Week 1

This week served as an introduction to the software training semester. We had a brief discussion on the important relationship between programming and communication. With the sheer size of the code bases for these projects, clearly written code and comments are critical.

We then reviewed the components of the most minimal of a C++ program, and worked through a demo about console input and output.

Finally, we installed the Arduino IDE and set it up for use with the LEGO mindstorms robots we'll be using this semester. See the slides folder for isntructions.

RoboJackets

BATTLEBOTS - OUTREACH - IGVC - ROBOCUP - IARRC

Welcome

Software Training

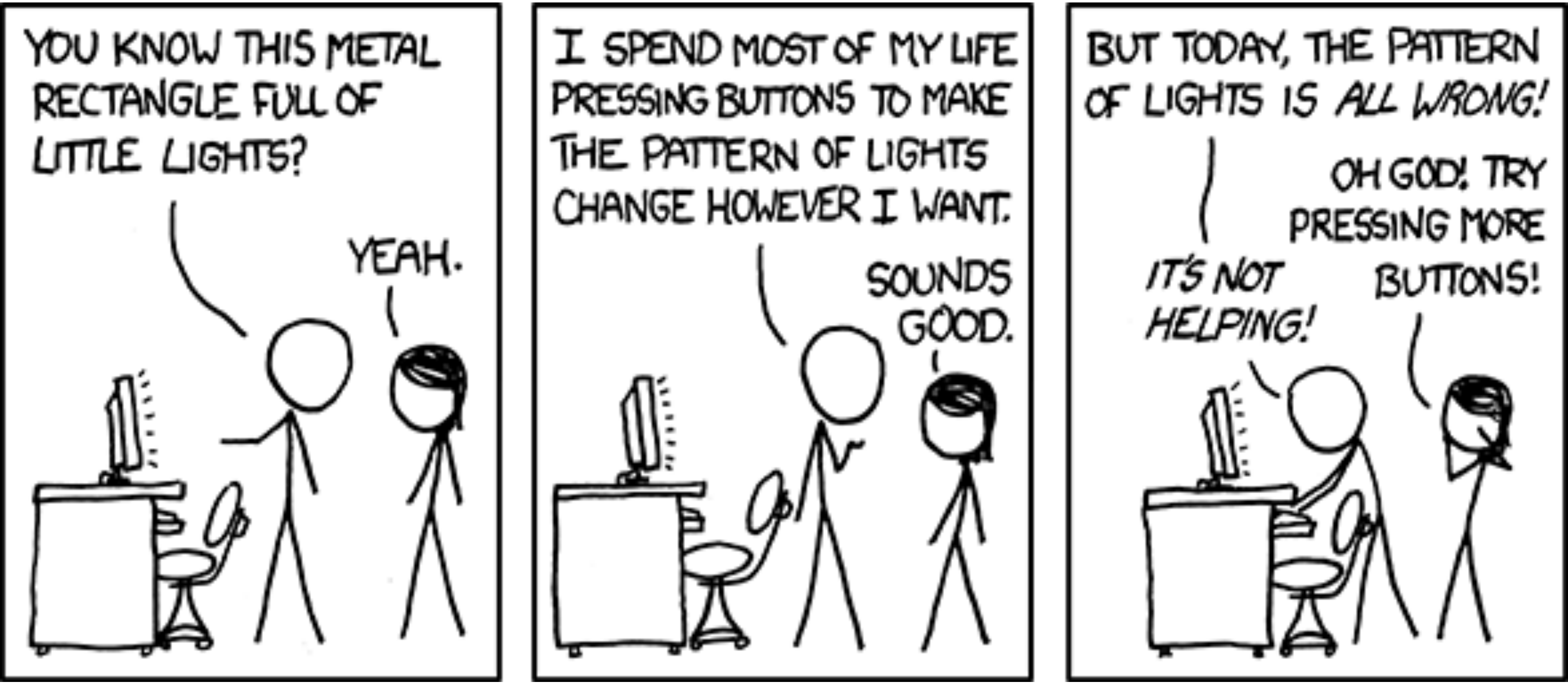
Topics We’ll Cover

- C++
- Git
- CMake
- OpenCV
- Robotics

About Me

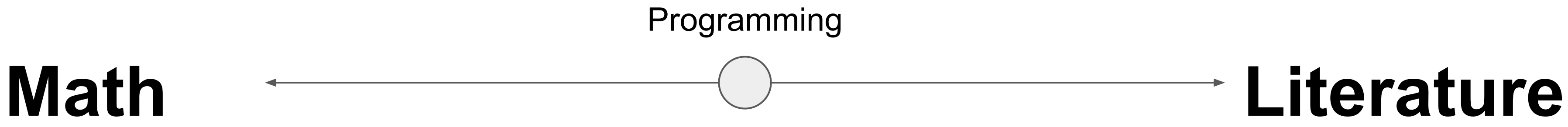
- 4th year CS - Devices & Intelligence
- >10 years in C++ & robotics
- HS experience in Vex, BEST, & FIRST
- History w/ RoboJackets
  - ‘12-’13 : Overactive software member (IGVC / RoboCup)
  - ‘13-’14 : IGVC Project Manager
  - ‘14-’15 : Treasurer
  - ‘15-’16 : President
- Work in CPL on Auto-Rally project

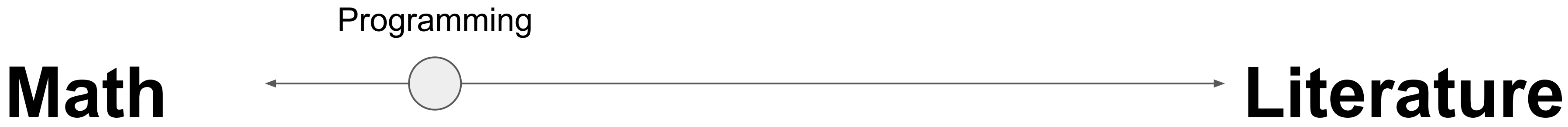
Programming

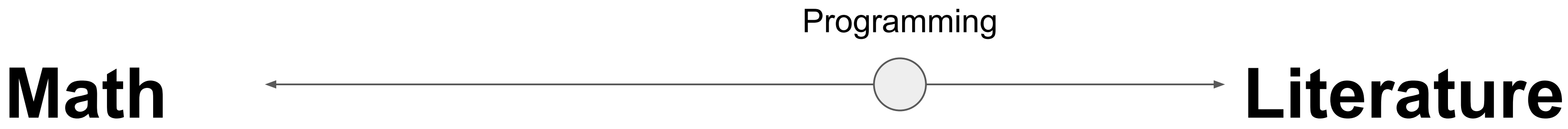

YOU KNOW THIS METAL RECTANGLE FULL OF LITTLE LIGHTS?
YEAH.
I SPEND MOST OF MY LIFE PRESSING BUTTONS TO MAKE THE PATTERN OF LIGHTS CHANGE HOWEVER I WANT.
SOUNDS GOOD.
BUT TODAY, THE PATTERN OF LIGHTS IS *ALL WRONG!*
OH GOD! TRY PRESSING MORE BUTTONS!
IT'S NOT HELPING!

ARDUINO™

+


Programming
Math
Literature


Programming
Math
Literature


Programming
Math
Literature

```
Packet::LogFrame::Robot *log = _state.logFrame->add_self();
*log->mutable_pos() = r->pos;
*log->mutable_vel() = r->vel;
// *log->mutable_cmd_vel() = r->cmd_vel;
// log->set_cmd_w(r->cmd_w);
log->set_shell(r->shell());
log->set_angle(r->angle);

if (r->radioRx().has_kicker_voltage())
{
          log->set_kicker_voltage(r->radioRx().kicker_voltage());
}

if (r->radioRx().has_kicker_status())
{
          log->set_charged(r->radioRx().kicker_status() & 0x01);
          log->set_kicker_works(!(r->radioRx().kicker_status() & 0x90));
}

if (r->radioRx().has_ball_sense_status())
{
          log->set_ball_sense_status(r->radioRx().ball_sense_status());
}

if (r->radioRx().has_battery())
{
          log->set_battery_voltage(r->radioRx().battery());
}

log->mutable_motor_status()->Clear();
log->mutable_motor_status()->MergeFrom(r->radioRx().motor_status());

if (r->radioRx().has_quaternion())
{
          log->mutable_quaternion()->Clear();
          log->mutable_quaternion()->MergeFrom(r->radioRx().quaternion());
} else {
          log->clear_quaternion();
}
```

124885

GitHub changes by barulicm

25

Lines per page

4995

Pages of code by barulicm

Programming is Communication

C++ Basics

An Introduction

Bare bones

```
int main()
{
    return 0;
}
```

Bare bones

```
int main()
{
    return 0;
}
```

Function Declaration
Defines a callable section or "block" of code.

Bare bones

```
int main()
{
    return 0;
}
```

Curly braces are used to mark the beginning and end of blocks.

Bare bones

```
int main()
{
    return 0;
}
```

Name
Gives the function a name so other code can reference and execute it.

Bare bones

```
int main()
{
    return 0;
}
```

Return Type
Sets the type of output that this function will give back to those who call it. In this case, an integer.

Bare bones

```
int main()
{
    return 0;
}
```

Parameter List
Defines the set of inputs needed to call this function.
In this case, no inputs are needed.

Bare bones

```
int main()
{
        return 0;
}
```

Method Body
The section of code executed by the calling of this method.

Bare bones

```
int main()
{
    return 0;
}
```

Return Statement
The ***return*** keyword is used to end the execution of the method and give the specified output back to the caller. In this case, the output is the number zero.

Now for something fun!

Demo

Output