a matrix manipulation language

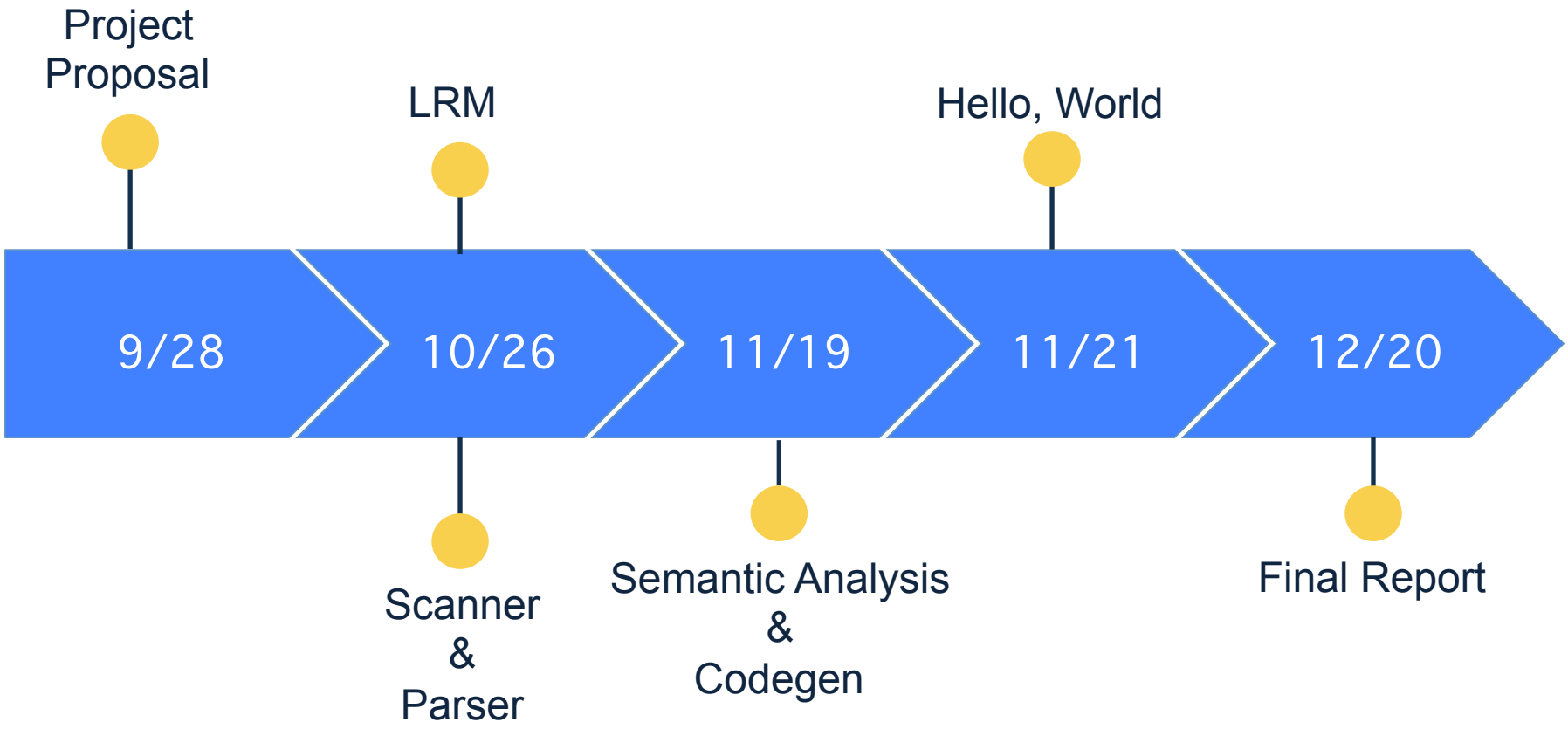
Daisy Chaussee (dac2183)
Anthony Kim (ak3703)
Rafael Takasu (rgt2108)
Ignacio Torras (it2216)

INTRODUCTION

DARN is a matrix manipulation language with native support for matrix data types.

- Strongly typed
- Imperative
- Supports if/else/for/while flow controls
- User control and freedom (inspiration from C)
- Robust matrix-oriented standard library

TIMELINE


Project Proposal
LRM
Hello, World
9/28
10/26
11/19
11/21
12/20
Scanner & Parser
Semantic Analysis & Codegen
Final Report

LANGUAGE OVERVIEW

Primitive Types:
int, float, bool, char, string

Data Types:
1-D and 2-D matrices

Declaration/Initialization:

```
int a;
a = 2;
```

1D matrix:

```
int[5] m;
m[1] = 0;
```

2D matrix:

```
int[5][5] w;
w[0][0] = 1;
```

Function Declaration:

```
int addMatrices(int[] x, int[]
y, int len) {
     /* function */
}
```

Operators:
Standard C arithmetic and logical operators

[ ] 1-D matrix access
[ ] [ ] 2-D matrix access

% access pointer
%%
dereference pointer
++ pointer increment

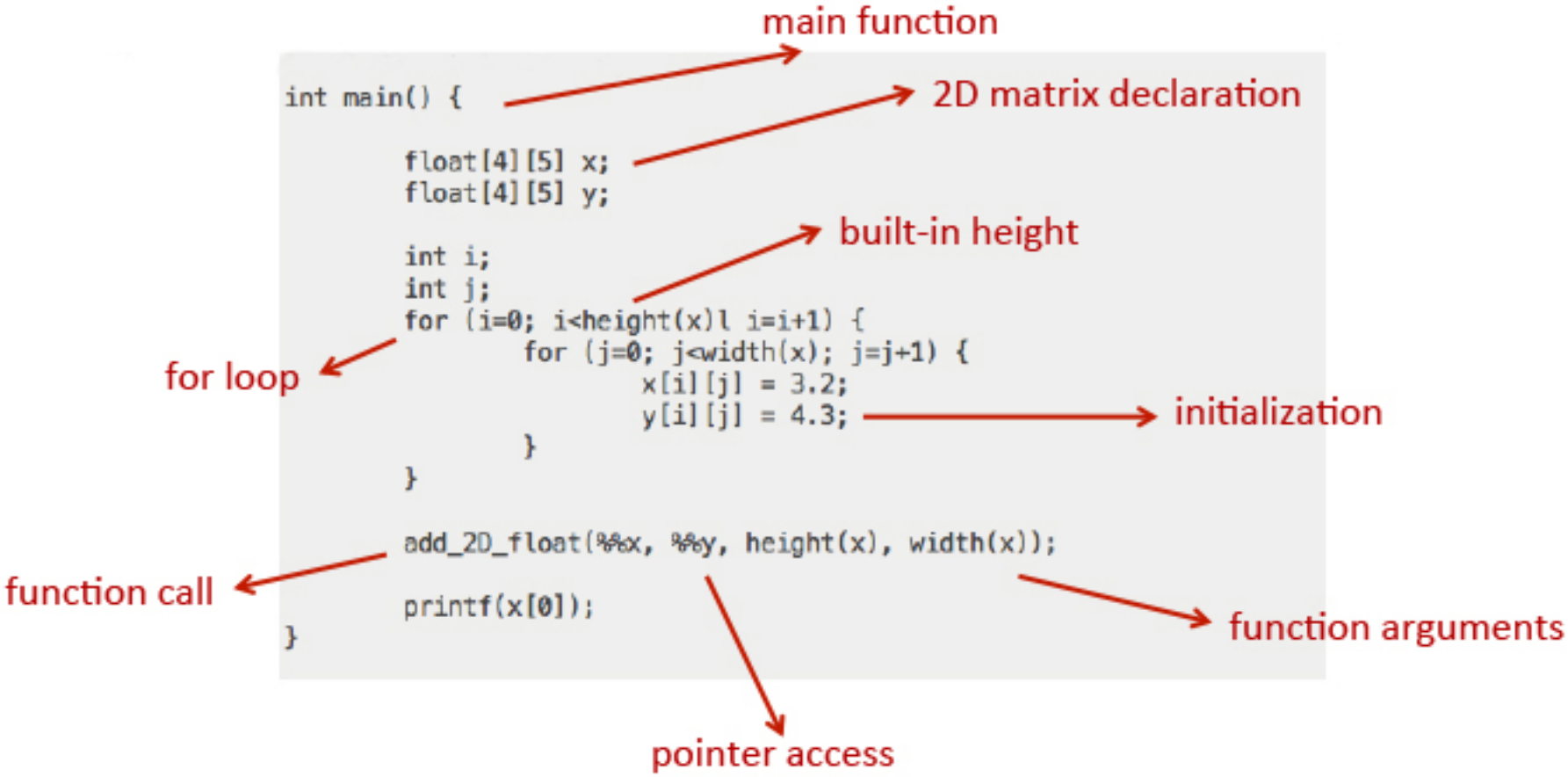
File Extension:
.darn

Control Flow:

```
if (true) {
        print(x);
} else {
        print(y);
}

while(x > y) {
        prints("hello\n");
}
int i;
for (i= 0; I < 3; i=i+1) {
        print(x);
}
```

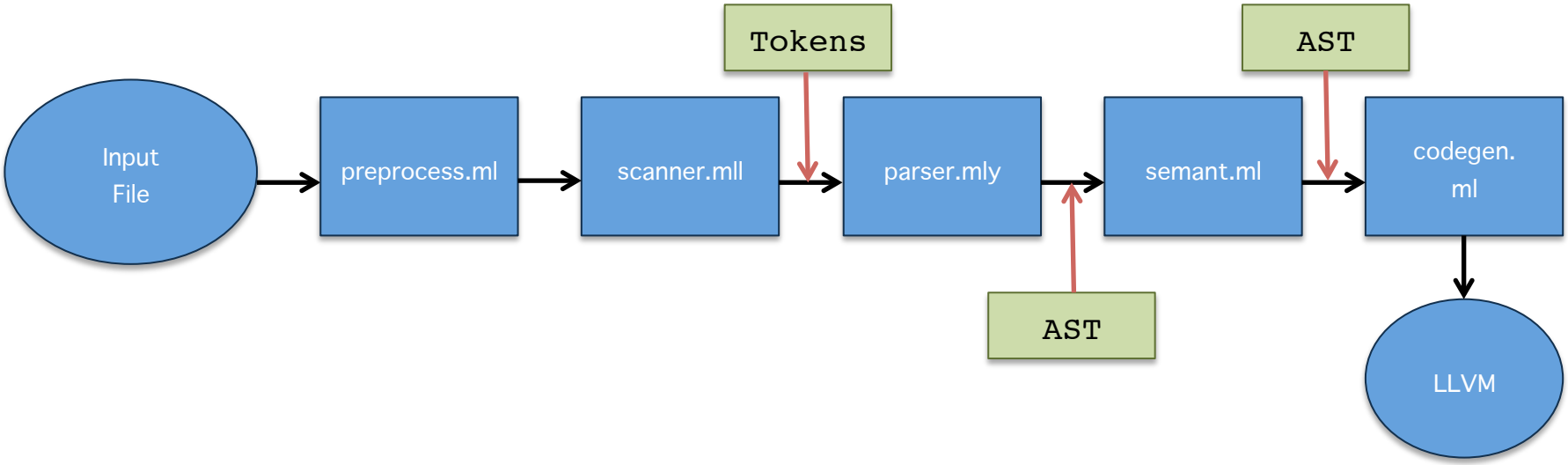
BRIEF TUTORIAL



HELLO, WORLD


```
1 int main() {
2   prints(“Hello, World!\n”);
3 }
```

ARCHITECTURE



TESTING

- **test directory** contains all tests & test scripts for compiler, parser, scanner, and compiler_fail tests
- **testing automation program** invoked separately for each directory, such as ./compiler_test.sh
- success & **fail tests** included
- continuous integration with **Travis CI**

LESSONS LEARNED

Daisy: *make SMART goals & follow through*

Anthony: *power of pair programming, overcome setbacks*

Rafa: *time and communication are valuable, learned to love matrices*

Nacho: *plan ahead, master OCaml early on, work on everything for better understanding*

DEMO