Session 4: Functions, Organization, and Sensors in VEX

Goals

- How header files work in C
- Analog vs Digital signals and sensors in VEX
- Understanding important PROS functions

PROS Project Structure

- PROS projects are made up of three parts
  - PROS Library (/firmware)
  - Header files (/include)
  - User code (/src)

PROS Header Files

- api.h: contains PROS API functions such as `motor_move()`
- main.h: contains user declared code i.e. anything you want to add

Main PROS Functions

- `initialize()`: For setup of sensors and motors
- `opcontrol()`: To control robot motion
- `autonomous()`: For self driving

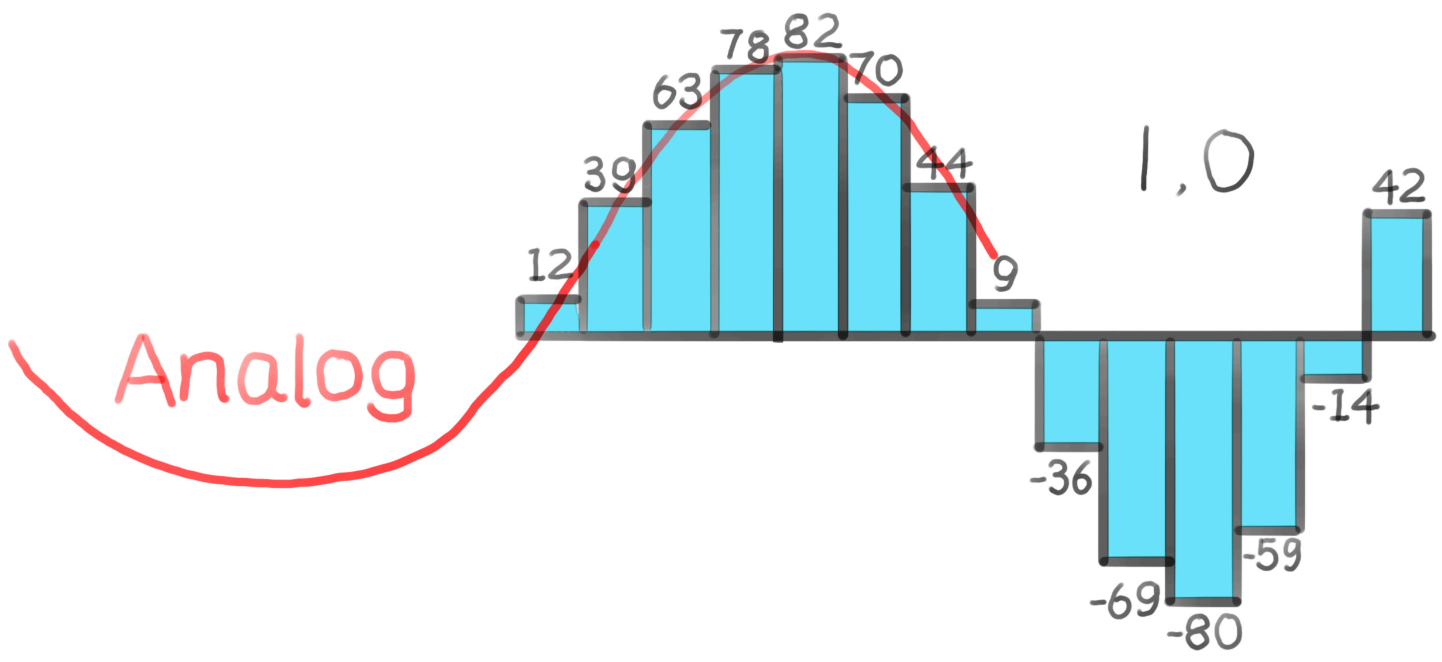
Discussion

- What does `#include "api.h"` do?
- Can you explain what would happen if we did not have `#include "api.h"` in our PROS file?

Demo

- Let's create a new PROS project and see how each of `initialize()`, `opcontrol()`, `autonomous()` work in practice
- We'll also define our own header file and include it in `api.h`

Analog vs. Digital Signals



- Sensors will read incoming signals. Signals can be either analog or digital
- Digital signals are either ON or OFF, only representing a specific set of values
    - Example: light switches

Discussion

- Can you give examples of other analog and digital signals?

Demo

- Let's see which sensors in VEX are considered analog and which are digital

Important Analog Sensors in VEX

- Potentiometer: Measures angular position.
- Light Sensor: Produces an analog value based on the amount of light it detects

Important Digital Sensors in VEX

- Bumper Switch: Acts as a simple digital switch. It's either pressed (1) or not pressed (0)
- Quad Encoder: Measures rotational position of an axle and speed of rotation

Discussion

- Why do you think the light sensor is considered analog, and the bumper switch is considered digital?

Demo

- Let's connect these sensors to our V5 and read their values to see how they are represented in our code. How do you think the values will be represented?