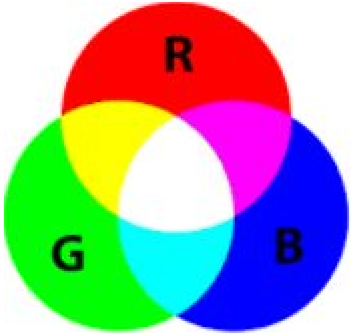
A raster graphics language.

Crayon

Naman Agrawal, Vaidehi Dalmia, Ganesh Ravichandran, David Smart

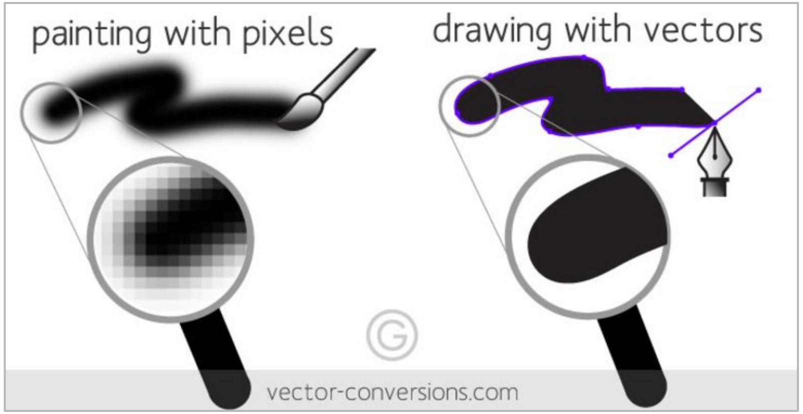
What is Crayon?

- Crayon is a raster-graphics creation language that simplifies the digital painting of images through code.
- Based on a matrix-layout of RGB pixels -> converted to a ppm file
- Allows artistic expression through mathematical and algorithmic means



Why Pixels?

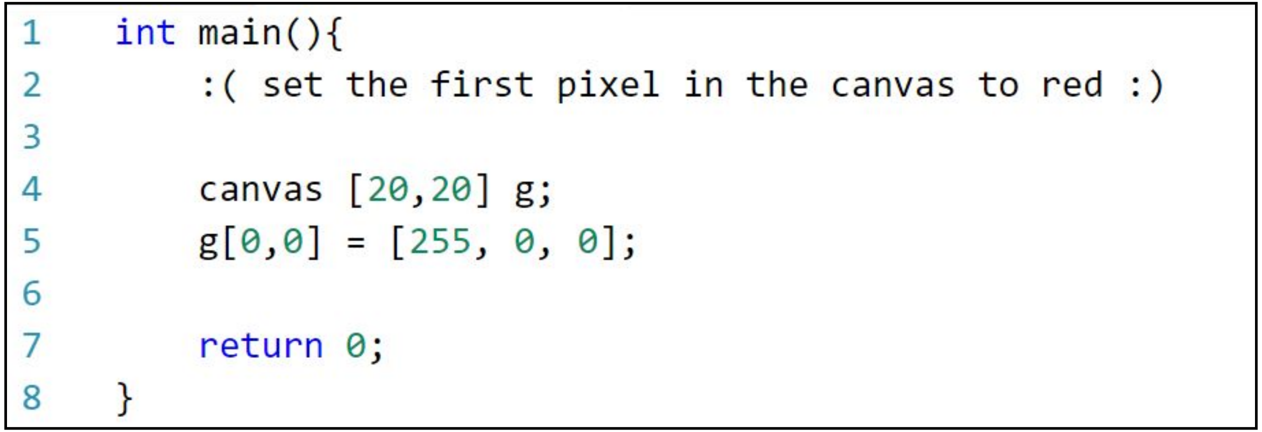
- Rasterization allows the manipulation of each pixel’s color.
- On the other hand, vector-images fill in objects with a single color.
- Vector images are more scalable, but pixels allow for interesting color blends and programmer-friendly manipulation!



Goals for Crayon

- **Transparency:** our intuitive *Canvas* type allows direct manipulation of the pixels of a ppm (Portable Pixmap) file.
- **Familiarity:** the syntactic learning curve is low for those that know C; manipulating RGB values is as easy as using arrays.
- **Creativity:** by making our language familiar and transparent, developers can create robust and interesting graphics programs.

So, What is a ‘Canvas’ Anyway?

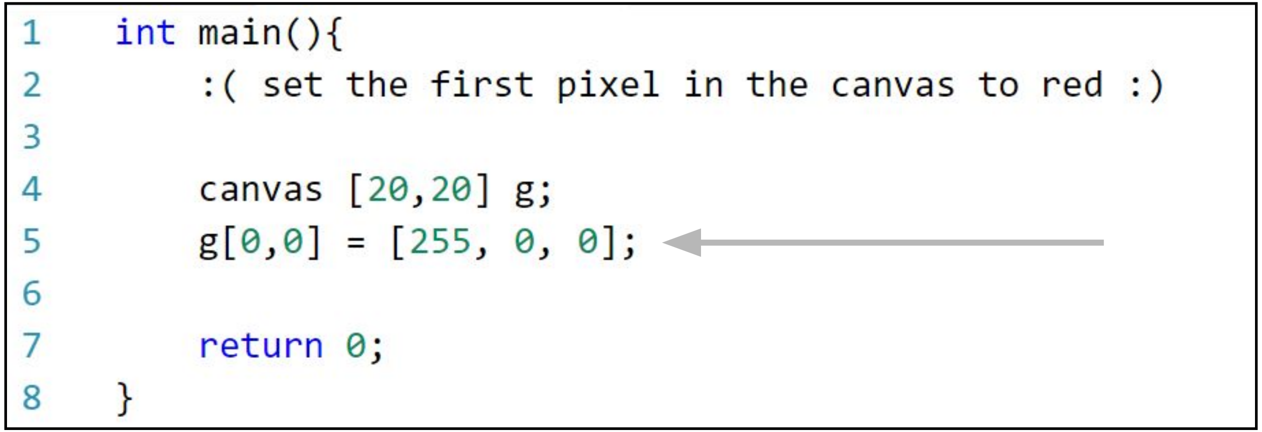


```
int main(){
    :( set the first pixel in the canvas to red :)

    canvas [20,20] g;
    g[0,0] = [255, 0, 0];

    return 0;
}
```

Essentially, it is a two-dimensional array, with 3-element arrays as RGB pixels.

So, What is a ‘Canvas’ Anyway?

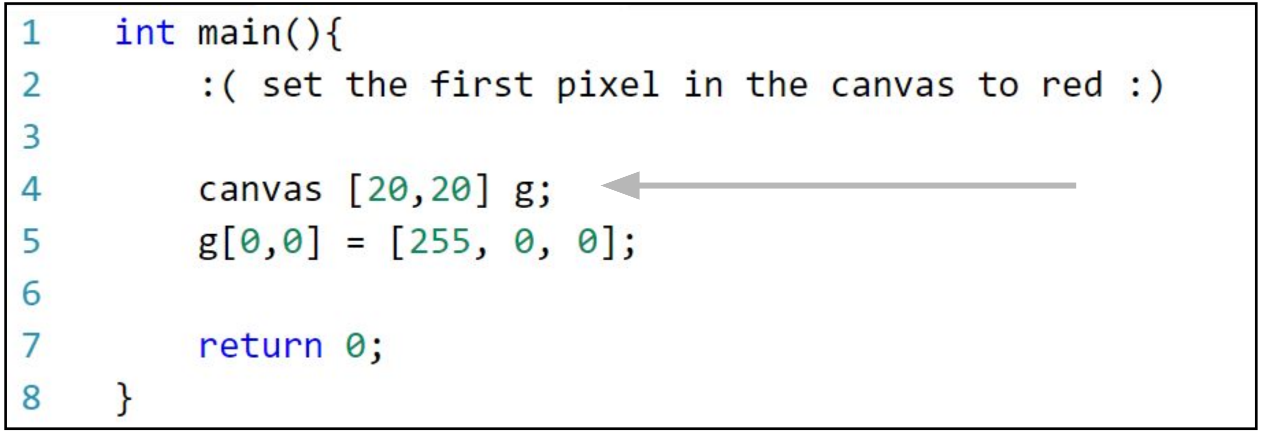
- The three element integer array (Pixel) represents an RGB value:
- E.g. **red** = **(255, 0, 0)** in RGB notation **= [255, 0, 0]** as an element of a Canvas.



```
int main(){
    :( set the first pixel in the canvas to red :)

    canvas [20,20] g;
    g[0,0] = [255, 0, 0];

    return 0;
}
```

So, What is a ‘Canvas’ Anyway?

- The Canvas is the exact same size as the ppm file that is generated.

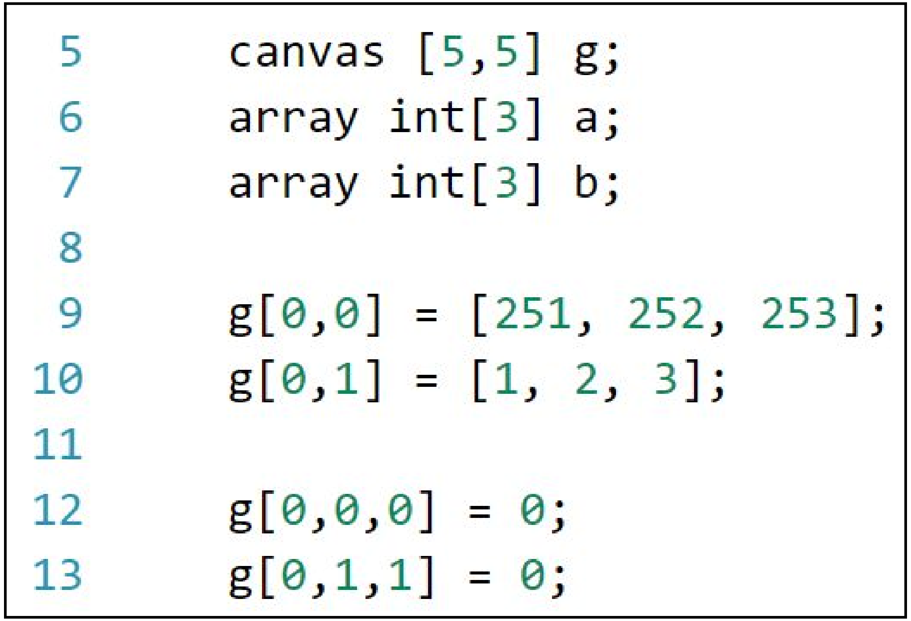


```
int main(){
    :( set the first pixel in the canvas to red :)

    canvas [20,20] g;
    g[0,0] = [255, 0, 0];

    return 0;
}
```

In this case 20x20 pixels.

So, What is a ‘Canvas’ Anyway?

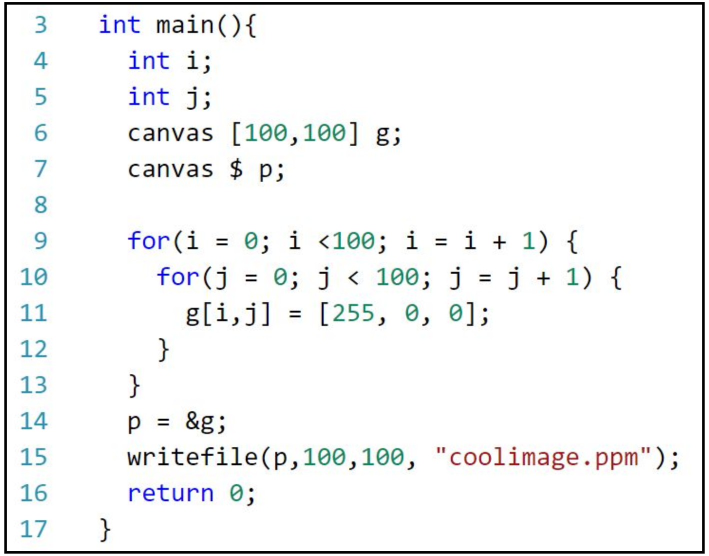
- Pixels can be accessed and assigned values quite intuitively.



```
canvas [5,5] g;
array int[3] a;
array int[3] b;

g[0,0] = [251, 252, 253];
g[0,1] = [1, 2, 3];

g[0,0,0] = 0;
g[0,1,1] = 0;
```

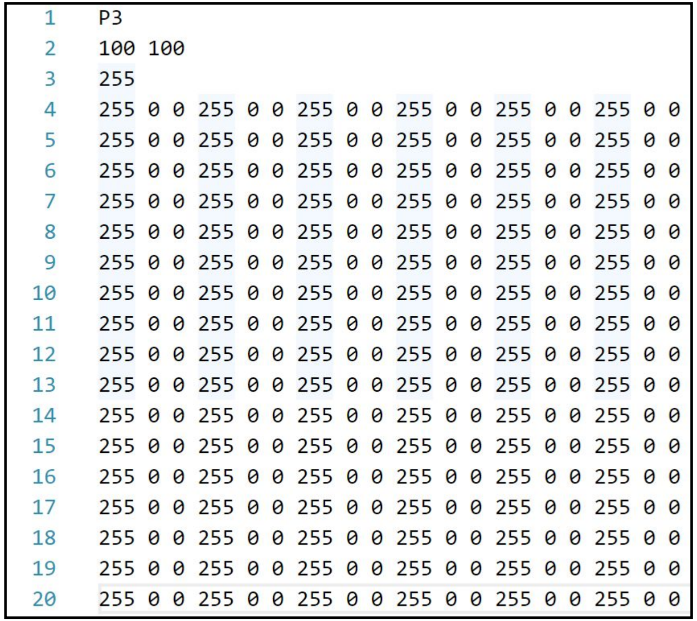
Dude, Where’s My File?

- Files can be created easily by passing in a Canvas pointer to our **writefile** function.

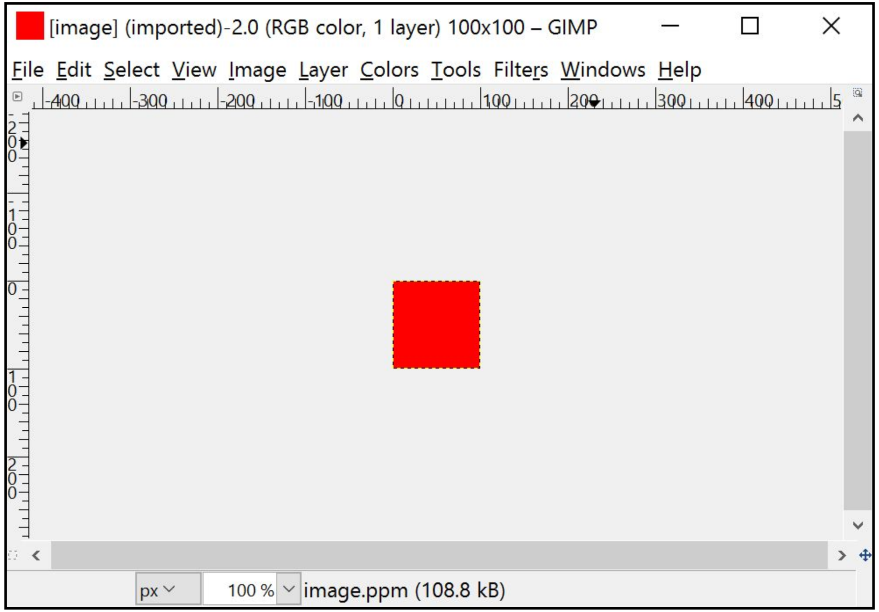


```
int main(){
  int i;
  int j;
  canvas [100,100] g;
  canvas $ p;

  for(i = 0; i <100; i = i + 1) {
    for(j = 0; j < 100; j = j + 1) {
      g[i,j] = [255, 0, 0];
    }
  }
  p = &g;
  writefile(p,100,100, "coolimage.ppm");
  return 0;
}
```

Dude, Where's My File?

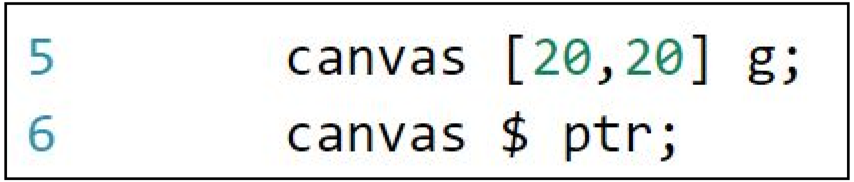


```
P3
100 100
255
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
```


[image] (imported)-2.0 (RGB color, 1 layer) 100x100 – GIMP
File Edit Select View Image Layer Colors Tools Filters Windows Help
-400
-300
-200
-100
0
100
200
300
400
px
100 %
image.ppm (108.8 kB)

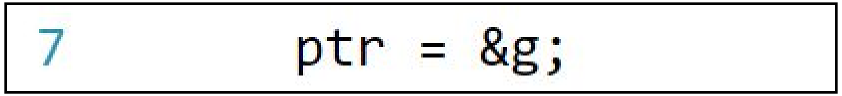
The computer text version and the human eye version.

A Gentleman’s Guide to Canvas Pointers



```
        canvas [20,20] g;
        canvas $ ptr;
```

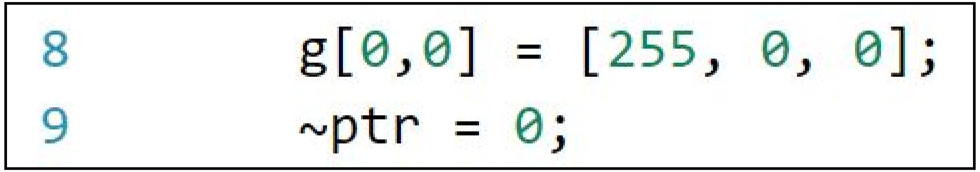
Declaring the pointer.



```
        ptr = &g;
```

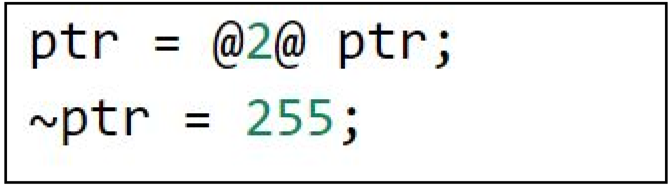
Defining the pointer.

A Gentleman’s Guide to Canvas Pointers



```
        g[0,0] = [255, 0, 0];
        ~ptr = 0;
```

Dereferencing the pointer.



```
ptr = @2@ ptr;
~ptr = 255;
```

Moving the pointer.

Our Types

Primitive types:

- Int
- String
- Boolean
- Void

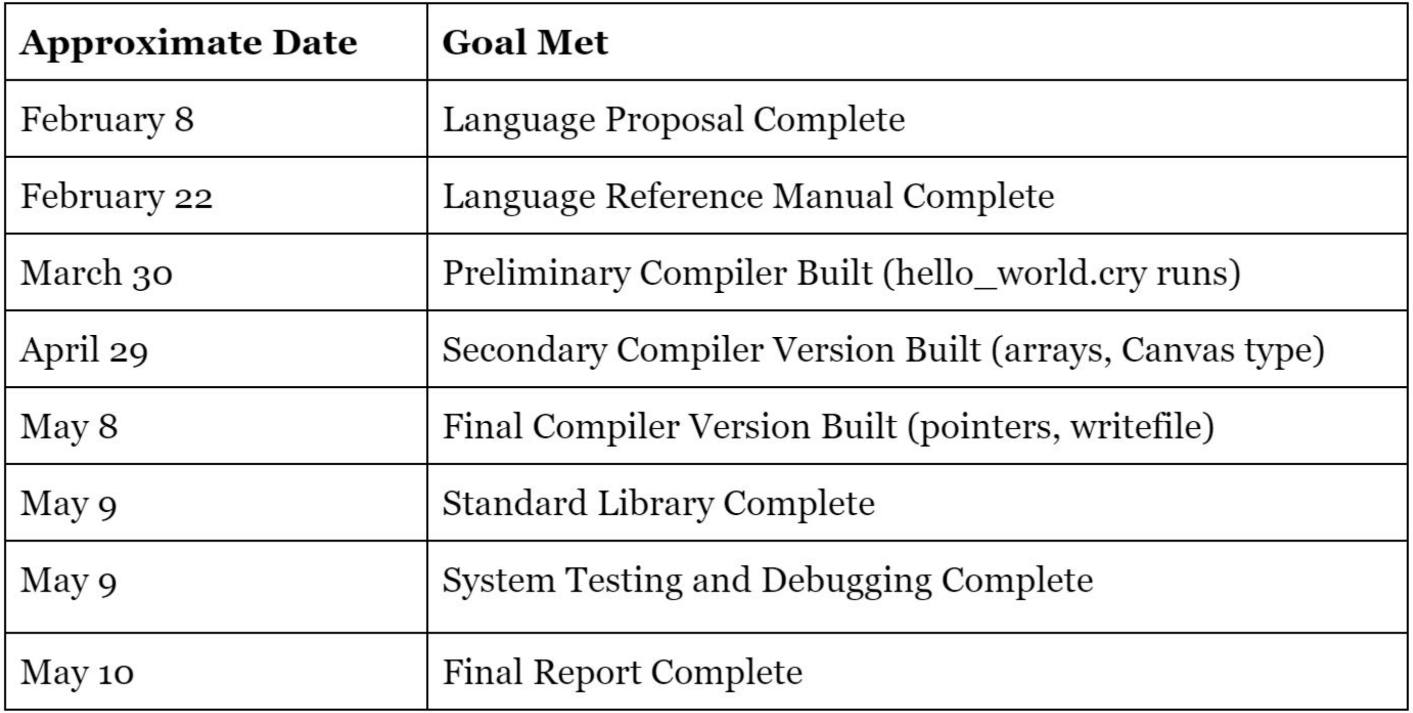
Non-Primitive types:

- Canvas
- Array
- Pointer

Project Plan

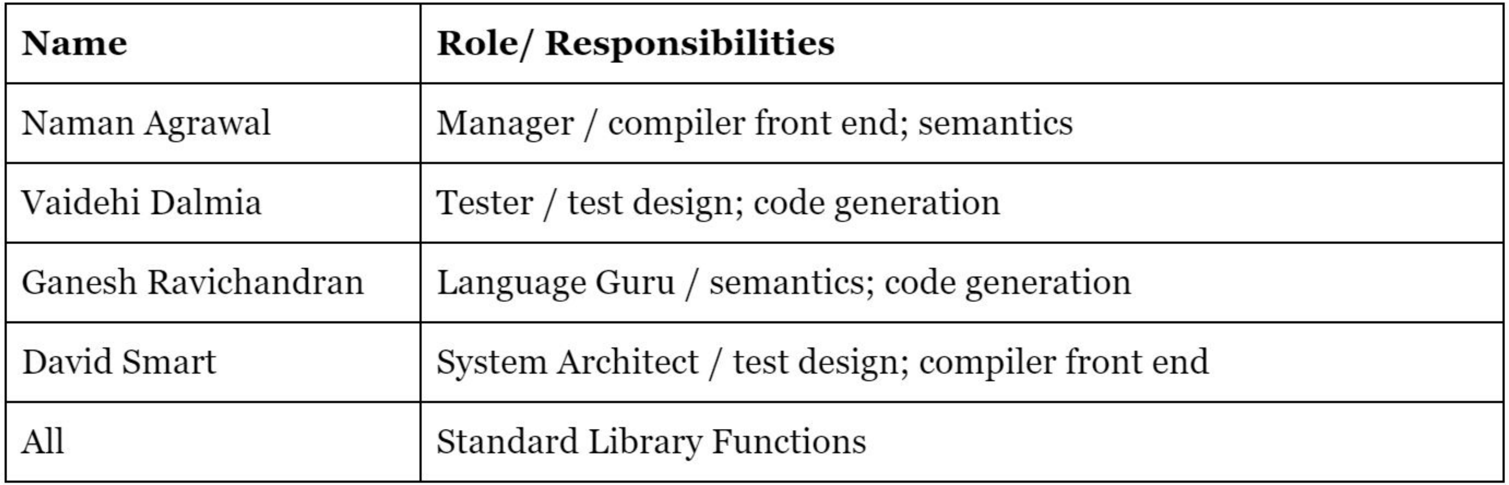
- Agile (iterative) development approach
- Lots of new decisions as new problems were encountered (e.g. adding pointers, not making Pixel a type)
- Informal and formal testing at each stage to ensure complete functioning.

Timeline



Approximate Date	Goal Met
February 8	Language Proposal Complete
February 22	Language Reference Manual Complete
March 30	Preliminary Compiler Built (hello_world.cry runs)
April 29	Secondary Compiler Version Built (arrays, Canvas type)
May 8	Final Compiler Version Built (pointers, writefile)
May 9	Standard Library Complete
May 9	System Testing and Debugging Complete
May 10	Final Report Complete

Responsibilities

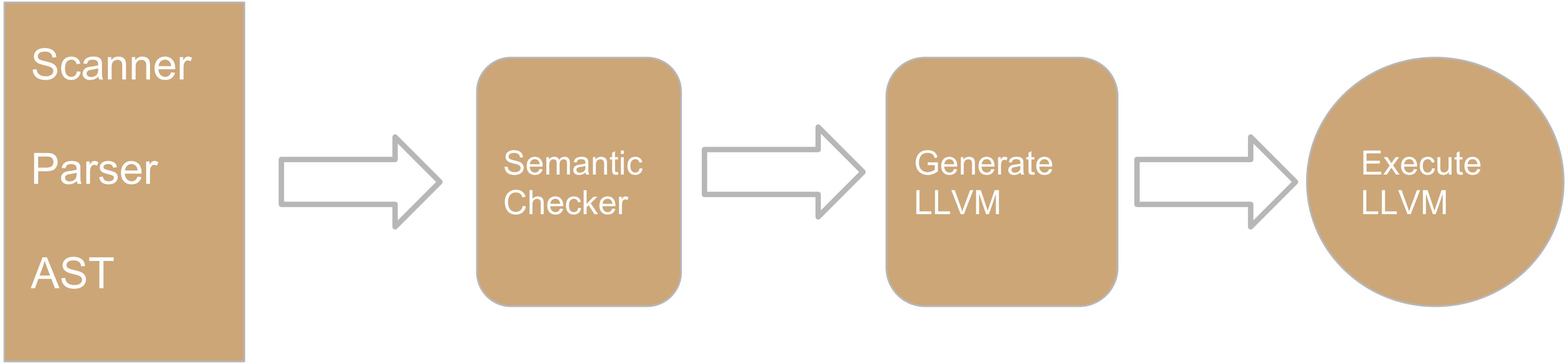


Name	Role/ Responsibilities
Naman Agrawal	Manager / compiler front end; semantics
Vaidehi Dalmia	Tester / test design; code generation
Ganesh Ravichandran	Language Guru / semantics; code generation
David Smart	System Architect / test design; compiler front end
All	Standard Library Functions

Testing

- Test suites were run at each stage.
- We adapted test cases from MicroC and added several of our own for types and standard library functions.
- We adapted the testall.sh script from MicroC for automation.

Architecture Diagram



Scanner

Parser

AST

Semantic Checker

Generate LLVM

Execute LLVM

Thank you! Enjoy the demo!

*Not created with Crayon, but maybe some day!