0400

    Merge branch 'master' of https://github.com/hwangks/gridlok into laura-semcheck

commit 951f3e7957d827e4f94ab846a0180042ec0ccfef
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Thu May 5 12:59:53 2016 -0400
```

```
    added semantic checking for declaring and assigning variables

commit f913d6b9b798afdfc62e5e1a261199d5780100d5
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Wed May 4 16:28:13 2016 -0400

    no changes, just testing

commit 454e66683780402c889bcca533bb5f888e98e967
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Wed May 4 16:25:48 2016 -0400

    Makefile

commit 872f7c30d7b4cdca88a230a3f14dbcfb720acf66
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Wed May 4 16:25:25 2016 -0400

    gridlok.ml now has semantic checker function check

commit 4af1c40c08db0fef874657300aa78fc7d40711de
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Wed May 4 16:24:47 2016 -0400

    semantic.ml checks for players and player duplicates

commit b836f37f3eb950c07b3b52d41b2f04c0f44a6fba
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Tue May 3 20:31:39 2016 -0400

    finished type checking for binops and unop, tested.

commit 98cd66ca630293e940297b4be8ffda8a5dbcefea
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Tue May 3 16:57:50 2016 -0400

    changed grammar to accomodate for global variables, and allow for using
expressions in the declarations of boards, pieces, turns, etc.

commit 1105e4079103bd81e04c5d3d0078a5f425ae2ecb
Merge: a4b67e2 2b955d4
Author: Laura Hu <lh2718@columbia.edu>
```

```
Date:   Tue May 3 11:04:02 2016 -0400

    Merge pull request #8 from hwangks/Laura

    Laura

commit 2b955d4cdba63ee1db613aae14aa6ce2db321d13
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Tue May 3 10:59:03 2016 -0400

    started on sast

commit 8ac8be102edc1d07df808e7b38e7d9fc5e4cc3a7
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Tue May 3 05:05:02 2016 -0400

    added comments

commit a4b67e2a700e6d27571da9ae8b2e60c4635ca09d
Author: Laura Hu <lh2718@columbia.edu>
Date:   Tue May 3 04:36:29 2016 -0400

    Update README.md

commit e3ccb1408c65d912bd1d6fee7e3653b0b488d036
Author: Laura Hu <lh2718@columbia.edu>
Date:   Tue May 3 04:03:25 2016 -0400

    Delete README

commit f1020ad6aa7b86d1dd0f0094e8bd29130ce292a5
Author: Laura Hu <lh2718@columbia.edu>
Date:   Tue May 3 04:02:33 2016 -0400

    Create README.md

commit 1573f0c64001630c22319d46c171212901c0e4fb
Merge: b0d2411 3f9930f
Author: Laura Hu <lh2718@columbia.edu>
Date:   Tue May 3 04:01:52 2016 -0400

    Merge pull request #7 from hwangks/Laura
```

```
    Laura

commit 3f9930f6a032f46dd26fd03bb5068d5785637225
Merge: b0d2411 d9e39f0
Author: lauwrahoo <lh2718@columbia.edu>
Date:   Tue May 3 03:58:52 2016 -0400

    finished win conditions, basic fields, started on some code gen for for loops,
but primarily started on SAST and semantic check

commit b0d241121dd4532d72c378379449affdc546d08d
Author: lauwrahoo <Laura.hu1996@gmail.com>
Date:   Fri Apr 29 21:37:53 2016 -0400

    fixed if statments, added turn and player tracking

commit b3247a3cb320439a9dc6fa105213dd4208a8a8ce
Author: lauwrahoo <Laura.hu1996@gmail.com>
Date:   Fri Apr 29 14:23:06 2016 -0400

    update

commit d9e39f04fcaf20fa7261e843f0c08168a9e0e1de
Author: lauwrahoo <Laura.hu1996@gmail.com>
Date:   Fri Apr 29 14:17:21 2016 -0400

    updated codegen with mouseclick

commit bda1025562696781b9d727a97aec4b8068c8472e
Author: lauwrahoo <Laura.hu1996@gmail.com>
Date:   Fri Apr 29 13:28:46 2016 -0400

    added grammar for assigning/declaring variables, as well as template code for
clikcing

commit f6698d292d1ec3bad8df866815f5f6404379be10
Author: lauwrahoo <Laura.hu1996@gmail.com>
Date:   Thu Apr 28 22:25:02 2016 -0400

    added grammar for if statmenets and for loops, also added code gen for
importing piece images

commit 11b8966e4473810185812f90857435e809b3d841
```

```
Author: lauwrahoo <Laura.hu1996@gmail.com>
Date:   Thu Apr 28 16:17:42 2016 -0400

    updated branch with working copy of my files

commit 1f3554c0e97b97b8770ad7064d75bf4ce75119e3
Author: lauwrahoo <Laura.hu1996@gmail.com>
Date:   Thu Apr 28 15:42:38 2016 -0400

    added sdl code for including background image, drawing grid of correct size

commit 492f9c5c2a1a1a0fb4f17fc13e33ab368f1d1995
Author: lauwrahoo <Laura.hu1996@gmail.com>
Date:   Thu Apr 21 07:38:38 2016 -0400

    updated the parser to include a general grammar (sans detailed statements and
lists) as well as a compilable codegen file that can be used as a template

commit 69b4b09de318001e3bffd38f81cc6376beb2e996
Author: lauwrahoo <Laura.hu1996@gmail.com>
Date:   Fri Apr 15 12:45:36 2016 -0400

    new parser/scanner/ast compiles, currently not compatible with iteration of
codegen and gridlok.mls

commit f944c4f29d45ce7e0235e2c7055f14680454b135
Author: lauwrahoo <Laura.hu1996@gmail.com>
Date:   Fri Apr 15 12:01:57 2016 -0400

    modified ast

commit 7490b61b6251671c033c1be457fa022de990975f
Merge: 9666299 0099310
Author: Alice Hwang <ah2813@columbia.edu>
Date:   Wed Apr 6 20:22:29 2016 -0400

    Merge pull request #6 from hwangks/Laura

    Laura

commit 0099310db77552969ca29cf0a35d6fa730fe74db
Merge: e6cc798 9666299
Author: lauwrahoo <Laura.hu1996@gmail.com>
```

```
Date:   Wed Apr 6 10:39:28 2016 -0400

    Merge branch 'master' of https://github.com/hwangks/gridlok into Laura

commit e6cc798dcc20cdbc2712f1c1edd2373307e62989
Author: lauwrahoo <Laura.hu1996@gmail.com>
Date:   Wed Apr 6 10:31:56 2016 -0400

    added binops and assignment, still no semantics check

commit c00592b907a6bd8387ce7c7a0d0128766bde30f1
Author: lauwrahoo <Laura.hu1996@gmail.com>
Date:   Wed Apr 6 02:38:27 2016 -0400

    added a simple gridlok to c code, without a semantic checking system, that only
looks for the print function and nothing else.

commit 966629944e878d52d701d981913d68084fd7165e
Author: hwangks <ah2813@columbia.edu>
Date:   Tue Apr 5 21:32:54 2016 -0400

    c-style codegen?

commit 3fcb2176007a1ac022182f5a702bb49cbbec9600
Merge: 233ef1c 520267a
Author: Alice Hwang <ah2813@columbia.edu>
Date:   Tue Apr 5 20:18:01 2016 -0400

    Merge pull request #5 from hwangks/julian

    fixed string_literal code

commit 520267a3c027029026cb916bcd41e3cfa04df256
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Tue Apr 5 20:16:27 2016 -0400

    fixed string_literal code

commit 233ef1cbd596893d45f6beeaa45a4bb6b7f699d7
Author: hwangks <ah2813@columbia.edu>
Date:   Tue Apr 5 20:05:28 2016 -0400

    small edit to parser
```

```
commit 67123e6ea767587291e5e90e34c5ef06bd74ee6b
Author: hwangks <ah2813@columbia.edu>
Date:   Tue Apr 5 19:44:56 2016 -0400

    edited call() line under expr

commit f5c5561ae61e36c693fb304520eef92620e88c1d
Merge: 476c891 542bed3
Author: Alice Hwang <ah2813@columbia.edu>
Date:   Tue Apr 5 19:40:22 2016 -0400

    Merge pull request #4 from hwangks/julian

    Julian

commit 542bed327c9c3d1c85fa198ec5b48a1e8554075a
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Tue Apr 5 19:36:19 2016 -0400

    Made some changes.

commit eb9c2e6665820c7898341179416fda6a8fb99f09
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Tue Apr 5 19:35:38 2016 -0400

    Added for loops

commit 476c8919ede9ff4b76c3d767a4e068c8dcc4afb7
Merge: 1cb849d 7936163
Author: hwangks <ah2813@columbia.edu>
Date:   Fri Apr 1 22:40:18 2016 -0400

    Merge pull request #3 from hwangks/julian

    Julian

commit 79361633735231ee5ab74f91d6ec80d0415b0298
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Fri Apr 1 21:07:55 2016 -0400

    Formatting in scanner.mll.
```

```
commit f4dcf89cb256a1fbe3bb22105f605aa219dc5ee9
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Fri Apr 1 21:01:13 2016 -0400

    Added for loops that loop over integers.

commit ad66617f346758a0b44758c6e270c29804fbb640
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Fri Apr 1 20:11:41 2016 -0400

    Fixed formatting in scanner.ml.

commit 1cb849db4e63529d5a3227a06eaed0af65562ed9
Merge: 5c80f34 dce2870
Author: hwangks <ah2813@columbia.edu>
Date:   Fri Apr 1 14:16:30 2016 -0400

    Merge pull request #2 from hwangks/Laura

    Scanner edits, some basic AST done

commit dce287026a3ed66c2ae4272d0fb86e600642fa98
Author: Laura Hu <lh2718@columbia.edu>
Date:   Fri Apr 1 04:46:50 2016 -0400

    added some to parser, fixed some scanner issues

    parser doesn't compile since haven't gotten rid of the SEMI declarations
    yet. currently have commented out a lot of the reserve words in our
    scanner and made sure the true/false returned boolean literals. more
    details in text

commit a6497c49fe1f93fdba43e70545e480897b32f028
Author: Laura Hu <lh2718@columbia.edu>
Date:   Fri Apr 1 04:23:23 2016 -0400

    Basic AST changes

    added some basic expressions and statements to the AST, and commented
    some of the scanner out. need to edit parser still.

commit 287de6ed748ef4c9dbeeeb81eefd3fe1c687d7ba
Merge: 2d251bc 5c80f34
```

```
Author: Laura Hu <lh2718@columbia.edu>
Date:   Thu Mar 31 21:12:25 2016 -0400

    Merge remote-tracking branch 'refs/remotes/origin/master' into Laura

    # Conflicts:
    #  README

commit 5c80f34d0b43f167a2c38155f8f2a510d41e0c98
Merge: 7529e0e 071cc83
Author: hwangks <ah2813@columbia.edu>
Date:   Thu Mar 31 20:41:25 2016 -0400

    Merge pull request #1 from hwangks/julian

    julian edited scanner and edited readme (test edit)

commit 071cc83f2ac0c4b002d50673e4af4590344a71bf
Author: Julian Edwards <jse2122@columbia.edu>
Date:   Thu Mar 31 20:21:08 2016 -0400

    test changed readme julian

commit 2d251bc65d20b2cc272b81edabffc3a93da006b3
Author: Laura Hu <lh2718@columbia.edu>
Date:   Thu Mar 31 20:17:03 2016 -0400

    test test test

    changed readme for test

commit 7e0b07cc7a3b6370ab03be2520748d914f9457b0
Author: Julian Edwards <julianedwards@dyn-160-39-240-62.dyn.columbia.edu>
Date:   Sun Mar 27 20:07:14 2016 -0400

    Edited scanner.

commit 7529e0e9ec3bd0edd95374fc7dddf5ee49bf8420
Author: hwangks <ah2813@columbia.edu>
Date:   Sun Mar 27 19:29:47 2016 -0400

    added microC files
```