Robot Arm:

↔ like a visual Basic "Class"

CREATING A STRUCTURE & Using it:

Stuff To store in it:

lengths of both "bones"/appendages/arms

~~current~~

~~etc~~

nest? {
- motors ~~etc~~ ports used – can refer to port directly, through name given in ~~motors~~ #pragma config (motors & sensors setup)
- motor encoder ports used
- calibration values
- gear ratios – CREATE A LIST of results!

nest? {
- current length
- start length
- end length

nest? {
- current coordinate (X, Y)
- start coordinate (X, Y)
- end coordinate (X, Y)

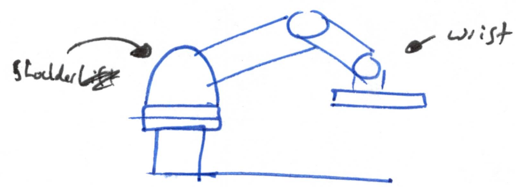
Data Types

- int – integer $2^{-1} \pm 32767$
- (not usable) → ~~double – decimal, more accurate than float~~
- float – floating-point. ~~not as accurate as double~~
- bool – boolean, t/f (true/false)
- byte – number from -128 to 127
- char – single character
- float – decimal, can be larger than 1.
- long – whole number ranging from ±2,147,483,648
- short – ~~is~~ like int
- String – "Lol, what?"

+ Sensors

"standard" RobotC Sensors & pins.

if you set up a touch sensor as "bump" on dgtl1, then you can set a variable (int, integers work) equal to that.



motor control structure –
- encoder port
- motor port
- calibration value
- gear ratio
- ~~encoder~~ value → probably shouldn't do, shouldn't need much updating.

arm length substructure
// – should I include math here?
// let's go with ~~yes~~.

From page on Maintaining Constant Height While Changing Length

☆ need a calibration function!

~~Target angle MA is can be found by~~

current angle MA can be found by getting the degrees from flat of the shoulder.