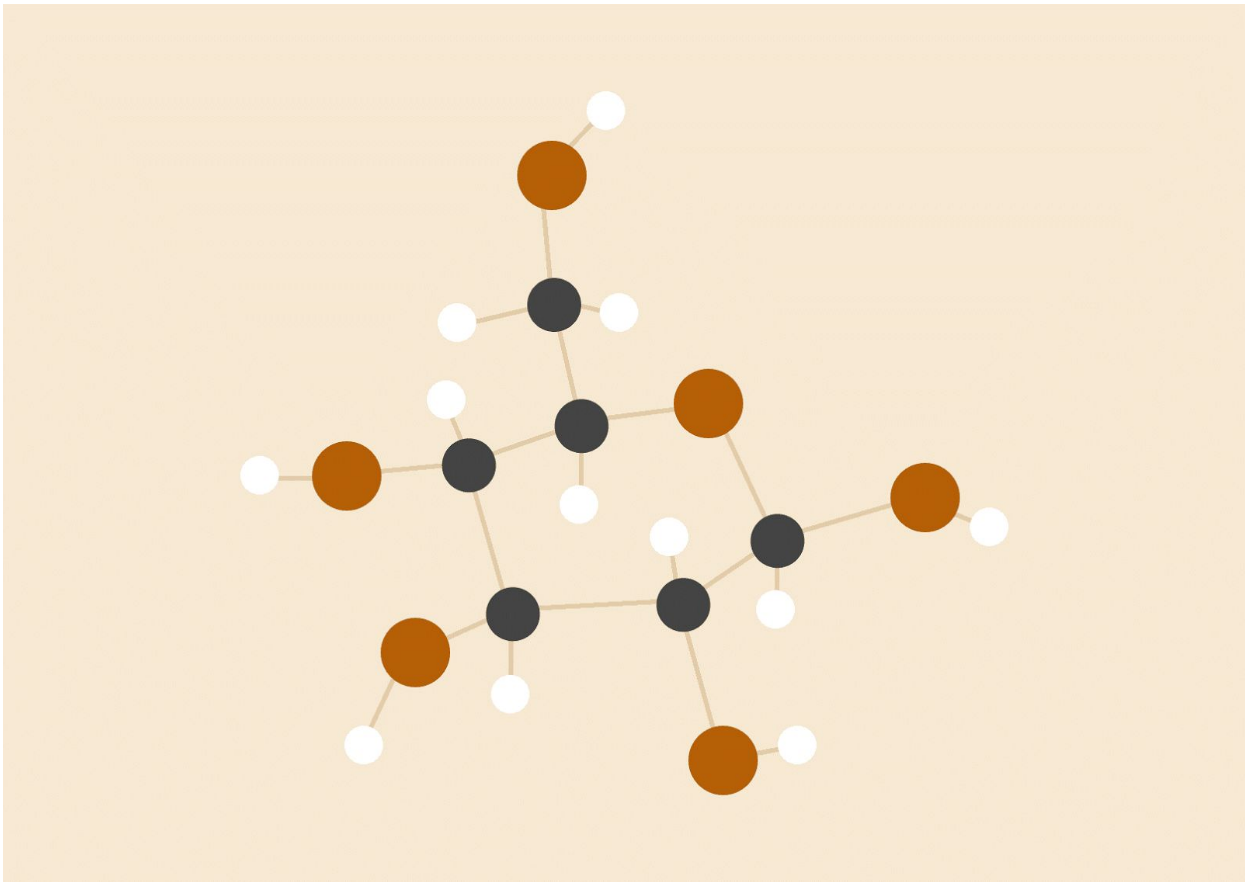
WARHOL



Martina Atabong | maa2247
Charvinia Neblett | cdn2118
Samuel Nnodim | son2105
Catherine Wes | ciw2109
Sarina Xie | sx2166

05.10.2016

Programming Languages and Translators

1. Introduction

Warhol is a functional and imperative programming computer language based on both C and Matlab. Warhol's types, syntax, and semantics are meant to help the user easily manipulate images. Images are uploaded from files and translated into our main primitive type, matrices. Our primitive types are designed to store pixel data from each image. Warhol allows frequent writes, reads, and computes of pixels. Built-in functions are provided to compute commonly used image algorithms while also giving users freedom to implement functions.

The Warhol Library contains methods to facilitate declarative programming. Library functions provide standard implementations of image editing tasks that can be performed on matrix types.

2. Language Tutorial

2.1 Setup

-Install OCaml using the OPAM(OCaml Package Manager).
-Install GIMP.

2.2 Using the Compiler

Commands to filter an image:

1. Inside the WARHOL project folder, type "`make`".
2. To run your own WARHOL program, type the following in the command line:
```
./warhol.native < filename.wl > filename.ll
```
3. Next, convert your .ll file into a .ppm file by executing the following command:
```
lli filename.ll > filename.ppm
```
4. You should now have a '`filename.ppm`' file in your directory. Open up this file through GIMP and your image will be filtered.

Commands to run other .wl files:

1. Inside the WARHOL project folder, type "`make`".
2. To run your own WARHOL program, type the following in the command line:
```
./warhol.native < filename.wl > filename.ll
```
3. Next, run your program by executing the following command:
```
lli filename.ll
```

Command to run .wl files using the built-in or stdlib functions:

1. Inside the WARHOL project folder, type "`make`".
2. Type the following in the command:
```
./run.sh filename.wl
```
3. Next, run the executable:
```
./filename.exe
```

2.3 Hello World

Since Warhol's main purpose is to manipulate integer arrays, *hello.wl* prints the integer 42.

hello.wl

```
fun int main(){

print(42);

return 0;


      }
```

2.4 Sample Program

Every program must include a main function. Within the main functions, users can define variables, call built-in and use defined functions, utilize control and make I/O calls. *filtercyan.wl* is a function that:

- Initializes an integer array by specifying the array type, the size of the array and the array name.
- Variables initialization cannot occur at the same time as variable declaration.
- Openfile is a built-in function that reads a file one number at a time and returns the integer read.
- Arrays can be accessed with [] and an integer index.
- cyan() is a library function that gives an image a cyan overcast.
- printppm() stores the new image in a output ppm file.

```
fun int main(){
   int[22188] marilyn;
   int val;
   int i;
   i = 0;
   while ((val=openfile("marilyn.ppm"))!= -1 ) {
       marilyn[i] = val;
       i = i + 1;
   }

     cyan(@marilyn, 22188);
```

```
    printppm(@marilyn, 22188, 86, 86);
}
```

3. Language Manual

3.1 Lexical Conventions

WARHOL follows consistent, syntax among all types and declarations as well as usage of functions. It aims to make the language predictable and clear and scalable to more functionalities. There will be no variables of the same names, no possible use of keywords, or multi-use of characters.

3.2 Comments

Comments are denoted by opening and closing dollar signs. They are always in the form of a block comment and can encompass extensive lines but an open comment cannot encompass more block comments.

- Example 1: $ this is my code $
- Example 2: $ this is also my code
  $ and this as well $

3.3 Identifiers

Identifiers are a sequence of at least one character and a number. The first character of an identifier must be alphabetic. Identifiers are case sensitive and cannot begin with an uppercase letter. Longest string search is used to determine if an identifier is identical to a key word. Tokens consist of identifiers, constants, separators, keywords, strings, and expression operators. Comments and tabs are completely ignored. Blanks are new lines are used to separate tokens.

3.1.2 Identifiers

Keyword	**Usage**
`int`	Define an integer type
`bool`	Define a boolean type
`void`	Denote a void type
`if/else`	Define a conditional
`for`	Define an iterative loop
`while`	Define a conditional loop
`fun`	Define a function

`char`	Define a character
`string`	Define a string data type

3.2 Types

Integers

- Integers constants are sequence of at least one digit.

Characters

- Character constants are one character enclosed with single quotes. Two character character constants are allowed only if the first character is a backslash. Characters are treated as integers.
  - Example: `'\n'`, `'a'`, `'\t'`, `'0'`

Strings

- Strings are implemented as a matrix of characters. The matrix is always the exact length of the string. Termination of string is determined by bounds of the matrix containing a string's characters.

Boolean

- Booleans are declared using keyword `bool`. Booleans are stored and equivalent to integers equal to 0 for false and greater than 0 for true.

Array

- Holds a 1-dimensional list of values all of the same consistent type. The variable points to a place in memory allocated to this list.
- Accessing an element: the array name points to the first places. Adding and subtracting takes traverses the list (i.e `arr++`).
- Referencing: `@arr`, gives the point in location the array resides
- Dereferencing: &arr, returns the value of the element at the current memory location

3.3 Expressions

3.3.1 Primary Expressions

Primary expressions are literals and names. Primary expressions involving subscripting and function calls group left to right.

- An identifier is a primary expression. Its type is specified by its declaration (ex: matrix, function)
- A decimal or character constant is a primary expression. Its type is int.
- A string is a primary expression. Its type is matrix of chars.
- (*expression)*
  A parenthesized expression is a primary expression. Its type and value are identical to the expression inside the parentheses.
- *primary-expression [expression]*
  A primary expression followed by an expression in square brackets is a subscript and is a primary expression.
- *primary-expression (expression-list* (optional) *)*

A function call is a primary expression that is immediately followed by parentheses containing the list of its arguments. It is of type "function returning ..." and the result of the function call is of type "...". Any arguments of type char are converted to type int before the call. Parameters are passed by reference.

3.3.2 Assignment Operator

Value = expression

- The value of the expression replaces that of the object referred to by the value.

3.3.3 Relational Operators

The operators < , > , <=, and >= all yield 0 if the specified relation is false, and 1 if it is true.

- Is-less-than
  - *expression_a < expression_b*
- Is-greater-than
  - *expression_a > expression_b*
- Is-less-than-or-equal-to
  - *expression_a <= expression_b*
- Is-greater-than-or-equal-to
  - *expression_a >= expression_b*

3.3.4 Arithmetic Operators

If both operands are int or char, the result is int. If one is int or char and the other is mat, then the result is mat. If both operands are mat, then the result is mat. No other combinations are allowed.

- Addition
  - *expression_a + expression_b*
    - The result is the sum of the expressions.

- Subtraction
  - *expression_a - expression_b*
    - The result is the difference of the expressions.

- Division
  - *expression_a / expression_b*
    - The binary / operator signifies division.

- Multiplication
  - *expression_a * expression_b*
    - The binary * operator signifies multiplication.
- - *expression*
  - The result is the negative of the expression. This operator is only applicable to ints and chars.

3.3.5 Logical Operators

- *! expression*
    - This negation operator returns 1 if the value of the expression is 0, 0 if the value of the expression is non-zero. The type of the result is bool. This operator is only applicable to bools.
- *expression && expression*
    - The && ("and") operator returns 1 if both its operands are non-zero. Otherwise, it returns 0.
- *expression || expression*
    - The || ("or") operator returns 1 if either of its operands is non-zero, and 0 otherwise.
- *expression == expression*
    - The == ("equal to") operator returns 1 if both its operands are non-zero. Otherwise, it returns 0.
- *expression != expression*
    - The != ("not equal to") operator returns 1 if both its operands are non-zero. Otherwise, it returns 0.

3.4 Declaration

Declarations in WARHOL are used to establish particular types. The variable must be declared before it can be assigned a value, and the statements must be on separate lines;

3.4.1 Integers, Booleans, Characters, and Strings

Primary types are declared by the delineating type followed by the variable name, the assignment operator, then the respective value. Integer goes with int, booleans use bool, characters use char and strings use string. Characters and string must have their value encompassed by '' and "", respectively.

Format: *`typ variableName = value`* *(i.e. int val = 4);*

3.4.2 Arrays

Arrays are declared and immediately allocated the necessary space so there must be a determined list length. The declaration also must include type in the format of type[size] variable name

3.5 Control Flow

3.5.1 If/Else

These clauses work as a control flow that are incorporated with a boolean operation to decide what to follow. Else-if is an extension of this method.

Ex:
```
int x;
x = -3;
if (x>4) {
      prints("good");
} else if (x> -4) {
      prints("bad");
} else {
      prints("ugly");
}
```

3.5.2 For-loop

This loop is comprised of three statements, with two being optional. It's a pre-test loop similar to C that is declared by *for*, followed by a possible assignment of a value, an expression that must resolve to a boolean, and then a possible increment. There must still be three semi-colons to separate the three components.

All arguments:
```
int x;
for (x=2;x<4; x++){
     prints("just counting");
}
```

With some arguments:
```
int x;
x=2;
for (;x<4;){
     prints("just counting");
    x++;
}
```

3.5.3 While-loop
While serves as the basis for as a continuous loop that depends on one expression that must resolve to a boolean.

Ex:
```
while (true) {
     prints("Infinite loop!");
}
```

3.6 Conversions
3.6.1 Characters and Integers
Certain operators perform implicit conversions from one type to another such as characters. Characters can be used the same as integers.

3.6.2 Booleans and Integers
When converting to a string, true takes on the value of 1 and false takes on the value of 0

3.7 Statements
3.7.1 Expressions
In Warhol there is one expression statement max, per semicolon.

i.e. form *expression ;*

3.7.2 Compound Statement
In Warhol, a compound statement consists of a list of statements where each statement ends in a semicolon.

i.e.: *expression1 ; expression2 ; expression3 ;*

3.7.3 Conditional Statement
In Warhol, a conditional statement consists of a conditional keyword, such as **if**, followed by an expression, **in parentheses** and an expression statement.

i.e.: ***keyword*** *(expression) statement;*

3.7.4 Return

The **return** statement returns the result of a function to the caller of the function.

i.e. return;

3.8 Functions

3.8.1 Declaration

```
     fun return_type nameOfFunction (Declarator1, Declarator2, …) {
          <statement1>
          <statement 2>
          …
          return result;
     }
```

3.8.2 Parameters

Default scope for any variable is local. If declared in the parameters of a function, the scope is the length of the function. All parameters are passed by value, leaving the original object passed into the function unchanged.

3.8.3 Built-in Functions

Reading a File:

`int Openfile(const char *filename)`

Takes in a image file format and returns the first token in the file then removes the token from the file. Running openfile for the length of a file returns a stream of integers in the file and leaves the original file empty.

Input
- String: name of the image file

Return Type
- Integer: returns the first integer in the file specified by filename

Printing:

`void print(int value)`

Prints the value of an int then goes to the next line.

`void printb(boolean value)`

Prints the value of a boolean. True is converted to 1 and false is converted to 0 for printing.

`void prints(string value)`

3.8.4 Standard Library

`Void printppm(int[] arr, int size, int width, int height)`

Prints the header for a ppm file as well as well outputs all integer values in the *arr* as a stream of integers.

-Input

- Integer array, arr:
- Mat/image - the matrix or image to be concatenated
- Direction - enumeration type of *LEFT, BOTTOM, RIGHT, TOP,* to choose where to concatenate

-Return Type

- void - prints out to file

`Filter(int[] image, int size)`

Takes an image and filters the image to specified color temperature. using a convolution matrix

-Input:

- 1-Dimensional Integer Array, image : image is the image to be filtered containing pixel values
- Integer, size: size is the length of the image

-Return Type:

- Void: filter prints the filtered pixels values. These values are piped from the executable into a file where the new image can be viewed.

3.9 Scoping

WARHOL uses static scoping. The life of the variable starts at declaration and terminates with the block in which it is declared. For values to reach other functions they must be passed.

3.10 Standard Library

```
fun void blue(int[] arr, int sz){

        int i;

        for(i = 0; i < sz - 3; i = i + 3){

                &arr=0;

                arr = ++arr;

                &arr=0;

                arr = ++arr;

                arr = ++arr;

        }


}

fun void cyan(int[] arr, int sz){

        int i;
        for(i = 0; i < sz - 3; i = i + 3){
                arr = ++arr;
                arr = ++arr;
                arr = ++arr;
                &arr=0;
        }


}
fun void green(int[] arr, int sz){
        int i;
        for(i = 0; i < sz - 3; i = i + 3){
                &arr=0;
                arr = ++arr;
                arr = ++arr;
                &arr=0;
                arr = ++arr;
        }
}
fun void pink(int[] arr, int sz){
        int i;
        for(i = 0; i < sz - 3; i = i + 3){
                arr = ++arr;
                &arr=0;
                arr = ++arr;
                arr = ++arr;
        }
}
fun void red(int[] arr, int sz){
        int i;
        for(i = 0; i < sz - 3; i = i + 3){
                arr = ++arr;
                &arr=0;
                arr = ++arr;
                &arr=0;
                arr = ++arr;
        }
}
fun void yellow(int[] arr, int sz){
        int i;
        for(i = 0; i < sz - 3; i = i + 3){
                arr = ++arr;
                arr = ++arr;
                &arr=0;
                arr = ++arr;
        }
}
fun void printppm(int[] arr, int sz, int width, int height) {
        prints("P3\n");
        print(width);
        print(height);
        print(255);
        int i;
        int[] ptr;
        ptr = arr;
        for (i = 0; i < sz; i = i + 1) {
                print(&ptr);
                ptr = ++ptr;
        }
}
```

4. Project Plan

4.1 Planning Process
To begin the project, we first identified team goals and individual responsibilities for the entire semester. We planned to meet once a week after class to discuss individual and team progress. We started the semester off strong by meeting once a week to discuss the specifications of our language, learn OCaml, and get our programming environment setup. We fell behind on the Hello World check mark, but with the help of our TA (Julie), we got back on track for the second half of the semester. We met much more frequently and made progress consistently each week.

4.2 Specifications
The beginning of the semester we wrote our Language Reference Manual (LRM) to guide our development process. During the development process, we added new specifications and deleted old specifications as our language grew and evolved. We updated our LRM to meet these changes accordingly, and to ensure that all members of the group were on the same page.

4.3 Development Environment
Each member of the team used Sublime and Vim through the Ubuntu VirtualBox. This ensured that we all were developing in the same environment. We used Github for our repository.

4.4 Timeline
Multiple meetings were spent brainstorming possible projects, speaking to TA's about what projects are doable and substantial. After deciding upon Warhol, the next brainstorm consisted of developing ideas and a standard for lexical conventions. Once there was a goal language, the project was prioritized into steps. The first checkpoint was HelloWorld. Because of the original lack of interest in string, there was a focus on developing the integer type, thus, the first program printed the number 42. This moment signified the first working program and a shell, in which to build the rest of WARHOL upon. The next step was to provide other types including boolean, char, and string. Type definitions, along with operations, gave us the capability to do control flow and have a basic program working. The biggest milestone for WARHOL was the proper development of arrays, along with pointers. Arrays serve as the basic functionality in which all functionalities are built upon. Once arrays were complete, the next steps involved developing a standard library, in turn proving our program worked as expected. And lastly, input/output streams were the last step so that WARHOL no longer had to be self-contained.

4.5 Style Guide
While each of us were developing, we tried our best to follow the same programming style guidelines:

- When indenting lines, use tabs.
- Use descriptive variable names.
- Comment code to make it easier for other group members to quickly understand.

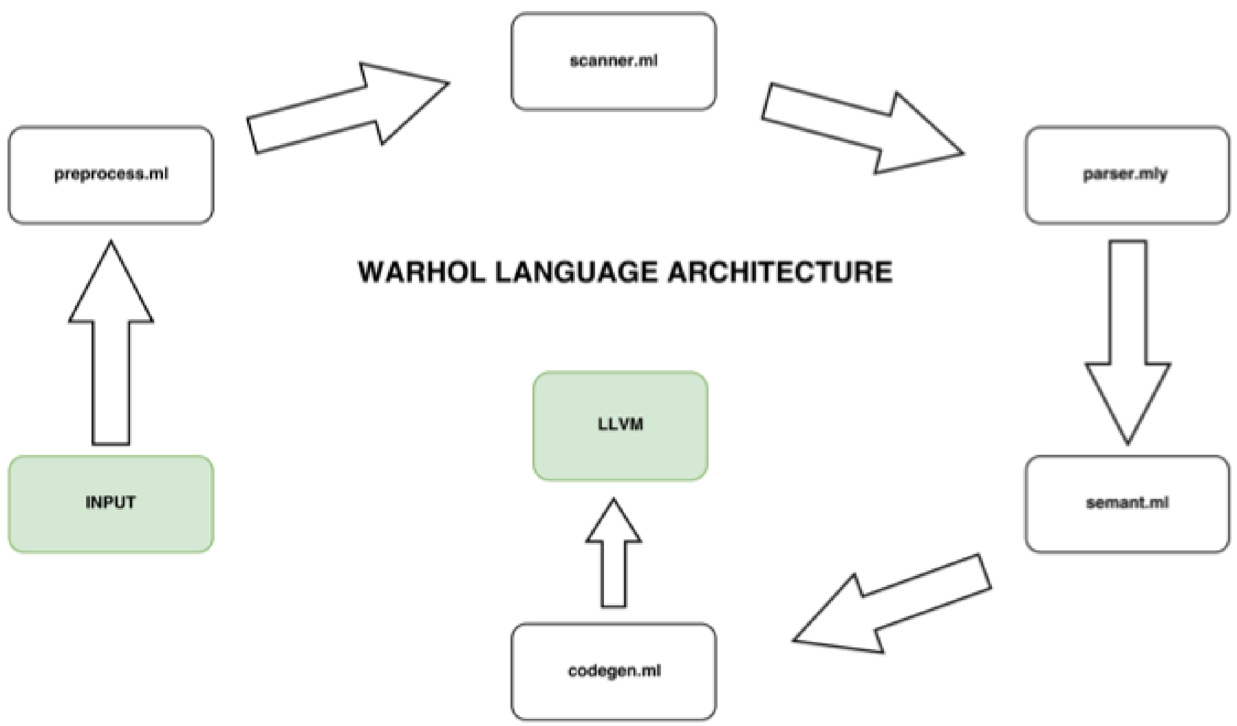
4.6 Roles and Responsibilities
Martina Atabong - *Manager*
Charvinia Neblett - *Language Guru*

Sarnia Xie - *Compiler Architect*
Catherine Wes - *Tester*
Samuel Nnodim - *Development Environment*

4.7 Software Development

OCaml: language we used to implement the compiler
Github: for version control
Gimp: to open our .ppm files
Google Docs: for all written work (LRM, Final Report)
Sublime, Vim: our chosen text editor
Ubuntu on Virtual Box: operating system

5. Architecture



5.1 Source Code

The source file is concatenated with .wl and syntax should adhere to the Language Reference Manual.

5.2 Pre-Processor

The preprocessor is used to link built-in library functions. If “include stdlib” is in the header of the source file, then then the built-in functions are inserted in the this position.

5.3 Scanner

The scanner performs lexical analysis on the source code. The source file is read in linear stream of characters. Characters are grouped into tokens based on a defined set of regular expression rules. Any character not defined in the scanner is therefore not recognized by the scanner, and an error will be thrown.

5.4 Parser

The parser takes in the stream of tokens outputted by the scanner. Tokens are matched against a set of rules set in the parser. The rules within the parser define a Context Free Grammar and determine how tokens should be grouped. If all tokens are matched into the rules set in the parser then the source code is syntactically correct. Based on groups of matched tokens, the parser outputs values stored with tokens in a specific format. The values, types of values, and format of the values outputted are determined by the Abstract Syntax Tree.

5.5 Ast

The Abstract Syntax Tree is a data structure that defines the program. The AST specifies the types of values that can be stored in token based on how each tokens are matched into grouping rules. The AST provides a specific structure for how to store matched tokens so that data within each structure can be accessed easily in the next architecture component. Rules set in the AST illustrates all possible types of programs that can be created while the source code program is a subset within the AST. We can equate the AST to a class in Object Oriented Programming and the source code program as an object of the class. The object of the AST is passed to the following phases within the Architecture. If we know the methods and instance variables of the object, data can be easily accessed from the object just as program information can be easily accessed by subsequent steps.

5.6 Semantic Checking

Semantic checking is an analyzer that takes in an Abstract Syntax Tree and provides an additional layer of checking after source code has been scanned and parsed. This includes evaluating if all values stored in variables match the variable types. Variables are checked for multiple declarations. Function and variable names are checked for being identical to keywords within the language. The analyzer returns a semantically checked AST.

5.7 Code Generation

Code Generation translates the source code program into Low Level Virtual Machine IR code. Code Generation uses the semantically checked Abstract Syntax Tree to access data types and values from the source code and builds semantically equivalent code in llvm. In codegen, built-in functions that are written in other languages outside our language are linked. Meaning, the type of the external function is defined so that the compiler knows that this function is external built-in function. The code generated to LLVM is then compiled using the LLVM compiler and an executable is produced.

6. Test Plan

6.1 Test Suite

a. `*tbrack.wl*` → Tests that you can use an expression <u>instead</u> of an int literal when accessing an array. This is necessary because it is vital in the way in which we step through the image representation held in matrix.

   i. Source Code

```
fun int main() {
       i=0;

       int[3] arr;

       arr[i+1] = 1;

       print(arr[1]);

}
```

   ii. Test Command

```
make clean && make
./warhol.native -c < tests/tbrack.wl > tbrack.ll
lli tbrack.ll
```

   iii. Output

```
plt@ubuntu-plt:~/work/warhol$ lli tbrack.ll
1
```

   iv. LLVM IR

```
; ModuleID = 'Warhol'

@fmt = private unnamed_addr constant [4 x i8] c"%d\0A\00"


declare i32 @printf(i8*, ...)


declare i32 @openfile(i8*, ...)
```

```
define i32 @main() {
      entry:
      %i = alloca i32
      %arr = alloca [3 x i32]
      store i32 0, i32* %i
      %i1 = load i32* %i
      %tmp = add i32 %i1, 1
      %arr2 = getelementptr [3 x i32]* %arr, i32 0, i32 %tmp
      store i32 1, i32* %arr2
      %arr3 = getelementptr [3 x i32]* %arr, i32 0, i32 1
      %arr4 = load i32* %arr3
     %printf = call i32 (i8*, ...)* @printf(i8* getelementptr inbounds ([4 x
i8]* @fmt, i32 0, i32 0), i32           %arr4)
      ret i32 0
}
```

b. `*tfun.wl*` → Checks that functions can be called in main. This is key because functions are the cornerstone of our language and being able to reuse functions makes for a bettr code experience

   i. Source Code

```
fun int prt3() {
  print(3);
  return 0;
}
fun int main() {
  prt3();
  return 0;
}
```

   ii. Test Command

```
make & make clean
./warhol.native -c < tests/tfun.wl > tfun.ll
```

```
lli tfun.ll
```

iii. Output

```
plt@ubuntu-plt:~/work/warhol$ lli tfun.ll
3
```

iv. LLVM IR

```
; ModuleID = 'Warhol'
@fmt = private unnamed_addr constant [4 x i8] c"%d\0A\00"
@fmt1 = private unnamed_addr constant [4 x i8] c"%d\0A\00"



declare i32 @printf(i8*, ...)



declare i32 @openfile(i8*, ...)



define i32 @main() {
      entry:
      %prt3_result = call i32 @prt3()
       ret i32 0
}



define i32 @prt3() {
   entry:
   %printf = call i32 (i8*, ...)* @printf(i8* getelementptr inbounds ([4 x
i8]* @fmt1, i32 0, i32 0), i32     3)
    ret i32 0
}
```

c. `*tppp.wl*` → Checks that pointers can be assigned and can be incremented. This is essential because the backbone of WARHOL is pointer

i. Source Code

```
fun int main() {
      int[2] k;
      k[0] = 0;

      k[1] = 1;

      int[] kp;

      kp = @k;

      kp = ++kp;

      print(&kp);
}
             }
```

ii. Test Suite

```
make clean && make
./warhol.native -c  < test/tppp.wl > tppp.ll
lli tppp.ll
```

iii. Output

```
plt@ubuntu-plt:~/work/warhol$ lli tppp.ll
1
```

iv. LLVM IR

```
; ModuleID = 'Warhol'

@fmt = private unnamed_addr constant [4 x i8] c"%d\0A\00"



declare i32 @printf(i8*, ...)



declare i32 @openfile(i8*, ...)
```

```
define i32 @main() {

      entry:

      %k = alloca [2 x i32]

      %kp = alloca i32*

      %k1 = getelementptr [2 x i32]* %k, i32 0, i32 0

      store i32 0, i32* %k1

      %k2 = getelementptr [2 x i32]* %k, i32 0, i32 1

      store i32 1, i32* %k2

      %k3 = getelementptr inbounds [2 x i32]* %k, i32 0, i32 0

      store i32* %k3, i32** %kp

      %kp4 = getelementptr inbounds i32** %kp, i32 0

      %kp5 = load i32** %kp4

      %kp6 = getelementptr inbounds i32* %kp5, i32 1

      store i32* %kp6, i32** %kp

      %kp7 = load i32** %kp

      %kp8 = load i32* %kp7

      %printf = call i32 (i8*, ...)* @printf(i8* getelementptr inbounds ([4 x
i8]* @fmt, i32 0, i32 0), i32 %kp8)
      ret i32 0

}
```

7. Lessons Learned

7.1 Catherine

I thought one of the most difficult aspects of this project was learning how to properly delegate work. Because many parts of the project depend on the other working parts, I found that we didn’t utilize/delegate our time as efficiently as we could have. If you find yourself waiting for another member to finish something, look ahead and prepare for the next steps. Additionally, because no one in our group was familiar with OCaml and the technology was initially foreign to us, I found that pair programming was especially helpful to work through the more difficult/tricky parts of the project. Having someone to bounce ideas off of is very valuable. Additionally, make sure that every week you accomplish at least one goal, big or small.

7.2 Charvinia

If you have never seen OCAML before nor have indulged in the ways of compilers, everything will seem like a huge web. The best way to start sorting through all the new concepts thrown at you is to sit down and learn ocaml. You will not have homework for the first few weeks, so give yourself ocaml programming assignments. Next, if you know your focus game during lectures is already shaky, watch the lectures online. I found this incredibly helpful because I could rewind and fastforward where needed. Understand how theory links to application. Making this connection helped make coding a lot easier. Lastly, know yourself and know your group. Know how each person in your group operates. Some people will code on their own time and some people will only code when they are with your group. If you know someone needs a push, make sure to keep them accountable.

7.3 Samuel

Make sure everyone's environment is setup from Day 0. This means making sure the group has knowledge of git, and more importantly knows git flows necessary to be a contributing member. Also, meet with frequent regularity. This is perhaps the most vital point because work does not happen by itself. Without meeting it is easy to lose sight of the larger plan as well as the cohesion group work requires. Lastly, constant, clear communication is the glue that holds a group together, meeting outside of class, in class, and during class are all things that any group project requires.

7.4 Martina

It's very easy to lose sight of goals, even if you start of strong. Task allocation should be dynamic and communication is so important in order to keep a group balanced and productive. Also, priorities change a lot so it's important to take a second and count your eggs.

7.5 Sarina

Look at Clang for learning llvm.

7.6 General Advice to Future Groups

Know your group members and how they operate very well. This will help you set expectations and strategies for reaching those expectations. Make a weekly schedule and stick to it. Even making small progress weekly makes a big difference. Learn Ocaml from day one. The first few lectures are introductory and a good time to start learning Ocaml on your own. Lastly, do not be afraid to be bestfriends with your TA. They know a lot and can save you time early on if you are having difficulties understanding the compilation process

8. Appendix

8.1 Prep.c

```
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[]) {
        char checkbuf[15];
        size_t checkcount = 15;
        FILE *fp;
        //fp = fopen(argv[1], "r");
        fp = fopen(argv[1], "r");
        fread(checkbuf, 15, 1, fp);
        //for tempfile
        FILE *tempfp;
        tempfp = fopen("tempfile.wl", "wb");
        //check if .wl file starts with import statement
        if (strstr("$import stdlib$", checkbuf)) {
                FILE *libfp;
                libfp = fopen("stdlib.wl", "r");
                //copy contents of .wl file and stdlib to tempfile.wl
                char buf[2048];
        size_t n;
        fwrite(checkbuf, checkcount, 1, tempfp);
                while ((n = fread(buf, 1, sizeof(buf), fp)) > 0) {
                        fwrite(buf, 1, n, tempfp);
                }
                while ((n = fread(buf, 1, sizeof(buf), libfp)) > 0) {
                        fwrite(buf, 1, n, tempfp);
                }
                fclose(libfp);
        }
        else {
```

```
                        //copy contents of .wl file to tempfile.wl
                        char buf[2048];
            size_t n;
            fwrite(checkbuf, checkcount, 1, tempfp);
                        while ((n = fread(buf, 1, sizeof(buf), fp)) > 0) {
                                    fwrite(buf, 1, n, tempfp);
                        }
            }
            fclose(fp);
            fclose(tempfp);
  }
```

8.2 Scanner.mll

```
{ open Parser
      let unescape s =
              Scanf.sscanf ("\"" ^ s ^ "\"") "%S%!" (fun x -> x)

  }
  let digits = ['0'-'9']
  let ascii = ([' '-'!' '#'-'[' ']'-'~'])
  let esc = '\\' ['\\' ''' '"' 'n' 'r' 't']
  let string = '"' ( (ascii | esc)* as s ) '"'
  let char = ''' ( ascii | digits ) '''

  rule token = parse
  [' '  '\t' '\r' '\n'] { token lexbuf } (* Whitespace *)
  | "$" { comment lexbuf } (* Comments *)

  (* Identifiers *)
  | '(' { LPAREN }
  | ')' { RPAREN }
  | '{' { LBRACE }
| '}' { RBRACE }

| '[' { LSQUARE }

| ']' { RSQUARE }

| ',' { COMMA }

| ';' { SEMI }

| "fun" { FUN }

| "return" { RETURN }

| "int" { INT }

| '@' { STAR }

| '&' { AMPERSAND }

| '+' { PLUS }

| '-' { MINUS }

| '*' { TIMES }

| '/' { DIVIDE }

| '=' { ASSIGN }

| "=="     { EQ }

| "!="     { NEQ }

| '<'      { LT }

| "<="     { LEQ }

| ">"      { GT }

| ">="     { GEQ }

| "!"      { NOT }

| "&&"     { AND }

| "||"     { OR }

| "if"     { IF }

| "else"   { ELSE }

| "while"  { WHILE }

| "for"    { FOR }

| "bool"   { BOOL }

| "true"   { TRUE }
| "false"  { FALSE }

| "void"   { VOID }

| ['0'-'9']+ as lxm { INTLITERAL(int_of_string lxm) }

| ['a'-'z' 'A'-'Z']['a'-'z' 'A'-'Z' '0'-'9' '_']* as lxm { ID(lxm) }

| string { STRINGLITERAL(unescape s) }

|

char    as lxm { CHARLITERAL(String.get lxm 1) }

| eof { EOF }

and comment = parse

"$" { token lexbuf }

| _ { comment lexbuf }
```

8.3: Parser.mly

```
%{ open Ast %}

/*Delimiters*/

%token LPAREN RPAREN LBRACE RBRACE LSQUARE RSQUARE COMMA SEMI

/* Operators */

%token PLUS MINUS TIMES DIVIDE NOT AND OR

%token EQ NEQ LT LEQ GT GEQ

%token STAR AMPERSAND

/*Types*/

%token INT VOID BOOL CHAR

/*Literals*/
```

```
%token <int> INTLITERAL

%token <string> STRINGLITERAL

%token <string> ID

%token <char> CHARLITERAL

%token TRUE FALSE

/*Control flow*/

%token IF ELSE NOELSE WHILE FOR

/*Misc*/

%token ASSIGN

%token FUN RETURN

%token EOF

%nonassoc NOELSE

%nonassoc ELSE

%nonassoc NOLSQUARE

%nonassoc LSQUARE

%right ASSIGN

%left OR

%left AND

%left EQ NEQ
```

```
%left LT GT LEQ GEQ

%left PLUS MINUS

%left TIMES DIVIDE

%right NOT NEG

%start program

%type <Ast.program> program

%%

program:

 decls EOF {$1}

decls:

 /*nothing*/ {[],[]}

| decls topvdecl {($2 :: fst $1), snd $1}

| decls fdecl {fst $1, ($2 :: snd $1)}

/*these two vdecl parts are for being able to mix vdecls and fdecls at the top level (outside main)*/

topvdecl_list:

 /*nothing*/ {[]}

| topvdecl_list topvdecl {$2 :: $1}

topvdecl:

 typ ID SEMI {$1, $2}
```

```
fdecl:

 FUN typ ID LPAREN formals_opt RPAREN LBRACE fbody RBRACE

 { {return_type = $2; fname = $3; formals = $5; fbody = $8 }} /*copied from slides*/

/*depending on whether the next line is a vdecl or a stmt, add it to the correct list: f_vdecl or f_stmts*/

fbody:

 /* nothing */ { { f_vdecls = []; f_stmts = []; } }

| fbody vdecl { { f_vdecls = $1.f_vdecls @ [$2]; f_stmts = $1.f_stmts; } }

| fbody stmt { { f_vdecls = $1.f_vdecls; f_stmts = $1.f_stmts @ [$2]; } }

formals_opt:

 /*nothing*/ {[]}

| formal_list {List.rev $1}

formal_list:

 typ ID  { [$1,$2] }

| formal_list COMMA typ ID { ($3,$4) :: $1 }



typ:

 INT {Int}

| BOOL {Bool}

| CHAR { Char }
| VOID { Void }

| inttyp LSQUARE INTLITERAL RSQUARE { Matrix1DType($1, $3) }

| inttyp LSQUARE RSQUARE { Matrix1DPointer($1) }



inttyp:

 INT {Int}

stmt_list:

 /*nothing*/ {[]}

| stmt_list stmt {$2 :: $1}




vdecl:

 typ ID SEMI {$1, $2}

stmt:

| expr SEMI {Expr $1}

| RETURN SEMI {Return Noexpr}

| RETURN expr SEMI {Return $2}

| LBRACE stmt_list RBRACE {Block(List.rev $2)}

| IF LPAREN expr RPAREN stmt %prec NOELSE { If($3, $5, Block([])) }

| IF LPAREN expr RPAREN stmt ELSE stmt    { If($3, $5, $7) }
| WHILE LPAREN expr RPAREN stmt { While($3, $5) }

| FOR LPAREN expr_opt SEMI expr SEMI expr_opt RPAREN stmt

  { For($3, $5, $7, $9) }

/*s--how does the list work*/

matrix_literal:

   expr { [$1] }

 | matrix_literal COMMA expr { $3 :: $1 }



expr_opt:

   /* nothing */ { Noexpr }

 | expr         { $1 }




expr:

 INTLITERAL {IntLiteral($1)}

| STRINGLITERAL { StringLiteral($1) }

| CHARLITERAL     { CharLiteral($1) }

| TRUE            { BoolLit(true) }

| FALSE           { BoolLit(false) }

| expr ASSIGN expr {Assign($1, $3)}
| expr PLUS expr  { Binop($1, Add, $3) }

| expr MINUS expr  { Binop($1, Subt, $3) }

| expr TIMES expr  { Binop($1, Mult, $3) }

| expr DIVIDE expr  { Binop($1, Div, $3) }

| expr AND    expr { Binop($1, And,   $3) }

| expr OR     expr { Binop($1, Or,    $3) }

| expr EQ     expr { Binop($1, Eq, $3) }

| expr NEQ    expr { Binop($1, Neq,   $3) }

| expr LT     expr { Binop($1, Less,  $3) }

| expr LEQ    expr { Binop($1, Leq,   $3) }

| expr GT     expr { Binop($1, Greater, $3) }

| expr GEQ    expr { Binop($1, Geq,   $3) }

| MINUS expr %prec NEG { Unop(Neg, $2) }

| NOT expr         { Unop(Not, $2) }

| LPAREN expr RPAREN {$2}

| LSQUARE matrix_literal RSQUARE { MatrixLiteral(List.rev $2) }   /*ie, [1,1,1]*/

| ID LSQUARE expr RSQUARE       { Matrix1DAccess($1, $3)}  /*ie, i[2]*/

| STAR ID                       { Matrix1DReference($2)}

| AMPERSAND ID                  { Dereference($2)}
| PLUS PLUS ID                  { PointerIncrement($3) }

| ID {Id($1)}

| ID LPAREN actuals_opt RPAREN {Call($1, $3)}




actuals_opt:

 /*nothing*/ {[]}

| actuals_list {List.rev $1}




actuals_list:

 expr {[$1]}

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

8.4 Ast.ml

```
type op = Add | Subt | Mult | Div |
                    Eq | Neq | Less | Leq | Greater | Geq |
                    And | Or

type uop = Not | Neg



(*Types*)

type typ = Int | Void | Bool | String | Char

| Matrix1DType of typ * int
| Matrix1DPointer of typ



type bind = typ * string (*bind means the same thing as datatype in CMAT*)



(*Expressions*)

type expr =

 IntLiteral of int

| StringLiteral of string

| CharLiteral of char

| BoolLit of bool

| Id of string

| Noexpr

| Binop of expr * op * expr

| Unop of uop * expr

| Assign of expr * expr

| MatrixLiteral of expr list     (*i.e. 1,1,1, I believe*)

| Matrix1DAccess of string * expr        (*i.e. i[2], I believe*)

| Matrix1DReference of string

| Dereference of string

| PointerIncrement of string

| Call of string * expr list



(*Statements*)

type stmt =

 Block of stmt list

| Expr of expr

| Return of expr

| If of expr * stmt * stmt

| While of expr * stmt
| For of expr * expr * expr * stmt



(*Function declarations*)

type func_body = {

  f_vdecls: bind list;

  f_stmts: stmt list;

 }



(*Function declarations*)

type func_decl = {

   return_type : typ;

   fname : string;

   formals : bind list;

   fbody : func_body;

}



(*Start symbol*)

type program = bind list * func_decl list



(* Pretty-printing functions *)

let string_of_op = function

   Add -> "+"

 | Subt -> "-"

 | Mult -> "*"

 | Div -> "/"

 | Eq -> "=="

 | Neq -> "!="

 | Less -> "<"
 | Leq -> "<="

 | Greater -> ">"

 | Geq -> ">="

 | And -> "&&"

 | Or -> "||"



let string_of_uop = function

   Neg -> "-"

 | Not -> "!"



let string_of_matrix m =

 let rec string_of_matrix_lit = function

     [] -> "]"

   | [hd] -> (match hd with

              IntLiteral(i) -> string_of_int i

            | _ -> raise( Failure("Illegal expression in matrix literal") )) ^ string_of_matrix_lit []

   | hd::tl -> (match hd with

                 IntLiteral(i) -> string_of_int i ^ ", "

               | _ -> raise( Failure("Illegal expression in matrix literal") )) ^ string_of_matrix_lit tl

 in

 "[" ^ string_of_matrix_lit m



let rec string_of_expr = function

   IntLiteral(l) -> string_of_int l

 | CharLiteral(i) ->  String.make 1 i

 | StringLiteral(i) -> i

 | BoolLit(true) -> "true"

 | BoolLit(false) -> "false"

 | Id(s) -> s
 | Binop(e1, o, e2) ->

     string_of_expr e1 ^ " " ^ string_of_op o ^ " " ^ string_of_expr e2

 | PointerIncrement(s) -> "++" ^ s

 | Unop(o, e) -> string_of_uop o ^ string_of_expr e

 | Assign(v, e) -> string_of_expr v ^ " = " ^ string_of_expr e

 | Call(f, el) ->

     f ^ "(" ^ String.concat ", " (List.map string_of_expr el) ^ ")"

 | Noexpr -> ""

 | MatrixLiteral(m) -> string_of_matrix m

 | Matrix1DAccess(s, r1) -> s ^ "[" ^ (string_of_expr r1) ^ "]"

 | Matrix1DReference(s) -> "@" ^ s

 | Dereference(s) -> "&" ^ s



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";

 | Return(expr) -> "return " ^ string_of_expr expr ^ ";\n";

 | If(e, s, Block([])) -> "if (" ^ string_of_expr e ^ ")\n" ^ string_of_stmt s

 | If(e, s1, s2) ->  "if (" ^ string_of_expr e ^ ")\n" ^

     string_of_stmt s1 ^ "else\n" ^ string_of_stmt s2

 | For(e1, e2, e3, s) ->

     "for (" ^ string_of_expr e1  ^ " ; " ^ string_of_expr e2 ^ " ; " ^

     string_of_expr e3  ^ ") " ^ string_of_stmt s

 | While(e, s) -> "while (" ^ string_of_expr e ^ ") " ^ string_of_stmt s



let rec string_of_typ = function

   Int -> "int"

 | Bool -> "bool"
 | Void -> "void"

 | String -> "string"

 | Matrix1DType(t, i1) -> string_of_typ t ^ "[" ^ string_of_int i1 ^ "]"

 | Matrix1DPointer(t) -> string_of_typ t ^ "[]"

let string_of_vdecl (t, id) = string_of_typ t ^ " " ^ id ^ ";\n"

let string_of_fdecl fdecl =

 string_of_typ fdecl.return_type ^ " " ^

 fdecl.fname ^ "(" ^ String.concat ", " (List.map snd fdecl.formals) ^

 ")\n{\n" ^

 String.concat "" (List.map string_of_vdecl fdecl.fbody.f_vdecls) ^

 String.concat "" (List.map string_of_stmt fdecl.fbody.f_stmts) ^

 "}\n"

let string_of_program (vars, funcs) =

 String.concat "" (List.map string_of_vdecl vars) ^ "\n" ^

 String.concat "\n" (List.map string_of_fdecl funcs)
```

8.5 Semant.ml

```
open Ast
module StringMap = Map.Make(String)

(* Semantic checking of a program. Returns void if successful,

  throws an exception if something is wrong.

  Check each global variable, then check each function *)

let check (globals, functions) =

 (* Raise an exception if the given list has a duplicate *)

 let report_duplicate exceptf list =

   let rec helper = function

 n1 :: n2 :: _ when n1 = n2 -> raise (Failure (exceptf n1))
```

```
    | _ :: t -> helper t
    | [] -> ()
  in helper (List.sort compare list)
in


(* Raise an exception if a given binding is to a void type *)
let check_not_void exceptf = function
    (Void, n) -> raise (Failure (exceptf n))
  | _ -> ()
in

(* Raise an exception of the given rvalue type cannot be assigned to
   the given lvalue type *)
let check_assign lvaluet rvaluet err =
   if lvaluet = rvaluet then lvaluet else raise err
in

(**** Checking Global Variables ****)
List.iter (check_not_void (fun n -> "illegal void global " ^ n)) globals;
report_duplicate (fun n -> "duplicate global " ^ n) (List.map snd globals);


(**** Checking Functions ****)


if List.mem "print" (List.map (fun fd -> fd.fname) functions)
then raise (Failure ("function print may not be defined")) else ();

report_duplicate (fun n -> "duplicate function " ^ n)
  (List.map (fun fd -> fd.fname) functions);

(* Function declaration for a named function *)
```

```
let built_in_decls =  StringMap.add "print"
   { return_type = Void; fname = "print"; formals = [(Int, "x")];
     fbody = { f_vdecls = []; f_stmts = [] } } (StringMap.add "printb"
   { return_type = Void; fname = "printb"; formals = [(Bool, "x")];
     fbody = { f_vdecls = []; f_stmts = [] } } (StringMap.add "prints"
   { return_type = Void; fname = "prints"; formals = [(String, "x")];
     fbody = { f_vdecls = []; f_stmts = [] } } (StringMap.add "openfile"
   { return_type =  Int; fname = "openfile"; formals = [(String, "x")];
     fbody = { f_vdecls = []; f_stmts = [] } }(StringMap.singleton "printbig"
   { return_type = Void; fname = "printbig"; formals = [(Int, "x")];
     fbody = { f_vdecls = []; f_stmts = [] } })))
 in
let function_decls = List.fold_left (fun m fd -> StringMap.add fd.fname fd m)
                   built_in_decls functions
in
let function_decl s = try StringMap.find s function_decls
     with Not_found -> raise (Failure ("unrecognized function " ^ s))
in
let _ = function_decl "main" in (* Ensure "main" is defined *)


let check_function func =
  List.iter (check_not_void (fun n -> "illegal void formal " ^ n ^
    " in " ^ func.fname)) func.formals;
  report_duplicate (fun n -> "duplicate formal " ^ n ^ " in " ^ func.fname)
    (List.map snd func.formals);
  List.iter (check_not_void (fun n -> "illegal void local " ^ n ^
    " in " ^ func.fname)) func.fbody.f_vdecls;


  report_duplicate (fun n -> "duplicate local " ^ n ^ " in " ^ func.fname)
```

```
    (List.map snd func.fbody.f_vdecls);

  (* Type of each variable (global, formal, or local *)
  let symbols = List.fold_left (fun m (t, n) -> StringMap.add n t m)
 StringMap.empty (globals @ func.formals @ func.fbody.f_vdecls )
  in

 let type_of_identifier s =
    try StringMap.find s symbols
    with Not_found -> raise (Failure ("undeclared identifier " ^ s))
  in

 let matrix_access_type = function
    Matrix1DType(t, _) -> t
    | _ -> raise (Failure ("illegal matrix access") )
  in

  let check_pointer_type = function
      Matrix1DPointer(t) -> Matrix1DPointer(t)
    | _ -> raise ( Failure ("cannot increment a non-pointer type") )
  in

  let check_matrix1D_pointer_type = function
    Matrix1DType(p, _) -> Matrix1DPointer(p)
    | _ -> raise ( Failure ("cannont reference non-1Dmatrix pointer type"))
  in

  let pointer_type = function
  | Matrix1DPointer(t) -> t
  | _ -> raise ( Failure ("cannot dereference a non-pointer type")) in

  let matrix_type s = match (List.hd s) with
  | IntLiteral _ -> Matrix1DType(Int, List.length s)
  | _ -> raise ( Failure ("Cannot instantiate a matrix of that type")) in
  let rec check_all_matrix_literal m ty idx =
    let length = List.length m in
      match (ty, List.nth m idx) with
    (Matrix1DType(Int, _), IntLiteral _) -> if idx == length - 1 then Matrix1DType(Int, length) else
check_all_matrix_literal m (Matrix1DType(Int, length)) (succ idx)
  | _ -> raise (Failure ("illegal matrix literal"))
 in


  (* Return the type of an expression or throw an exception *)
  let rec expr = function
      IntLiteral _ -> Int
      | BoolLit _ -> Bool
      | Id s -> type_of_identifier s
      | CharLiteral _ -> Char
      | StringLiteral _ -> String
      | PointerIncrement(s) -> check_pointer_type (type_of_identifier s)
      | MatrixLiteral s -> check_all_matrix_literal s (matrix_type s) 0
      | Matrix1DAccess(s, e1) -> let _ = (match (expr e1) with
        Int -> Int
        | _ -> raise (Failure ("attempting to access with a non-integer type"))) in
      matrix_access_type (type_of_identifier s)
      | Dereference(s) -> pointer_type (type_of_identifier s)
      | Matrix1DReference(s) -> check_matrix1D_pointer_type( type_of_identifier s )

    (*  | Len(s) -> (match (type_of_identifier s) with
        Matrix1DType(_, _) -> Int
        | _ -> raise(Failure ("cannot get the length of non-1d-matrix")))
    *)
      | Binop(e1, op, e2) as e -> let t1 = expr e1 and t2 = expr e2 in
    (match op with
      Add | Subt | Mult | Div when t1 = Int && t2 = Int -> Int
      | Eq | Neq when t1 = t2 -> Bool
      | Less | Leq | Greater | Geq when t1 = Int && t2 = Int -> Bool
      | And | Or when t1 = Bool && t2 = Bool -> Bool
      | _ -> raise (Failure ("illegal binary operator " ^
        string_of_typ t1 ^ " " ^ string_of_op op ^ " " ^
        string_of_typ t2 ^ " in " ^ string_of_expr e)))
    | Unop(op, e) as ex -> let t = expr e in
    (match op with
      Neg when t = Int -> Int
      | Not when t = Bool -> Bool
      | _ -> raise (Failure ("illegal unary operator " ^ string_of_uop op ^
        string_of_typ t ^ " in " ^ string_of_expr ex)))
    | Noexpr -> Void
    | Assign(e1, e2) as ex -> let lt = ( match e1 with
      | Matrix1DAccess(s, _) -> (match (type_of_identifier s) with
        Matrix1DType(t, _) -> (match t with
          Int -> Int
          | _ -> raise ( Failure ("illegal matrix of not ints") )
        )
        | _ -> raise ( Failure ("cannot access a primitive") )
      )
    | _ -> expr e1)
    and rt = expr e2 in
    check_assign lt rt (Failure ("Illegal assignment " ^ string_of_typ lt ^
      " = " ^ string_of_typ rt ^ " in " ^
    string_of_expr ex))
   | Call(fname, actuals) as call -> let fd = function_decl fname in
     if List.length actuals != List.length fd.formals then
```

```
        raise (Failure ("expecting " ^ string_of_int
          (List.length fd.formals) ^ " arguments in " ^ string_of_expr call))
      else
        List.iter2 (fun (ft, _) e -> let et = expr e in
          ignore (check_assign ft et
            (Failure ("illegal actual argument found " ^ string_of_typ et ^
            " expected " ^ string_of_typ ft ^ " in " ^ string_of_expr e))))
          fd.formals actuals;
        fd.return_type
  in
  let check_bool_expr e = if expr e != Bool
   then raise (Failure ("expected Boolean expression in " ^ string_of_expr e))
   else () in


  (* Verify a statement or throw an exception *)
  let rec stmt = function
 Block sl -> let rec check_block = function
        [Return _ as s] -> stmt s
      | Return _ :: _ -> raise (Failure "nothing may follow a return")
      | Block sl :: ss -> check_block (sl @ ss)
      | s :: ss -> stmt s ; check_block ss
      | [] -> ()
     in check_block sl
   | Expr e -> ignore (expr e)
   | Return e -> let t = expr e in if t = func.return_type then () else
      raise (Failure ("return gives " ^ string_of_typ t ^ " expected " ^
                      string_of_typ func.return_type ^ " in " ^ string_of_expr e))

   | If(p, b1, b2) -> check_bool_expr p; stmt b1; stmt b2
   | For(e1, e2, e3, st) -> ignore (expr e1); check_bool_expr e2;
```

```
                              ignore (expr e3); stmt st

       | While(p, s) -> check_bool_expr p; stmt s

     in

     stmt (Block func.fbody.f_stmts)

   in

   List.iter check_function functions
```

8.6 Warhol.ml

```
type action = LLVM_IR | Compile
let _ =

 let action = if Array.length Sys.argv > 1 then

   List.assoc Sys.argv.(1) [

          ("-l", LLVM_IR);  (* Generate LLVM, don't check *)

          ("-c", Compile) ] (* Generate, check LLVM IR *)

 else Compile in

 let lexbuf = Lexing.from_channel stdin in

 let ast = Parser.program Scanner.token lexbuf in

   Semant.check ast;

 match action with

   LLVM_IR -> print_string (Llvm.string_of_llmodule (Codegen.translate ast))

 | Compile -> let m = Codegen.translate ast in

   Llvm_analysis.assert_valid_module m;

   print_string (Llvm.string_of_llmodule m)
```

8.7 Makefile

```
all: warhol.native printbig.o openfile.o prep
warhol.native :

              ocamlbuild -use-ocamlfind -pkgs llvm,llvm.analysis -cflags -w,+a-4 \
                           Warhol.native

# "make clean" removes all generated files

.PHONY : clean

clean :

              ocamlbuild -clean

              rm -rf testall.log *.diff warhol scanner.ml parser.ml parser.mli prep

              rm -rf printbig openfile

              rm -rf *.cmx *.cmi *.cmo *.cmx *.o *.s *.ll *.out *.exe

printbig : printbig.c

              cc -o printbig printbig.c

openfile : openfile.c

              cc -o openfile openfile.c

prep: prep.c

              gcc -o prep prep.c
```

8.8 Run.sh

```
#!/bin/bash

basename=`echo $1 | sed 's/.*\///

                  s/.wl//'`





libs="printbig.o openfile.o"



./prep ${basename}.wl

./warhol.native < tempfile.wl > ${basename}.ll

rm tempfile.wl
```

```
llc ${basename}.ll > ${basename}.s

cc -o ${basename}.exe ${basename}.s ${libs}



#./${basename}.exe to run file



#move to piping add a -c flag?
```

8.9 Git Log

```
commit f8359df75d0bf888ea9b2252aa8dae7d5154b43d
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Wed May 10 16:44:14 2017 -0400



    cleaned up, demo



commit beba8e71437a62c25108b9dfffe3310a732a243b

Author: Sarina Xie <sx2166@columbia.edu>

Date:   Wed May 10 16:32:22 2017 -0400



    made demo of stdlib



commit 60940f3ea553f9444d801ad902ffe00260295121

Author: Sarina Xie <sx2166@columbia.edu>

Date:   Wed May 10 16:31:45 2017 -0400



    made demo of stdlib
```

```
commit 4d350454898226d47ef9d0f07cc35741aa9c7a7c
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Wed May 10 16:07:52 2017 -0400

    fixed bug in prep

commit 12a9c8447fdb702c56a74d3850ec5b634b73573c
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Wed May 10 15:40:03 2017 -0400

    std lib stuff

commit 42487d4f0aea7837878b0f1d7a03707773da9ec6
Author: Sam Nnodim <sam.nnodim@gmail.com>
Date:   Wed May 10 01:18:41 2017 -0400

    linked openfile to filter in warhol

commit eb57c97655b9b8e79e3b0e02cd265ce0cd021f61
Author: Sam Nnodim <sam.nnodim@gmail.com>
Date:   Wed May 10 00:48:06 2017 -0400

    fixed comma parsing

commit 797f77159fe0f41c2523e7e38fbec367be029dbb
Author: Sam Nnodim <sam.nnodim@gmail.com>
```

```
Date:   Wed May 10 00:40:54 2017 -0400

    working openfile()

commit 19d53384e00fda0295061fba33eef65983d8a8bc
Author: cassiewes <ciw2109@columbia.edu>
Date:   Tue May 9 23:46:32 2017 -0400

    quick fix

commit 968226172a2bc589232152d5e32db884a8059457
Author: Cassie Wes <ciw2109@columbia.edu>
Date:   Tue May 9 22:53:07 2017 -0400

    cleaned up files

commit 5ef0123ba2809bddcba7bfacff2d5e7b7f33dfe5
Author: Cassie Wes <ciw2109@columbia.edu>
Date:   Tue May 9 22:30:16 2017 -0400

    marilyn filters all working woohooooo

commit edb6719ddfb587fdfd53b50d5ae88b496c63808f
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Tue May 9 21:00:09 2017 -0400
```

```
    cyanblue

commit 499aa4b05a0941f94ca8426e7e3f03618a487f99
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Tue May 9 20:58:19 2017 -0400

    changedfilternames

commit 3b350e868623a1f5c16c0712bc82769d8b1d2c60
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Tue May 9 20:23:36 2017 -0400

    filters again

commit 49201ae2e1bad022a6c3c93a0713ccd776318055
Author: Sam Nnodim <sam.nnodim@gmail.com>
Date:   Tue May 9 19:43:20 2017 -0400

    late msg

commit 906e5c0ee8002947524eb2155ae6d94595189b4f
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Tue May 9 17:22:30 2017 -0400
```

```
    minor changes



commit b32307d46237d080d60f780fc7c1c264ae821161

Author: Cassie Wes <ciw2109@columbia.edu>

Date:   Tue May 9 17:01:54 2017 -0400



    testing marilyn



commit 4fa1e4298670b0939ef306a87ba0374f176ba1ed

Author: Cassie Wes <ciw2109@columbia.edu>

Date:   Tue May 9 16:29:18 2017 -0400



    thank you sarina



commit 7aad05417cd5165b24cf8a44841a191606f16612

Author: Martina Atabong <maa2247@columbia.edu>

Date:   Tue May 9 07:19:03 2017 -0400



    Got a hang of how to do printing. Also fixed how we treat ppm matrices



commit d38d8c900cb50bda2a48b576f423db921a023b2b

Merge: 14bc671 7a8ca4b

Author: Cassie Wes <ciw2109@columbia.edu>

Date:   Mon May 8 23:41:10 2017 -0400



    praying this works
```

```
commit 14bc671b217fa42ebd18fb45d71ec4a606ffa4c4
Author: Cassie Wes <ciw2109@columbia.edu>
Date:   Mon May 8 23:37:08 2017 -0400

    praying this works

commit 7a8ca4b2554141ec68ab5e217854ac23c047dbf1
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Mon May 8 22:16:43 2017 -0400

    merged Kesi's changes in

commit 81cc111f45cc2991249ad71c3f04fefb03c20032
Author: Cassie Wes <ciw2109@columbia.edu>
Date:   Mon May 8 21:48:11 2017 -0400

    for sarina

commit 6918e4e0f62bb930a17526430ed17fcc6277af6b
Author: Cassie Wes <ciw2109@columbia.edu>
Date:   Mon May 8 14:19:03 2017 -0400

    changed *->@
```

```
commit 15ff4910fb2bc721a9ab9168b3f9e00f9a34f27d
Author: Cassie Wes <ciw2109@columbia.edu>
Date:   Mon May 8 14:17:51 2017 -0400

    changed to @s

commit e6ad500ecf5f9353e3b6dece69c589b65865cfcb
Author: Cassie Wes <ciw2109@columbia.edu>
Date:   Mon May 8 14:03:09 2017 -0400

    filterall function

commit 9adda8bc7e60ed0dbc3fe1e18b4b2bdf9322905c
Author: Cassie Wes <ciw2109@columbia.edu>
Date:   Mon May 8 14:01:53 2017 -0400

    ppm

commit 3f7a948cf2c5f36719664235250396a60597a7a2
Merge: 3a090a3 4767603
Author: Cassie Wes <ciw2109@columbia.edu>
Date:   Mon May 8 11:53:43 2017 -0400

    Merge branch 'master' of https://github.com/samnnodim/warhol

commit 3a090a3662281b670a92d59030772579867d436f
```

```
Author: Cassie Wes <ciw2109@columbia.edu>
Date:   Mon May 8 11:53:37 2017 -0400

    filters

commit 19f07b8cc46a7a38c3a408d94fe8e2746909b15e
Author: cdn2118 <cdn2118@columbia.edu>
Date:   Mon May 8 11:48:58 2017 -0400

    Openfile: Working

commit d00f5a6fbc51c00a82594c3890b986030d3889fd
Author: Martina Atabong <maa2247@columbia.edu>
Date:   Mon May 8 09:21:31 2017 -0400

    As far as make and basic functionality goes, this is what you should be using

commit af08ae6f868e49927d54d28e4d73b4f8934cbbe8
Author: Martina Atabong <maa2247@columbia.edu>
Date:   Mon May 8 07:57:16 2017 -0400

    Why is make broken...

commit 47676039435c29f40d08db532f0680d9abb7fdf1
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Mon May 8 02:10:20 2017 -0400
```

```
    readme

commit dca9663b52473c26806b408a1647c416bd4b087e
Merge: 5012e0f 75021bf
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Mon May 8 01:25:39 2017 -0400

    filters

commit 5012e0f6439f239d43113096436209fa6c4cc523
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Mon May 8 01:16:04 2017 -0400

    filters folder

commit 75021bf66aa5f34eedcace3ea263df3d52b801f5
Author: Martina Atabong <maa2247@columbia.edu>
Date:   Mon May 8 00:38:18 2017 -0400

    Comments in tfpink.wl

commit 0cf5298975bdace5c3f967c8ba597d3a6541f4c1
Author: cdn2118 <cdn2118@columbia.edu>
Date:   Mon May 8 00:24:12 2017 -0400
```

```
    Openfile: Fatal exception error

commit 9f30f58435a8a9fc89b5207fb9eb2f0c9af43954
Author: Cassie Wes <ciw2109@columbia.edu>
Date:   Mon May 8 00:03:41 2017 -0400

    filter functions

commit 7aa0951051904d5399b149c9f91771c964ac7be5
Author: Cassie Wes <ciw2109@columbia.edu>
Date:   Sun May 7 23:33:35 2017 -0400

    marilyn.ppm

commit 4ff7bfed0a8582f92aae3e64ac5a5e79cc340f6b
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun May 7 22:54:47 2017 -0400

    filter pink

commit 9bb942cbaa93edf67a5aab0bfeec6948f56abf5a
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun May 7 22:29:05 2017 -0400
```

```
    for cassie


commit 4512fb7cd221fc0e9e8621dc1dcac3e4479bad88
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun May 7 22:26:40 2017 -0400


    fyellow


commit 9a758509f7c0bcc60a7db75b0afb870b583bc75c
Author: cdn2118 <cdn2118@columbia.edu>
Date:   Sun May 7 20:25:42 2017 -0400


    Openfile: adding function definitions. Not working yet


commit 6a9e1b55281174498bd4c3ce2c420d66ff7656c1
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun May 7 20:04:29 2017 -0400


    test images


commit a0d27b5406ef8d66f54d3692a8a1b6fe0be3256c
Merge: 79397dd b446d24
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun May 7 19:56:03 2017 -0400


    git problems
```

```
commit 79397dd25f649b40c4102fdb8570dc9e184918d2
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun May 7 19:50:42 2017 -0400

    2dmatrices ptrs

commit 0f612395ab98f7fb12ec5e6f799774c57fd21e21
Author: Martina Atabong <maa2247@columbia.edu>
Date:   Sun May 7 18:53:15 2017 -0400

    Use script to run any file cuz links libraries

commit 69cd6aaf99203a78384a6f3b95f065c6ea08b2f7
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun May 7 18:35:56 2017 -0400

    added ptrs from DARN

commit 4dc52e4fb311d6501c51c1ae8102a957c0667081
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun May 7 18:35:36 2017 -0400

    added pointers from DARN
```

```
commit b446d2444109fef8dfe23ec8638d6baa551d9c35

Author: Cassie Wes <ciw2109@columbia.edu>

Date:   Sun May 7 18:25:24 2017 -0400



   semant 1D matrices



commit 623ba1d5ad87717398329d4c9384a88c1bc9e67d

Author: cdn2118 <cdn2118@columbia.edu>

Date:   Sun May 7 18:02:16 2017 -0400



   Openfile: Openfile links to warhol. Only prints string



commit 66c9532eb57a4d7a22d5b446bf5f60a92045de58

Author: cdn2118 <cdn2118@columbia.edu>

Date:   Sun May 7 16:02:17 2017 -0400



   Accident: removed hello world and adding back



commit ce49488dcec3ba9b840b35af66f77fe129ed635e

Merge: 0db625f 57e111a

Author: Sarina Xie <sx2166@columbia.edu>

Date:   Sun May 7 15:54:49 2017 -0400



   merged in controlflow



commit 57e111a384b5140f67f341dcf57b59e18659c84a
```

```
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun May 7 15:45:40 2017 -0400

    edited semant, ready for merge

commit 0db625f60f3345a54d4cbaf0fb611b8afe3bcb12
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun May 7 13:18:31 2017 -0400

    location of variable decl

commit 3357fc6f7628249ceb971e3b3aaa7b2cbb13d12b
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun May 7 13:09:52 2017 -0400

    declare variable anywhere

commit c4f67fd050a77287f54d465b3fe5a9722a24424d
Author: cdn2118 <cdn2118@columbia.edu>
Date:   Sun May 7 09:15:30 2017 -0400

    Strings: Print Strigns function working

commit 036ffc6077ee7013cdc7db5321ab75c1b245e7b9
Author: Martina Atabong <maa2247@columbia.edu>
Date:   Sat May 6 18:48:11 2017 -0400
```

```
    Semant works, doesn't include string/char/matrices though

commit 675625192133fefa74971542a3b0d6497b26087e
Author: Martina Atabong <maa2247@columbia.edu>
Date:   Sat May 6 13:19:24 2017 -0400

    Actually merged with string literals

commit 1fea175e7c01a59abdf292edcca7366c6a14a819
Merge: a601866 77a0375
Author: Martina Atabong <maa2247@columbia.edu>
Date:   Sat May 6 13:16:16 2017 -0400

    Shit

commit 77a0375d1758da45d7bfa7cb585c2a66ee487b51
Merge: e206064 67d457c
Author: cdn2118 <cdn2118@columbia.edu>
Date:   Sat May 6 13:10:17 2017 -0400

    Merge Conflict: conflict between math and controlflow

commit 67d457c3fcb5c05b10b0832825dcb089c3517b2d
Author: cdn2118 <cdn2118@columbia.edu>
Date:   Sat May 6 12:57:23 2017 -0400
```

```
    Strings: added String and Char Literals

commit a6018661d454ba27ffee666f800da519450b3539
Merge: e206064 7c97349
Author: Martina Atabong <maa2247@columbia.edu>
Date:   Sat May 6 12:51:30 2017 -0400

    Attempts at merging

commit a389b7fbf40f0ddb23bfd27488b439a78408c9a6
Author: cdn2118 <cdn2118@columbia.edu>
Date:   Sat May 6 11:23:31 2017 -0400

    Strings: Only prints ascii value of first char in string

commit e2060645f60d5cae4f8a231a928cbdb06d705737
Author: Martina Atabong <maa2247@columbia.edu>
Date:   Sat May 6 03:33:32 2017 -0400

    Working relational operators

commit 55204a1a9d8d971aa85c64b4730f343f5d09102f
Author: Martina Atabong <maa2247@columbia.edu>
Date:   Sat May 6 01:34:12 2017 -0400
```

```
    Added logic operators and arithmetic operators that went missing

commit 7c9734917051f41122215572503c146c52416659
Author: Sam Nnodim <sam.nnodim@gmail.com>
Date:   Fri May 5 22:40:35 2017 -0400

    updated gitignore

commit e849461e08d47862b498a52144029d543b2e5b86
Merge: 85ece61 3ed47e9
Author: Sam Nnodim <sam.nnodim@gmail.com>
Date:   Fri May 5 22:28:16 2017 -0400

    Merge branch 'controlflow' of https://github.com/samnnodim/warhol into math

commit 339d9eca1f7d9c2e59db1ef343ea9cd0cfc2d4b6
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Fri May 5 20:32:27 2017 -0400

    commented ll files

commit e81c7cc67c73b534ea6b40c6a89a2d3f2c461537
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Fri May 5 20:11:10 2017 -0400
```

```
    looking at array llvm

commit 949295b81dfa45a9757ccd04d00c899df6fc8354
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Fri May 5 19:48:26 2017 -0400

    more comments in codegen

commit aba8f5f121563d566c6ba48ac1fae01217fd2e4e
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Fri May 5 19:33:47 2017 -0400

    started variable size arrays

commit da524f6a22eba49d11a0a407fc774f1bf406ba15
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Fri May 5 18:10:05 2017 -0400

    can declare variables in middle of function

commit 7fb4753c050da95ed5a8ac65b52749af551e8426
Merge: 75cf239 699d0bb
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Thu May 4 22:06:15 2017 -0400
```

```
    var decls

commit 75cf239cbe89162534fcd83405c161cacfe0b091
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Thu May 4 21:45:43 2017 -0400

    working on var decl

commit 3ed47e9c805d61b712f96a26891cc779ad5b7738
Author: Martina Atabong <maa2247@columbia.edu>
Date:   Thu May 4 10:52:01 2017 -0400

    Boolean, If/Else, For, and While all work

commit 85ece61cb9bd50802c4a08cb7fd23527e3f61054
Author: Sam Nnodim <sam.nnodim@gmail.com>
Date:   Thu May 4 03:28:18 2017 -0400

    updated parts of scanner [old]

commit ea894268b7eb75cf2e97fb5a56b46f8c69ac42ef
Author: Sam Nnodim <sam.nnodim@gmail.com>
Date:   Fri Apr 28 17:46:22 2017 -0400
```

```
    fixed shift/reduce error

commit 0fb068999c5f718c672a8d1177e7480a67bff4bd
Author: Sam Nnodim <sam.nnodim@gmail.com>
Date:   Fri Apr 28 16:28:36 2017 -0400

    added left associativity

commit 6a162c8547aa2a8aaa803168ecb3bf9a6ba340b8
Author: Sam Nnodim <sam.nnodim@gmail.com>
Date:   Fri Apr 28 15:51:34 2017 -0400

    added: add,subtract,multiply,divide to scanner,parser,and AST

commit 67dd4f8f352c40f11f6c7e119a0dbb4f4d815cf2
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Fri Apr 28 00:34:02 2017 -0400

    test for declaration, assignment, array

commit 128f984beaeee95ec3904418e47b362127d213b1
Merge: 00236cc 87169d4
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Fri Apr 28 00:30:26 2017 -0400

    Merge branch 'master' of https://github.com/samnnodim/warhol
```

```
commit 00236ccd87fe9cb25966c92cd0fbda73ff0cacd6

Author: Sarina Xie <sx2166@columbia.edu>

Date:   Fri Apr 28 00:29:47 2017 -0400



   used microc to get variable dec and assignment working, used darn to get arrays working



commit 699d0bbf96e100eb47296cd1777d126f294a412c

Merge: 03f3751 87169d4

Author: cassiewes <ciw2109@columbia.edu>

Date:   Wed Apr 12 20:50:53 2017 -0400



   Merge branch 'master' into sarina


   # Conflicts:

   #        ast.ml

   #        parser.mly

   #        scanner.ml



commit 87169d42cefcef6397fa30a0a9599914c78a9a26

Author: cassiewes <ciw2109@columbia.edu>

Date:   Wed Apr 12 20:30:25 2017 -0400



   how to run hello world



commit 5e21f500b538d427b99a9e133c96cce43639a0dd
```

```
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Tue Apr 11 15:52:09 2017 -0400

    printed 42 albeit with many warnings

commit 39fdf552c741e9873506fbf82527300f6badb2d6
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Fri Apr 7 15:17:24 2017 -0400

    parser working

commit 06a1c2865e106497f6d4afdf51ea4acda097a157
Author: cassiewes <ciw2109@columbia.edu>
Date:   Fri Apr 7 13:26:50 2017 -0400

    lessons from other groups

commit 8c1345bffabb5435893633db325ca816744ba322
Merge: a52b03a 7cb1b0a
Author: cassiewes <ciw2109@columbia.edu>
Date:   Wed Apr 5 16:29:44 2017 -0400

    Merge branch 'cassie'

commit 7cb1b0a2e8fcbb1a7e69d0e6397ef124869d93f5
Merge: 575dee0 a52b03a
```

```
Author: cassiewes <ciw2109@columbia.edu>

Date:   Wed Apr 5 16:27:37 2017 -0400



   Merge branch 'master' into cassie



commit 575dee0f8883e57397921ec1627b834c00ca800d

Author: cassiewes <ciw2109@columbia.edu>

Date:   Wed Apr 5 16:27:34 2017 -0400



   added to parser



commit a52b03acf6cf4e1d93f1b8bbeac822b42e89ed30

Author: samnnodim <sam.nnodim@gmail.com>

Date:   Tue Apr 4 22:41:09 2017 -0400



   added semant.ml and updated test script.



commit 8e7783590cce1134925de27bcc72096dedebc6af

Author: samnnodim <sam.nnodim@gmail.com>

Date:   Tue Apr 4 21:45:50 2017 -0400



   added test.sh



commit e5b5a09570fc7b3062d2312ed4e92bfd555245fe

Author: samnnodim <sam.nnodim@gmail.com>

Date:   Tue Apr 4 21:36:13 2017 -0400
```

```
    rm the scanner

commit 433ff489e94204cb5209b5bb560a3388486c58b1
Merge: 7054536 59817e1
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Tue Apr 4 21:35:06 2017 -0400

    Merge branch 'master' into sam

commit 70545362ae0f6f2d5f4064c47c7720847ef61fc6
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Tue Apr 4 21:14:07 2017 -0400

    addded a build script

commit 59817e1eeccadb73cc3eebb6eff0c3f6433c5f6a
Merge: 83e5bd6 5ad48a3
Author: cassiewes <ciw2109@columbia.edu>
Date:   Tue Apr 4 21:06:03 2017 -0400

    Merge branch 'cassie'

commit 5ad48a30089084415ba44f9ff31ff2f326a89329
Author: cassiewes <ciw2109@columbia.edu>
Date:   Tue Apr 4 21:01:45 2017 -0400
```

```
    condensed


commit b4a40eecde5e9bb94c552e88eefe1d0f80d8f95d
Merge: a95e03c 83e5bd6
Author: cassiewes <ciw2109@columbia.edu>
Date:   Mon Mar 27 09:42:17 2017 -0400


    Merge branch 'master' into cassie


commit 03f3751389600cdf9ffdca1ebcb68b118b27fd8a
Author: Charvinia Neblett <cdn2118@columbia.edu>
Date:   Sun Mar 26 19:48:23 2017 -0400


    Testing Declarations: current state of rejection


commit 83e5bd6b39d8bce0c66849e181bea634a97ed920
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Sun Mar 26 18:44:08 2017 -0400


    added new test line to the README.


commit 202ac2dfb15f467c14e448b670672aa3255aacea
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Sun Mar 26 18:34:57 2017 -0400
```

```
    updated parser

commit a95e03c5c983f17d3d1f98552a7c0231b1efa68d
Merge: 7d91390 36ab8ca
Author: cassiewes <ciw2109@columbia.edu>
Date:   Sun Mar 26 18:30:12 2017 -0400

    Merge branch 'master' into cassie

commit 36ab8cab8a9ea57963990a7d47355a35b6ca6903
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Sun Mar 26 18:28:02 2017 -0400

    added a test help document.

commit 305c531ce573c1b486ca5ac17be7bda705f57cd6
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Sun Mar 26 18:23:38 2017 -0400

    unneccessary folder

commit 3b072b98449a7cc1c2eb44a72682b5febe059dee
Merge: 6df79bf 7d91390
Author: cassiewes <ciw2109@columbia.edu>
Date:   Sun Mar 26 18:18:38 2017 -0400
```

```
    Merge branch 'cassie'

commit 7d91390767680017d27bf69611e002b7824244f3
Author: cassiewes <ciw2109@columbia.edu>
Date:   Sun Mar 26 18:18:18 2017 -0400

    update scanner.mll

commit 3d9a55d72725b89b8cfc600b159a326a8a9a50d5
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Sun Mar 26 18:11:03 2017 -0400

    -a

commit 36067b29c07fb1fd2fa6a3cb41d52bf223d76c6d
Merge: 57a6be8 6df79bf
Author: cassiewes <ciw2109@columbia.edu>
Date:   Sun Mar 26 18:07:29 2017 -0400

    Merge branch 'master' of https://github.com/samnnodim/warhol into cassie

commit 6df79bf50a2deeab7bed8a79c83131bacbbae51f
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Sun Mar 26 18:01:08 2017 -0400
```

```
    added the online example.

commit a682789cde01ac78950c2903a5eaa5738ee6a64a
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun Mar 26 18:00:07 2017 -0400

    started looking at the ast too

commit 9866dc1c561808b7061014ab2aeda090abead520
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Sun Mar 26 17:30:48 2017 -0400

    working tests.

commit 7cdfe657df86eac2fcfdd966832dcdc788eae8dd
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Sun Mar 26 17:28:48 2017 -0400

    updated the gitignore

commit d579f9e3555e2aad4b804d89e1e63fb3520437b6
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Sun Mar 26 17:21:56 2017 -0400
```

```
    made a test directory. and put a test in it.



commit f143f637d1daf3e72c1baed9be8b30252b35cc68

Merge: 28ad16c 66af78f

Author: Sarina Xie <sx2166@columbia.edu>

Date:   Sun Mar 26 11:34:28 2017 -0400



    Merge branch 'master' into sarina

    Trying to update sarina



commit 28ad16c9a9300e32be3b644479352d346f9de13d

Author: Sarina Xie <sx2166@columbia.edu>

Date:   Sun Mar 26 11:34:18 2017 -0400



    tiny change



commit 66af78f2c95fe92d6debae515318f8f8942eeaaf

Merge: d7e49a5 6986c63

Author: Sarina Xie <sx2166@columbia.edu>

Date:   Sun Mar 26 11:17:52 2017 -0400



    Merge branch 'master' of https://github.com/samnnodim/warhol



commit d7e49a5576fd8a208d126141de354260e8dd9e93

Author: Sarina Xie <sx2166@columbia.edu>

Date:   Sun Mar 26 11:16:37 2017 -0400
```

```
    started parser, working on vars

commit 5169f8e3a9bc390b969f88b0e844f78dc7764808
Merge: 52a3505 9cc2637
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun Mar 26 11:14:09 2017 -0400

    whoops merge del some lines

commit 52a3505c4bcc1cadb7fe4d004ad19495a7bf5a1e
Author: Sarina Xie <sx2166@columbia.edu>
Date:   Sun Mar 26 11:11:52 2017 -0400

    started parser, doing vars first

commit 6986c6357bcbe7ebe35ed476e1bd8cc043930a4f
Author: cassiewes <ciw2109@columbia.edu>
Date:   Fri Mar 24 22:58:21 2017 -0400

    change to scanner

commit 57a6be84076c1a0a836e875b5c0706ac8f39a576
Merge: fd5fe0e 39414e3
Author: cassiewes <ciw2109@columbia.edu>
Date:   Fri Mar 24 22:57:45 2017 -0400
```

```
    Merge branch 'master' into cassie

commit 39414e328aee2feb82d81f0661de41ef52a72a31
Author: cassiewes <ciw2109@columbia.edu>
Date:   Fri Mar 24 22:52:13 2017 -0400

    scanner progress

commit fd5fe0e762f28608466733a8d3fc578bcf78ed51
Author: cassiewes <ciw2109@columbia.edu>
Date:   Fri Mar 24 22:42:50 2017 -0400

    scanner progress

commit 7b0cb89bf7505eb5f738e4cd66fd9af449db6767
Merge: 17bcb51 9cc2637
Author: cassiewes <ciw2109@columbia.edu>
Date:   Fri Mar 24 21:04:20 2017 -0400

    Merge branch 'master' into cassie

commit 17bcb51c216b20f30368263a4f13ef02a178843a
Author: cassiewes <ciw2109@columbia.edu>
Date:   Fri Mar 24 21:04:17 2017 -0400
```

```
    !

commit 9cc2637773353115d054c3d7b25b29ef3d8a5963
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Fri Mar 24 21:00:52 2017 -0400

    little change to the scanner.

commit a8aab257b5ad5057c47d8dac8cd999c9d9e9547e
Merge: 7f188ff 4a40458
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Fri Mar 24 20:41:51 2017 -0400

    Merge branch 'master' of https://github.com/samnnodim/warhol

commit 7f188ff7e77059676cb351d3d5d5ea659c5499d1
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Fri Mar 24 20:41:30 2017 -0400

    updated the scanner

commit 4a404586911924d085bbe0ee5840111c96af6919
Author: sam <sam.nnodim@gmail.com>
Date:   Fri Mar 24 20:41:07 2017 -0400
```

```
    update that readme

commit 10c1ac07f0e6bb8c19a01f2bbbba83ed4602e26d
Merge: a4044e5 2fd9a33
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Fri Mar 24 20:37:28 2017 -0400

    Merge branch 'sam'

commit 2fd9a334ac47ebb7c9aa6f134e9e51b94b32b72b
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Fri Mar 24 20:36:57 2017 -0400

    semi-working scanner.

commit a4044e5d85cd5d8baecf1c7239e0a58791ca1fdd
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Fri Mar 24 20:01:23 2017 -0400

    readme tweak

commit 104f9f6aec5cad76c76147d5b3669c6c1a31b646
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Fri Mar 24 19:43:57 2017 -0400
```

```
    updated the le readme

commit 2abbc95243886d648bd6ff0e29e1b4bcfb1a3205
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Fri Mar 24 19:40:28 2017 -0400

    working scanner

commit a7ad81bae6f03643a1a35f4eb96fa30b5cec5aea
Merge: b06a30d 0697e79
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Fri Mar 24 19:31:18 2017 -0400

    Merge branch 'master' of https://github.com/samnnodim/warhol

commit b06a30da78399a864eda9069a311e807d6b57aee
Author: samnnodim <sam.nnodim@gmail.com>
Date:   Fri Mar 24 19:31:01 2017 -0400

    updates to the readme

commit 0697e79f9fbf7b3690267aeef1555d7f53ab2701
Author: cassiewes <ciw2109@columbia.edu>
Date:   Fri Mar 24 19:30:58 2017 -0400
```

```
    this should work

commit f39b7bff114a918ff4cdf2309065cda651613b92
Author: cassiewes <ciw2109@columbia.edu>
Date:   Fri Mar 24 19:22:16 2017 -0400

    here ya go

commit a545ff8cce2ec1bcd73adc45c78cd23289a3188e
Author: cassiewes <ciw2109@columbia.edu>
Date:   Fri Mar 24 19:10:57 2017 -0400

    ast.ml, main.ml, parser.mly, scanner.mll

commit 0d00f1e4300006c2b672ba3a88d719b637b2fa8f
Author: sam <sam.nnodim@gmail.com>
Date:   Fri Mar 24 18:44:51 2017 -0400

    language reference manual

commit 1bca23b69ba37ef611c1c4611b107bf383ebd9eb
Author: sam <sam.nnodim@gmail.com>
Date:   Fri Mar 24 18:38:35 2017 -0400
```

```
    deleted it b/c it didn't work.



commit 06c87107d77b1e25a314291e574b032459c24382

Author: sam <sam.nnodim@gmail.com>

Date:   Fri Mar 24 18:38:13 2017 -0400



    added language ref manual



commit 91a11a27cc04f8d1a10744c4e085abdd5beb61b1

Author: sam <sam.nnodim@gmail.com>

Date:   Fri Mar 24 18:30:25 2017 -0400



    Initial commit
```