Blur

A Programming Language for ASCII and Image Manipulation

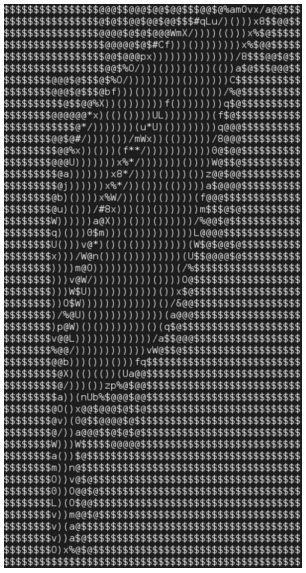
github.com/dextercallender/blur

Programming Languages and Translators
COMS W4115
Fall 2016

Dexter Callender	dec2148
Timothy Goodwin	tlg2132
Daniel Hong	sh3266
Melissa Kaufman-Gomez	mhk2149

1. Introduction

1.1 Language Description



Blur is a simple programming language for processing and manipulating image data and for creating text-based visual art. Blur draws inspiration from the traditional ASCII computer art that predated digital images, and incorporates this legacy computer art form with the digital image formats that are ubiquitous today. The Blur language provides a set of familiar and intuitive syntactic conventions for handling pixel data and pixel-to-character conversions. Blur provides the building blocks for the programmer to easily implement their own image processing algorithms, image to ASCII conversion schemes, and algorithmic/generative ASCII artworks.

1.2 Installing Blur

To download the Blur compiler, clone the Blur source repository, hosted at https://github.com/dextercallender/blur.

Installation under Ubuntu 15.10:

```
sudo apt-get install -y ocaml m4 llvm opam
opam init
opam install llvm.3.6 ocamlfind
eval `opam config env`

sudo apt-get install freeglut3-dev
sudo apt-get install binutils-gold ( for ubuntu >= 11.10 )
sudo apt-get install libdevil-dev
```

1.3 Running Blur Programs

After installing the system dependencies for the Blur compiler, you are ready to start programming in Blur. To compile a Blur program, one must first build the compiler by executing **make blur** in the top level directory. To compile your-blur-program.blr , run **make your-blur-program-ll** to produce the executable **your-blur-program.blx.**

2. Language Tutorial

2.1 A Basic Example

- A main function defined by the signature `int main()` is required.
- The return value of the main function is 0 or 1, and will return 0 by default if nothing is wrong.
- Statements are delimited by semicolons.
- println() prints the argument to standard out with newline.

```
/* add() function to print the sum of two numbers */
int add(string x, int y) {
    int z;
    z = x + y;
    println(z);
    /* note that nothing is returned - this is allowed */
}

int main() {
    add(3, 7); /* this call prints 10 to stdout*/
}
```

3. Language Reference Manual

3.1 Lexical Conventions

3.1.1 Comments

Comments are introduced by **/*** characters and terminated by ***/**. Everything after the comment introduction, /*, is considered to be a part of the comment until the comment termination symbols, */, are seen. This comment syntax supports both single-line and multi-line comments.

```
/* multi-line comment
   still a comment
last line of comment */
```

3.1.2 Identifiers

An identifier is a sequence of letters and digits. The first character must be alphabetic. Identifiers must start with an alphabetic character, including the '_' character. Blur identifiers are case-sensitive.

3.1.3 Reserved Words

The following identifiers are reserved for use as keywords, and may not be used as identifiers otherwise.

```
for, if, else, while, true, false, return
```

3.1.4 Punctuation

In Blur, each line of code is terminated with a semicolon to mark the end of an expression or statement. Multiple statements or expressions can be executed on the same line in the file if each statement or expression is separated by a semicolon. Semicolons are not needed at the end of function definitions after the closing curly braces or at the end of conditional statements after the closing curly brace.

Additionally, the following punctuators have semantic significance. They must occur and pairs and be balanced by the end of the code block or program.
`( )` — Parentheses are used to demarcate function arguments and conditional predicates.
`{ }` — Curly braces are used to specify scope of conditional statements and functions.
`[ ]` — Square brackets are used to declare array dimensions, access arrays, and declare array literals.

The following is a chart of all relevant punctuators with description and usage:

Punctuator	***Description***	***Usage***
`,`	separator for array and function arguments	`int a[][] = [[1,2],[3,4]];` `add(1, 2);`
`()`	arguments	`funcArgs(x, y, z);`
`{}`	scope	`int main() {...}`
`[]`	array init and access	`int a[][] = [[1,2],[3,4]];` `int b = a[1][1];`
`;`	end of statement	`int a;`

‘ ’	char literal delimiters	`char c = ‘z’;`
“ ”	string literal delimiters	`string s = “hello”;`

3.2 Types

Blur types are composed of primitive types and array types.

3.2.1 Primitives

The following primitive types apply to local and global variables, function parameters, and return types.

Type	Syntax	Implementation
Boolean	bool	Integer of bitwidth 1
Character	char	Integer of bitwidth 8
Integer	int	Integer of bitwidth 32
Double	double	IEEE 64-bit floating point type
String	string	Constant array of Character type
Void	void	Creates a type of a function with no return value

The following Blur snippet demonstrates usage of all types:

```
/* an example of all types above,
excluding void variable, which is illegal */
void main() {
    int a;
    double b;
    char f;
    bool d;
    string e;

    a = 5;
    b = 3.14;
    f = 'c';
    d = false;
    e = "Hello World.";
    println(a);
    println(e);
```

```
}
```

3.2.1 Arrays

The Sized Array

The sized array type is declared with explicit dimension values, but cannot be initialized to an array literal. The Sized Array creates contiguous space on the stack for the programmer to read to and write from. The sized array provides a fast, lightweight array for use in dynamic programming or sorting algorithms. This space will not persist beyond the declaration's scope, thus this array type should not be passed between functions.

The Unsized Array

The Unsized Array type is declared with dimensions but without dimension values, and is immediately assigned to an array literal value of the same dimension. Blur allocates an assigned unsized array assignment dynamically, and handles the variable name by *reference*. Thus, this datatype can be easily passed into and returned from functions without risk of undefined behavior. Dimension sizes are determined at runtime, allowing the programmer to assign array-structured data to an array variable without having to know the file size beforehand.

```
int main() {
    /* Sized array with explicit dimensions initializes all cells to 0 */
    int[5] a;
    int[5][5] b;

    /* Unsized array, immediately assigned. */
    double[] gpa = [4.0, 3.8, 3.67, 3.5, 3.34];
    int[][] img = readGrayscaleImage("images/mom_and_dad.jpg");
}
```

3.3 Declarations

3.3.1 Variable Declarations

Variables are declared using the following syntax:
Simple variable declaration: `<type> <variable_name>;`
Variable declaration with instantiation: `<type> <variable_name> = expression;`
A variable may have its value updated, as long as its type remains consistent.

Example:

```
int i = 5;
i = i + 3; /* Results in 8. */
i = 5.67; /* Semantic error. */
```

Global Variables

Global variables are declared at the top of the Blur file outside the scope of any function. The scope of global is the entire file; the value of the variables will be updated by any function that uses them.

Local Variables

The scope of local variables is bound by the set of curly braces in which the variables are contained. Blur is statically scoped, as further explained in **§3.5**.

3.3.2 Function Declarations

Functions are declared and identified using the following C-like syntax:
`<return_type> <function_name> ( <arguments> ){ <function_body> }`

First, a return type for the function is specified, followed by the function name. A return statement is required within every function definition. The function arguments are enclosed in parentheses and the code block enclosed in the curly braces is executed.

All arguments passed into functions are passed in by value, i.e. their values are copied into the function. Arguments are not passed by reference.

3.4 Operators

3.4.1 Operator Precedence

Precedence of operators follows a standard order of operations (GEMDAS - Grouping symbols, Exponents, Multiplication, Division, Addition, Subtraction). Blur is a left-associative language (evaluated left to right, after the application of order of operations).

3.4.2 Unary Operators

-expression
Negation: can be applied to the int and double types. The result is the negative of the expression.
!expression

Logical NOT operator. Applicable for type boolean.

| expression |

Magnitude: Converts an integer to a character type or vice versa, using a built-in ASCII grayscale ramp.

3.4.3 Multiplicative Operators

expression * expression

The binary * operator indicates multiplication. Applicable to types int and double. Both operands must be of the same type.

expression / expression

The binary / operator indicates division. Applicable to types int and double. Both operands must be of the same type.

expression % expression

The mod operator yields the remainder from the division of the first expression by the second. The remainder keeps the sign of the dividend. Both operands must be of the same type.

3.4.4 Additive Operators

`expression + expression`

The result is the sum of the expressions. Applicable to type int or double. Both operands must be of the same type.

expression – expression

The result is the difference of the operands. Applicable to type int or double. Both operands must be of the same type.

3.4.5 Relational Operators

`expression < expression` (less than)
`expression > expression` (greater than)
`expression <= expression` (less than or equal to)
`expression >= expression` (greater than or equal to)

The operators < , > , <= , and >= all yield a boolean result as 1 or 0. These operators take types int, double, char, and string. Both operands must be of the same type.

3.4.6 Equality Operators

`expression == expression` (equal to)
`expression != expression` (not equal to)

The == and the `!=` operators are analogous to the relational operators except for their lower precedence. (Thus, `((a<b) == (c<d))` is true whenever `a<b` and also `c<d`).

3.4.7 Logical Operators

and and **or** are logic operators that compute the boolean AND and OR of two boolean-valued operands.

3.4.8 Assignment Operator

The left value is an identifier with a type. The stored value is on the righthand side, and is stored after the assignment operation.

<type> <identifier> = expression;

The value of the expression is of type <type>, and is stored in the <identifier> variable.

The following is a chart of all relevant operators with description and usage:

Operator	***Description***	***Usage***
+	addition	`1 + 2;`
-	subtraction	`10 - 5;`
*	multiplication	`4 * 5;`
/	division	`10 / 2;`
%	modular	`10 % 3;`
=	assignment	`int a = 5;`
==	equality comparison	`5 == 5;`
!=	inequality comparison	`5 != 4;`
<	less than	`5 < 10;`
<=	less than or equal to	`5 <= 10`
>	greater than	`10 > 5;`
>=	greater than or equal to	`10 >= 5;`
and	logical and	`if(true and true) {...}`
or	logical or	`if(true or false) {...}`
!	negation	`!true;`
| `<int or char>` **|**	Bar on each side of expression	`int a = |'k'|;`

		char c = |255|; char z = | |'%'| +5 |;

3.5 Statements

A statement is a unit of execution. Statements are executed in sequence.

3.5.1 End of Statement

Statements in Blur are delimited by a single semicolon '**;**'.

3.5.2 Expression Statements

The majority of statements are expression statements, taking the form: `expression;`

3.5.3 Compound Statements

Compound statements allow for multiple statements where one is expected.

compound statement:

{statement-list}

3.5.4 Control Statements and Loops

Conditional Statement

```
if (expression) {
      statement
}
```

If the expression evaluates to true, then the statement is executed. The statement is not executed if the expression evaluates to false.

```
if (expression) {
      statement1
} else {
      statement2
}
```

If the expression evaluates to true, then statement1 is executed. In this case, statement2 within the else { } is not executed.

If the expression evaluates to false, then only statement2 within the else { } is executed.

```
int main() {
    int a;
```

```
    int b;
    a = 1;
    b = 9;

    /* single if condition */
    if(a<b) {
       println("check 1");
    }

    /* if else condition */
    if(a>b) {
       println("should not println");
    }
    else {
       println("check 2");
    }
}
```

While Loop

`while (predicate) statement … statement;`
The statement is executed as long as the predicate is true. The predicate is reevaluated after each execution of the while body (a statement block).

For Loop

```
for (expression1; expression2; expression3) {
      statement;
      …
      statement;
}
```

expression1 initializes the base of the loop range, expression2 is a predicate condition (evaluated before each iteration), and expression3 is an increment specification. The loop exits when expression2 is no longer true.

```
/* while loop */
int main() {
    int i;
    i = 5;
    while(i > 0) {
        i = i - 1;
        println(i);
    }
    println(42);
}

/* nested for loops */
int main() {
    int i;
```

```
    int j;
    for(i=0; i<3; i=i+1) {
       for(j=0; j<2; j=j+1) {
           println(i);
       }
    }
}
```

3.5.5 Return statements

A function returns to its caller via a return statement. The second case returns the value of the expression. If the type expected by the caller does not match that of the return statement, an error will be thrown. In the case of void functions, nothing should be returned. Simply writing `return;` will throw an error.

```
bool retBool() {
    return true;
}

char retChar() {
    return 'c';
}

double retDouble() {
    return 1.12345;
}

int retInt() {
    return 5;
}

string retString() {
    return "str";
}

void voidFunc() {
       /* should not have return; */
}
```

3.6 Scoping Rules

3.6.1 Variable Scope

Variables declared outside of functions have global scope and can be accessed anywhere within the program. If declared within a function, variables only remain in scope for the duration of the function's execution. Parameters passed into a function as arguments are declared as local variables within the scope of the function.

3.6.2 Function Scope

A function may not be called before it has been declared. This suggests that main() should be the last function declared. All functions have global scope by default.

3.7 Parameter Passing

Blur function arguments are all passed by value, with the one exception of unsized arrays, which are automatically passed by reference. All Blur types, with the exception of the **void** type and the **SizedArray** type, can be passed as parameters.

3.8 Functions

Every Blur program must have a main function.

3.8.1 Function Structure

```
Structure:
<return type> <function name> (<args and types>) {
      <local vars>
      <function body>
      <return>          /* required *
}
```

```
/* Example: */
int recurse (int x) {
    if (x == 0) {
       return 1;
    }
    else {
       return 1 + recurse(x);
    }
}
```

3.8.2 Built-In Functions

Dithering: Converting Between Pixels and ASCII

Blur contains a built-in ASCII grayscale ramp, implemented as a character ordering, to map pixel intensity values to ASCII characters and vice-versa. The built-in grayscale ramp is based off of the Monospace font face found in most shells. The accuracy of the grayscale

ramp is both font and display dependent, and additional grayscale ramps should be implemented by the programmer as needed. Blur provides three functions that access the built-in grayscale ramp:

`int charToIntensity(char c):` converts the argument ASCII character into its corresponding intensity value.

`char intensityToChar(int a):` converts the argument intensity value into its corresponding ASCII character.

`char adjustPX(char c, int offset):` accesses the Blur builtin library's grayscale ASCII ramp and returns the character in the ramp at offset **`i`** from the input character **`c.`**

Reading Images: The Canvas Paradigm

Blur contains a suite of built in functions that return two-dimensional arrays. These functions allow the Blur programmer to directly load image file types into a data structure that maintains an image's native two-dimensional format.
The Canvas paradigm describes an array sized to the same pixel dimensions of a loaded image. The Canvas paradigm both provides the programmer an automatic and intuitive representation of image data, as well as a standard for writing functions and operations between different "canvas" arrays sized against the same image file.

`char[][] canvas(string filename):` returns a character array of the same dimensions as the argument image file. Used to return a blank "canvas" on which the user can write their ASCII art.

`int[][] readGrayscaleImage(string filename):` returns a 2D int array of intensity values (in range 0-255) from input image file.

`int[][] readColorImage(string filename):` returns a 2D int array of intensity value triplets, where each [i][j] index contains 3 consecutive integers representing R, G, and B intensities. The user can access a specific color by incrementing the j index by 0, 1, or 2 for R, G, and B respectively.

`int len(<array_type>):` Returns the length of any array type in Blur. Also works on array access in the case of multi-dimensional arrays.

`int intcast(double d):` Returns the integer cast (rounded) of a double type.

double doublecast(int a): Returns the floating point value of an int type.

```
int main() {
    int[][] a = readGrayscaleImage("images/leaf.jpg");
    char[][] canv = canvas("images/leaf.jpg");
    println(len(a));
    println(len(a[0]));
    char c = intensityToChar(115);
    int darkness = charToIntensity(c);

}
```

3.8.3 The Standard Library

The blur standard library uses grayscale ramp of characters to perform dithering by mapping each character to a range of intensity values between 0-255. This grayscale ramp is font dependent but generalizes well to most standard fonts found in a bash terminal.

Edge Detection

edgeDetect(string filename, int threshold) : detects high contrast edges on an image and creates a bitmask in which 1's represent hard edges and 0's represent non-edges.

Dither

dither(string filename) : dithers an image using the intensityToChar() function.

Impose

impose(char[][] ditheredAsciiArt, int[][] edges, char edgeCharacter) : impose the edges detect using edgeDetect onto the ditheredAsciiArt image, filled with the edgeCharacter.

3.9 Input / Output

In addition to the library functions that allow the programmer to read files as image data from the file system, Blur provides two simple functions that allow the programmer to write data to stdout.

print(<blur_expression>): writes the value of a Blur expression to stdout. No format string is required.

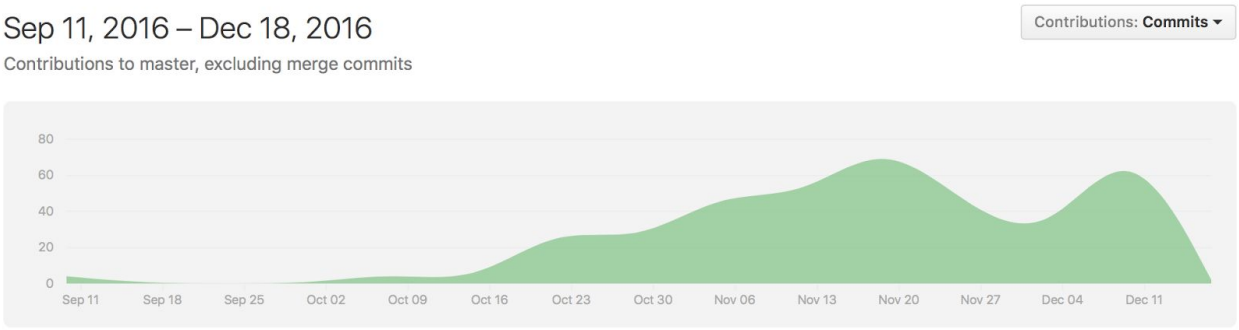
println(<blur_expression>): writes the value of a Blur expression to stdout on a new line.

Displaying formatted ASCII art is an implementation decision for the Blur programmer, and it can be easily achieved via a combination of the print() and println() functions.

4. Project Summary

4.1 Plan

Our development of Blur is akin to climbing a mountain or running a marathon.



As is evident from the chart above, once we ramped up, we contributed continuously to Blur. We met three times a week — brainstorming, discussing, and coding. Our feature development was largely driven by writing tests. For each feature, we created a branch, wrote tests, completed the feature, submitted a pull request, and merged back into master. Rachel, our TA, helped us set milestones to complete parser, scanner, semantic analyzer, and generator.

We were in constant communication over Slack, which was connected to our Blur repo on Github. We had a channel dedicated to questions we would ask Rachel during our weekly TA meetings. We also had channels for updates to the LRM and final report, to which we contributed throughout the semester.

4.2 Style Guide

Each member of the team did their best to adhere to the following style guidelines:

- Lines should not exceed 80 characters.
- Pattern-matching should be indented and aligned.
- For indentation, use tabs, not spaces. Tab width should be 4 spaces.

4.3 Timeline

Please find an outline of our milestones below:

Date	Milestone
9/28/16	Project Proposal
10/7/16	Scanner, Parser, AST
10/14/16	Pretty-print
11/1/16	Semantic Analysis — at least skeleton/microc functionality
12/9/16	Completely integrated all components of compiler pipeline
12/14/16	Handling arrays by reference
12/17/16	“Screaming” Demo Complete

4.4 Roles and Responsibilities

Team Member	Responsibility
Melissa Kaufman-Gomez	Team Leader Scanner, Parser, AST, Pretty-print, Semantic analyzer, Tests
Timothy Goodwin	System Architect. Designed the way arrays work in the backend. Implemented the codegen stage and integrated it with the C backend. Designed and implemented the Mag operator.
Dexter Callender	Language Wizard C-backend/OPENGL, Standard Library, Makefile
Daniel Hong	Testing Guru Testscript

4.5 Development Environment

Our code development took place largely in Sublime and Vim through VirtuaBox, to ensure that we all had consistent dependencies. We used GitHub for version control.

5. Language Design

5.1 Syntax Decisions

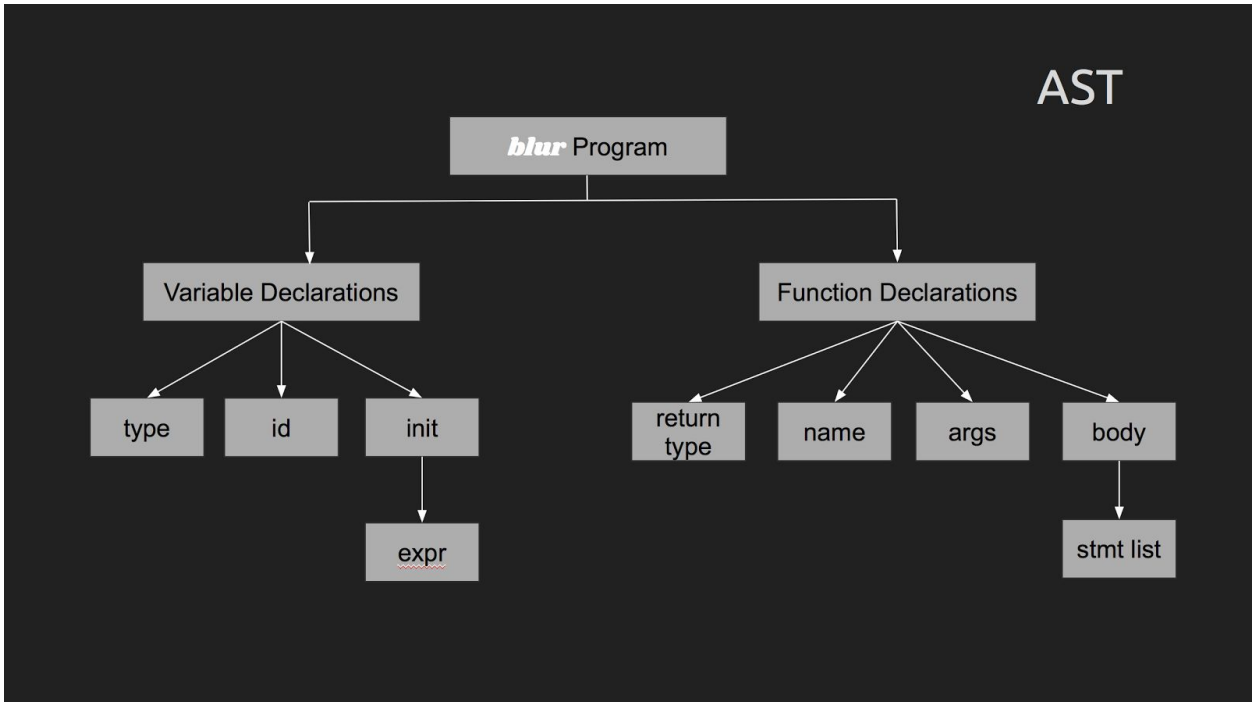
We wanted to provide familiar syntax for array indexing, loop iteration, and control statements. Blur's syntax combines the syntax of C with the simplicity of Python.

5.2 Feature Decisions

Arrays -- we didn't want the Blur user to have to handle complicated pointer syntax like in C, yet we also wanted to provide the programmer with the ability to return and pass arrays as they would a primitive datatype. We decided to provide two array types, the stack-based Sized Array and the heap-based Unsized Array.

A conscious decision was made to separate the functionality between the standard library and built-in functions.

5.3 Design Considerations of Semantic Architecture



At the highest level, a Blur program consists of variable declarations and function declarations. An important feature of Blur is the ability to declare variables both globally and anywhere in a function — not just at the top of the function, but anywhere throughout the function. This led us to the decision to use an environment record, composed of a series of symbol tables and parent symbol tables, to keep track of a Blur program during semantic analysis.

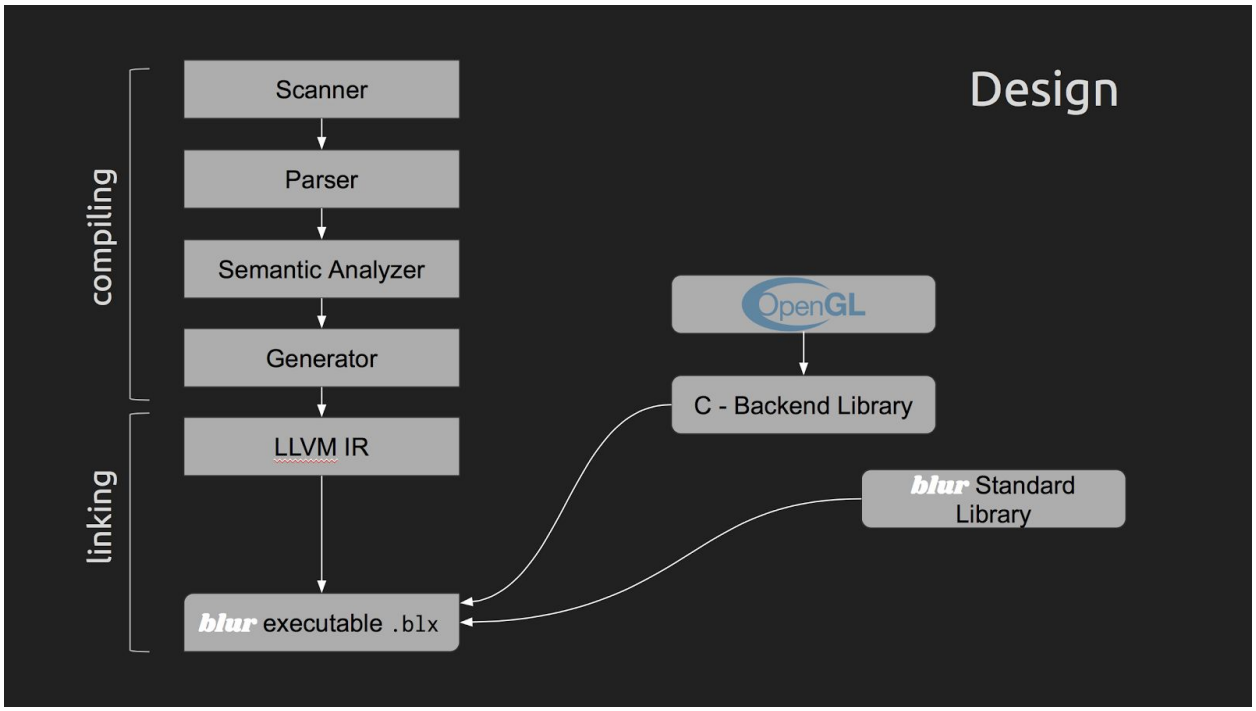
```
type symbol_table = {
   parent: symbol_table option;
   args: argdecl list;
   variables: vardecl list
}

type func_entry = {
   name: string;
   arg_types: datatype list;
   return_type: datatype
}

type env = {
   symtab: symbol_table;
   funcs: func_entry list;
   return_type: datatype option
}
```

In order to enable Blur users to both declare and initialize variables on one line, we developed two different types of variable declarations, vardecl_simple and init_vardecl. This allows the user to declare a variable, and optionally initialize it on the same line. Both of these variable declarations are of type vardecl, so in semantic analysis, we can seamlessly update the environment whenever we come across any type of variable declaration.

After checking and adding global variables to the environment, the main entry point for semantic analysis is adding the function declarations. We check function declarations, and then launch into tests of the arguments and function body.

5.4 System Design

Major Component Block Diagram



5.4.1 Interfaces Between Components

The Parsing stage passes in an abstract syntax tree representation of a Blur program to the Semantic Analyzer stage. The AST represntation is a tuple of two OCaml lists, a list of function declarations and a list of global declarations). The Semantic Analyzer stage passes this same AST representation into the Generator stage, along with flags to indicate which Blur libraries to include in the LLVM Module. The Blur generator stage produces an LLVM module of an input Blur program, which is then assembled into an object file using llc. The C library, as well as the Blur builtin library, are compiled into a library archive, which is then statically linked with the Blur program object file (using **`gcc -c`**) to produce the final **`.blx`** executable.

5.4.2 Implementation Roles

Scanner: Implemented by Melissa, Tim, Daniel, and Dexter
Parser: Implemented by Melissa and Tim
Semantic Analyzer: Implemented by Melissa and Tim
Generator: Implemented by Tim
C backend library: Implemented by Tim and Dexter

6. Testing Procedure

Unit Testing

Blur team strongly believes in test-driven development and efficient communication to be the fundamental building blocks that should persist and motivate the entire project pipeline. From the frontend to the backend generator and libraries - from the scanner, parser, ast, and semantic to the generator, libraries, built-in files - every step of our language generation has been unit tested.

Testing Approach

Following our development method of implementing every major feature on a new branch and merging upon completion, each feature was promptly tested using a bash shell script.

6.1 Representative Tests

```
/* tests/test-funcCall.blr */
int sub(int x, int y) {
        int z = x-y;
        return z;
}

int add(int x, int y) {
        int z = x + y;
        sub(3,1);
        return z;
}

int mult(int x, int y) {
        int z = x*y;
        return z;
}

int main() {
        println(add(5, 4));
        println(sub(3, 2));
        println(mult(2, 5));
}
```

```
/* tests/test-funcCall.out */
9
1
10
```

```
/* tests/fail-arg-scope.blr */
int scope() {
    int x = 5;
    return x;
}

int main() {
    scope();
    println(x);
}
```

```
/* tests/fail-arg-scope.out */
Fatal error: exception Failure("undeclared identifier x")
```

6.2 Test Suite

There are two possible outputs of a test script's `code()` function checking individual test cases:

`tests/test-while: check`
`tests/test-return-void: Wrong Output`

The final test script can be run from the blur directory using the following commands:

```
# make sure that the compiler has been built
make
# first, source the test script:
source testscript.sh
# for single file compile + run testing:
code <tests/filename.blr>
# this will output whether the test passed or failed
# check the output stored in output.txt
cat output.txt
# for automated test suite compile + run testing for all files:
codeAll
# check the outcome of each test case by checking codeResults.out file
cat codeResults.out
```

Test scripts for blur were created using bash shell scripts. Each version of test script described below was used to test a different aspect of the compilation pipeline. Below, we

show an old version of the test script used for pretty printing and semantically checking our code for development, along with code compilation. This version was used for developing **semantic_analyzer.ml** shown in §8.12 and excludes links to built-in and standard libraries.

6.2.1 Old Test Script

This original version of the test script below was used to check pretty printing, semantic, and generator processes of the compilation pipeline. **check** command pretty prints a single file and **testAll** automates the pretty print functionality for the entire test suite. The commented segment in the check() function is an alternative way of comparing two files for differences using the bash **cmp** command. This may be useful for future development purposes because cmp stops comparing at the first byte difference. For blur's development purposes, the diff command was more helpful in understanding which parts of two files are different.

```
#!/bin/bash

# check() and testAll() functions below are for pretty print testing

check(){
filebase=$(echo ${1} | cut -f 1 -d '.')

# commented code below is an alternative way to compare files;
# cmp stops at the first byte difference

#if [ "$#" -ne 1 ]; then
#    echo "Usage: check filename"
#else
#    diff "${filebase}.blr" "${filebase}.blr" # .blr.pp
#    echo "${filebase} check: checked! "
#    cmp --silent "${filebase}.blr" "${filebase}.blr" || echo "Wrong Output"
#fi

echo "${filebase} pretty print: "
./blur -p < "${filebase}.blr"
}

testAll(){
#rm results.out
for i in tests/*.blr
do
    check $i >> results.out;
done
}
```

```
# code() and codeAll() functions below are for compile + run testing

code(){
    filebase=$(echo ${1} | cut -f 1 -d '.')
    { ./blur -l < "${filebase}.blr" > "${filebase}.ll" && lli "${filebase}.ll"; } &>
output.txt
    #cmp --silent output.txt helloWorld.out || echo "Wrong Output"
    DIFF=$(diff -bBw output.txt "${filebase}.out")
    if [ "$DIFF" == "" ]; then
       echo "${filebase}: check"
    else
       echo "${filebase}: Wrong Output"
    fi
    rm -rf tests/*.ll
}

codeAll(){
if [ -f "codeResults.out" ]; then
    rm "codeResults.out"
fi
for i in tests/*.blr
do
    code $i >> codeResults.out;
done
}
```

6.2.2 Final Test Script

The following script is the final version of the testscript.sh that includes links to blur's built-in and standard libraries:

```
#!/bin/bash

code(){
    filebase=$(echo ${1} | cut -f 1 -d '.')
    { ./blur -ls < "${filebase}.blr" > "${filebase}.ll"; } &> output.txt
       if [ -s output.txt ]; then
              :
       else
       make "${filebase}-ls" &> garb.txt # now have an executable with .blx extension
       ./${filebase}.blx &> output.txt
              #echo "checking" >> output.txt
       fi
    #cmp --silent output.txt helloWorld.out || echo "Wrong Output"
    DIFF=$(diff -bBw output.txt "${filebase}.out")
    if [ "$DIFF" == "" ]; then
       echo "${filebase}: check"
    else
       echo "${filebase}: Wrong Output"
```

```
    fi
    rm -rf tests/*.ll
       rm -rf tests/*.s
       rm -rf tests/*.blx
}

codeAll(){
if [ -f "codeResults.out" ]; then
    rm "codeResults.out"
fi
for i in tests/*.blr
do
    code $i >> codeResults.out;
done
}
```

6.2.3 Assessing Test Results

As shown below, our regression suite thoroughly evaluates both positive and negative test cases:

```
tests/fail-add2: check
tests/fail-add: check
tests/fail-add-int-double: check
tests/fail-add-strings: check
tests/fail-add-sub: check
tests/fail-arg1: check
tests/fail-arg-dup: check
tests/fail-arg-scope: check
tests/fail-arr-bounds1: check
tests/fail-arr-bounds2: check
tests/fail-built-in-call-args: check
tests/fail-comp-and: check
tests/fail-comp-eq: check
tests/fail-comp-neq: check
tests/fail-comp-or: check
tests/fail-definePrint: check
tests/fail-definePrintln: check
tests/fail-duplicateFuncs: check
tests/fail-dup-var-arr: check
tests/fail-for: check
tests/fail-funcCall-args: check
tests/fail-funcCall: check
tests/fail-funcCall-expr: check
tests/fail-global-dup: check
tests/fail-globals-only: check
tests/fail-if: check
tests/fail-local-dup2: check
tests/fail-local-dup: check
```

```
tests/fail-mag: check
tests/fail-main-args: check
tests/fail-main-missing: check
tests/fail-overload: check
tests/fail-return-funcCall: check
tests/fail-returnType: check
tests/fail-sizedArr-decl: check
tests/fail-types: check
tests/fail-undeclared-id-while: check
tests/fail-undeclaredVarInit: check
tests/fail-unexpected-arg-types: check
tests/fail-uninit-unsizedArr: check
tests/fail-unsizedArr-decl: check
tests/fail-var1: check
tests/fail-var2: check
tests/fail-var-asn: check
tests/fail-var-in-arrLiteral: check
tests/fail-while: check
tests/float-arith: check
tests/helloWorld: check
tests/test-abs-char1: check
tests/test-abs-char2: check
tests/test-add1: check
tests/test-add: check
tests/test-add-sub: check
tests/test-args: check
tests/test-arr1D-char: check
tests/test-arr1D-double: check
tests/test-arr1D-int: check
tests/test-arr2D-char: check
tests/test-arr2D-double: check
tests/test-arr2D-int: check
tests/test-arr2D-lit: check
tests/test-array2: check
tests/test-array: check
tests/test-arr-lit: check
tests/test-bool: check
tests/test-canvas: check
tests/test-cast-int: check
tests/test-charToInt: check
tests/test-comment2: check
tests/test-comment: check
tests/test-comparison1: check
tests/test-comparison2: check
tests/test-cond-and: check
tests/test-conditional: check
tests/test-conditional-mult2: check
tests/test-conditional-mult: check
tests/test-cond-or: check
tests/test-diff-return-types: check
tests/test-dither: check
tests/test-edge: check
tests/test-forloop1: check
```

```
tests/test-forloop2: check
tests/test-forloop3: check
tests/test-funcCall-args: check
tests/test-funcCall: check
tests/test-if1: check
tests/test-if2: check
tests/test-if3: check
tests/test-if-else: check
tests/test-intArr: check
tests/test-intToChar: check
tests/test-localArgs: check
tests/test-mag: check
tests/test-math-ops: check
tests/test-mod: check
tests/test-negation: check
tests/test-not: check
tests/test-print: check
tests/test-print-func: check
tests/test-readGrayImg: check
tests/test-rec1: check
tests/test-return-bool: check
tests/test-return-char: check
tests/test-return-double: check
tests/test-return-funCall: check
tests/test-return-int: check
tests/test-return-str: check
tests/test-sizedArr-decl: check
tests/test-str: check
tests/test-types: check
tests/test-var-asn: check
tests/test-var-scope1: check
tests/test-var-scope2: check
tests/test-var-scope3: check
tests/test-while1: check
tests/test-whileloopMC: check
```

6.2 Blur Code Generation

6.2.1 Using the Compiler

Building the compiler produces the **blur** executable. The **blur** executable allows the user to generate their code in using flags specified upon compilation. The user is given the option to pretty print the code, compile it to llvm, compile it to check (ie valid) llvm, or to compile it to llvm and link in the Blur standard library. This functionality is controlled via the following flags.

Flags	Compilation Action
-p	Pretty print
-l	Compile to Llvm
-c	Compile to Checked Llvm
-ls	Compile to Checked Llvm and link Standard Library

6.2.2 Building an Executable

The compiler flags are abstracted behind dedicated Makefile targets for compiling Blur programs and linking in the desired Blur libraries. The **make <blurfile>-ll** target compiles a Blur program and links the C-backend and the Blur builtin library to produce the **<blurfile>.blx** executable.

The **make <blurfile>-ls** target takes all the steps of the previously mentioned **<blurfile>-ll** target and additionally links in the Blur standard library.

7. Lessons Learned

7.1 Dexter

Linking static and dynamic libraries into a compiler's pipeline can be quite difficult. Trying to convert dynamically linked libraries into static libraries can also be quite difficult. In grappling with many different image processing libraries of both the statically linked and dynamically linked variety, I learned that building libraries that are dependent on other libraries is best done by simply installing the requisite library locally instead of trying to package a static version of the requisite library into the final build. Libraries have dependencies on other libraries which have dependencies on other libraries. It's best to just install them all and avoid the issue of trying to include the proper source files in your programs build.

7.2 Tim

I learned the dangers of siloing knowledge when building a project with a team. Towards the end of the Blur development process, our knowledge of our codebase was pretty divided along who worked on what. This ultimately hindered our ability to collaborate effectively despite our willingness to do so. I also experienced the value in iterating on a design, and in saving and recording your mistakes and past attempts when working on

long term projects. Our final array implementation, which took nearly 6 weeks to arrive at, ended up being a synthesis of 3 prior LLVM implementations that I had scrapped since they each could only achieve a portion of the functionality we wanted for Blur arrays.

7.3 Daniel

Test-driven development is crucial to building a robust program, and assists every aspect of a team project. Accommodating each new feature as it moves through the development and compilation pipeline, from the scanner, parser, and ast to semantic and llvm, and reporting the errors and improvements back up to the developers, the language guru and the architect, for the problem to get resolved, was a difficult process. Often, one issue discovered in a single test case propagates to many other relevant features; therefore, prompt testing and reporting is integral to the project's timely completion. Even with the use of version control like git, communication is extremely important to avoid any conflicts in how our language is built and so that errors found in tests do not further snowball into larger problems.

7.4 Melissa

As much as is possible, make decisions about the language and stick with them. We implemented several portions of Blur, and then realized that we needed to re-implement them in order to make adjustments for new language decisions. For example, initially, we decided to use maps, as microc does, to run semantic analysis. However, we also wanted to be able to declare variables anywhere in the program, not just at the top of functions. Because of this, we needed a more robust system of keeping track of values within semantic analysis, leading to our final implementation using an environment record. We implemented much of semantic analysis using the map approach, and then had to backtrack and re-implement with the environment record. While it was a good learning process to go through semantic analysis twice, it set us a bit behind.

Later on in the process, I discovered that it would have been better to make arguments a variable type, rather than their own type. After re-implementing semantic analysis, we decided to stick with the arguments the way they were so that we wouldn't have to change the frontend and have it cascade through the rest of our code.

Finally, don't be afraid. I started off feeling intimidated of the project, but once I got my feet wet, it was fun and rewarding to play around and learn as I went. If you have a strong test suite, you won't run into a disaster as long as you keep experimenting and learning.

8. Source Code

Our stack contains components written in Ocaml, C, and blur. Proper compilation and use of blur requires installing the prerequisite image processing libraries noted in the ReadMe. The makefile is responsible for linking the compiler which is written in OCaml with the image processing backend library written in C. The user can specify whether they would like to link in the standard library via a flag when compiling their program. It should be noted that ASCII art output in the terminal is font dependent.

8.1 Makefile (authored by Dexter, Tim, Melissa)

```
LIBDIR = clib

OBJS = ast.cmx parser.cmx scanner.cmx semantic_analyzer.cmx exceptions.cmx
configuration.cmx generator.cmx prettyprint.cmx blur.cmx

blur: $(OBJS)
	ocamlfind ocamlopt -linkpkg -package llvm -package llvm.analysis -package
llvm.bitwriter -package llvm.bitreader -package llvm.linker $(OBJS) -o blur

scanner.ml : scanner.mll
	ocamllex scanner.mll

parser.ml parser.mli : parser.mly
	ocamlyacc parser.mly

%.cmo : %.ml
	ocamlc -c $<

%.cmi : %.mli
	ocamlc -c $<

%.cmx : %.ml
	ocamlfind ocamlopt -c -package llvm $<

ast.cmo :
ast.cmx :
exceptions.cmo :
exceptions.cmx :
generator.cmo : ast.cmo exceptions.cmo
generator.cmx : ast.cmx exceptions.cmx
prog.cmo: scanner.cmo parser.cmi ast.cmo exceptions.cmo generator.cmo sast.cmi
prettyprint.cmo semantic_analyzer.cmo
prog.cmx : scanner.cmx parser.cmx ast.cmx exceptions.cmx generator.cmx sast.cmx
prettyprint.cmx semantic_analyzer.cmx
parser.cmo : ast.cmo parser.cmi
parser.cmx : ast.cmx parser.cmi
```

```
scanner.cmo: parser.cmi
scanner.cmx : parser.cmx
semantic_analyzer.cmo : ast.cmo sast.cmo
semantic_analyzer.cmx : ast.cmx sast.cmx
parser.cmi: ast.cmo

.PHONY: %.ll
%-ll:
	./blur -l < $(*F).blr > $(*F).ll
	make libs
	llc $(*F).ll > $(*F).s
	gcc -I ${LIBDIR} -o $(*F).blx $(*F).s -L${LIBDIR} -lclib -lGL -lglut -lGLU
-lIL

# for including the Blur standard library.
.PHONY: %.ll
%-ls:
	./blur -ls < $(*D)/$(*F).blr > $(*D)/$(*F).ll
	cd ${LIBDIR} && make stdlib && cd ../
	llc $(*D)/$(*F).ll > $(*D)/$(*F).s

	gcc -I ${LIBDIR} -o $(*D)/$(*F).blx $(*D)/$(*F).s -L${LIBDIR} -lclib -lGL
-lglut -lGLU -lIL

.PHONY : libs
libs :
	cd ${LIBDIR} && make clib

.PHONY : clean
clean :
	rm -f prog scanner.ml parser.ml parser.mli blur
	rm -f *.cmo *.cmi *.cmx *.o *.bc *.ll *.s *.out *.blx
	rm -f *~
```

8.2 scanner.mll (authored by Melissa, Dexter, Tim, Daniel)

```
(* Ocammlex scanner for Blur lang *)
{ open Parser }

let character = ['a'-'z' 'A'-'Z' '$' '@' '%' '&' '#' '*' '/' '|' '(' ')' '{' '}' '['
']' '?' '-' '_' '+' '~' '<' '>' '!' ';' ':' '^' ',' '.' ' ' ]
let number = ['0'-'9']
let double = ((number+ '.' number*) | ('.' number+))

rule token = parse
  [' ' '\t' '\r' '\n'] { token lexbuf }
| "/*" { comment lexbuf }
| '('  { LPAREN }
```

```
| ')'  { RPAREN }
| '{'  { LBRACE }
| '}'  { RBRACE }
| '['   { LBRACK }
| ']'   { RBRACK }
| ';'  { SEMI }
| ','  { COMMA }
| '+'  { PLUS }
| '-'  { MINUS }
| '*'  { TIMES }
| '/'  { DIVIDE }
| '%'   { MOD }
| '='  { ASSIGN }
| "=="  { EQUAL }
| "!="  { NEQUAL }
| '<'   { LT }
| "<="  { LEQ }
| '>'   { GT }
| ">="  { GEQ }
| "and"  { AND }
| "or"  { OR }
| '!'   { NOT }
| '|'  { BAR }
| "int"      { INT }
| "double"      { DOUBLE }
| "string"      { STRING }
| "char"        { CHAR }
| "bool"     { BOOL }
| "for"            { FOR }
| "while"    { WHILE }
| "if"       { IF }
| "else"     { ELSE }
| "void"     { VOID }
| "return"   { RETURN }

(* literals for each data type *)
| "true"        { BOOL_LITERAL(true) }
| "false"       { BOOL_LITERAL(false) }
| number+ as lxm { INT_LITERAL(int_of_string lxm) }
| number* '.' number+ as lxm { DOUBLE_LITERAL(float_of_string lxm) }
| '"' (([^ '"'] | "\\\"")* as lxm) '"' { STRING_LITERAL(lxm) }
| '\'' ([' '-'&' '('-'[' ']'-'~'] as lxm) '\'' { CHAR_LITERAL(lxm) }

| "break"       { BREAK }
| "continue"    { CONTINUE }
| '_'?['a'-'z' 'A'-'Z']['a'-'z' 'A'-'Z' '0'-'9' '_']* as lxm { ID(lxm) }
| eof { EOF }

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

8.3 parser.mly (Authored by Melissa and Tim)

```
/* PARSER.MLY for BLUR */

%{ open Ast %}

%token LPAREN RPAREN LBRACE RBRACE LBRACK RBRACK
%token SEMI COMMA FUNC
%token INT DOUBLE STRING CHAR BOOL
%token IF ELSE FOR WHILE VOID RETURN TRUE FALSE BREAK CONTINUE
%token PLUS MINUS TIMES DIVIDE ASSIGN MOD
%token EQUAL NEQUAL LT LEQ GT GEQ AND OR NOT
%token BAR
%token <string> ID
%token <int> INT_LITERAL
%token <float> DOUBLE_LITERAL
%token <bool> BOOL_LITERAL
%token <string> STRING_LITERAL
%token <char> CHAR_LITERAL
%token EOF

%nonassoc NOELSE
%nonassoc ELSE
%right ASSIGN
%left AND OR
%left EQUAL NEQUAL
%left LT GT LEQ GEQ
%left PLUS MINUS
%left TIMES DIVIDE MOD
%right NOT
%nonassoc UNOP /* for unary op precedence */

%start program
%type <Ast.program> program

%%

program:
    decls EOF { $1 }

decls:
    /* nothing */ { [], [] }
  | decls vardecl { ($2 :: fst $1), snd $1 }
  | decls funcdecl { fst $1, ($2 :: snd $1) }

funcdecl:
    datatype ID LPAREN args_opt RPAREN LBRACE stmt_list RBRACE
    {
        {
            typ = $1;
            fname = $2;
```

```
            args = $4;
            body = List.rev $7;
        }
    }

args_opt:
    /* nothing */ { [] }
  | args_list     { List.rev $1 }

args_list:
    argdecl                 { [$1] }
  | args_list COMMA argdecl { $3 :: $1 }

argdecl:
    datatype ID
    {
        {
            argdeclType = $1;
            argdeclID = $2;
        }
    }

primitive:
    INT { Int }
  | DOUBLE { Double }
  | CHAR { Char }
  | STRING { String }
  | BOOL { Bool }
  | VOID { Void }

type_tag:
    primitive { $1 }

array_type:
    unsized_array { $1 }
  | sized_array   { $1 }

unsized_array:
  type_tag LBRACK brackets RBRACK { UnsizedArray($1, $3) }

literal_dimension_args:
    LBRACK INT_LITERAL          { [$2] }
  | literal_dimension_args RBRACK LBRACK INT_LITERAL { $4::$1 }

sized_array:
    type_tag literal_dimension_args RBRACK { SizedArray($1, List.rev $2) }

datatype:
    type_tag    { Datatype($1) }
  | array_type  { $1 }

brackets:
    /* nothing */ { 1 }
```

```
  | brackets RBRACK LBRACK {$1 + 1}

vardecl:
    vardecl_simple { $1 }
  | init_vardecl   { $1 }

vardecl_simple:
    datatype ID SEMI
    {
        {
            declTyp = $1;
            declID = $2;
            declInit = Noexpr;
        }
    }

init_vardecl:
    datatype ID ASSIGN expr SEMI
    {
        {
            declTyp = $1;
            declID = $2;
            declInit = $4;
        }
    }


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

stmt:
    expr_stmt           { $1 }
  | vardecl             { Decl($1) }
  | condit_stmt         { $1 }
  | loop_stmt           { $1 }
  | RETURN expr SEMI    { Return($2) }
  | CONTINUE SEMI       { Continue }
  | BREAK SEMI          { Break }
  | LBRACE stmt_list RBRACE { Block(List.rev $2) }

expr_stmt:
    expr SEMI { Expr $1 }

condit_stmt:
    IF LPAREN expr RPAREN stmt %prec NOELSE { If($3, $5, Expr(Noexpr)) }
  | IF LPAREN expr RPAREN stmt ELSE stmt   { If($3, $5, $7) }

loop_stmt:
    FOR LPAREN expr_opt SEMI expr SEMI expr_opt RPAREN stmt { For($3, $5, $7, $9) }
  | WHILE LPAREN expr RPAREN stmt { While($3, $5) }

expr_opt:
```

```
        { Noexpr }
  | expr { $1 }

expr_list:
    expr                { [$1] }
  | expr COMMA expr_list { $1::$3 }

mag:
    BAR expr BAR        { Unop(Mag, $2) }

expr:
  /* literals */
    INT_LITERAL       { IntLit($1) }
  | DOUBLE_LITERAL    { DoubleLit($1) }
  | STRING_LITERAL    { StrLit($1) }
  | CHAR_LITERAL      { CharLit($1) }
  | BOOL_LITERAL      { BoolLit($1) }
  | ID                { Id($1) }

  /* binops */
  | 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 EQUAL expr   { Binop($1, Eq, $3) }
  | expr NEQUAL expr  { Binop($1, Neq, $3) }
  | expr LT expr      { Binop($1, Lt, $3) }
  | expr LEQ expr     { Binop($1, Leq, $3) }
  | expr GT expr      { Binop($1, Gt, $3) }
  | expr GEQ expr     { Binop($1, Geq, $3) }
  | expr AND expr     { Binop($1, And, $3) }
  | expr OR expr      { Binop($1, Or, $3) }
  | mag { $1 }

  /* unary operators */
  | NOT expr %prec UNOP         { Unop(Not, $2) }
  | MINUS expr %prec UNOP       { Unop(Neg, $2) }
  | LPAREN expr RPAREN { $2 }

  | ID ASSIGN expr    { Binop(Id($1), Asn, $3) }
  | array_access ASSIGN expr {Binop($1, Asn, $3) }


  /* lists */
  | array_access { $1 }
  | func_call { $1 }

  /* lists */
  | LBRACK expr_list RBRACK { ArrayListInit($2) }
  /* | ID LBRACK INT_LITERAL RBRACK { ArrayAccess($1, $3) } */

dimension_args:
```

```
    LBRACK expr { [$2] }
  | dimension_args RBRACK LBRACK expr { $4::$1 }

array_access:
    ID dimension_args RBRACK { ArrayAccess($1, List.rev $2) }

func_call:
    ID LPAREN RPAREN            { FuncCall($1, []) }
  | ID LPAREN expr_list RPAREN  { FuncCall($1, $3) }
```

8.4 ast.ml (authored by Melissa and Tim)

```
(* Abstract Syntax Tree *)
type binopr =
    Add
  | Sub
  | Mult
  | Div
  | Mod
  | Eq
  | Neq
  | Lt
  | Leq
  | Gt
  | Geq
  | And
  | Or
  | Asn

type unopr = Not | Neg | Mag

(* BLUR TYPES *)
type primitive =
    Int
  | Double
  | Char
  | String
  | Bool
  | Void

type expr =
    Binop of expr * binopr * expr
  | Unop of unopr * expr
  | IntLit of int
  | DoubleLit of float
  | StrLit of string
  | CharLit of char
  | BoolLit of bool
  | Id of string
  | ArrayListInit of expr list
```

```
  | ArrayAccess of string * expr list
  | FuncCall of string * expr list
  | Noexpr

type datatype =
    SizedArray of primitive * int list
  | UnsizedArray of primitive * int
  | Datatype of primitive

type argdecl = {
  argdeclType : datatype;
  argdeclID : string;
}

type vardecl = {
  declTyp : datatype;
  declID : string;
  declInit : expr;
}

type stmt =
    Block of stmt list
  | Expr of expr
  | Decl of vardecl
  | Return of expr
  | If of expr * stmt * stmt
  | For of expr * expr * expr * stmt
  | While of expr * stmt
  | Continue
  | Break

type funcdecl = {
    typ : datatype;
    fname : string;
    args : argdecl list;
    body : stmt list;
  }

type program = vardecl list * funcdecl list
```

8.5 blur.ml (authored by Melissa, Dexter, Tim)

```
open Prettyprint
open Ast
open Generator

(* open Llvm *)
(* open Llvm_analysis *)
```

```
type action = Pretty | Llvm | Checked_Llvm | StdLib_Llvm

let _ =
       let action = if Array.length Sys.argv > 1 then
              List.assoc Sys.argv.(1) [ ("-p", Pretty);
                     ("-l", Llvm);
                         ("-c", Checked_Llvm); ("-ls", StdLib_Llvm) ]
              else Checked_Llvm in
       let lexbuf = Lexing.from_channel stdin in
       let ast = Parser.program Scanner.token lexbuf in
       (*Semantic_analyzer.check_prog ast;*)
       ast;
       match action with
       Pretty -> print_endline (Prettyprint.string_of_prog ast)
       | Llvm -> print_string (Llvm.string_of_llmodule (Generator.translate ast
false))
        | StdLib_Llvm -> print_string (Llvm.string_of_llmodule (Generator.translate
ast true))
       | Checked_Llvm -> let m = Generator.translate ast false in
              Llvm_analysis.assert_valid_module m;
              print_string (Llvm.string_of_llmodule m)
```

8.6 prettyprint.ml (authored by Melissa, Tim)

```
open Ast

(* Pretty-printing functions *)

let rec string_of_op = function
    Add -> "+"
  | Sub -> "-"
  | Mult -> "*"
  | Div -> "/"
  | Mod -> "%"
  | Asn -> "="
  | Eq -> "=="
  | Neq -> "!="
  | Lt -> "<"
  | Leq -> "<="
  | Gt -> ">"
  | Geq -> ">="
  | And -> "and"
  | Or -> "or"

and string_of_unop o e = match o with
    Not -> "!" ^ string_of_expr e
  | Neg -> "-" ^ string_of_expr e
  | Mag -> "|" ^ string_of_expr e ^ "|"
```

```
and string_of_typ = function
    Int -> "int"
  | Double -> "double"
  | Char -> "char"
  | String -> "string"
  | Bool -> "bool"
  | Void -> "void"

and  str_brackets d str = if d > 0 then str_brackets (d - 1) ("[]" ^ str) else str

and string_of_array t d = string_of_typ t  ^ (str_brackets d "")


and string_of_datatype = function
    UnsizedArray(t, d) -> string_of_array t d
  | SizedArray(t, el)   -> string_of_typ t ^ "[" ^ String.concat "][" (List.map
string_of_int el) ^ "]"
  | Datatype(t) -> string_of_typ t

and string_of_expr = function
    IntLit(l) -> string_of_int l
  | DoubleLit(l) -> string_of_float l
  | StrLit(l) -> "\"" ^ l ^ "\""
  | CharLit(l) -> "'" ^ Char.escaped l ^ "'"
  | BoolLit(l) -> string_of_bool l
  | Id(s) -> s
  | Binop(e1, o, e2) -> "\t" ^ string_of_expr e1 ^ " " ^ string_of_op o ^ " " ^
string_of_expr e2
  | Unop(o, e) -> string_of_unop o e
  | ArrayListInit(l) -> "[" ^ String.concat ", " (List.map string_of_expr l) ^ "]"
  (*| ArraySizeInit(t, n) -> string_of_typ t ^ "[" ^ String.concat "][" (List.map
string_of_expr n) ^ "]" *)
  | ArrayAccess(id, dl) -> "\t" ^ id ^ "[" ^ String.concat "][" (List.map
string_of_expr dl) ^ "]"
  | FuncCall(n, p) -> n ^ "(" ^ String.concat ", " (List.map string_of_expr p) ^ ")"
  | Noexpr -> ""

let string_of_argdecl a = string_of_datatype a.argdeclType ^ " " ^ a.argdeclID

let string_of_vardecl_simple vdecl =
    "\t" ^ string_of_datatype vdecl.declTyp ^ " " ^
    vdecl.declID ^
    string_of_expr vdecl.declInit ^ ";\n"

let string_of_init_vardecl vdecl =
    "\t" ^ string_of_datatype vdecl.declTyp ^ " " ^
    vdecl.declID ^ " = " ^
    string_of_expr vdecl.declInit ^ ";\n"

let string_of_vardecl vdecl = match vdecl.declInit with
    Noexpr ->  string_of_vardecl_simple vdecl
  | _ -> string_of_init_vardecl vdecl
```

```
let rec string_of_stmt = function
    Block(stmts) ->
      "{\n" ^ String.concat "" (List.map string_of_stmt stmts) ^ "}\n"
  | Expr(expr) -> string_of_expr expr ^ ";\n"
  | Decl(decl) -> string_of_vardecl decl
  | Return(expr) -> "return" ^ " " ^ string_of_expr expr ^ ";\n"
  | If(e, s1, s2) -> "if (" ^ string_of_expr e ^ ") {\n" ^ string_of_stmt s1 ^ "}\n
else {\n" ^ string_of_stmt s2 ^ "}\n"
  | For(e1, e2, e3, s) -> "for (" ^ string_of_expr e1 ^ ";" ^ string_of_expr e2 ^ ";"
^ string_of_expr e3 ^ ")\n" ^ string_of_stmt s ^ ";\n"
  | While(e, s) -> "while (" ^ string_of_expr e ^ ")\n" ^ string_of_stmt s ^ ";\n"
  | Continue -> "continue;"
  | Break -> "break;"

let string_of_funcdecl fdecl =
  string_of_datatype fdecl.typ ^ " " ^
  fdecl.fname ^ "(" ^
  String.concat ", " (List.map string_of_argdecl fdecl.args) ^ ")\n{\n" ^
  String.concat "" (List.map string_of_stmt fdecl.body) ^ "}\n"

let string_of_prog (vars, funcs) =
    String.concat "" (List.map string_of_vardecl vars) ^ "\n" ^
    String.concat "\n" (List.map string_of_funcdecl funcs)
```

8.7 semantic_analysis.ml (authored by Melissa, Tim)

```
open Ast

module A = Ast

module StringMap = Map.Make(String)

type symbol_table = {
    parent: symbol_table option;
    args: argdecl list;
    variables: vardecl list
}

type func_entry = {
    name: string;
    arg_types: datatype list;
    return_type: datatype
}

type env = {
    symtab: symbol_table;
    funcs: func_entry list;
    return_type: datatype
}
```

```
let string_of_typ = function
    Int -> "int"
  | Double -> "double"
  | Char -> "char"
  | String -> "string"
  | Bool -> "bool"
  | Void -> "void"

let string_of_datatype = function
  | Datatype(t) -> string_of_typ t

let check_prog (globals, functions) =
  (* Add global variable declarations to the symbol table *)

  let built_in_functions =
    (* print and println actually take expr, not a Datatype. We deal with this in
check_func_call*)
    [ {name = "print"; arg_types = [Datatype(String)]; return_type =
Datatype(Void);};
      {name = "println"; arg_types = [Datatype(String)]; return_type =
Datatype(Void);};
      {name = "len"; arg_types = [Datatype(String)]; return_type = Datatype(Int);};
      {name = "readGrayscaleImage"; arg_types = [Datatype(String)]; return_type =
UnsizedArray(Int, 2);};
      {name = "readColorImage"; arg_types = [Datatype(String)]; return_type =
UnsizedArray(Int, 2);};
      {name = "charToIntensity"; arg_types = [Datatype(Char)]; return_type =
Datatype(Int);};
      {name = "intensityToChar"; arg_types = [Datatype(Int)]; return_type =
Datatype(Char);};
      {name = "intcast"; arg_types = [Datatype(Double)]; return_type =
Datatype(Int);};
      {name = "doublecast"; arg_types = [Datatype(Int)]; return_type =
Datatype(Double);};
      {name = "canvas"; arg_types = [Datatype(String)]; return_type =
UnsizedArray(Char, 2);};
      {name = "dither"; arg_types = [Datatype(String)]; return_type =
UnsizedArray(Char, 2);}; ]
  in

  let is_arith (t : datatype) :bool =
    match t with
      | Datatype(Int) | Datatype(Double) -> true
      | _ -> false
  in

  let is_logical (t : datatype) :bool =
    match t with
    | Datatype(Int) | Datatype(Double) | Datatype(Char) | Datatype(String) |
Datatype(Bool) -> true
    | _ -> false
  in
```

```
  let is_bool (t : datatype) :bool =
    match t with
    Datatype(Bool) -> true
    | _ -> false
  in

  (* A global variable cannot have type void. *)
  let check_not_void (vdecl : vardecl) =
    (* Get the types of the globals *)
    let global_typ = (fun v -> v.declTyp) vdecl in
        if global_typ = Datatype(Void) then raise
          (Failure ("illegal void variable " ^  vdecl.declID))
        else ()
  in
  List.iter check_not_void globals;

  let rec get_variable_decl (symtab : symbol_table) (id : string) :datatype =
    try
      let decl = List.find (fun vdecl -> vdecl.declID = id) symtab.variables in
decl.declTyp
      with
      | Not_found -> try let decl = List.find (fun adecl -> adecl.argdeclID = id)
symtab.args in decl.argdeclType
            with
            | Not_found -> (match symtab.parent with
              | Some parent -> get_variable_decl parent id
              | _ -> raise Not_found) in

        let check_arr_access_type symtab s elist =
            let num_dims = (List.length elist) in
            let arr_type =
                (try get_variable_decl symtab s with Not_found -> raise (Failure
("undeclared identifier " ^ s)))
            in
            match arr_type with
              UnsizedArray(p, d) ->
                  if (d > num_dims) then
                      UnsizedArray(p, d - num_dims)
                  else if (d < num_dims) then
                      raise (Failure ("Array accessing more dimensions than exist"))
                  else
                      Datatype(p)
            | SizedArray(p, dl) ->
                    let tot_dims = (List.length dl) in
                    if (tot_dims > num_dims) then
                        let result_dim = (List.nth dl (tot_dims - num_dims)) in
                        SizedArray(p, [result_dim])
                    else if (tot_dims < num_dims) then
                        raise (Failure ("Array accessing more dimensions than
exist"))
                    else
                        Datatype(p)
```

```
        in

  (* Returns datatype of expression. *)
  let rec check_expr (env : env) (expr : expr) =
    match expr with
      IntLit i -> Datatype(Int)
    | DoubleLit d -> Datatype(Double)
    | CharLit c -> Datatype(Char)
    | StrLit s -> Datatype(String)
    | BoolLit b -> Datatype(Bool)
    | Noexpr -> Datatype(Void)
    | ArrayListInit elist -> check_arr_literal env elist
    | ArrayAccess (s, elist) -> check_arr_access_type env.symtab s elist
      (* Check that you're accessing something available*)
    | Id s ->
        (* This gets the type of the variable. *)
        (try get_variable_decl env.symtab s
        with | Not_found -> raise (Failure ("undeclared identifier " ^ s))
        )
    | Unop (op, e) ->
      let t = check_expr env e in
      (match op with
      | Mag ->
        if (t <> Datatype(Char) && t <> Datatype(Int)) then raise (Failure("illegal
operation"))
        else
          (if t = Datatype(Int) then Datatype(Char)
          else Datatype(Int))
      | Not -> if t <> Datatype(Bool) then raise (Failure("illegal operation"))
        else Datatype(Bool)
      | Neg -> if (t <> Datatype(Int) && t <> Datatype(Double)) then raise
(Failure("illegal operation"))
        else t
      | _ -> raise(Failure("illegal unop")))
    | FuncCall (s, arglist) -> check_func_call s arglist env
    | Binop (e1, op, e2) ->
      let t1 = check_expr env e1
      and t2 = check_expr env e2 in
      match op with
      | Add | Sub | Mult | Div | Mod ->
        if is_arith t1 && t1 = t2 then t1
        else raise (Failure ("illegal operation"))
      | Lt | Leq | Gt | Geq | Eq | Neq | And | Or ->
        if is_logical t1 && t1 = t2 then Datatype(Bool)
        else raise (Failure("invalid operands"))
      | Asn ->
        if t1 = t2 then t1
        else raise (Failure ("illegal assignment"))

        (* get type of an expr list, potentailly nested. Blur supports up to 2D
arrays. *)
        and check_arr_literal env elist =
            let tot_dims = match (List.hd elist) with
```

```
            ArrayListInit(el) -> 2
          | _                 -> 1
          in let data_typ = match (List.hd elist) with
            ArrayListInit(el) -> check_expr env (List.hd el)
          | e                 -> check_expr env e
          in let prim_typ = match data_typ with Datatype(p) -> p in
UnsizedArray(prim_typ, tot_dims)

  (* Checking function call returns the type of the function. *)
  and check_func_call (id : string) (args : expr list) (env : env) =
    try
      let func_entry = List.find (fun f -> f.name = id) env.funcs in
      (* Get the types of the arg expressions. *)
      let arg_types = List.map(fun arg -> check_expr env arg) args in
      (* Ensure that arguments match. *)
      if List.length func_entry.arg_types <> List.length args then
      raise (Failure ("Incorrect number of args for function call " ^ id ^
        ". Expecting " ^ (string_of_int (List.length func_entry.arg_types)) ^ " args
but got "
        ^ (string_of_int (List.length args)))) else
      if id <> "print" && id <> "println" && id <> "len" && arg_types <>
func_entry.arg_types then
      raise (Failure("unexpected arg types")) else
      func_entry.return_type
    with | Not_found -> raise (Failure ("undeclared function " ^ id))
  in

  let var_add (env : env) (decl : vardecl) =
    let etype = check_expr env decl.declInit in
    if etype = decl.declTyp || decl.declInit = Noexpr then (* declInit must be same
type as declTyp. *)
      (try
        let _ =
          (* Error out if local variable with same name already exists. *)
          List.find
            (fun vdecl -> vdecl.declID = decl.declID) env.symtab.variables
        in raise (Failure ("Duplicate variable " ^ decl.declID))
      with
      | Not_found ->
        let new_symbol_table =
          {
            (env.symtab)
            with
            variables = decl :: env.symtab.variables;
          } in
        let new_env = { (env) with symtab = new_symbol_table; }
        and vdecl =
          {
            declTyp = decl.declTyp;
            declID = decl.declID;
            declInit = decl.declInit;
          }
        in (new_env, vdecl))
```

```
    else raise (Failure("variable declaration type mismatch"))
in

(* Add ArrayListInit as declInit of vardecl to env. *)
let adding_arr (env : env) (decl : vardecl) (p : primitive) =
  (try
    let _ =
      (* Error out if local variable with same name already exists. *)
      List.find
        (fun vdecl -> vdecl.declID = decl.declID) env.symtab.variables
    in raise (Failure ("Duplicate variable " ^ decl.declID))
  with
  | Not_found ->
    let new_symbol_table =
      {
        (env.symtab)
        with
        variables = decl :: env.symtab.variables;
      } in
    let new_env = { (env) with symtab = new_symbol_table; }
    and vdecl =
      {
        declTyp = decl.declTyp; (* UnsizedArray(p, int) *)
        declID = decl.declID;
        declInit = decl.declInit; (* ArrayListInit(elist) *)
      }
    in (new_env, vdecl))
  in

  (* Add array when it is initialized by a function that returns an array. *)
  let adding_arr_func_call (env : env) (decl : vardecl) (p : primitive) =
  (try
    let _ =
      (* Error out if local variable with same name already exists. *)
      List.find
        (fun vdecl -> vdecl.declID = decl.declID) env.symtab.variables
    in raise (Failure ("Duplicate variable " ^ decl.declID))
  with
  | Not_found ->
    let new_symbol_table =
      {
        (env.symtab)
        with
        variables = decl :: env.symtab.variables;
      } in
    let new_env = { (env) with symtab = new_symbol_table; }
    and vdecl =
      {
        declTyp = decl.declTyp; (* UnsizedArray(p, int) *)
        declID = decl.declID;
        declInit = decl.declInit; (* ArrayListInit(elist) *)
      }
    in (new_env, vdecl))
```

```
    in

  (* When an unsized array is declared, the RHS must be
  an ArrayListInit, or a function that returns ArrayListInit. *)
  let var_add_arr (env : env) (decl : vardecl) (p : primitive) =
    match decl.declInit with
    | ArrayListInit(elist) -> adding_arr env decl p
    | FuncCall(s, elist) -> adding_arr_func_call env decl p
    | _ -> raise (Failure("illegal array initialization"))
  in

  let check_variable_declaration (env : env) (decl: vardecl) =

    (* A variable cannot have type void. *)
    let check_not_void_var (decl : vardecl) =
      let var_typ = (fun v -> decl.declTyp) decl in
          if var_typ = Datatype(Void) then raise (Failure ("illegal void variable " ^
decl.declID))
          else ()
    in
    ignore(check_not_void_var (decl));

    match decl.declTyp with
    | UnsizedArray(p,d) ->
      if decl.declInit = Noexpr then raise(Failure("unsized array must be
initialized"))
      else var_add_arr env decl p
    | SizedArray(p, intlist) -> if decl.declInit = Noexpr then adding_arr env decl p
else raise (Failure("illegal array initialization"))
    | _ -> var_add env decl

  in

  (* Return env and stmt tuple. *)
  let rec check_stmt (env : env) (stmt : stmt) :(env * stmt) =
    match stmt with
      Expr e -> ignore(check_expr env e); (env, stmt) (* Expression cannot mutate the
environment. *)
    (* Return current env since Blocks have their own scope. *)
    | Block stmt_list ->
      let new_symbol_table = { parent = Some env.symtab; variables = []; args = []; }
in
      let (_, checked_stmts) = check_stmt_list { (env) with symtab =
new_symbol_table; } stmt_list in
      (env, stmt)
    | Decl vdecl -> (* Return new env*)
      let (new_env, vdecl) = check_variable_declaration env vdecl
      in (new_env, stmt)
    | If (e, s1, s2) ->
      let checked_expr = check_expr env e
      and (_, checked_s1) = check_stmt env s1
      and (_, checked_s2) = check_stmt env s2 in
      if is_bool checked_expr then (env, stmt)
```

```
    else raise(Failure("illogical if"))
  | For (e1, e2, e3, s) ->
    let checked_e1 = check_expr env e1
    and checked_e2 = check_expr env e2
    and checked_e3 = check_expr env e3 in
    if is_bool checked_e2 then (env, stmt)
    else raise(Failure("illogical for"))
  | While (e, s) ->
    let checked_expr = check_expr env e
    and (_, checked_stmt) = check_stmt env s in
    if is_bool checked_expr then (env, stmt)
    else raise(Failure("illogical while"))
  | Return e -> let e_type = check_expr env e in
    match env.return_type with
      | return_type ->
        if e_type = return_type then (env, stmt)
        else raise (Failure ("incorrect return type"))
(* Each statement takes the environment updated from the previous statement. *)
and check_stmt_list (env : env) ( slist : stmt list ) :(env * stmt list) =
  let(new_env, stmts) =
    List.fold_left (fun acc stmt ->
      let (nenv, s) = check_stmt (fst acc) stmt
      in (nenv, (s :: (snd acc)))) (env, []) slist
  in (new_env, List.rev stmts)
in

(* Check arguments *)
let check_argdecl (env : env) (adecl : argdecl) =

  (* An argument cannot have type void. *)
  let check_not_void_arg (adecl : argdecl) =
    let arg_typ = (fun a -> a.argdeclType) adecl in
        if arg_typ = Datatype(Void) then raise (Failure ("illegal void arg"))
        else ()
  in
  ignore(check_not_void_arg (adecl));

  (try
    let _ =
      (* Error out if local variable with same name already exists. *)
      List.find
        (fun argdecl -> argdecl.argdeclID = adecl.argdeclID) env.symtab.args
    in raise (Failure (";Duplicate variable " ^ adecl.argdeclID))
  with
  | Not_found ->
    let new_symbol_table =
      {
        (env.symtab)
        with
        args = adecl :: env.symtab.args;
      } in
    let new_env = { (env) with symtab = new_symbol_table; }
    and arg =
```

```
      {
        argdeclType = adecl.argdeclType;
        argdeclID = adecl.argdeclID;
      }
    in (new_env, adecl))
in

(* Add function declaration to the environment. *)
let add_function_declaration (env : env) (fdecl : funcdecl) :(env * funcdecl) =
  if fdecl.fname="main" && (List.length fdecl.args) > 0
    then raise (Failure("main() may not take args")) else
  if (List.mem fdecl.fname (List.map (fun f -> f.name) built_in_functions)) then
  raise (Failure ("Cannot overwrite built-in function!!")) else
  if (List.mem fdecl.fname (List.map (fun f -> f.name) env.funcs)) then
  raise (Failure ("Duplicate function.")) else
  (* Get the types of the function's arguments. *)
  let a_types = List.map (fun adecl -> adecl.argdeclType) fdecl.args in
  (* Make a function entry for the function. *)
  let func_entry =
    {
      name = fdecl.fname;
      arg_types = a_types;
      return_type = fdecl.typ;
    } in
  let new_funcs = func_entry :: env.funcs in
  (* Make a new symbol table for the function scope. *)
  let new_symbol_table =
    {
      parent = Some env.symtab;
      args = [];
      variables = [];
    } in
  (* Add the function to the environment
  For now, the symbol table and return type have empty local scope. *)
  let new_env =
  {
    (env)
    with
    symtab = new_symbol_table;
    funcs =  new_funcs;
    return_type = fdecl.typ;
  } in
  (* Add the args to the function scope. *)
  let (env_with_args, argdecl_list) =
    List.fold_left (fun acc argdecl ->
      let (nenv, arg) = check_argdecl (fst acc) argdecl
      in (nenv, (arg :: (snd acc)))) (new_env, []) fdecl.args in
  let (_, func_body) =
    check_stmt_list env_with_args fdecl.body in
  let func_body = func_body in
  let f =
  {
    typ = fdecl.typ;
```

```
    fname = fdecl.fname;
    args = List.rev argdecl_list;
    body = func_body;
  } in
  (* Return the environment with this added function. *)
  ({ (env_with_args) with funcs = new_funcs; }, f)
in

(* Establish initial environment *)
let env =
  {
    symtab = { parent = None; variables = []; args = []; };
    funcs = built_in_functions;
    return_type = Datatype(Int);
  } in

(* Add global variables to the environment. *)
let check_global_var (env : env) (vdecl : vardecl) =
  (try
    let _ =
      (* Error out if global variable with same name already exists. *)
      List.find
        (fun v -> v.declID = vdecl.declID) env.symtab.variables
    in raise (Failure ("Duplicate variable " ^ vdecl.declID))
  with
    | Not_found ->
      let new_symbol_table =
        {
          (env.symtab)
          with
          variables = vdecl :: env.symtab.variables;
        } in
      let new_env = { (env) with symtab = new_symbol_table; }
      and vardecl =
        {
          declTyp = vdecl.declTyp;
          declID = vdecl.declID;
          declInit = vdecl.declInit;
        }
      in (new_env, vardecl))
in

(* Add globals to env. *)
let(new_env, vars) =
  List.fold_left (fun acc v ->
    let (nenv, v) = check_global_var (fst acc) v
    in (nenv, (v :: (snd acc)))) (env, []) globals
in

(* Adding func decl to env, which also adds args to env.*)
let (new_env, funcs) =
  List.fold_left (fun acc f ->
    let(nenv, f) = add_function_declaration (fst acc) f
```

```
      in (nenv, (f :: (snd acc)))) (new_env, []) (List.rev functions)
  in

  let env_func_names = List.map (fun f -> f.name) env.funcs in

  let (new_env, funcs) =
    try
    let _ = List.find (fun func -> func.name = "main") new_env.funcs in
    (new_env, funcs)
    with | Not_found -> raise (Failure("no main"));
  in

  let check_function functions =

    (* Return list of functions after checking functions. *)
    functions in

    (* After semantically checking, we return the program -
    a tuple of a list of globals and a list of functions. *)
    (globals, functions);
```

8.8 generator.ml (authored by Tim)

```
(* code generation: translate takes semantically checked AST and produces LLVM IR *)

open Ast
open Llvm
open Exceptions

module L = Llvm
module A = Ast

module StringMap = Map.Make(String)

let translate (globals, functions) use_stdLib =
    let context = L.global_context() in
    let the_module = L.create_module context "Blur" in

    let i32_t = L.i32_type context
    and iFl_t = L.double_type context
    and i8_t = L.i8_type context
    and i1_t = L.i1_type context
    and void_t = L.void_type context in


    let string_t = L.pointer_type i8_t in
    let int_ptr_t = L.pointer_type i32_t in
    let array_t = L.array_type in
    let zero_t = L.const_int i32_t 0 in
```

```
    (* UnsizedArray types *)
    let img_t = L.struct_type context [| i32_t; i32_t; i32_t; int_ptr_t |] in
    let char_struct = L.struct_type context [| i32_t; i32_t; i32_t; string_t |] in
    let float_struct = L.struct_type context [| i32_t; i32_t; i32_t; (L.pointer_type
iFl_t) |] in


    let rec ltype_of_sized_array t el =
        match (List.length el) with
            3 -> array_t (array_t (array_t (ltype_of_typ (Datatype(t))) (List.nth el
2)) (List.nth el 1)) (List.nth el 0)
          | 2 -> array_t (array_t (ltype_of_typ (Datatype(t))) (List.nth el 1))
(List.nth el 0)
          | 1 -> array_t (ltype_of_typ (Datatype(t))) (List.hd el)
          | _ -> raise (Exceptions.NotADatatype)

    and ltype_of_typ (d: A.datatype) = match d with
        Datatype(A.Int) -> i32_t
      | Datatype(A.Double) -> iFl_t
      | Datatype(A.Char) -> i8_t
      | Datatype(A.String) -> string_t
      | Datatype(A.Bool) -> i1_t
      | Datatype(A.Void) -> void_t
      | UnsizedArray(t, d) -> ltype_to_struct t (* wow can u believe it lol *)
      | SizedArray(t, el)  -> ltype_of_sized_array t el
      | _            -> raise (Exceptions.NotADatatype)

    and get_struct_type ptrTyp = match ptrTyp with
      i32_t    -> img_t
    | i8_t   -> char_struct
    | iFl_t   -> float_struct

    and ltype_to_struct (p: A.primitive) = match p with
      A.Int     -> img_t
    | A.Double  -> float_struct
    | A.Char    -> char_struct
    in

    let global_vars =
        let global_var map (vdecl : A.vardecl) =
            let typ = vdecl.declTyp in
            let name = vdecl.declID in

            let init = L.const_int (ltype_of_typ typ) 0
            in StringMap.add name (L.define_global name init the_module) map in
        List.fold_left global_var StringMap.empty globals in

    let builtin_decls = StringMap.empty in

    (* DECLARE EXTERNAL C LIBRARY FUNCTIONS *)
    let printf_t = L.var_arg_function_type i32_t [| L.pointer_type i8_t |] in
    let printf_func = L.declare_function "printf" printf_t the_module in
    let builtin_decls = StringMap.add "printf" printf_func builtin_decls in
```

```
    let getimg_t = L.var_arg_function_type img_t [| L.pointer_type i8_t |] in
    let getimg_func = L.declare_function "readGrayscaleImage" getimg_t the_module in
    let builtin_decls = StringMap.add "readGrayscaleImage" getimg_func builtin_decls
in

    let canvas_t = L.var_arg_function_type char_struct [| string_t |] in
    let canvas_func = L.declare_function "canvas" canvas_t the_module in
    let builtin_decls = StringMap.add "canvas" canvas_func builtin_decls in

    (* DECLARE BLUR BUILT-INS *)
    let charToInt_t = L.var_arg_function_type i32_t [| i8_t |] in
    let charToInt_f = L.declare_function "charToIntensity" charToInt_t the_module in
    let builtin_decls = StringMap.add "charToIntensity" charToInt_f builtin_decls in

    let intensityToChar_t = L.var_arg_function_type i8_t [| i32_t |] in
    let intensityToChar_f = L.declare_function "intensityToChar" intensityToChar_t
the_module in
    let builtin_decls = StringMap.add "intensityToChar" intensityToChar_f
builtin_decls in

    let adjust_px_t = L.var_arg_function_type i8_t [| i8_t; i32_t |] in
    let adjust_px_f = L.declare_function "adjustPX" adjust_px_t the_module in
    let builtin_decls = StringMap.add "adjustPX" adjust_px_f builtin_decls in

    let builtin_decls =
        (* CONDITIONALLY DECLARE BLUR STANDARD LIBRARY FUCTIONS *)
        if use_stdLib then
            let edgeDetect_t = L.var_arg_function_type img_t [| string_t; i32_t |] in
            let edgeDetect_f = L.declare_function "edgeDetect" edgeDetect_t
the_module in
            let builtin_decls = StringMap.add "edgeDetect" edgeDetect_f builtin_decls
in

            let pixelDistance_t = L.var_arg_function_type i32_t [| i32_t; i32_t |] in
            let pixelDistance_f = L.declare_function "pixelDistance" pixelDistance_t
the_module in
            let builtin_decls = StringMap.add "pixelDistance" pixelDistance_f
builtin_decls in

            let dither_t = L.var_arg_function_type char_struct [| string_t |] in
            let dither_f = L.declare_function "dither" dither_t the_module in
            let builtin_decls = StringMap.add "dither" dither_f builtin_decls in

            let impose_t = L.var_arg_function_type char_struct [| char_struct; img_t;
i8_t |] in
            let impose_f = L.declare_function "impose" impose_t the_module in
            StringMap.add "impose" impose_f builtin_decls
        else builtin_decls
    in

    (* define each function w/ args and return type so we can call it *)
    let function_decls =
```

```
        let function_decl map fdecl =
            let name = fdecl.A.fname
            and formal_types = Array.of_list (List.map (fun (typ) -> ltype_of_typ
typ.argdeclType) fdecl.A.args) in
            let ftype = L.function_type (ltype_of_typ fdecl.A.typ) formal_types in
            StringMap.add name (L.define_function name ftype the_module, fdecl) map
in

        List.fold_left function_decl StringMap.empty functions in

    let codegen_func func_decl =
        let (f, _) = StringMap.find func_decl.A.fname function_decls in

        let llbuilder = L.builder_at_end context (L.entry_block f) in


        (* format strings for println() *)
        let int_format_str = L.build_global_stringptr "%d\n" "int_fmt" llbuilder
        and str_format_str = L.build_global_stringptr "%s\n" "str_fmt" llbuilder
        and chr_format_str = L.build_global_stringptr "%c\n" "chr_fmt" llbuilder
        and flt_format_str = L.build_global_stringptr "%f\n" "flt_fmt" llbuilder

        in

        let local_vars = StringMap.empty in
        let arr_dims = StringMap.empty in

        let add_formal map (typ, name) fml =
            L.set_value_name name fml;
            let local = L.build_alloca (ltype_of_typ typ) name llbuilder in
            ignore (L.build_store fml local llbuilder);
            StringMap.add name local map
        in
        let add_local (vdecl: A.vardecl) local_vars =
            let typ = vdecl.declTyp in
            let name = vdecl.declID in
            let local_var = L.build_alloca (ltype_of_typ typ) name llbuilder in
            StringMap.add name local_var local_vars
        in

        (* Only add each function's args for now, will add to map when we encounter a
varDecl in the functions body,
         * which is a statement list *)

        let local_vars = List.fold_left2 add_formal local_vars (List.map (fun (t) ->
(t.argdeclType, t.argdeclID)) func_decl.A.args) (Array.to_list (L.params f)) in

        let maps = (local_vars, arr_dims) in

        (* see if a variable has been declared already *)
        let rec lookup name locals =
            try StringMap.find name locals
            with Not_found -> try StringMap.find name global_vars
```

```
            with Not_found -> raise (Exceptions.UnknownVariable name)
        in

         let rec codegen_binop e1 op e2 maps llbuilder =
            let int_ops lh op rh  =
                match op with
                A.Add   -> L.build_add lh rh "tmp" llbuilder
              | A.Sub   -> L.build_sub lh rh "tmp" llbuilder
              | A.Mult  -> L.build_mul lh rh "tmp" llbuilder
              | A.Div   -> L.build_sdiv lh rh "tmp" llbuilder
              | A.Mod   -> L.build_srem lh rh "whocares" llbuilder
              | A.And   -> L.build_and lh rh "tmp" llbuilder
              | A.Or    -> L.build_or lh rh "tmp" llbuilder
              | A.Eq    -> L.build_icmp Icmp.Eq lh rh "tmp" llbuilder
              | A.Neq   -> L.build_icmp Icmp.Ne lh rh "tmp" llbuilder
              | A.Lt    -> L.build_icmp Icmp.Slt lh rh "tmp" llbuilder
              | A.Leq   -> L.build_icmp Icmp.Sle lh rh "tmp" llbuilder
              | A.Gt    -> L.build_icmp Icmp.Sgt lh rh "tmp" llbuilder
              | A.Geq   -> L.build_icmp Icmp.Sge lh rh "tmp" llbuilder

            in

            let float_ops lh op rh =
                match op with
                A.Add   -> L.build_fadd lh rh "flt_addtmp" llbuilder
              | A.Sub   -> L.build_fsub lh rh "flt_subtmp" llbuilder
              | A.Mult  -> L.build_fmul lh rh "flt_multmp" llbuilder
              | A.Div   -> L.build_fdiv lh rh "flt_divtmp" llbuilder
              | A.Mod   -> L.build_frem lh rh "frem" llbuilder
              | A.Eq    -> L.build_fcmp Fcmp.Oeq lh rh "flt_eqtmp" llbuilder
              | A.Neq   -> L.build_fcmp Fcmp.One lh rh "flt_neqtmp" llbuilder
              | A.Lt    -> L.build_fcmp Fcmp.Ult lh rh "flt_lesstmp" llbuilder
              | A.Leq   -> L.build_fcmp Fcmp.Ole lh rh "flt_leqtmp" llbuilder
              | A.Gt    -> L.build_fcmp Fcmp.Ogt lh rh "flt_sgttmp" llbuilder
              | A.Geq   -> L.build_fcmp Fcmp.Oge lh rh  "flt_sgetmp" llbuilder
              | _       -> raise (Exceptions.FloatOpNotSupported)
            in

            let arith_binop e1 op e2 =
                let lh = codegen_expr (maps, llbuilder) e1
                and rh = codegen_expr (maps, llbuilder) e2
                in
                (* peanut brittle! *)
                let op_typ = L.string_of_lltype (L.type_of lh) in match op_typ with
                  "i32" -> int_ops lh op rh
                | "double" -> float_ops lh op rh
                | "i8"     -> int_ops lh op rh (* chars are treated as ints *)
                | "i1"     -> int_ops lh op rh
                | _        -> raise (Exceptions.NotSupported)
            in
            let assign_binop e1 e2 =
                (* if the assignment involves array dimensions, update the arr_dims
map *)
```

```
                let maps = ((fst maps), arr_dims) in
                match e1 with
                  A.Id s        -> codegen_asn s (codegen_expr (maps, llbuilder) e2)
maps llbuilder
                | A.ArrayAccess(n, dl) -> codegen_asn_arr e1 e2 maps llbuilder
                | _                    -> raise (Exceptions.NotAssignable)
            in

            let handle_binop e1 op e2 =
                match op with
                A.Asn         -> assign_binop e1 e2
              | _             -> arith_binop e1 op e2

            in
            handle_binop e1 op e2

        and codegen_unop op e maps llbuilder =
            let exp = (codegen_expr (maps, llbuilder)) e in
            if (L.type_of exp) = iFl_t then
                L.build_fneg exp "flt_unoptmp" llbuilder
            else
            match op with
            A.Neg       -> L.build_neg exp "int_unoptmp" llbuilder
          | A.Not       -> L.build_not exp "bool_unoptmp" llbuilder
          | A.Mag       -> (match (L.string_of_lltype (L.type_of exp)) with
                             "i32" -> L.build_call intensityToChar_f [| exp |]
"mag_call" llbuilder
                           | "i8"  -> L.build_call charToInt_f [| exp |]
"mag_callchar" llbuilder)
          | _           -> raise (Exceptions.NotSupported)

        (* helper to get the raw string from an ID expression type. MOVE TO A UTILS
FILE *)
        and id_to_str id = match id with
            A.Id s      -> s
          | A.ArrayAccess(n, dl) -> n
          | _           -> raise Exceptions.NotAnId


        and codegen_asn_arr e1 e2 maps llbuilder =
            let gen_e1 = (match e1 with
              A.ArrayAccess(n, dl)    -> build_array_access n dl maps llbuilder true
              | _                      -> raise Exceptions.IllegalAssignment)
            in
            let gen_e2 = codegen_expr (maps, llbuilder) e2 in
            ignore(L.build_store gen_e2 gen_e1 llbuilder); gen_e2

        and codegen_asn n gen_e maps llbuilder =
            let locals = fst maps in
            ignore(L.build_store gen_e (lookup n locals) llbuilder); gen_e

        and codegen_print e maps llbuilder newLine =
            let param = (codegen_expr (maps, llbuilder) e) in
```

```
            let theType = L.string_of_lltype (L.type_of param) in
            if newLine then
                let fmt_str = match theType with
                  "i32"     -> int_format_str
                | "double"  -> flt_format_str
                | "i8"      -> chr_format_str
                | "i8*"     -> str_format_str
                | "i1"      -> int_format_str
                | _         -> raise (Exceptions.NotSupported)

                in
                L.build_call printf_func [| fmt_str; param |] "println" llbuilder
            else
                let fmt_str = match theType with
                  "i32"    -> "%d"
                | "double" -> "%f"
                | "i8"     -> "%c"
                | "i8*"    -> "%s"
                | "i1"     -> "%d"
                | _        -> "%d" (* default *)

                in
                let str_ptr = L.build_global_stringptr fmt_str "print_fmt" llbuilder
in
                L.build_call printf_func [| str_ptr; param |] "print" llbuilder

        and codegen_call f el (maps, llbuilder) =
            let args = List.rev (List.map (codegen_expr (maps, llbuilder)) (List.rev
el)) in
            if StringMap.mem f builtin_decls then
                let func = StringMap.find f builtin_decls in
                L.build_call func (Array.of_list args) "" llbuilder
            else
                let (fdef, fdecl) = StringMap.find f function_decls in
                L.build_call fdef (Array.of_list args) "" llbuilder

        and get_img_handler e (maps, llbuilder) =
            let img_name = codegen_expr (maps, llbuilder) e in
            let img_loc = L.build_alloca (img_t) "imgloc" llbuilder in
            let res = L.build_call getimg_func [| img_name |] "grayScaleImgfunccall"
llbuilder in
            ignore(L.build_store res img_loc llbuilder); res

        and get_canvas_handler e (maps, llbuilder) =
            let img_name = codegen_expr (maps, llbuilder) e in
            let img_loc = L.build_alloca (char_struct) "canvasloc" llbuilder in
            let res = L.build_call canvas_func [| img_name |] "canvasFuncCall"
llbuilder in
            ignore(L.build_store res img_loc llbuilder); res

        and arr_len_handler arr (maps, llbuilder) =

            if (StringMap.mem (id_to_str arr) (snd maps)) then
```

```
                let exp = codegen_expr (maps, llbuilder) arr in
                L.const_int i32_t (L.array_length (L.type_of exp))
            else
                let arr_ref = lookup (id_to_str arr) (fst maps) in
                match arr with
                  A.Id s ->
                    let width_ptr = L.build_gep arr_ref [| zero_t; zero_t |] "width"
llbuilder in
                    let width = L.build_load width_ptr "widthval" llbuilder in
                    width
                | A.ArrayAccess(n, dl) ->
                      (match (List.length dl) with
                        1 ->
                          let height_ptr = L.build_gep arr_ref [| zero_t;
L.const_int i32_t 1 |] "height" llbuilder in
                          let height = L.build_load height_ptr "heightval"
llbuilder in
                          height
                      (* this case should never be allowed *)
                      | _ -> raise (Exceptions.UnsupportedDimensions))
                | _ -> raise (Exceptions.NotAnArray)

        and codegen_expr (maps, llbuilder) e =
            match e with
            A.IntLit i                  -> L.const_int i32_t i
          | A.DoubleLit i               -> L.const_float iFl_t i
          | A.StrLit s                  -> L.build_global_stringptr s "tmp" llbuilder
          | A.CharLit c                 -> L.const_int i8_t (Char.code c)
          | A.BoolLit b                 -> if b then L.const_int i1_t 1 else
L.const_int i1_t 0
          | A.Id id                     -> L.build_load (lookup id (fst maps)) id
llbuilder
          | A.Binop(e1, op, e2)         -> codegen_binop e1 op e2 maps llbuilder
          | A.Unop(op, e)               -> codegen_unop op e maps llbuilder
          (* --- built in functions --- *)
          | A.FuncCall ("print", [e])   -> codegen_print e maps llbuilder false
          | A.FuncCall ("println", [e]) -> codegen_print e maps llbuilder true
          | A.FuncCall ("len", [arr])   -> arr_len_handler arr (maps, llbuilder)
          | A.FuncCall ("readGrayscaleImage", [e])   -> get_img_handler e (maps,
llbuilder)
          | A.FuncCall ("canvas", [e])  -> get_canvas_handler e (maps, llbuilder)
          | A.FuncCall ("intcast", [e])             -> L.build_fptosi (codegen_expr
(maps, llbuilder) e) i32_t "intcast" llbuilder
          | A.FuncCall ("doublecast", [e])          -> L.build_sitofp (codegen_expr
(maps, llbuilder) e) iFl_t "doublecast" llbuilder
          (* --- end built-ins --- *)
          | A.FuncCall (n, el)          -> codegen_call n el (maps, llbuilder)
          | A.ArrayListInit el          -> build_array_of_list el (maps, llbuilder)
          | A.ArrayAccess(n, dl)        -> build_array_access n dl maps llbuilder
false
          | A.Noexpr                    -> L.const_int i32_t 0

        (* codegen_vdecl: handle variable declarations *)
```

```
        and codegen_vdecl (vdecl: A.vardecl) (maps, llbuilder) =

            let local_vars = (fst maps) in
            match vdecl.declTyp with
              A.UnsizedArray(p, d) ->

                let gen_exp = codegen_expr (maps, llbuilder) vdecl.declInit in
                let local_vars, arr_src  =
                let typ = L.type_of gen_exp in

                (* determine if array is of a heap allocated type or not *)
                if ((typ = img_t) || (typ = char_struct) || (typ = float_struct))
then

                    let local_img_var = L.build_alloca typ vdecl.declID llbuilder in
                    let arr_ptr = L.build_gep local_img_var [| zero_t |] "arr_ptr"
llbuilder in
                    let arr_ptr_a = L.build_alloca (L.type_of arr_ptr) vdecl.declID
llbuilder in

                    ignore(L.build_store arr_ptr arr_ptr_a llbuilder);
                    let local_vars = StringMap.add vdecl.declID local_img_var
local_vars in
                    ignore(codegen_asn vdecl.declID gen_exp (local_vars, (snd maps))
llbuilder); local_vars, (snd maps)

                (* normal case, i.e. int[] a = [1,2]; *)
                else
                    let exp_typ = (L.type_of gen_exp) in (* a LLVM array type, i.e.
[3 x i32] *)
                    let local_var = L.build_malloc exp_typ vdecl.declID llbuilder in

                    let struct_typ = ltype_to_struct p in
                    let pointer_typ = L.pointer_type (ltype_of_typ (Datatype(p))) in

                    let arr_ptr = L.build_gep local_var [| zero_t |] "arr_ptr2"
llbuilder in
                    let arr_ptr = L.build_pointercast arr_ptr pointer_typ "idk"
llbuilder in
                    let arr_ptr_a = L.build_alloca (struct_typ) vdecl.declID
llbuilder in
                    ignore(L.build_store gen_exp local_var llbuilder);

                    (* use LLVM to measure cost_array dimensions, then store them in
the array struct type *)
                    let width = L.const_int i32_t (L.array_length exp_typ) in
                    let height = if d = 2 then
                        L.const_int i32_t (L.array_length exp_typ)
                    else
                        zero_t
                    in
                    (*ignore(L.build_store arr_ptr arr_ptr_a llbuilder); *)
                    let arr_struct = L.const_named_struct struct_typ [| width;
```

```
height; zero_t; (L.undef pointer_typ) |] in
                    let arr_struct1 = L.build_insertvalue arr_struct arr_ptr 3 "pls"
llbuilder in
                    let local_vars = StringMap.add vdecl.declID arr_ptr_a local_vars
in (* adding struct type *)
                    ignore(codegen_asn vdecl.declID arr_struct1 (local_vars, (snd
maps)) llbuilder); local_vars, (snd maps)
                in
                let maps = (local_vars, arr_src) in maps, llbuilder

            (* cannot be initialized, only declared *)
            | A.SizedArray(p, d) ->
                    let local_vars = add_local vdecl local_vars in
                    (local_vars, (snd maps)),  llbuilder

            | _         ->
                let local_vars = add_local vdecl local_vars in
                let maps = (local_vars, (snd maps)) in
                match vdecl.declInit with
                  A.Noexpr      -> maps, llbuilder
                | e             -> let exp = (codegen_expr (maps, llbuilder) e) in
ignore(codegen_asn vdecl.declID exp maps llbuilder); maps, llbuilder


        (* BUILD 1-dimensional array from Literal *)
        and build_array_of_list el (maps, llbuilder) =
            let llvalues = List.map (codegen_expr (maps, llbuilder)) el in
            let typ = (L.type_of (List.hd llvalues)) in
            let cool_array = Array.of_list llvalues in
            let res = L.const_array typ cool_array in res

        (* BUILD ARRAY ACCESS *)
        and build_array_access name idx_list maps llbuilder isAssign =

            let arr_handle = (lookup name (fst maps)) in
            let typ = L.type_of arr_handle in
            if ((typ = L.pointer_type img_t) || (typ = L.pointer_type char_struct) ||
(typ = L.pointer_type float_struct)) then
                (let idx_list = List.map (codegen_expr (maps, llbuilder)) idx_list in
                let depth_ptr = L.build_gep arr_handle [| zero_t; L.const_int i32_t 2
|] "depth" llbuilder in
                let depth = L.build_load depth_ptr "depthval" llbuilder in

                if depth = zero_t then (* Unsized Array initialized within a Blur
program. *)

                    let data_ptr = L.build_gep arr_handle [| zero_t; L.const_int
i32_t 3 |] "data" llbuilder in
                    let data = L.build_load data_ptr "dataval" llbuilder in (* actual
dataptr *)
                    let datatyp = L.type_of (L.build_load data "val" llbuilder) in
                    let struct_typ = get_struct_type datatyp in
                    let data = L.build_pointercast data (L.pointer_type (struct_typ))
```

```
"datacast" llbuilder in

                    let idx_list = (L.const_int i32_t 0)::[]@idx_list in
                    let idx_arr = Array.of_list idx_list in
                    let gep = L.build_gep data idx_arr name llbuilder in
                    if isAssign then
                        gep
                    else
                        L.build_load gep name llbuilder

                else (* UnsizedArray type returned from the C-Backend (follows depth
> 0 standard) *)

                    let the_arr_pointer = arr_handle in

                    let width_ptr = L.build_gep the_arr_pointer [| zero_t; zero_t |]
"width" llbuilder in
                    let data_ptr = L.build_gep the_arr_pointer [| zero_t; L.const_int
i32_t 3 |] "data" llbuilder in
                    let width = L.build_load width_ptr "widthval" llbuilder in
                    let data = L.build_load data_ptr "dataval" llbuilder in

                    let gep =
                    if List.length idx_list = 2 then
                        let offset = L.build_mul width (List.hd idx_list) "base"
llbuilder in
                        let offset = L.build_add offset (List.nth idx_list 1)
"offset" llbuilder in
                        let idx_ptr = L.build_gep data [| offset |] "idx_ptr"
llbuilder in idx_ptr

                    else
                        let idx_ptr = L.build_in_bounds_gep data [| (List.hd
idx_list) |] "idx_ptr" llbuilder in idx_ptr
                    in
                    if isAssign then
                        gep
                    else
                        let load = L.build_load gep name llbuilder in load)
            else
                (let idx_list = List.map (codegen_expr (maps, llbuilder)) idx_list in
                ignore(print_endline("; ok"));
                let idx_list = (L.const_int i32_t 0)::[]@idx_list in
                let idx_arr = Array.of_list idx_list in
                let gep = L.build_gep arr_handle idx_arr name llbuilder in
                if isAssign then
                    gep
                else
                    L.build_load gep name llbuilder)

        (* used to add a branch instruction to a basic block only if one doesn't
already exist *)
        and codegen_conditional pred then_stmt else_stmt (maps, llbuilder) =
```

```
            let bool_val = (codegen_expr (maps, llbuilder) pred) in

            let merge_bb = L.append_block context "merge" f in
            let then_bb = L.append_block context "then" f in
            let then_builder = (L.builder_at_end context then_bb) in
            let then_tup = (codegen_stmt (maps, then_builder) then_stmt) in
            add_terminal (snd then_tup) (L.build_br merge_bb);

            let else_bb = L.append_block context "else" f in
            let else_builder = (L.builder_at_end context else_bb) in
            let else_tup = (codegen_stmt (maps, else_builder) else_stmt) in
            add_terminal (snd else_tup) (L.build_br merge_bb);
            ignore (L.build_cond_br bool_val then_bb else_bb llbuilder);
            L.builder_at_end context merge_bb

        (* WHILE LOOP *)
        and codegen_while pred body (maps, llbuilder) =
            let pred_bb = L.append_block context "while" f in
            ignore (L.build_br pred_bb llbuilder);

            let body_bb = L.append_block context "while_body" f in
            add_terminal (snd (codegen_stmt (maps, (L.builder_at_end context
body_bb)) body)) (L.build_br pred_bb);

            let pred_builder = L.builder_at_end context pred_bb in
            let bool_val = (codegen_expr (maps, pred_builder) pred) in

            let merge_bb = L.append_block context "merge" f in
            ignore (L.build_cond_br bool_val body_bb merge_bb pred_builder);
            L.builder_at_end context merge_bb

        (* FOR LOOP *)
        and codegen_for e1 e2 e3 body (maps, llbuilder) =
            codegen_stmt (maps, llbuilder) (A.Block [A.Expr e1; A.While (e2, A.Block
[body; A.Expr e3])])

        and add_terminal llbuilder f =
            match L.block_terminator (L.insertion_block llbuilder) with
              Some _ -> ()
            | None      -> ignore (f llbuilder)

        and codegen_return ret_e (maps, llbuilder) =
            match func_decl.A.typ with
              A.Datatype(A.Void)        -> L.build_ret_void llbuilder
            | _                         -> L.build_ret (codegen_expr (maps,
llbuilder) ret_e) llbuilder

        (* build instructions in the given builder for the statement,
         * return the builder for where the next instruction should be placed *)
        and codegen_stmt (maps, llbuilder) = function
            A.Block sl              -> List.fold_left codegen_stmt (maps, llbuilder)
sl
          | A.Decl e                -> codegen_vdecl e (maps, llbuilder)
```

```
          | A.Expr e                -> ignore (codegen_expr (maps, llbuilder) e);
maps, llbuilder
          | A.Return e              -> ignore (codegen_return e (maps, llbuilder));
maps, llbuilder
          | A.If(p, s1, s2)         -> let builder = (codegen_conditional p s1 s2
(maps, llbuilder)) in maps, builder
          | A.While(p, body)        -> let builder = (codegen_while p body (maps,
llbuilder)) in maps, builder
          | A.For(e1, e2, e3, body) -> codegen_for e1 e2 e3 body (maps, llbuilder)

        (* build the code for each statement in the function *)
        in
        let tuple = codegen_stmt (maps, llbuilder) (A.Block func_decl.A.body) in
        let llbuilder = (snd tuple) in
        add_terminal llbuilder (match func_decl.A.typ with
                A.Datatype(A.Void) -> L.build_ret_void
              | typ -> L.build_ret (L.const_int (ltype_of_typ typ) 0))
    in
    List.iter codegen_func functions;
    the_module
```

8.9 builtin.blr (authored by Tim and Dexter)

```
char intensityToChar(int i){

    char[] map = [' ', '.', '^', ',', ':', ';', 'I', 'l', '!', 'i', '>', '<', '~',
'+', '_', '-', '?', ']', '[', '}', '{', '1', ')', '(', '|', '/', 't', 'f', 'j', 'r',
'x', 'n', 'u', 'v', 'c', 'z', 'X', 'Y', 'U', 'J', 'C', 'L', 'Q', 'O', '0', 'Z', 'm',
'w', 'q', 'p', 'd', 'b', 'k', 'h', 'a', 'o', '*', '#', 'M', 'W', '&', '8', '%', 'B',
'@', '$'];

    int intensity = i % 255;
    int maplen = len(map);
    int factor = intensity * maplen; /* scale factor to map intensity (0-255) to char
(0-len(map) */
    int index = factor / 255;
    return map[maplen - index];
}

int charToIntensity(char c){

    char[] map =
['$','@','B','%','8','&','W','M','#','*','o','a','h','k','b','d','p','q','w','m','Z',
'0','O','Q','L','C','J','U','Y','X','z','c','v','u','n','x','r','j','f','t','/','|','
(',')','1','{','}','[',']','?','-','_','+','~','<','>','i','!','l','I',';',':',',','^
','.',' '];

    int maplen = len(map);
    int idx = 0;
    int i;
```

```
    int intensity;

    for(i = 0; i < maplen; i = i + 1) {
        if(map[i] == c) {
            idx = i;
        }
    }

    intensity = (255 * idx) / maplen;
    return 255 - intensity;
}

/* offset = 0 for darken, offset = 1 for lighten */
char adjustPX(char c, int offset) {

    char[] map =
['$','@','B','%','8','&','W','M','#','*','o','a','h','k','b','d','p','q','w','m','Z',
'0','O','Q','L','C','J','U','Y','X','z','c','v','u','n','x','r','j','f','t','/','|','
(',')','1','{','}','[',']','?','-','_','+','~','<','>','i','!','l','I',';',':',',','^
','.',' '];

    int maplen = len(map);
    int idx = 0;
    int i;
    for(i = 0; i < maplen; i = i + 1) {
        if(map[i] == c) {
            idx = i;
        }
    }
    offset = offset - 1;
    return map[idx + offset];
}
```

8.10 stdlib.blr (authored by Tim and Dexter)

```
void display(char[][] cv) {
     int width = len(cv);
     int height = len(cv[0]);
     for(i = 0; i < width; i = i + 1){
       for( j = 0; j < height; j = j + 1){
            print( cv[i][j] );
       }
       println(" ");
    }
}

int[][] edgeDetect(string x, int edgeDist){
    int[][] image = readGrayscaleImage(x);
    int width = len(image);
    int height = len(image[0]);
```

```
    int leftPixel = -1;
    int rightPixel = -1;
    int bottomPixel = -1;

    int distance = -1;
    int black = 0;

    int row;
    int col;
    for(row=0; row<width; row=row+1){
       for(col=0; col<height; col=col+1){
           black = 0;
           leftPixel = image[row][col];

           if (col < height-1){
              rightPixel = image[row][(col+1)];
               distance = pixelDistance( leftPixel, rightPixel );
              if(distance > edgeDist){
                  black = 1;
              }
           }

           if(row < width-1){
              bottomPixel = image[(row+1)][col];
               distance = pixelDistance( leftPixel, bottomPixel );
              if (distance > edgeDist){
                  black = 1;
              }
           }

           if( black == 1 ){
              image[row][col] = 1;
           }else{
               image[row][col] = 0;
           }

       }
    }

    for(row=0; row<width; row=row+1){
       for(col=0; col<height; col=col+1){
            print( image[row][col]  );
       }
       println("");
    }

    return image;
}


int pixelDistance( int x, int y ){
    int distance;
```

```
    if( x > y ){
       distance = x - y;
    }else{
       distance = y - x;
    }
    return distance;
}

char[][] dither(string imageFile) {
        int[][] a = readGrayscaleImage(imageFile);
        char[][] b = canvas(imageFile);
        int width = len(a);
        int height = len(a[0]);
        char c = intensityToChar(255);
        println(c);
        println(width);
        println(height);
        int i;
        int j;
        char px;
        for(i = 0; i < width; i = i + 1) {
                for(j = 0; j < height; j = j + 1) {
                        px = intensityToChar(a[i][j]);
                    b[i][j] = px;
                }
        }
       return b;
}

char[][] impose(char[][] asciiArt, int[][] edges, char edgeChar){
    int width = len(asciiArt);
    int height = len(asciiArt[0]);
    int i;
    int j;
    for(i = 0; i < width; i = i + 1){
       for( j = 0; j < height; j = j + 1){
           if( edges[i][j] == 1 ){
              asciiArt[i][j] = edgeChar;
              print( edgeChar );
           }
           else{
              print( asciiArt[i][j] );
           }
       }
       println(" ");
    }
    return asciiArt;
}
```

8.11 bindings.h (authored by Dexter)

```
#ifndef BINDINGS_H
#define BINDINGS_H

void initGL(int w, int h);

int LoadImage(char *filename);

struct imgData {
    int width;
    int height;
    int sad;
    int *data;
};

int foo(int x);

int *getArr();

struct imgData getImg();

ILubyte * getImageData(char *filename);

void initializeGlDevIL(char *filename);

struct ImageStruct readColorImage(char *filename);

int* readDimensions(char *filename);

struct ImageStruct readGrayscaleImage(char *filename);

struct CanvasStruct canvas(char *filename, char* option);

#endif
```

8.12 bindings.c (authored by Dexter)

```
#include <stdio.h>
#include <string.h>
#include <GL/glut.h>
#include <IL/il.h>


#define DEFAULT_WIDTH  640
#define DEFAULT_HEIGHT 480

int glutInitialized = 0;    // Ensure glutInit() is not called twice
```

```
                            // the only function that calls it is readDimensions()

/* Handler for window-repaint event. Called back when the window first appears and
   whenever the window needs to be re-painted. */
void display()
{
    // Clear color and depth buffers
       glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
       glMatrixMode(GL_MODELVIEW);     // Operate on model-view matrix

    /* Draw a quad */
       glBegin(GL_QUADS);
           glTexCoord2i(0, 0); glVertex2i(0,   0);
           glTexCoord2i(0, 1); glVertex2i(0,   DEFAULT_HEIGHT);
           glTexCoord2i(1, 1); glVertex2i(DEFAULT_WIDTH, DEFAULT_HEIGHT);
           glTexCoord2i(1, 0); glVertex2i(DEFAULT_WIDTH, 0);
       glEnd();

    glutSwapBuffers();
}

/* Initialize OpenGL Graphics */
void initGL(int w, int h)
{
     glViewport(0, 0, w, h); // use a screen size of WIDTH x HEIGHT
     glEnable(GL_TEXTURE_2D);     // Enable 2D texturing

    glMatrixMode(GL_PROJECTION);     // Make a simple 2D projection on the entire
window
     glLoadIdentity();
     glOrtho(0.0, w, h, 0.0, 0.0, 100.0);

     glMatrixMode(GL_MODELVIEW);    // Set the matrix mode to object modeling

     glClearColor(0.0f, 0.0f, 0.0f, 0.0f);
     glClearDepth(0.0f);
     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT); // Clear the window
}

/* Load an image using DevIL and return the devIL handle (-1 if failure) */
int LoadImage(char *filename)
{
    ILboolean success;
     ILuint image;

    ilGenImages(1, &image); /* Generation of one image name */
     ilBindImage(image); /* Binding of image name */
     success = ilLoadImage(filename); /* Loading of the image filename by DevIL */

    if (success) /* If no error occured: */
    {
        /* Convert every colour component into unsigned byte. If your image contains
alpha channel you can replace IL_RGB with IL_RGBA */
```

```
            success = ilConvertImage(IL_RGBA, IL_UNSIGNED_BYTE);

        if (!success)
            {
                  return -1;
            }
    }
    else
        return -1;

    return image;
}

ILubyte * getImageData(char *filename)
{
    ILboolean success;
     ILuint image;

    ilGenImages(1, &image); /* Generation of one image name */
     ilBindImage(image); /* Binding of image name */
     success = ilLoadImage(filename); /* Loading of the image filename by DevIL */

    if (success) /* If no error occured: */
    {
        /* Convert every colour component into unsigned byte. If your image contains
alpha channel you can replace IL_RGB with IL_RGBA */
            success = ilConvertImage(IL_RGBA, IL_UNSIGNED_BYTE);

        if (!success){ "IMAGE FAILED TO BE SUCCESFULLY READ!"; }
    }

    ILubyte * bytes = ilGetData();
    return bytes;
}

void initializeGlDevIL(char *filename){

  // GLUT init if not already initialized
  if( glutInitialized == 0){

    int *num_files_to_read = (int *) malloc(sizeof(int));
    *num_files_to_read = 1;

    glutInit(num_files_to_read, &filename);            // Initialize GLUT
    glutInitDisplayMode(GLUT_DOUBLE); // Enable double buffered mode
    /*
    glutInitWindowSize(DEFAULT_WIDTH, DEFAULT_HEIGHT);   // Set the window's initial
width & height
    glutCreateWindow(filename);      // Create window with the name of the executable
    glutDisplayFunc(display);       // Register callback handler for window re-paint
event
    glutReshapeFunc(reshape);       // Register callback handler for window re-size
event
```

```
    */

    /* OpenGL 2D generic init */
    initGL(DEFAULT_WIDTH, DEFAULT_HEIGHT);

    // Initialization of DevIL
     if (ilGetInteger(IL_VERSION_NUM) < IL_VERSION)
     {
           printf("wrong DevIL version \n");
           exit(0);
     }
     ilInit();
  }

  glutInitialized = 1;
}

int* readDimensions(char *filename){

  // Initialize GL and DevIL
  initializeGlDevIL(filename);

  // Load the image into DevIL
  int image;
  image = LoadImage(filename);
  if ( image == -1 ){
    printf("Can't load picture file %s by DevIL \n", filename);
  }

  // Get a data pointer to the image
  ILubyte * bytes = getImageData(filename);

  // Get the dimensions of the image
  ILuint width, height;
  width = ilGetInteger(IL_IMAGE_WIDTH);
  height = ilGetInteger(IL_IMAGE_HEIGHT);

  int *dimensions = (int*) malloc(sizeof(int) *2);
  dimensions[0] = width;
  dimensions[1] = height;
  return dimensions;
}

struct ImageStruct{
  int width;
  int height;
  int depth;
  int *imageData;
};

struct CanvasStruct{
  int width;
  int height;
```

```
  int depth;
  char *asciiData;
};

struct ImageStruct readColorImage(char *filename){

    // Get the dimensions of the image
  int* dimensions = readDimensions(filename);
  struct ImageStruct is;
  is.width = dimensions[0];
  is.height = dimensions[1];
  is.depth = 3;     // three color values (R,G,B)

  // Load the image into DevIL
  int image;
  image = LoadImage(filename);
  if ( image == -1 ){
    printf("Can't load picture file %s by DevIL \n", filename);
  }

  // Get a data pointer to the image
  ILubyte * bytes = getImageData(filename);

  int *colorImage = (int *) malloc(is.width * is.height * 3 * sizeof(int)); // 3 for
rgb values

  for(int i = 0; i < is.height; i++){
    for(int j = 0; j < is.width; j++){
      colorImage[(i*is.width+j)*3 + 0] = bytes[(i*is.width +j)*4 + 0];
      colorImage[(i*is.width+j)*3 + 1] = bytes[(i*is.width +j)*4 + 1];
      colorImage[(i*is.width+j)*3 + 2] = bytes[(i*is.width +j)*4 + 2];
    }
  }

  is.imageData = colorImage;
  return is;
}

struct ImageStruct readGrayscaleImage(char* filename){

  struct ImageStruct colorImageStruct = readColorImage(filename);
  int width = colorImageStruct.width;
  int height = colorImageStruct.height;
  int* colorImage = colorImageStruct.imageData;

  int* grayImage = malloc(width * height * sizeof(int));
  for(int i = 0; i < height; i++){
    for(int j = 0; j < width; j++){
      grayImage[(i*width)+j] = (colorImage[(i*width +j)*3 + 0] * .33) +
                               (colorImage[(i*width +j)*3 + 1] * .33) +
                               (colorImage[(i*width +j)*3 + 2] * .34);
    }
  }
```

```
  struct ImageStruct is;
  is.width = width;
  is.height = height;
  is.depth = 1;            // gray scale image. 1 intensity value
  is.imageData = grayImage;
  return is;
}

struct CanvasStruct canvas(char *filename){

  struct ImageStruct image = readGrayscaleImage(filename);
  struct CanvasStruct canvas;
  canvas.width = image.width;
  canvas.height = image.height;
  canvas.depth = 1;
  char *characters = (char *) malloc( sizeof(char) * canvas.width * canvas.height );
  canvas.asciiData = characters;
  return canvas;

}
```

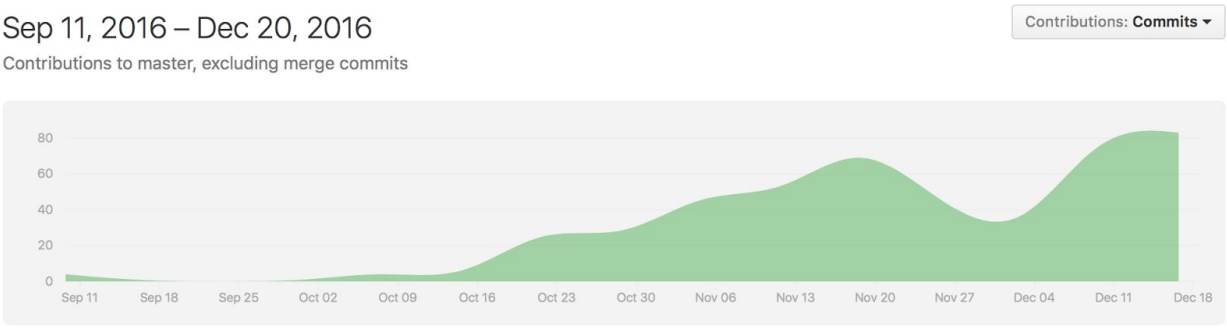
8.13 Makefile (backend libraries) (authored by Tim and Dexter)

```
clib:
	gcc -c bindings.c -lGL -lglut -lGLU -lIL
	ar -cvq libclib.a bindings.o builtins.o

stdlib:
	gcc -c bindings.c -lGL -lglut -lGLU -lIL
	ar -cvq libclib.a bindings.o builtin.o stdlib.o

exec:
	gcc bindings.c -lGL -lglut -lGLU -lIL -o test

.PHONY: clean
clean:
	rm -f bindings.o
	rm libclib.a

.PHONY: cleantest
cleantest:
	rm -f bindings.o stdlib.o
	rm libclib.a test
```

9. Project Log

9.1 Graphs and Stats

Commit graph of master branch since September 2016:



This graph does not show commits made on all other branches, where significant portions of our project was developed. As mentioned in the project plan, each major implementation of Blur was developed on a separate branch to be merged upon completion.

9.2 Git Commit History

master branch:

```
commit 0edcf427de36506908e2d64fd068b8fcd1411b95
Author: restocle <sh3266@columbia.edu>
    removed discard
commit 3f994e138296e79bd1775f0426db0dc3dba391f2
Author: restocle <sh3266@columbia.edu>
    arr2D char test fix
commit b266e50863d060c34aeea0c7096d6e95e5ea84d8
Author: restocle <sh3266@columbia.edu>
    tests
commit b468e2f7906c40af1f16b50951dc94c523da4109
Author: Tim Goodwin <tlg2132@columbia.edu>
    last
commit 506b2d6980a76fb733ac39692c2a233344507f6e
Author: restocle <sh3266@columbia.edu>
    abs tests
commit 4298abbd6b355355a824cce6f97815692fcb3710
Merge: 365884a 57d1f45
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 365884aa74af0de09997aca8103a9d62768cd98d
Author: restocle <sh3266@columbia.edu>
```

```
    arr2D + discards
commit 57d1f453d8ee7fd0bf54319c0cefefb3a114d1e4
Merge: f631e46 548959a
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit f631e46bddfba0a89635e6de37ff76bb914a4316
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing abs char.
commit 548959a00f1220cd6a7b6208b2ab39b2aeb35641
Author: Tim Goodwin <tlg2132@columbia.edu>
    new tests
commit 98c3a25e502ee74c61aedb93a4a161d5a0543694
Merge: 35dfeb8 49fdec9
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 35dfeb814d3157229340b417f4f22e7d693595fe
Author: restocle <sh3266@columbia.edu>
    arr2D test fix
commit 49fdec94d32b00cdd5304ae67541920e845b82ee
Merge: 2ce3601 9c94414
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 2ce360177de440a401fbbdaa2ee4dcdcff797418
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing output for comp or.
commit 5c450fdaed109f5cf2eba2509d1fac6ea07dfb23
Author: restocle <sh3266@columbia.edu>
    test-mag
commit 9c944141668438cbbce1d58c28cf854cd3a3a893
Author: restocle <sh3266@columbia.edu>
    tests
commit 9ee2738915ab2258d5ec26b41e6a4154b1abfe16
Author: melissaKG <mkaufmangomez@gmail.com>
    Not and negation.
commit 3cd1acf21e5bf655cffd8239aa1442bdf8cee389
Author: melissaKG <mkaufmangomez@gmail.com>
    Main may not take args.
commit 1439d1139645ea9cdcacf50d780ac4465cc07321
Author: melissaKG <mkaufmangomez@gmail.com>
    Rolling back last change to fix tests.
commit b3a7fb8f232b68445f2b0954bb2fa20b69449ed0
Author: tim goodwin <tlg2132@columbia.edu>
    fixed test-arr-nonasn-get. catching illegal assignment of SizedArray types.
commit 098c302d46ab490d507970d46aefbe236f1ee7ac
Author: tim goodwin <tlg2132@columbia.edu>
    fixed test-arr2D-int.blr
commit 038101ce6c52b065c7f796c7f868cbfa543e1d4e
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding more while tests.
commit e86fb54111a5918b2c56fa29d80187f92beddafc
Author: melissaKG <mkaufmangomez@gmail.com>
    Deleting subarray constr tests.
commit b273cf73827267cb993ccd602ecb0b7548dd574e
```

```
Author: tim goodwin <tlg2132@columbia.edu>
    checking arrayListInit
commit 92a04c7e0e2819d9e1eb41b50c05b9e528a0ad4f
Author: melissaKG <mkaufmangomez@gmail.com>
    Removing unnecessary tests.
commit 10f453057cf40079d0a8427763741c25c4d4efaf
Author: tim goodwin <tlg2132@columbia.edu>
    checking array access dimension stuff
commit 502eaea0baf73025482a7d6ef84b42b5b32108ea
Author: tim goodwin <tlg2132@columbia.edu>
    major clean
commit 5203be792531b240910e7a32e653b61138132d2c
Author: tim goodwin <tlg2132@columbia.edu>
    idk
commit 1e6fca1076f2b312a7fd33e0dd8269e7bf6e55aa
Author: melissaKG <mkaufmangomez@gmail.com>
    More for tests.
commit 42e8a6a447a32d0ee0b9fc33bc67302208e8ec02
Author: melissaKG <mkaufmangomez@gmail.com>
    More tests for ifs.
commit f5c4524e232b1e4ec6a7f25bb57217662702173a
Author: melissaKG <mkaufmangomez@gmail.com>
    More checks for ifs.
commit e9a8b31720daa5353b84038e7db2140cacc3b75a
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding mod.
commit a2e40c2201b4deda6a7e085cb39f9980fee97ad6
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding mag.
commit 3769de4223b89a4ac7788fa793dcd0b36a172b4b
Merge: d82141c 7abb222
Author: melissaKG <mkaufmangomez@gmail.com>
    merging.
commit d82141c455c3e498e2b67ad649d6f0ce293d69ca
Author: melissaKG <mkaufmangomez@gmail.com>
    Final blur.ml.
commit 7abb222a921729e20e40ffc095a21683ebfb47b4
Author: dextercallender <dec2148@columbia.edu>
    demo
commit 64aa13ad8b8425b83067b2dcb29a54b864549331
Merge: 89d7ea3 b6f3b26
Author: dextercallender <dec2148@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 89d7ea3b545df9fed1d4cd51ee268981f372235a
Author: dextercallender <dec2148@columbia.edu>
    added demo-files
commit b6f3b263c45bf40313c9af4bc59be596137d00c8
Author: restocle <sh3266@columbia.edu>
    dither test fix
commit 84c97d1c8265087bd776f00d8a40f6ca8222a2aa
Merge: 1b356f4 e694b44
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
```

```
commit 1b356f42be96db5fdaaeb775f24aea4f06e6f5ae
Author: restocle <sh3266@columbia.edu>
    tests
commit e694b44f0d53ee115b2efc6b008b9eca85f9833b
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing bug.
commit 44d37b723d59e185c576f36ab7700315e7a30228
Author: melissaKG <mkaufmangomez@gmail.com>
    Every prog must have main func.
commit 25af21a06005ea26135270af6d272ebeb96c3ffb
Author: melissaKG <mkaufmangomez@gmail.com>
    Checking for duplicate funcs.
commit cc35e382617ed157eda97d9e2423665bbc44a42a
Author: restocle <sh3266@columbia.edu>
    PP example
commit a70cb5dfca8a9267d1190ff079a08aca26e075fe
Merge: 0686403 7f9afc9
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 06864033672610300 61cfc16bfcdf47a6b6d3c8d
Author: restocle <sh3266@columbia.edu>
    ppt output files
commit 7f9afc96b415645b7846c7b75ae06cbb70417f5d
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing print.
commit 282ae3008ab0d607227f65bf40e3dbfc8d94e00e
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing bad push.
commit da51429a7d42d196f5a2eb91cd5729af7b00b102
Merge: e8d3460 d50417d
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit e8d34606dc5bbfbb2a6b164b33bdf7c6ca6dae2e
Author: melissaKG <mkaufmangomez@gmail.com>
    tried to do print void check.
commit d50417de25cb56e477616b0dd5d264383501f2d8
Author: restocle <sh3266@columbia.edu>
    tests
commit b4fd3cd5e80cf4b189b6ea5cdab8ca315b54d10e
Author: restocle <sh3266@columbia.edu>
    tests
commit 04d6a261bb7b043b7f4a9689da592a520c0313b3
Author: restocle <sh3266@columbia.edu>
    readGrayImg.blr
commit 44170175f06d2250d32233849731897cb1a8664e
Author: restocle <sh3266@columbia.edu>
    changed void main to int main
commit 3160245fef2f61f3c6ecd06d6e5b4ef7c5f35e5a
Author: restocle <sh3266@columbia.edu>
    tests
commit c5dcd413f3c23c7c9a7114ec887d3b9c51f6cc07
Merge: 2e19ac9 5410b44
Author: melissaKG <mkaufmangomez@gmail.com>
```

```
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 2e19ac993845acdd677abba476a37ba2e3dd3afa
Author: melissaKG <mkaufmangomez@gmail.com>
    Cannot print void.
commit 5410b4414f96b1153b11406372c86e0e4314e046
Author: restocle <sh3266@columbia.edu>
    removed return void tests
commit 3dda65e0c2d42548f53456667cad5f881cbebfd3
Merge: 69d9154 9544ea2
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 69d915400e04e088aa662dc2040db41ac521fdbe
Author: restocle <sh3266@columbia.edu>
    testscript_cp update
commit 9544ea2de344bdf336491383e12866b987117402
Author: melissaKG <mkaufmangomez@gmail.com>
    Changing test so that functions can only be called if they have previously been
declared.
commit 5f2cd10e3f455e154a106728cc505118d79de69b
Merge: 3ec5d80 575bc74
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 575bc7491e28cd07ee8ab3e60e71a95c551829c4
Merge: cad109d 730c9b8
Author: dextercallender <dec2148@columbia.edu>
    fixed merge conflict
commit cad109da675a48b28701655569e3b1cce3de8829
Author: dextercallender <dec2148@columbia.edu>
    added a demo pattern
commit 730c9b8c778040dff3a603d38faee8eb126238d9
Author: restocle <sh3266@columbia.edu>
    test-funcCall
commit 3ec5d80e5e26b2405c42b4123170586179eaf482
Merge: 8485d9b 8c836c0
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 8485d9b1ae0720e202dfa6e12998d64b5ae244b9
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing tests.
commit 8c836c02f0ae8963570511b08466dbabe097adc0
Author: restocle <sh3266@columbia.edu>
    testscript2 update
commit fe84a060158cff1ba563da619371542263c69f6e
Author: tim goodwin <tlg2132@columbia.edu>
    temp
commit da40dcdba7623f4ea98ac5186f27c6c7fa20d401
Author: tim goodwin <tlg2132@columbia.edu>
    ok
commit 7ebb98eeb8c2b38194b246555d12e12aa4f552b9
Author: tim goodwin <tlg2132@columbia.edu>
    returned support to sized arr
commit 42ab0918be3dc55506ed7eabbb9557457bbbf8a9
Merge: 7ec1db4 e5c6b39
```

```
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 7ec1db4c1c837daaa553e60d733e3c9a118630fb
Author: restocle <sh3266@columbia.edu>
    makefile, blur, testscript update for testing
commit e5c6b39c86b91b3051dcf731fd96620d379f8561
Merge: 979e466 f084404
Author: dextercallender <dec2148@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 979e466e934798ec4dc3c6127a52f0cd3bf81188
Author: dextercallender <dec2148@columbia.edu>
    added demo/ directory
commit ce10ae7a8c8d6150964e35d855a5a2e3383646e8
Author: dextercallender <dec2148@columbia.edu>
    modified demo program
commit f084404bbb08fdfe6da517db5c2b2b0882229f19
Author: tim goodwin <tlg2132@columbia.edu>
    testscript2.sh
commit fb9d6ea9416054c4117c257a2297e80e3c91fdad
Author: restocle <sh3266@columbia.edu>
    testscripts
commit fb879cfdee49714d60823382d332ec4438b13a40
Author: restocle <sh3266@columbia.edu>
    testscript update
commit 0c199c317dfdc7384dcf6f1863f1f5c303b83192
Author: restocle <sh3266@columbia.edu>
    abs char test
commit bd5916e6f74ab2c88473d47c738a70aaade2aee8
Author: restocle <sh3266@columbia.edu>
    dither test path fix
commit 5847c7c0cc8b72fcdc826b88b059c427f81b84f1
Merge: 6d7a981 e7704ac
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 6d7a9818dff6dc5f8a663d1898db9018a04aef07
Author: restocle <sh3266@columbia.edu>
    makefile change for path
commit e7704ac063202f41da7e9d8f89d35430581cbe83
Merge: 1df61dc d3fcddc
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 1df61dc6718dc490df5ae258284f5bf167f87244
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing print statement and test to move main to end.
commit d3fcddc6de798c199cb9125ef65895922772c29f
Merge: a422f6b d187805
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit a422f6b3b163bf7ec1254f1ca6a44322c9a9a19e
Author: restocle <sh3266@columbia.edu>
    updated NEW LIB INCLUDING! testscript
commit d187805a797a731609ca236813248ade05b96fc9
Merge: e72c9c2 7609a61
```

```
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit e72c9c28e2355061d6188daa9871ed2a73d740ee
Author: melissaKG <mkaufmangomez@gmail.com>
    Reversing function list so that main() must be at the bottom.
commit 72b62d89be58594c0be4637b422f528c7ff36f31
Author: restocle <sh3266@columbia.edu>
    saving old testscript
commit 7609a611249d2a897a8535cf666492886b6bc019
Author: Tim Goodwin <tlg2132@columbia.edu>
    reverse builtin list
commit 73e2973f815d173478677f276b7d07d0983bcac5
Merge: fef9a12 d3dea6b
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit fef9a12650601da49fb8bd52aa98d59a6b9ada57
Author: melissaKG <mkaufmangomez@gmail.com>
    Semantic analysis working on demo.
commit d3dea6b98d2d738b78588b0ad8fd5858a8140cca
Author: tim goodwin <tlg2132@columbia.edu>
    bool test
commit 4eb10363ff8c4edf2859259c9885da50e6da350d
Author: melissaKG <mkaufmangomez@gmail.com>
    Improvement to env for the sake of mulitple functions and return types.
commit ed558af3df4999fc4fdcae5e1e2237928144088c
Author: tim goodwin <tlg2132@columbia.edu>
    generator cleanup unused stuff
commit cf014a5ae05172d40ca222f7a1393d3de2bf7645
Author: tim goodwin <tlg2132@columbia.edu>
    take out darken adn lighten
commit ef92965c054948d025b86558010e59b8bd91619e
Author: Tim Goodwin <tlg2132@columbia.edu>
    for headers
commit e027f5e2918e06c3eb8e4f2cb9c940285e6a0dc9
Author: Tim Goodwin <tlg2132@columbia.edu>
    reflecting frontend updates
commit fb000c2dac06820a6f261a2bced837c7dd669ff2
Author: tim goodwin <tlg2132@columbia.edu>
    makefile update
commit 102ba019d73cdfbf94a6fbf3448d0d4a8edba260
Author: tim goodwin <tlg2132@columbia.edu>
    last change to front end i promise
commit 3035101c863f761892f6bfcfbec21d377783d265
Author: tim goodwin <tlg2132@columbia.edu>
    new builtin.o sry we have to track this its complicated
commit c87844f3d3bbe57a3e6abeb87889fd01e45ae603
Author: tim goodwin <tlg2132@columbia.edu>
    normal array literals work again, pass them by value as much as u want
commit 6188dbc116f8c2bf47d6b139ffe4fae38c085d2a
Author: melissaKG <mkaufmangomez@gmail.com>
    This much is working. Going to slowly add more until all of test-getImg works.
commit 424538ea7ec0b8bfaed2f374ca456489794af37c
Merge: c2a7454 19dc5db
```

```
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit c2a74543f0b89e6bbe7346f95e349795e4e17fcb
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing define println test.
commit f91707a77df17e752d1156059981646ce94363a1
Author: melissaKG <mkaufmangomez@gmail.com>
    Duplicate var arr check.
commit 19dc5dbb76f337e83f2a3f5bab2ff712c326ea15
Author: tim goodwin <tlg2132@columbia.edu>
    dont overwrite
commit e6b7c609420bca8ea3a45f2b9a259fb6c9827a74
Author: tim goodwin <tlg2132@columbia.edu>
    conditional stdlib handling
commit 8429e34fbcb433de117e8823b6a3e69c9d5e21b9
Author: restocle <sh3266@columbia.edu>
    incomplete master log
commit 59c3740e0ccf39a01d34ebd82be7a217da89cdea
Merge: 4ab0ee2 50a13d0
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 4ab0ee28cdc7c9c2283a4f4c2543bb46e6e06eac
Author: restocle <sh3266@columbia.edu>
    log_master_cp removed empty lines
commit 50a13d05c1b9fc8b18a0f561f28ccc31991c4f3b
Author: dextercallender <dec2148@columbia.edu>
    stdlib.o in clib. demo_dither.blr created
commit df8495c00e9456efd68f61de08b4761d93285348
Author: restocle <sh3266@columbia.edu>
    log master_cp short version
commit 6a3e6709676212fb0643e0a35dedd4e675e214d4
Author: restocle <sh3266@columbia.edu>
    git logs
commit b81038a14c6237a8875aab7cba5a58f886979005
Merge: b79f67d d9214b2
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit b79f67df1a9d7d40946115889371e79eea75947c
Author: restocle <sh3266@columbia.edu>
    demo fixes
commit 9c8e8824a5f1b790a45f0ea5c281cd3dbe77ef2e
Author: restocle <sh3266@columbia.edu>
    dither, edge tests
commit d9214b29c8230efb4b2023cf4121774c427fe49a
Author: tim goodwin <tlg2132@columbia.edu>
    idk
commit 86afcf1a500c8575a04f37a9c0f1a0f1976c4c18
Author: restocle <sh3266@columbia.edu>
    sample single code fail output
commit a07bf8cb2407e22ae4ed444c52fa41f4907f37c2
Author: restocle <sh3266@columbia.edu>
    sample single code check output
commit db9b1784fce5a5bc688ace850ac3ae03bf2fb086
```

```
Merge: 2db77b0 ab75782
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 2db77b0b77b6ec2c815723c0ba036dc130916257
Author: restocle <sh3266@columbia.edu>
    not operator
commit ab7578218f508eabc555568f1daed74e00532a1b
Author: dextercallender <dec2148@columbia.edu>
    ldkajf
commit d70c7e0812d3a1dc3e48fc639be5a39ec2b8e6ac
Author: tim goodwin <tlg2132@columbia.edu>
    demo
commit e0de6ac972650258ac1bf13332762cea62731b2c
Author: tim goodwin <tlg2132@columbia.edu>
    handling canvas
commit 54c02c5404ebcfe56ed1aa309a1b45a5b7648d06
Merge: 52126b0 6cdcf43
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 52126b0ebfbd5d250eea8b62cbe4d69c2900ffbe
Author: restocle <sh3266@columbia.edu>
    arr 1D 2D init fix
commit 6cdcf439ecbdd726b1f3f51099847d8d9cba77e6
Merge: 5dd893e fd552d1
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 5dd893e9c0f3661bceaacf3cb50ca9707d50b006
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing return types. Only int used to work before lol but now all function return
types should work.
commit fd552d1a74af3027617fecb62140571931dc6dc3
Author: dextercallender <dec2148@columbia.edu>
    impose function implemented. impose-test.blr created. demo.blr almost done
commit fdbff48c74d569c279ce3f57a7d195952d939b72
Merge: 75ff93f 7b8e1c2
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 75ff93f7e4dd0a7031fb8978ac85267f7eed2d56
Author: restocle <sh3266@columbia.edu>
    negation test
commit 7b8e1c21ebf3dbd9797d59ccd003e2b287cb4247
Merge: 15ebb22 abace7f
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 15ebb22bff2502c325b0facd64024395c948b668
Author: melissaKG <mkaufmangomez@gmail.com>
    Checking and assiging return types.
commit abace7f3cb21c02a5ea53c5bea896cf3b299d824
Author: dextercallender <dec2148@columbia.edu>
    added more to demo.blr
commit 073f71948fc57ee68db4376e5cb2660592e9ecdb
Author: melissaKG <mkaufmangomez@gmail.com>
    Pattern matching is the answer to checking the type of arr lit. Pattern matching
```

```
is the answer to most things.
commit 22a5973c72861cfa1955fa62d50b23cdf3facd97
Author: restocle <sh3266@columbia.edu>
    test-rec-fib
commit 1e17d8c3e78efd77cc14c8485a1df3b6183d7cc7
Author: tim goodwin <tlg2132@columbia.edu>
    nice, fun example for demo test-getImg.blr
commit a78128a00f06b155130610517865356f6feba904
Merge: 4e0279b 41ddae7
Author: dextercallender <dec2148@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 41ddae76418f8c104fa0db0463985531b546c123
Author: tim goodwin <tlg2132@columbia.edu>
    passing structs all over
commit 5f7b9c96b14229c076c6b515ce7a40b6718a8643
Author: tim goodwin <tlg2132@columbia.edu>
    runtime segfault aka impossible bug
commit fe0ff6f8a97770de38c08d57cf730dceaf677acb
Author: tim goodwin <tlg2132@columbia.edu>
    blur literal arrays casting to int pointers, working for 1D
commit 9303f50fbd55004111f3e3aba8d6be1333ba69d9
Author: tim goodwin <tlg2132@columbia.edu>
    casting C struct pointer to int pointer
commit 02c9d93e9e79a37855ed469cbfb43b7150224b48
Author: tim goodwin <tlg2132@columbia.edu>
    c struct stored in map as pointers, works
commit 18e2c0f94fdbd59e9f6fce87c5e47be653875610
Author: tim goodwin <tlg2132@columbia.edu>
    mag operator working
commit 4e0279b45cfd1b24164702b821792cb907e357de
Merge: d37551c 451e45d
Author: dextercallender <dec2148@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit d37551ca0071ae123d4695ae93c97fc5f9296487
Author: dextercallender <dec2148@columbia.edu>
    started demo.blr
commit 54b91d6a5f16ddec29f8caba425811fa0b75aefd
Author: restocle <sh3266@columbia.edu>
    canvas test
commit 451e45d74223cda02b8a1629d1d82131af50dbbb
Author: restocle <sh3266@columbia.edu>
    blur.ml semantic
commit 339f96bde5060e0b78b05e417dfd209f51000d72
Author: restocle <sh3266@columbia.edu>
    tests
commit 99d82a131b8f8a4b3d178ee9bad051f39ff3a34e
Merge: 08e728d c52c3c2
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 08e728db454c7d193e6ad573f942bf8d937908c4
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding pattern matching when checking vardecl for arrays.
commit c52c3c24712711c2a9350887a01c2257ed9b443f
```

```
Merge: 7086ee3 2e915aa
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 2e915aa555a0f5ae45814a1c6119e046800a1db4
Author: melissaKG <mkaufmangomez@gmail.com>
    Ensuring that unsized arr is immediately initialized.
commit 7086ee3e3714cdd78feb0b47ebadb83fbd738d38
Author: restocle <sh3266@columbia.edu>
    conditionals, abs-op tests
commit d7efc07fe545bc4618f96701d4a0163b0d5102f4
Author: melissaKG <mkaufmangomez@gmail.com>
    Unsized arr decl type mismatch works.
commit dbd95f9643936aee93b6154f8b12116c99656146
Author: melissaKG <mkaufmangomez@gmail.com>
    Array type mismatch works.
commit b148e273fe29222444e439478a6a3a70206e7b3c
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing expected output for comp tests.
commit c3d811cecac1bc9bdf100a599f7f728322d8f936
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing funcCall-args test expected output.
commit d75b90172cb58ca7d3d815e195e9f53e6d945dda
Author: melissaKG <mkaufmangomez@gmail.com>
    Type mismatch is not thrown if variable is declared but not initialized.
commit c8342bcfd6750b87ee2d7b4d97f86462188efbe5
Author: melissaKG <mkaufmangomez@gmail.com>
    Can only do arithmetic on ints and doubles.
commit e4961609a1bda574910d1fa91153750cf4a657bd
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing conditional checks.
commit 3e18cb67e079f58316ed78a6217fc2b6a1cfa932
Merge: 851c7c5 c42fdfd
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 851c7c557cdb084d2ba7d7f8ac12c49649ac1ef6
Author: melissaKG <mkaufmangomez@gmail.com>
    Commenting code.
commit 9870a39fe63113e7e8d0deba7a8864d65bffecf6
Author: melissaKG <mkaufmangomez@gmail.com>
    Checking that decl init matches decl type.
commit c42fdfdee4a756faba802c4b0b31b783bf70c46b
Author: restocle <sh3266@columbia.edu>
    op fails - eq neq and or
commit 8b5ae074f365835b96cd3689019d49bd481d7742
Merge: 231e86b a9a2092
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 231e86b309fc9688161fecd0d154ffd86762e09a
Author: melissaKG <mkaufmangomez@gmail.com>
    checking eq neq and or.
commit a9a209261422ba6b05f1bdb4880deeb654d6e469
Merge: 2240558 8c69190
Author: dextercallender <dec2148@columbia.edu>
```

```
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 22405580c5712aedf1904f8f024e4f030b5bb6f3
Author: dextercallender <dec2148@columbia.edu>
    modified clib makefile
commit 8c69190264c7a449407bc721b610bf60641d91af
Author: melissaKG <mkaufmangomez@gmail.com>
    Remove add-int-float. We're not doing casts.
commit a4bd6ba99cd8f73eff064746c5522084c07e1b39
Merge: d4dc2e1 db5c437
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit db5c4372b75b4c4e4f47db6049d6d350dae603c1
Author: tim goodwin <tlg2132@columbia.edu>
    builtin.o backup
commit d4dc2e107982cb746cc19c0f8e9da684ebb12982
Author: melissaKG <mkaufmangomez@gmail.com>
    Updating semantic analysis.
commit 4da9fdf1799001471f69e6d117902a8dff3fd3e3
Author: tim goodwin <tlg2132@columbia.edu>
    new linking builtin lib
commit 5d4f7f88a006a58610f5dbef312be0f8576115e5
Author: tim goodwin <tlg2132@columbia.edu>
    ok
commit 296cb6e6b3c67af84a39f6950d14de2f6bf1fa8a
Merge: 6f4722f ba5ed8c
Author: melissaKG <mkaufmangomez@gmail.com>
    mergin.
commit 6f4722feb2ffe917a58b6ce5f9fa65c0fe700130
Author: melissaKG <mkaufmangomez@gmail.com>
    updating semantic analyzer in master.
commit ba5ed8c79cebba2d93d68d1cd6d57fe15f3e2c5f
Author: restocle <sh3266@columbia.edu>
    semantic & test update
commit 08058c1f29268f3766b3e686155b0a8e644822a7
Merge: 8be670f 5956687
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 595668795cf4e6e7dbc083f4e9a7069aa185800e
Author: dextercallender <dec2148@columbia.edu>
    builtin.blr
commit 1f29762cdeba9b53cb4de0ebf5a6c2eace7229cd
Author: tim goodwin <tlg2132@columbia.edu>
    stdlib updates with repaired char operations
commit 835ddf3ffb64f207bdcf48cb14c391b67961f0f9
Author: dextercallender <dec2148@columbia.edu>
    edge detection complete
commit cd3a58db673a1f850ad8da284099b37042eababe
Author: dextercallender <dec2148@columbia.edu>
    initial edge detection function complete
commit 842d14e1b832638a6797180901b1693459866dad
Author: Timothy Goodwin <timg.goodwin@gmail.com>
    Update README.md
commit 63194bf1a788c42899cf3b667ea40b24b1571533
```

```
Author: Tim Goodwin <tgoodwin@graphiq.com>
    updated charToIntensity arith
commit 9ffe8b87be9164454b65d8995d0479608c6230a4
Author: tim goodwin <tlg2132@columbia.edu>
    implemented int cast and double cast
commit 061327b4314fbff1dab44f9624041207150539dc
Author: tim goodwin <tlg2132@columbia.edu>
    char boolean operators
commit 17f4857192ac65a4e10fbce687fd358092b0187b
Author: tim goodwin <tlg2132@columbia.edu>
    maybe
commit 3d702866c2e2f9ae3643b45c42116256b247f601
Author: tim goodwin <tlg2132@columbia.edu>
    test text output
commit e9c587d2d362104bebb5d1dc8b620420aa3b8431
Author: tim goodwin <tlg2132@columbia.edu>
    commented out main in bindings.c
commit 6df51d16a45eeb51d92d6927eac6a9382e3fe881
Author: Timothy Goodwin <timg.goodwin@gmail.com>
    emojis
commit c79709528974d3ad5d69347eb46c683409070752
Author: tim goodwin <tlg2132@columbia.edu>
    images folder
commit da12a44e96e6b5af4e8f52acf0fd372edd6e1fb8
Author: tim goodwin <tlg2132@columbia.edu>
    leaf
commit 7cbba172e7979778c9d48154fe3ea26d07addb9e
Author: tim goodwin <tlg2132@columbia.edu>
    differentiated print() and println()
commit 11d8f6995d66e45d4b35c1feff8d8b89a058e1a5
Author: tim goodwin <tlg2132@columbia.edu>
    initial ascii example
commit c60c7155a1749956adbec25552c4df475caa9a57
Author: tim goodwin <tlg2132@columbia.edu>
    grayScaleImg
commit 8be670fb2ee4e2475749d3611d0bbcde25524a2e
Merge: 6208abc 7afd1ba
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit ef306f87e99608825d1b708823f64f3c4ab1958b
Author: dextercallender <dec2148@columbia.edu>
    canvas(), readgrayscale, and readcolor() all return structures
commit 7afd1ba8316b461035c8d07a032cd8694ea51af8
Author: dextercallender <dec2148@columbia.edu>
    readGrayscaleImage() and canvas() function modified
commit 5f11c0d1d5d064de7036d602fa8351efe3d2698b
Author: tim goodwin <tlg2132@columbia.edu>
    array length compatability with C backend arrays
commit 5f442fb45bcf11af4d480a15d22de6566ea56f10
Author: tim goodwin <tlg2132@columbia.edu>
    getImg test example
commit ecc77444bd177af4feab447c034a29ad9552b5ab
Author: tim goodwin <tlg2132@columbia.edu>
```

```
    struct pointer access working for returned data from C lib
commit 7b8eb408869aff208a27638ba060bef1232481ed
Author: tim goodwin <tlg2132@columbia.edu>
    width height struct
commit 8a0133215e23be6fb8d5d62ec4c75af3924feb7a
Author: dextercallender <dec2148@columbia.edu>
    modified readGrayscaleImage() to return struct
commit f4d11e3072a9c209047c75617b6f4345a0544665
Author: tim goodwin <tlg2132@columbia.edu>
    add backend connection test
commit 04669df80b3d677dc0eb5a13531bdbc40af5e169
Merge: 211564b 05b25ca
Author: Timothy Goodwin <timg.goodwin@gmail.com>
    On the quiet night of Dec 7th the holy c_backend2 was merged into master
commit 05b25ca684012ab1cc94c39b38893f92ca825404
Author: tim goodwin <tlg2132@columbia.edu>
    dynamic arr len function handler
commit 553bd0c4009210f7d969333910012e95fe7e1d8c
Author: tim goodwin <tlg2132@columbia.edu>
    supporting two different array types based on initialization enviornment
commit 0ff51710557b536fd5e73e091f8618f2944c1348
Author: tim goodwin <tlg2132@columbia.edu>
    updated unsized array vdecl to new approach
commit af627435bcd781c6e43de2f3ac2b51190ca8810e
Author: tim goodwin <tlg2132@columbia.edu>
    handling 2 kinds of arrays
commit d59f0e21456ac53331a49155deb6a03793b0bfd6
Author: Tim Goodwin <tgoodwin@graphiq.com>
    generator comments
commit fd757d1fd29842f716376ec14621c980e500c98b
Author: tim goodwin <tlg2132@columbia.edu>
    genesis
commit 211564b57fde684ac5f259e2f5b5f3c95f1e3f0c
Author: tim goodwin <tlg2132@columbia.edu>
    added mod arith
commit b6d0fb7709009ca150faa7b4e65a81a4729aa97e
Author: tim goodwin <tlg2132@columbia.edu>
    added darken and lighten
commit 3caaec0ba16b1c7f6dd9f39d6d77801c308dd2b9
Author: tim goodwin <tlg2132@columbia.edu>
    implemented framework for |x| calling charToIntensity on x
commit 29b44d3e105328099571a603271bb2fdc1d6bffc
Merge: 45fca89 e6dbb5a
Author: dextercallender <dec2148@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 45fca891b149e0580c5576a3b877ff462d1bf641
Author: dextercallender <dec2148@columbia.edu>
    added charToIntensity() function
commit b2251126572e907adc9968fb2e895a78c33bc82e
Author: tim goodwin <tlg2132@columbia.edu>
    dont freak out dont freak out
commit b6646f84d6cb3bef90b2f503e482f71de43d7e48
Author: tim goodwin <tlg2132@columbia.edu>
```

```
    1D access works
commit e6dbb5a5177d2f615c0e184b45a19853c24dc2b8
Author: tim goodwin <tlg2132@columbia.edu>
    nesting pointers for array access
commit 4aa224f10d6aa14045a244d1058b7525616f2a50
Author: dextercallender <dec2148@columbia.edu>
    function to map intensity to character complete
commit bdfe25b95fb583637ff7f4a39fd4f2aa937267a0
Merge: 0fc34f3 db77733
Author: dexter_callender_iii <dec2148@columbia.edu>
    Merge pull request #11 from dextercallender/c_backend
commit 0fc34f3e312df1e451ad797671c2187d580e30e4
Author: restocle <sh3266@columbia.edu>
    testscript
commit db77733e889827d21a33fa398253ea2aaa4c2c1e
Merge: c732cd5 0b7d419
Author: dextercallender <dec2148@columbia.edu>
    Merge branch 'c_backend' of https://github.com/dextercallender/blur into
c_backend
commit c732cd58493a7459076833c213717189654c723b
Author: dextercallender <dec2148@columbia.edu>
    canvas function in c backend complete
commit 17e7b0050f2e5cdcc1419b8a7d97510904c57372
Merge: b8581ca a36fae8
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit b8581ca7f5def159bfbec3e143e2d3f2eed2565a
Author: restocle <sh3266@columbia.edu>
    working on intensity bar
commit a36fae84f70274a5ffcbe708e23f4904a7a5c135
Merge: a832ca8 0b7d419
Author: dexter_callender_iii <dec2148@columbia.edu>
    Merge pull request #10 from dextercallender/c_backend
commit 0b7d4192025d02883dc7adb991c2101c6e7f6e2f
Author: tim goodwin <tlg2132@columbia.edu>
    building pointer to array type when array is returned from func
commit 6426433190748303a9116fe6b44a2c0c04c40783
Author: tim goodwin <tlg2132@columbia.edu>
    foo test, try it out
commit b66385f9e3f5572af85b303a726c1e8033d17a1c
Author: tim goodwin <tlg2132@columbia.edu>
    bindings
commit e4dbc5e5ebcb3046881ede97b34530e6cc474661
Merge: b94ce46 0b52ee8
Author: tim goodwin <tlg2132@columbia.edu>
    Merge branch 'master' into dex_makefile
commit 0b52ee86abc3e502e2598fbb88c3e2f28fab6847
Author: tim goodwin <tlg2132@columbia.edu>
    declare foo external
commit a832ca851bc4d3c11dafdc19f08bc51cf3676ced
Author: tim goodwin <tlg2132@columbia.edu>
    unsized array 3D
commit 5af8af3e396f602eda643a6423535cf1e90278b9
```

```
Author: tim goodwin <tlg2132@columbia.edu>
    2D array iter example using len func
commit d0ddfdc9bddc8aceb8bafc9bad2e740ec420ef0d
Author: tim goodwin <tlg2132@columbia.edu>
    supporting 3D arrays
commit b94ce46ada80e97f81e338f81a27931aa49896c9
Author: dextercallender <dec2148@columbia.edu>
    final makefile step
commit 923bf220fae31d30a8680192aff26f9ecdbeb5da
Merge: 28d986b bd51dcb
Author: Timothy Goodwin <timg.goodwin@gmail.com>
    Merge pull request #9 from dextercallender/new.array
commit bd51dcb9f0c5a8388ccb8d2f7f448b696153eff5
Author: tim goodwin <tlg2132@columbia.edu>
    some new array tests exhibiting updated usage
commit 28d964425acbcfa66e4f249055510dbe03f18de4
Author: tim goodwin <tlg2132@columbia.edu>
    tired of arrays no variable sized arrays too tricky
commit 329ef10108da1aea2f9058f3d0da074435936082
Author: tim goodwin <tlg2132@columbia.edu>
    added built-in array length function
commit 4cafe39722d0f567a6e9153556a25215d3edaf29
Author: tim goodwin <tlg2132@columbia.edu>
    1D, 2D size init and literal init working
commit eeeb2cdc7bd71a15d5654f4c5d41e4b60414ec37
Author: tim goodwin <tlg2132@columbia.edu>
    just compiled new array approach..
commit 980f68c822f255b0b98c102f667835dfd4ce3e28
Author: tim goodwin <tlg2132@columbia.edu>
    SizedArray not arraysizeinit
commit 28d986b99a25ef7e5117c38ca15d541a42a21b48
Merge: 8a98ce2 98c30af
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 8a98ce2160e7c0d45b775a00a701edf6294f53ae
Author: restocle <sh3266@columbia.edu>
    tests
commit 2e0eac6235fb642c1617f537384d9091fc7a4f80
Author: tim goodwin <tlg2132@columbia.edu>
    fix hello world return type
commit 98c30af0e8375464ae5161f41a15721ea49bb3ab
Author: dextercallender <dec2148@columbia.edu>
    new makefile to test. en route to full pipeline
commit c5d76b296e2a8e26c534d76ef6f5f099111f5935
Author: restocle <sh3266@columbia.edu>
    array fix
commit 8131a1bf283e6f253ec1fbcfeedb72f67c80bb42
Merge: 6389bb4 c21311a
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 6389bb478ad2be626c72c313e29e60c30c9a0e50
Author: restocle <sh3266@columbia.edu>
    Makefile edit
```

```
commit c21311aa4a845dffc50d8193ea6d32cf242897c9
Author: dextercallender <dec2148@columbia.edu>
    readColorImage(), readDimensions(), readGrayscaleImage() complete.
commit 0177cc80396dffdd823677c05a96bdf973f107e4
Author: Tim Goodwin <tgoodwin@graphiq.com>
    more test updates
commit 51e1787e4b8b0247644ca70b957405bfa4590885
Author: Tim Goodwin <tgoodwin@graphiq.com>
    updated loop tests to account for new line printign
commit 2bc0ad73d6152202a09a4c62c57b74ac736a5073
Author: tim goodwin <tlg2132@columbia.edu>
    keep track of arraytype dimensions
commit 0bc239d734736a4607d6f73f092a3f23f79505f8
Author: tim goodwin <tlg2132@columbia.edu>
    supporting float type binops now (see included test)
commit e77c7fb69467b45e378ae0440d5a7c1283dbd875
Author: Tim Goodwin <tgoodwin@graphiq.com>
    updated print formatting in tests
commit 44c9edabb69f8e8059c5c805a915ca2d970aa593
Author: tim goodwin <tlg2132@columbia.edu>
    No longer need print format strings :)
commit 02f1aa23844886acf12f08648a265de65deced00
Author: tim goodwin <tlg2132@columbia.edu>
    who cares
commit 4aaa030fa4892ea0f40815c88fbf4d77cf4d0e92
Merge: 1898a40 dbc25d4
Author: dextercallender <dec2148@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 1898a4009613c606ff9242039f71ac0f583c1152
Author: dextercallender <dec2148@columbia.edu>
    fixed merge conflicts
commit dbc25d42519a27a1256fdafb6858dbf617f65616
Author: restocle <sh3266@columbia.edu>
    tests
commit 61fb6d351a3cfe83c5dc69e9c821858641f941d6
Merge: babc5e3 ec4ffcc
Author: dextercallender <dec2148@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit babc5e36b502f3c3be60059e039324acbc50627f
Author: dextercallender <dec2148@columbia.edu>
    completed function to return pointer to pointer of image and dimensions
commit ec4ffcc47f0f83d3a41847d513ba58b55384b232
Author: restocle <sh3266@columbia.edu>
    removed Canvas from ast
commit be457240b3dc367a1bfece907f57a296b742113e
Author: restocle <sh3266@columbia.edu>
    removed Canvas from parser
commit 01bdb7c78c872c8fed844401a79830d62e7cf2c9
Author: restocle <sh3266@columbia.edu>
    removed Canvas from prettyprint
commit 57fe7bf9b9877f5ada27a205d5919ea478f98ddf
Author: restocle <sh3266@columbia.edu>
    removed Canvas from ast
```

```
commit 44407404f3fbb2bdbd041a8d2d9a9a4b894fb6a1
Author: restocle <sh3266@columbia.edu>
    removed Canvas from parser
commit 02f1e67cca8262a808a90dc0b6a09e9c0b661c4b
Author: restocle <sh3266@columbia.edu>
    removed Canvas from scanner
commit 83b68f4395b8a25c76a543e5ccf7bef73b53198d
Author: restocle <sh3266@columbia.edu>
    tests
commit 8757c3cff9e4be4a36f383f26450afdc74dfaa20
Author: restocle <sh3266@columbia.edu>
    tests
commit cd2ef61cce067d934d57800ef1a411fcadb52df3
Author: restocle <sh3266@columbia.edu>
    tests
commit b33e843e694414e88d349a1da85e5d0e4d570e8a
Author: tim goodwin <tlg2132@columbia.edu>
    some error handling in generator
commit 951f012924ce2671610ca94f7bb1b778dc515030
Author: tim goodwin <tlg2132@columbia.edu>
    1D array literals working
commit 2c1bc789b58f7aea540e7154cd472bf3a3122e79
Author: tim goodwin <tlg2132@columbia.edu>
    example array tests
commit 02477d620b2a1db684c0e7d4ce4aba5cab0f8077
Author: tim goodwin <tlg2132@columbia.edu>
    array access works now
commit 9cf37c4402d6a02c6adff797ba61c5d7ade5c9cf
Author: tim goodwin <tlg2132@columbia.edu>
    small fix
commit 0de674248a5872efbb684f5f872a1d35a01a6854
Author: tim goodwin <tlg2132@columbia.edu>
    fix id_to_str to handle array ids
commit 95ccdeac38dea9b6b99eabb1414931d008290778
Author: restocle <sh3266@columbia.edu>
    testscript error redirection
commit 88f4840ce76cb35aeb56fedcde260fb0d16655e3
Author: restocle <sh3266@columbia.edu>
    tests
commit 291f3bf5ac15c309b0ca50367be0f792b8ba37ad
Author: restocle <sh3266@columbia.edu>
    testscript
commit 6fa081fddae57fd14c2cd37b237ac5faa330c466
Author: restocle <sh3266@columbia.edu>
    tests
commit d54e3db69ac90cdcd314f962b4f5915d76a21895
Author: restocle <sh3266@columbia.edu>
    tests
commit 6a6ac87c58bfe468427c7c5bf04a02b226da4f5d
Merge: 194a26b dc994c2
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 194a26b4d7028b17f7ce2c367b7e0b775e266900
```

```
Author: restocle <sh3266@columbia.edu>
    updated testscript
commit dc994c2f7cfe49d265e07b4d33071f41d4e00a5c
Author: Tim Goodwin <tgoodwin@graphiq.com>
    renaming type getters for clarity
commit 04b1ab5ca456bbef842049224745fb4c19b3c9f2
Author: tim goodwin <tlg2132@columbia.edu>
    arr access test example
commit 648011fa7d0b4e695e5ef51c922532ca50c42ee3
Author: tim goodwin <tlg2132@columbia.edu>
    fixed shift reduce conflicts
commit 7199f81fe94c43ac20beaf00304e322ff6769673
Author: tim goodwin <tlg2132@columbia.edu>
    notes
commit e32b6dbd31057c4fd6efcfa593dd4beba56ba648
Author: tim goodwin <tlg2132@columbia.edu>
    trying arrayaccess using expr instead of ID
commit c108f7655b545773fbc9d74a4ac74cada2452459
Author: tim goodwin <tlg2132@columbia.edu>
    array access compiling
commit 8db3bd1825443f9a777bcdd101a5ef30ac6fbdd3
Author: tim goodwin <tlg2132@columbia.edu>
    compiling 2D arr support
commit adae72a8c326bbda016ba123e6e02f7b89cecf18
Author: tim goodwin <tlg2132@columbia.edu>
    arr size pretty print
commit 8acb144880527190dba7acd284a44c768bf64833
Merge: fcab198 30613ad
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit fcab198c11ade79c33f9933491064c12b983848e
Author: restocle <sh3266@columbia.edu>
    tests no semi
commit 30613ad8aa1bc59e595215d47b253787e1eeba90
Author: tim goodwin <tlg2132@columbia.edu>
    while loop test
commit 8bad661f10a2bc7519b424c21429571db97e24ce
Author: tim goodwin <tlg2132@columbia.edu>
    frontend changes to arr size in it, arr access
commit 25949c92c0d1bc78e175f0bb7b237e2f8a4c0b08
Author: restocle <sh3266@columbia.edu>
    tests
commit 6208abcf0cf4bdff0cae0369965542eea518acba
Author: melissaKG <mkaufmangomez@gmail.com>
    Starting to add variables to semantic analysis.
commit 02e5022b25df55fe4dbf5a60ee9eac3d30bb4e7a
Author: tim goodwin <tlg2132@columbia.edu>
    modularized arrmap update + added float binops
commit c4337f0e2e7dadb5b29df4f1e0ff5055e2d3e7b0
Merge: 1dd3a25 7034945
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 1dd3a2589ba8fbcff279f7cf82346cac73aa54b4
```

```
Author: melissaKG <mkaufmangomez@gmail.com>
    TEMPORARY blur.ml without semantic analysis.
commit 7034945314accea5d87ed15c7cf61b9abcf2315e
Author: tim goodwin <tlg2132@columbia.edu>
    1D array literals working
commit 20938184ee541f171c71e7d8c00be311b08bc28c
Author: tim goodwin <tlg2132@columbia.edu>
    ok
commit 2d0dfeea59216397eef5785c25b1fabb8e899666
Merge: ce2ee10 5c14fa7
Author: Timothy Goodwin <timg.goodwin@gmail.com>
    Merge pull request #7 from dextercallender/tim_arr
commit 5c14fa74bcd7165be838b2e42257fb4d0911f2e9
Author: tim goodwin <tlg2132@columbia.edu>
    while loops work it was a tricky error
commit ce2ee10af32371e2dbfc74eefc4514f41f25785e
Author: melissaKG <mkaufmangomez@gmail.com>
    Undeclared identifier check is working.
commit 5b25bde2b3682f2c9b9de00f1a5bf6fa980c9b06
Author: melissaKG <mkaufmangomez@gmail.com>
    Beginning to implement checks for expressions and statements. Writing test to
ensure that nothing follows a return statement.
commit f8bc0fde5d1dbbd0323f5b4a2286792035cfd2df
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding function that will return type of a variable.
commit ee5afac17d47ec85e56d1d0c379ea95a9043c334
Merge: b74ecab 59bbe91
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit b74ecabb48554fa49fe81d31b6f0ba463196532f
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding global variables to a map and writing functions to add local variables to
a map.
commit 59bbe91314db2997d4baef3074319a3118ec71ba
Author: restocle <sh3266@columbia.edu>
    parser & tests
commit 9b0b37a01170ff8f156f36de1bb5eb9105db8460
Merge: 26d4d79 6298ac1
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 26d4d794da9e4f8181611c6317e5a5d0431789ac
Author: restocle <sh3266@columbia.edu>
    fixed while loop pretty print
commit 92d4c47c4d6bf129294fca65872b42243fd2bbee
Author: restocle <sh3266@columbia.edu>
    fixed for loop pretty print
commit 6298ac187e1eab80faff9df05b450cadb666429e
Author: tim goodwin <tlg2132@columbia.edu>
    compiling arraysizeinit, needs testing
commit d47702a97490f81388611bce150260a5080c89c3
Author: tim goodwin <tlg2132@columbia.edu>
    added noexpr to codegen expr
commit a4a26ada6fd13e19e5b6cc62bace53be4e3d8b89
```

```
Author: tim goodwin <tlg2132@columbia.edu>
    keeping track of array dimensions in map (1D right now)
commit b82a6cdb508c4249128431a66874eeef27b19842
Author: tim goodwin <tlg2132@columbia.edu>
    trying array stuff
commit da55d59b1e3ecbfae704167b381244e438162e80
Author: restocle <sh3266@columbia.edu>
    for loop
commit 7f726653019275150f9aa449e856e01a51ffbebf
Author: melissaKG <mkaufmangomez@gmail.com>
    Ensuring that main function exists.
commit f4c79b821e75f43ed4efca6b40cca3eccd727063
Author: melissaKG <mkaufmangomez@gmail.com>
    Cleaning up array code.
commit 92f768969760082c2581714500ed8a207710d3bb
Merge: 595c802 2b68c5d
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 595c80237dfa19eafe45b2c7bbb2467615ba47d4
Author: melissaKG <mkaufmangomez@gmail.com>
    Making array size declarations take int list. Note that pretty printing is not
formatted properly.
commit 2b68c5d0b8c1308ec3f0cf17ab3d144a2fdd1fe2
Author: restocle <sh3266@columbia.edu>
    test update
commit 5927bd8ac5db9934fe1585912eb3cc679d73be6b
Merge: e2497f2 52aa84d
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit e2497f25f7a8297fcfebfb7c29ef0ba49712d989
Merge: d8674bc 81d43f1
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge.
commit 81d43f142589747e3189e6966cf2f5c20b31c495
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing formatting.
commit 03ee4683ed0d6679cc1a48de3fe7e70bdc5892ed
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing array type in generator.
commit 1c5d46c11227bb2b0df4353b0d148bec4297c5cb
Author: melissaKG <mkaufmangomez@gmail.com>
    Working toward array types.
commit 923b87809b8aca0ef4f73f0bc2a3b009d8559f0b
Author: tim goodwin <tlg2132@columbia.edu>
    no more errors datatypes work
commit 90b910b7543a9199b556dc956bfb7bfe3497fda4
Author: Tim Goodwin <tgoodwin@graphiq.com>
    fake global maps i hate functional programming
commit bd35911a9ecd1b25912b4c906fe0e4b768475510
Author: tim goodwin <tlg2132@columbia.edu>
    temp
commit 0a4d334c30b23b888211032737120fcaea31c31c
Author: tim goodwin <tlg2132@columbia.edu>
```

```
    ugh
commit 52aa84db6602d1b6683f745c573ca00bdd7d69ad
Author: restocle <sh3266@columbia.edu>
    fail output
commit e0a43ccc901d2466a5f828b544b8d6a04455d483
Author: restocle <sh3266@columbia.edu>
    fail test
commit 69aa53228788b9bdaa4180a7ad02a7f3406527ce
Author: restocle <sh3266@columbia.edu>
    updated test output
commit d4d2510161af9e2c106244647a6fc356912c3b85
Merge: 78c4512 ae7f710
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 78c45123ae9c06afe71dec5d679c151b006be5fc
Author: restocle <sh3266@columbia.edu>
    tests print lowercase
commit ae7f71043c40a5f7183ffe3c2f277b1a90ba6800
Merge: e43cf04 3bc2180
Author: dextercallender <dec2148@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit e43cf0418ec6a10988a304a46e774c65fa3ba26b
Author: dextercallender <dec2148@columbia.edu>
    linked opengl and created c library that can read, display, and return data
pointer to image. Not yet linked into Makefile. to test, cd into clib/, run 'make
exec', and then run './test Golden_Mushroom.jpg'
commit d1aae36902af274dcfe49e2251ebedf060c89d35
Author: tim goodwin <tlg2132@columbia.edu>
    arraty init framework
commit 3bc218088e509aecbe22de6914a4adb246aec1f1
Author: restocle <sh3266@columbia.edu>
    code tests
commit 8860ccc682ddfc42d1dd7146a267a07febe95233
Author: restocle <sh3266@columbia.edu>
    code compile test automation
commit 2b562951768f8cd1109ef9e1352c5befad2fffd2
Author: restocle <sh3266@columbia.edu>
    fixed testscript
commit c458b2ffc33252d767d4c463854dec06e7d70e1d
Author: restocle <sh3266@columbia.edu>
    updated testscript for compile test automation
commit 5a04a6deb75aaf93a461108ad129104815c6084a
Author: restocle <sh3266@columbia.edu>
    new tests and testsPP folders
commit 221d44e6ef480333c762b499a2679264361ff990
Author: restocle <sh3266@columbia.edu>
    update testscript
commit d9e93803eeeff55f6bea342a2883f81929c3cf8f
Author: restocle <sh3266@columbia.edu>
    testscript without blanks
commit 484b4cbcd6e969fe83f721c6e7758c30676e71da
Author: restocle <sh3266@columbia.edu>
    added hello output file
```

```
commit d8674bc2b620e0883cd79f0422c966e4bcb3373b
Author: melissaKG <mkaufmangomez@gmail.com>
    Beginning to make changes to gen for arrays.
commit f94a78e59672a5efc4053e24b7e77b14378b334a
Author: restocle <sh3266@columbia.edu>
    updated testscript
commit c24508b8f730f954945e834038986b53e85d2827
Merge: 6a83f97 4accabb
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 4accabb9d3a9d269babdc2bf4fbea25727b7fa40
Author: dextercallender <dec2148@columbia.edu>
    removed all llvm linking code, attempt 2
commit b3aa0a24f5ad8ca12a634392617e9c0dbed16aea
Merge: bc3d6ce 364d7e5
Author: dextercallender <dec2148@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit bc3d6ceb97a967d153f45b6d95835b707aa92de3
Author: dextercallender <dec2148@columbia.edu>
    removed llvm linker code and library
commit 364d7e558ebdf1bb5696bb3b3f181d45daa2ca7d
Author: restocle <sh3266@columbia.edu>
    updated testscript
commit 5c00eec2c8223587280f6b35064503ab81ea5b8f
Author: restocle <sh3266@columbia.edu>
    updated testscript - helloworld
commit a172bfb59adac1a0b603d68ac4ce38d208464785
Author: restocle <sh3266@columbia.edu>
    tescript update
commit c1f0e397d226d54b5c2688606b8d68d99666e50b
Author: restocle <sh3266@columbia.edu>
    first automated tests
commit 6a83f97eb9d462fe8d5234d6584127cc7732e978
Merge: f22ba93 400ffbb
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 400ffbb033c246c0fa9462a7e6ca653d90702aae
Author: dextercallender <dec2148@columbia.edu>
    modified readme with installation instructions
commit f22ba93d4fa1cd0ac52caf388285a77f9672bb01
Merge: 038a01a 701a2ee
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 038a01a92c2cd9d342a801d87f4b6e1c382d2dc8
Author: melissaKG <mkaufmangomez@gmail.com>
    Restructuring array type in order to distinguish from primitive types.
commit 701a2ee077efb524c06f28b2c89cb4b823bc206b
Author: dextercallender <dec2148@columbia.edu>
    read image in bindings.c nearly complete but seg fault
commit 7582a499c8f17ba68e370ba9dcd2ce201f00c746
Author: dextercallender <dec2148@columbia.edu>
    slowly fixing more errors with linking an image reading library
commit 078a49bbd336bce8fd281530b72f2507dabafac3
```

```
Author: dextercallender <dec2148@columbia.edu>
    added commands to install llvm for ocaml in readme
commit c63553fc0ba6d009679c1ba4f690bdfceaad7644
Merge: 3cb198e 980d2c3
Author: dextercallender <dec2148@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 3cb198ec929950a4c348878ba15ce4ba32a663c2
Author: dextercallender <dec2148@columbia.edu>
    now using png++ image library
commit 980d2c36b172af474ae616dc1f5ab83529e96b07
Merge: b229267 c3296f3
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit b2292679b7907d53361a6f1a32f0df1cf2bc9e35
Author: restocle <sh3266@columbia.edu>
    testscript update
commit c3296f3deeffcad5bc0a28aaf806043b0ec582bc
Author: dextercallender <dec2148@columbia.edu>
    linking cpp files works now
commit 779b0f7df60898b0ec8f4c594ca590562dd7b81b
Merge: 4a684a4 d08b049
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 4a684a4634424a8d10c7459f4b57b6c6dfa3ae50
Author: melissaKG <mkaufmangomez@gmail.com>
    Removingdeprecated array test.
commit d08b049a83e6a563f153a62d71c0d6ae3e75e695
Merge: b7b48bb 2241c28
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit b7b48bb42ebb811cc8b6c05dd36c1c5d5a35d07e
Author: restocle <sh3266@columbia.edu>
    updated parser, ast, prettyprint, test for array init with size
commit 2241c28cbf6994b5fbc64261fc60605a01be7267
Author: tim goodwin <tlg2132@columbia.edu>
    WHILE and FOR statements generation. TODO -- scope behavior checking with tests
commit ae8b341764064be38f5fa1d52062bea5040e33d5
Author: tim goodwin <tlg2132@columbia.edu>
    generating IF statements. TODO: testing
commit 729b133e6263be03c21ca1ddc3fd8d3ad1ec475f
Author: restocle <sh3266@columbia.edu>
    trying pretty print comparison
commit 16821f495aa4d8d66253b533915a28cf2643f880
Author: melissaKG <mkaufmangomez@gmail.com>
    Implementing array access.
commit 79dfc57335d6e9a2e66b9ef04a528dff937af0b9
Author: tim goodwin <tlg2132@columbia.edu>
    TODO: finish statement handling. if, for, while, continue, break.
commit eac7a51ab9c772ab9b403afc5263b8e59548d081
Author: tim goodwin <tlg2132@columbia.edu>
    unops
commit 91f7991a56d4d2102f1454b44077d738a0780f51
Author: tim goodwin <tlg2132@columbia.edu>
```

```
    ~~ hello world ~~
commit f446a105baf925b0c62302ba1e94e43b2ec6ec9e
Author: tim goodwin <tlg2132@columbia.edu>
    supporting string pointers. 'string s = hello;'
commit ba6b8e1372211600a46e0d8c531c1a12d95977a0
Author: tim goodwin <tlg2132@columbia.edu>
    exceptions framework
commit 17134ea6198d0832035bf7571f411e3a89136799
Author: tim goodwin <tlg2132@columbia.edu>
    generator works with assigments now. that was not trivial
commit 6c718a77a82bb056924e6b048bf7f0793a721ff2
Author: tim goodwin <tlg2132@columbia.edu>
     'int a = 5' working in llvm module
commit c10df21e55f59f70d363fbd89f7ef72f26e55f63
Author: tim goodwin <tlg2132@columbia.edu>
    incorporated var declaration into the codegen_stmt function.
commit 90e069a6cf8c8b0f29472d883edf9d04721087d8
Author: tim goodwin <tlg2132@columbia.edu>
    temp commit
commit ef6a11101e4dafdf900bea39568c4771b04812a0
Author: tim goodwin <tlg2132@columbia.edu>
    try
commit 94eb5d983b2d226a07454d266fbeb4c9036463b8
Author: tim goodwin <tlg2132@columbia.edu>
    minor fixes, code cleaning. prettyprint charLiteral update.
commit 2deccc67dc2fe0c8fe94e8bada480ce5b2e8af94
Author: tim goodwin <tlg2132@columbia.edu>
    id_to_str hack for Asn binary operator
commit 85a53ed2de6c78c4f3179d40a6053567395eb5fc
Author: tim goodwin <tlg2132@columbia.edu>
    wasteland
commit 784a9160b87fea8f2ee50d0dcdf8d840f00fd40c
Author: tim goodwin <tlg2132@columbia.edu>
    idk
commit 0ac535941ac98b2938a6d8f701a97f530378268e
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding more to local variables in generator.
commit 0e01e6b1bf85543ff42824e10e3f811f5dcb738c
Author: melissaKG <mkaufmangomez@gmail.com>
    Implementing updated local variable declarations for generator.
commit 64cd202bd211e0b291ecc38bb882498b01b3d34f
Author: melissaKG <mkaufmangomez@gmail.com>
    Removing redundant Asn in ast.
commit 4025e52678189219c89e68a91e1068700e42339b
Merge: 3b5e266 f929215
Author: melissaKG <mkaufmangomez@gmail.com>
    Testing.
commit f929215593f3df00627c21921978648c2b99807b
Merge: cd428ac 6e7bc56
Author: melissaKG <mhk2149@columbia.edu>
    Merge pull request #5 from dextercallender/func_struct
commit 6e7bc56c631b7110435988799875626b1fbc447e
Author: melissaKG <mkaufmangomez@gmail.com>
```

```
    This is printing vars correctly.
commit cd428ac95eb2875672c628e45567af5c7b3799df
Merge: aa88c02 77ae099
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt into func_struct
commit aa88c025eb06e602f63c9227020645d4861498f1
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing format of pretty print for var init.
commit 3b5e26619e3e3adcff885f41e73d67be3eaa187e
Author: melissaKG <mkaufmangomez@gmail.com>
    Making test for func more complex.
commit 77ae0991be5e652d7e257bbd90f85158d252faf1
Merge: d84c907 3f0c9d9
Author: melissaKG <mhk2149@columbia.edu>
    Merge pull request #4 from dextercallender/func_struct
commit 3f0c9d948fd3556947c341aff449f1424fb16756
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing pretty print alignment.
commit 36753b45799c04f0f825054fcf7b630ffd1d5191
Author: melissaKG <mkaufmangomez@gmail.com>
    Implementing new local variable structure.
commit d84c90702e371ae7f7e3ceaf80446fe32afe458e
Author: tim goodwin <tlg2132@columbia.edu>
    added TODOs for all incomplete pattern matching in generator.ml
commit 46d38d8dccd7de84026bad051066020e9c259afb
Author: tim goodwin <tlg2132@columbia.edu>
    properly included ast module into namespace
commit 946ce4d98c169b728988a745655d4eac82c00172
Author: melissaKG <mkaufmangomez@gmail.com>
    Changing func decl to have statement list instead of locals and the body.
commit b2b3d1a73ab6df2f007baeca46ff9458744154fb
Author: melissaKG <mkaufmangomez@gmail.com>
    Cleaning up.
commit 107cb2dbfc6fe575a2d6564eb4cdf4b3cfe6291d
Author: tim goodwin <tlg2132@columbia.edu>
    code generation for a blur function call
commit 576c4fdefd793dc6088f21c79e54c382e7d34b24
Merge: 2c83e38 1dd07db
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 2c83e381dfe91f1858ca6e0121537ddc53764a91
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding test for var decl and init on one line. Also starting to check assignments
in semantic analyzer.
commit 1dd07dbf0c04e97c7ec1ec3cb02a40a95579eb75
Author: dextercallender <dec2148@columbia.edu>
    can link c files!
commit 3dce50d3c4614599e98290dd4a81f46ad9cf5afd
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding one line init to var decls.
commit e11eb1fc5b00a0e65cc61cc4460f4e83998f0fe4
Author: tim goodwin <tlg2132@columbia.edu>
    2D array test
```

```
commit 5516fc400627f04a4fc0d401a8768f4251b6bd0a
Author: tim goodwin <tlg2132@columbia.edu>
    fixed global_var codegen
commit fb2be61e05ea691f6a686367d696fc4589c8d361
Author: dextercallender <dec2148@columbia.edu>
    modified makefile to link generator. Modified blur file to accept flags for
different compile options
commit d610921d52d1d53bf8c7742ba16b50addd933d3f
Author: tim goodwin <tlg2132@columbia.edu>
    added very basic support for print function code generation
commit 954e8ce67585920b5be6b6b835cf5064d8bc16d1
Author: tim goodwin <tlg2132@columbia.edu>
    generator compiles with only warnings, no errors
commit df8cf1f523f92c96e44c3ae32928a4ae73aec840
Author: tim goodwin <tlg2132@columbia.edu>
    error on line 126
commit 1c8fdd5737b51aedb9600dd726602f6a89b71dad
Author: melissaKG <mkaufmangomez@gmail.com>
    Ensuring that user cannot define a print function.
commit d6c85db3f181ed024efafb4b403e288f2ad37c18
Merge: dd09722 8f3c52e
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit dd097226f085191906cf813bc35fa612546a2734
Author: melissaKG <mkaufmangomez@gmail.com>
    Checking for duplicate and void arguments.
commit 8f3c52e628f839acb3440a9cd1367a26d5cd6534
Merge: 3026088 31fff53
Author: tim goodwin <tlg2132@columbia.edu>
    Merge branch 'llvm.try' hope this helps
commit 31fff537609ec61d404936788801192204e624eb
Merge: 5ef9475 3026088
Author: tim goodwin <tlg2132@columbia.edu>
    makefile merge errors
commit 5ef9475a95f331925ecf64134eb922015c0ff376
Author: tim goodwin <tlg2132@columbia.edu>
    first iter
commit 2a0d4cce2d0ef6e9064a4148e9b498fd295f49f0
Author: tim goodwin <tlg2132@columbia.edu>
    v temp
commit 01224bc8fe637fedc45dfafc626aea20c47dfa9f
Author: melissaKG <mkaufmangomez@gmail.com>
    Checking for duplicate function names.
commit 30260889514281aa0ac7731a91a6107bcfd42d98
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding .cmx to Makefile.
commit 71781aa4c736c6a668529d1e0e455e0e49e98689
Author: melissaKG <mkaufmangomez@gmail.com>
    Renaming .mli files to .ml files.
commit d470011755270c165f8054f5f558f707d09bab5f
Merge: 8bc8155 acb1e3a
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
```

```
commit 8bc8155e83635a87f8f42012a8ec2b0fa342c207
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding semantic test for duplicate variables.
commit acb1e3a900ac1749e6ba94026c15c796d2d775ae
Author: restocle <sh3266@columbia.edu>
    updated funccall test
commit cf988226ef3a3217ee0401a6db943319a320222a
Author: tim goodwin <tlg2132@columbia.edu>
    no syntax errors, but llvm module not linked properly
commit 1cded3c3bf6410bfaf6723d803cac495c52d89db
Author: restocle <sh3266@columbia.edu>
    added var test
commit 404529b2a3fecc3eb79afec8240414693da7f694
Author: tim goodwin <tlg2132@columbia.edu>
    try
commit a4374fa948256b0169f3a6a8e3c54948d4c1b83a
Merge: 4693867 22ed5e9
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 46938670da48c5c6fe40df496c4fb52ccd5f7546
Author: restocle <sh3266@columbia.edu>
    added positive while tests
commit 22ed5e9583687185c02fec4afbef277ebcccbf5e
Merge: 4922cf6 b997536
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit 4922cf6d9513b5f94097adafd80ad05c8af85a18
Author: melissaKG <mkaufmangomez@gmail.com>
    Updating Makefile to include generator. Eventually we need to add generator to
OBJS.
commit b997536508a4789f86ca670283f9c346cb550c7f
Author: restocle <sh3266@columbia.edu>
    added positive forloop tests
commit 8c944b75cdfa07a4ec8190518cb4b946446671b6
Author: restocle <sh3266@columbia.edu>
    added fail return tests
commit 82c89dadbf9990f970cbaa94870f8bccfc059ab2
Merge: 89bf77e 10ad6a8
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 89bf77e35cf225a4971e4333144cfd52e61da734
Author: restocle <sh3266@columbia.edu>
    testscript.sh
commit 10ad6a80ebcc80ee19fa1be2d68e1809f1a290ce
Merge: 3c95db8 a241143
Author: Timothy Goodwin <timg.goodwin@gmail.com>
    Merge pull request #2 from dextercallender/llvm.stage
commit 3c95db89bce2426d77839632596391ae689d96a7
Author: restocle <sh3266@columbia.edu>
    added parenthesis balance test
commit a241143e05ee4ca386527133fce6817cd4adfe62
Author: tim goodwin <tlg2132@columbia.edu>
    initial work on llvm generation file
```

```
commit 5d1af5c2a4e8518ffe29fd88ff635a97019e8f46
Author: tim goodwin <tlg2132@columbia.edu>
    starting llvm
commit e5f9d4d89001d1e5feac3c640994999fe5380523
Author: melissaKG <mkaufmangomez@gmail.com>
    Implementing initialization of Canvas.
commit d3ad01169c37664038f2cda11391f7c908fbdd1f
Author: melissaKG <mkaufmangomez@gmail.com>
    Implementing Canvas declaration.
commit c6e271f52e725a5dd3bbc39e68514f8036b85b51
Merge: a042367 3b344b6
Author: melissaKG <mkaufmangomez@gmail.com>
    Merge branch 'master' of https://github.com/dextercallender/plt
commit a04236754fe808e06bc650c26a2ea4bf47a7e0d6
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding void check for global variables in semantic analysis.
commit 3b344b64afd2282757a29d22ffb13029978d973e
Author: restocle <sh3266@columbia.edu>
    added positive conditional test
commit 763fe57ac28facf6cf184020ceb30e97cb6231ca
Author: restocle <sh3266@columbia.edu>
    added positive variable test
commit 5f96f7bfd866a073eb5b9cd6c7fb82f9c9c0ce90
Author: restocle <sh3266@columbia.edu>
    added positive return test for all types
commit 9636d998953a0a98c3f1dd2b18f9c2c5b166b304
Author: restocle <sh3266@columbia.edu>
    updated print test & added for/while loop tests
commit 70704f4d98172b6564d59aa7f8c47125e025d288
Author: restocle <sh3266@columbia.edu>
    updated type tests
commit 0baee814c119470d7005f9f462065e02db078228
Merge: 5a65728 ba9cee1
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 5a6572876dd66cfa7dc2380f1e4a98060d8d24e6
Author: restocle <sh3266@columbia.edu>
    just conflict resolve
commit ba9cee1382b9d4c46733baf4631f09396097db26
Author: tim goodwin <tlg2132@columbia.edu>
    remove temp files with make clean
commit e1b8ef0d4667bdfd9c089140619cb55cd361b676
Author: melissaKG <mkaufmangomez@gmail.com>
    Trying to figure out type. This compiles.
commit caa89ca40310da3cb2f22d8de435d9ace76ada76
Author: tim goodwin <tlg2132@columbia.edu>
    fixed pretty print error
commit 505e37a7de12a6e332c6c5b9a9d14ebf60c4f7ca
Author: tim goodwin <tlg2132@columbia.edu>
    integrated new array_type into master, got rid of arr keyword
commit 6dad7660a69ef25cdc27c630fd8ebdfe51026606
Author: tim goodwin <tlg2132@columbia.edu>
    screw calc
```

```
commit f3747237ba362210e7bbf0354ce81f84bd55c16b
Merge: 1af6759 114c1cc
Author: Timothy Goodwin <timg.goodwin@gmail.com>
    Merge pull request #1 from dextercallender/2D.please
commit 114c1cc446dfcf28b7b04f4cf123218f1d89a7ea
Author: tim goodwin <tlg2132@columbia.edu>
    trying new 2D array
commit 1af67591d61d9360afbcb412f38de88f44b9aa0b
Merge: 3b5735f 4be121c
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 3b5735f41c5a4f9a64d4f34ed8e840f78ff221d6
Author: restocle <sh3266@columbia.edu>
    added type test - only int and void types work at the moment
commit d6079c5fce11bae201936ff68ff4cf1d8cbbde9c
Author: restocle <sh3266@columbia.edu>
    updated comment tests - ignores comments
commit 4be121cef098074e6ea5e5235f3cc59b4370459a
Author: tim goodwin <tlg2132@columbia.edu>
    first design of sast.mli
commit 48d05f15f32e880fe981de775e8b0d682aa0ebe4
Author: restocle <sh3266@columbia.edu>
    added comment tests but need to be implemented in pretty print
commit 2a3ea8ef0fae42edb53a4816afc2788f803fc85f
Author: melissaKG <mkaufmangomez@gmail.com>
    Beginning static analysis. Checking for duplicates.
commit 10abbc8a3d050d41e5812300d2d4bbf2cdf2aacd
Author: melissaKG <mkaufmangomez@gmail.com>
    De-cluttering Makefile.
commit 74666b10fdd2f231a26fe68a8404d77a87b96aca
Author: melissaKG <mkaufmangomez@gmail.com>
    Clarifying arg and var structure.
commit 08bd0a21573ff0b92fecd65896e8e94cf3cef50b
Author: melissaKG <mkaufmangomez@gmail.com>
    Implementing array declaration and initialization with no shift/reduce conflicts.
commit bca56794bfd987b719ec6c7bcb9b09dc1c565598
Author: melissaKG <mkaufmangomez@gmail.com>
    Array declaration works without shift/reduce conflicts.
commit 3b9188ab02e5d2cbb1b748ac98ef0c3363e2a30c
Author: tim goodwin <tlg2132@columbia.edu>
    goodbye arrays
commit b869e37cc18463ef645f9461ee332493a5e2ba71
Author: tim goodwin <tlg2132@columbia.edu>
    init semantic analysis
commit 5f9a60c89524c99f9d662488f8cf87859b564144
Author: melissaKG <mkaufmangomez@gmail.com>
    Ensuring that arrays still work with new variable structure.
commit 94f915d7fd56146bd2d5cdef7718291cb63cebc1
Author: melissaKG <mkaufmangomez@gmail.com>
    Formatting binop.
commit 9c8b65d0a73f25cfd8a4d853911d79cdbe082bd8
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding var specifications to pretty print.
```

```
commit 9fab8461faedc39335bd783356b3bc76b8f88c02
Author: melissaKG <mkaufmangomez@gmail.com>
    Defining function arguments as variable declarations.
commit 4ed3b322165752b80a98dff30751744505eb817d
Author: melissaKG <mkaufmangomez@gmail.com>
    Continuing to change the structure of var.
commit 1de9f62228f190f49226c779e7c3ca8150fa0568
Merge: e963618 8a9b300
Author: melissaKG <mkaufmangomez@gmail.com>
    Resolving merge conflicts.
commit e963618faabc7b77362663bbdd14d9e67ff9f9e4
Author: melissaKG <mkaufmangomez@gmail.com>
    Beginning to change var structure.
commit 8598065ab3c2d679128f4071cff05910325c4f41
Author: melissaKG <mkaufmangomez@gmail.com>
    Changing the structure of the program in an effort to fix the bug with ASSIGN.
commit 8a9b30074df16817391056267d593cda6ba4fff9
Author: tim goodwin <tlg2132@columbia.edu>
    prettyprint should have everything but it neeeds testing sorry for commiting
before testing
commit bcdd7e6bdd556517e3065644445f3e1e6c21948f
Author: tim goodwin <tlg2132@columbia.edu>
    fixed errors with unary op printing, removed whatever Seq was doing for now
commit ca9f3f30889faf325a4b44fbe0fd7519ceeac55d
Author: tim goodwin <tlg2132@columbia.edu>
    pretty print function calls
commit a024ac865655c50f2550dc05785dcc3340d53440
Author: tim goodwin <tlg2132@columbia.edu>
    added unary operators, some comments
commit 466b19ff684887bc1aa80473ac6727978d015805
Author: melissaKG <mkaufmangomez@gmail.com>
    Implementing arrays via curly braces.
commit 7aeb4a80fd7fa2ac2d28b1c6e2cbc82ecd6de248
Author: melissaKG <mkaufmangomez@gmail.com>
    Implementing array initialization by size.
commit 32342729f860aed63483face27c212d925885d89
Author: melissaKG <mkaufmangomez@gmail.com>
    Implementing array declarations.
commit 546dc6075f6bfba271a790f16f45b4a23236e109
Merge: fed90e0 af0ed83
Author: melissaKG <mkaufmangomez@gmail.com>
    Resolving merge conflicts.
commit fed90e07cd2b6a568a2af0781a5c459685605a4a
Author: melissaKG <mkaufmangomez@gmail.com>
    Starting to implement arrays.
commit af0ed83b11545897b929ef85da59b0fd0aca27d0
Author: tim goodwin <tlg2132@columbia.edu>
    added ability to make function calls
commit 3cc322117c47a23a9753ffedcd9500776473b7f8
Author: tim goodwin <tlg2132@columbia.edu>
    added conditional statements and loop statements to parser and ast
commit c83f957121f2c0711e2e299bcad8d6afd3f7fc87
Author: tim goodwin <tlg2132@columbia.edu>
```

```
    added all binary operators to ast
commit 80c0d358a1d743dfe2217739fb25b6127f7e85ae
Author: tim goodwin <tlg2132@columbia.edu>
    added type literals up to AST level. needs integration testing daniel daniel
daniel
commit 647367858c57b38b345feb1f530ca3d1a82041ff
Author: tim goodwin <tlg2132@columbia.edu>
    added type literals to scanner and parser
commit 6e8ac723aec70c632123aadaa9649c74fc396f27
Author: tim goodwin <tlg2132@columbia.edu>
    types?
commit cf57d182b6cc51b2e4837789fc28d09dce840c60
Author: melissaKG <mkaufmangomez@gmail.com>
    Implementing binary operators add sub mult div.
commit 49bb58009cf17b7d405130551b2708d68757c6c5
Author: tim goodwin <tlg2132@columbia.edu>
    added rest of things in LRM to scanner and as parser tokens
commit ca46bdded281dabad25501a1b82793f58b001cef
Author: melissaKG <mkaufmangomez@gmail.com>
    Implementing ASSIGN. Note that we cannot declare and assign on the same line yet.
commit 32d7d09a6383b94438e30a71467c47c96b5624c8
Author: melissaKG <mkaufmangomez@gmail.com>
    Function pretty prints.
commit bb57a33a0c94a13e2dc2fcc6ecbbc5ee08404786
Author: tim goodwin <tlg2132@columbia.edu>
    first pretty print
commit 4f6a9de0e061f9fd861577d64e0534614364b81e
Author: tim goodwin <tlg2132@columbia.edu>
    trial
commit 5c80c98bdc460a0677e5239acea0c956d00f87c8
Author: melissaKG <mkaufmangomez@gmail.com>
    Pretty print will print an empty file . . . but that's it atm.
commit c6e071e26aa4a0a9197cb054981fb14b815edcbe
Author: melissaKG <mkaufmangomez@gmail.com>
    Working toward pretty print. Need to fix fatal parsing error.
commit f562778b2488a196a0c6f4a2fe646a29714a763e
Author: melissaKG <mkaufmangomez@gmail.com>
    Pretty print can take a file as an arg.
commit b06fe13476417ad9cfc7dbf55cc4ae7278934d4b
Author: Tim Goodwin <tgoodwin@graphiq.com>
    added missing tokens to scanner and parser
commit 91d2b715150f4010cef6e3ecbaf0e8f01ff74aea
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixing bug.
commit 0872c9b72772cb64b5abb6275c4e950ccadd156e
Author: melissaKG <mkaufmangomez@gmail.com>
    Need to fix this, but working toward pretty printing.
commit f39aecde3d4e1cfa1db5ea6474a2d73653ebaef1
Author: melissaKG <mkaufmangomez@gmail.com>
    Cleaning up file structure for pretty printing.
commit e9dcb8037a91b49a0a70238bfd59480653461af0
Merge: ce42413 48eee7c
Author: melissaKG <mkaufmangomez@gmail.com>
```

```
    Fixing merge conflicts.
commit ce42413f7d13c58d390c568402e7682a7a19d295
Author: melissaKG <mkaufmangomez@gmail.com>
    Fixed Makefile by adding deps.
commit 48eee7c46924fbbb0d4acbe668e48799a8a1aee1
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding clean to Makefile.
commit dad269c4cd43f2d13a33f1beb7a3d2342f2f363f
Author: melissaKG <mkaufmangomez@gmail.com>
    SOMETHING PRINTS
commit 4ae2cd88f8c5e0c215d9cbdb931a4ff0bb067cf8
Author: melissaKG <mkaufmangomez@gmail.com>
    Modifying the Makefile to link files.
commit cfa8ce94d70865d4b9ecfc85a7f1a8d61f38cd75
Author: melissaKG <mkaufmangomez@gmail.com>
    Changing file structure to get pretty print going.
commit 85a9b3386640478a8e8ee9a9ed33fe27f54fbafe
Merge: 657b290 dd3168f
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 657b2903932f38f6e28a4c8f80e67b6d5d2c4c34
Author: restocle <sh3266@columbia.edu>
    added print test
commit dd3168f162ddf49c787ceb357ef85b115de74c90
Author: dextercallender <dec2148@columbia.edu>
    now we're ready to add the pretty printing functions into blur.ml
commit 5793e20d2d31936ef2bcd44469ff709f5464a4e5
Merge: 61135b2 63de1d4
Author: dextercallender <dec2148@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 61135b26f533964f938b55e16a53b0a9e40a48c2
Author: dextercallender <dec2148@columbia.edu>
    added blur.ml file so we can separate pretty printing functions from the ast.ml
file
commit 63de1d4114ec4e8ca9a4ee03deff3ba263493386
Author: restocle <sh3266@columbia.edu>
    added testscript.sh
commit 0f65314702403ade6131ad015a24d77710756df0
Author: restocle <sh3266@columbia.edu>
    test add edits
commit 3d58df0e7a27d400ff51b192751f2cb499c9ae6b
Author: restocle <sh3266@columbia.edu>
    Added test directory
commit c0480ac9f1ddec17e617a75691b9fcbd2d1f9621
Author: melissaKG <mkaufmangomez@gmail.com>
    Resolving inconsistencies between files. Had to delete some code, but we can add
back in once we get everything hooked up. Need to resolve 1 rule not reduced.
commit df172e7dd58f8e9c09b57151a2841f96efdc94cf
Author: melissaKG <mkaufmangomez@gmail.com>
    Modifying abstract syntax tree.
commit b1e71ce20acbed8ab9b910c7cd8ddc91e40db08c
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding precedence to resolve shift/redue conflicts.
```

```
commit e694b6935498e7f39eeecf28dbec7b313e7d1444
Author: melissaKG <mkaufmangomez@gmail.com>
    Updating parser. Need to resolve shift/reduce conflicts.
commit d8226334c3530fa10f4127053f50fe7e8fbc3bde
Author: restocle <sh3266@columbia.edu>
    parser.mly edit
commit c4ea2e22179f838f4abc9833b2496b7a0c1caf79
Merge: 439988d 861a113
Author: restocle <sh3266@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 439988d27f8219f16af6421b858d455c810c8d33
Author: restocle <sh3266@columbia.edu>
    added operators from scanner to ast.ml; minor fixes in scanner.mll; added
sample.ml for testing
commit 861a113fc7f8ccebfacce73bf1aef84b4ae23c3c
Merge: 6e68dd2 2f89189
Author: tgoodwin <tlg2132@columbia.edu>
    Merge branch 'master' of github.com:dextercallender/blur
commit 6e68dd2456e2577b958c25212b155fed278b8759
Author: tgoodwin <tlg2132@columbia.edu>
    fixed scanner typo
commit 2f8918907a10f9f4c6b64d83e93a0ad745bbdf38
Merge: 5a397a0 d65c220
Author: dextercallender <dec2148@columbia.edu>
    Merge branch 'master' of https://github.com/dextercallender/blur
commit 5a397a02fc0a831ffd65aa5c835e887b7ec7b5dd
Author: dextercallender <dec2148@columbia.edu>
    added makefile
commit d65c220efaf920424cdc53c2529df867e60c9045
Merge: 58b73c8 39d259f
Author: tgoodwin <tlg2132@columbia.edu>
    Merge branch 'master' of github.com:dextercallender/blur
commit 58b73c89e23fb09d6816cb1be2e9d04b08a7a438
Author: tgoodwin <tlg2132@columbia.edu>
    pushing scanner
commit 39d259f7eb2167846f8428bc7447dd5f16b9b060
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding more details to expr.
commit fee666f31509e9ac86e18ce2e5be60547a0c267f
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding skeleton for expr.
commit 2de816f6f5eb82d97e90613c74d915cdb54fc84b
Author: tgoodwin <tlg2132@columbia.edu>
    filled out scanner.mll
commit 25de79200727a009804ca66235c4f712685ed29c
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding pretty printing.
commit 29b6512827ce2716866e4b96eec4f47ac8e7ba54
Merge: c12d4e6 e79dbdb
Author: tgoodwin <tlg2132@columbia.edu>
    fixed merge conflicts
commit c12d4e62dcd1c23bb25406c43310132edf969da5
Author: tgoodwin <tlg2132@columbia.edu>
```

```
    scanner filling out
commit e79dbdb378caf5898e77c182508e8813a8705c6b
Merge: 5c3221f 091829b
Author: melissaKG <mkaufmangomez@gmail.com>
    Resolving merge conflicts.
commit 5c3221fc30177472712f7e4dd7a869e0a87dd27b
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding function and variable decls. TODO: Add statement decls.
commit 091829b350ece913b08c3933bf4dc2a9df72e323
Author: tgoodwin <tlg2132@columbia.edu>
    removed test
commit 4966934f32e27c42e663b410241f70532f8fe636
Merge: b859968 7edecce
Author: Timothy Goodwin <tlg2132@columbia.edu>
    Merge branch 'master' of github.com:dextercallender/blur
commit b8599680dc16f6e5bd237d5c0cad7dc2fd5115d8
Author: Timothy Goodwin <tlg2132@columbia.edu>
    hey
commit 7edeccef2403c0eeb7bd5c99726bd8e4f5730738
Author: dextercallender <dec2148@columbia.edu>
    scanner.mll modified to include variable names
commit a325abc235b5ae39571f8b490734d9cf471bf61e
Author: melissaKG <mkaufmangomez@gmail.com>
    Cleaning up directory.
commit 8ee21976f4c29338f6de0135bfff5256ed990af7
Author: melissaKG <mkaufmangomez@gmail.com>
    Compiles without errors.
commit 87046761de16fc3c06f05cc997a8a5cae25c2903
Author: melissaKG <mkaufmangomez@gmail.com>
    This compiles! With reduce conflicts.
commit 559aacfa84c5c2dd67afeb2df8b57cc7376738cc
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding scanner.mll.
commit c26b2672f5c3617ba83645745135fe92f3c4c16e
Author: melissaKG <mkaufmangomez@gmail.com>
    Adding a parser file.
commit 216a657c70bc15e547944534e752cb35f79077df
Author: dexter_callender_iii <dec2148@columbia.edu>
    Update README.md
commit e25bc8d9ffc7f67d77a28a0d098f18460142f146
Author: Tim Goodwin <tlg2132@columbia.edu>
    calculator
commit d22c4bc27b7558e8a44942ea60dd8908c361b6c2
Author: melissaKG <mkaufmangomez@gmail.com>
    Testing vm.
commit 09b1c2b884ce50ec0614a421b759f8c2532d5150
Author: Tim Goodwin <tlg2132@columbia.edu>
    problem5 output
commit 756435c85bdce34e5acde39d4840e23acf7aacdc
Author: Timothy Goodwin <tlg2132@columbia.edu>
    hey
commit 748df9180abb5fd17e357312458f71a964734adf
Author: Melissa Kaufman-Gomez <mhk2149@columbia.edu>
```

```
    Testing.
commit affc306312bf428c5d9268a995873164fdbd1b87
Author: dexter_callender_iii <dec2148@columbia.edu>
    Update README.md
commit 049efc587089eff6c5d9bb4cd86c265446710ea5
Author: dexter_callender_iii <dec2148@columbia.edu>
    Initial commit
```