Pre-Lab #7: Implementing a Simple Counter

L2D
March 30, 2021
Prepared by: Maia, Amanda

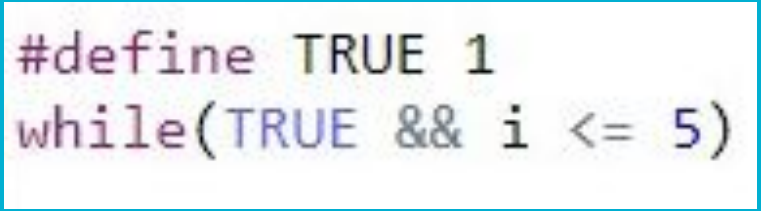
Today we will...

- Review of relevant Part 2 material
  - A note from worksheets regarding boolean expressions
  - Practice writing to 7-seg displays
  - What does channelNumber really mean?
- Review of some code style best practices
- Intro to Pre-Lab/Lab 7
- Look at the DAQlib.h header file

A Note From Worksheet Grading

What’s wrong with this?

- Technically works
- How can we make it better?



```
#define TRUE 1
while(TRUE && i <= 5)
```

Practice Writing to the 7-Segment Displays

Let's write "hello"

To write "h", we can use

0b00101110

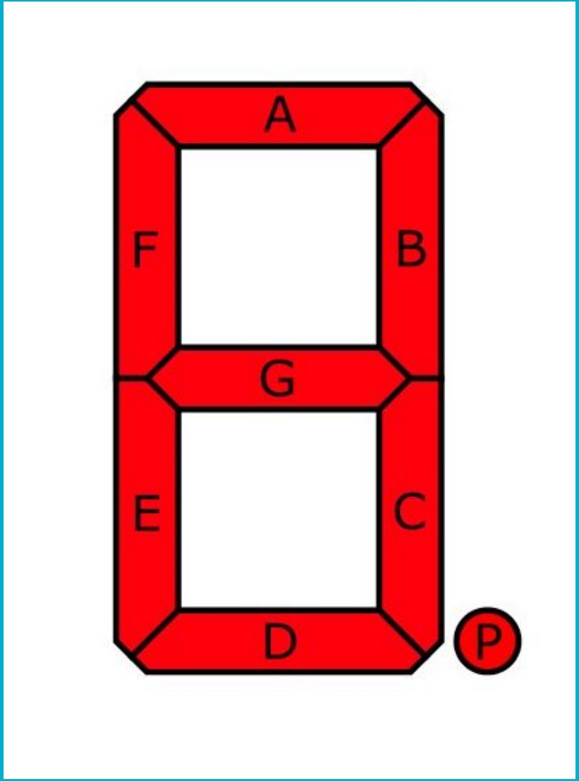
or

46

or even

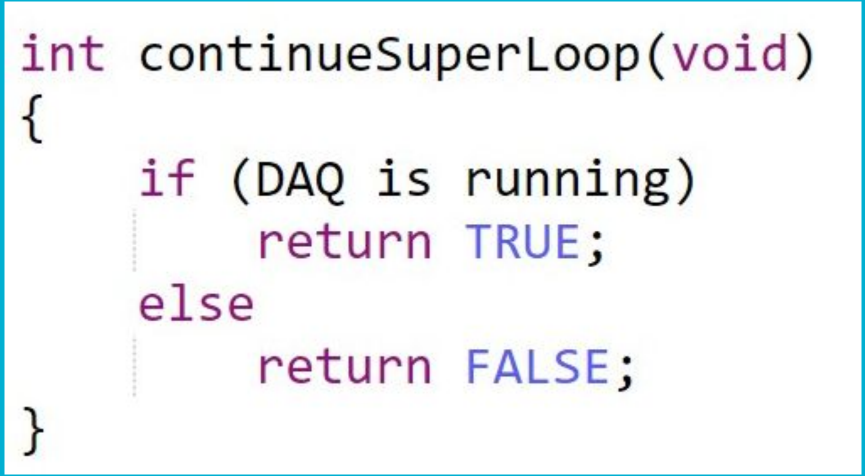
0x2E

^ They all mean the same thing to the computer!



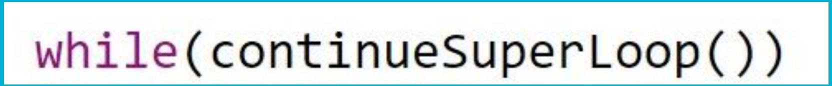
What is continueSuperLoop()?

A way of checking if the DAQ is still runnning



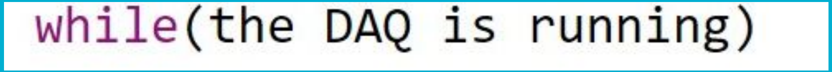
```
int continueSuperLoop(void)
{
    if (DAQ is running)
        return TRUE;
    else
        return FALSE;
}
```

*not actually how continueSuperLoop() is defined.
*not even valid c code



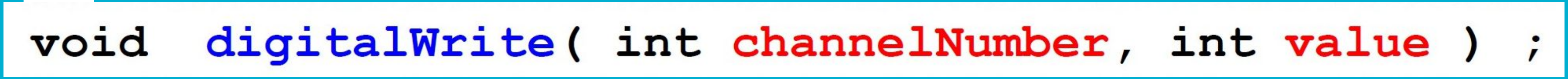
```
while(continueSuperLoop())
```

==



```
while(the DAQ is running)
```

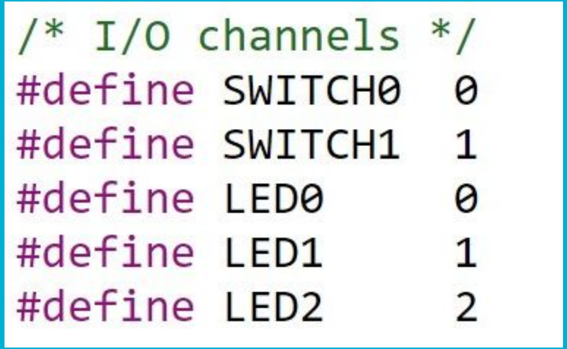
What Does channelNumber Mean?



```
void  digitalWrite( int channelNumber, int value ) ;
```

DAQ channels numbers are pre-set

We will tell you what channel everything is plugged into

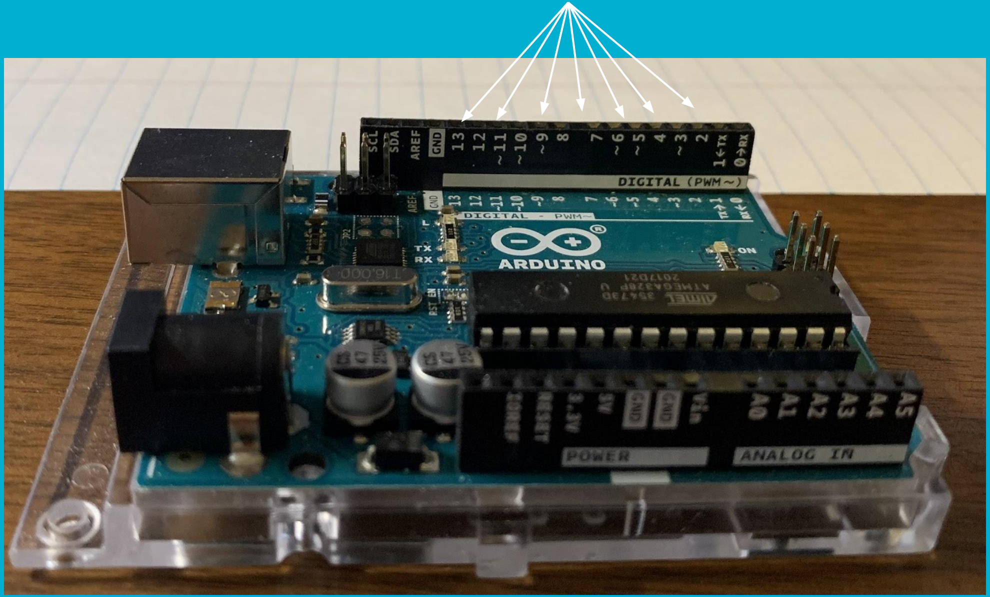


```
/* I/O channels */
#define SWITCH0  0
#define SWITCH1  1
#define LED0     0
#define LED1     1
#define LED2     2
```

What Does channelNumber Mean?

On a physical system...

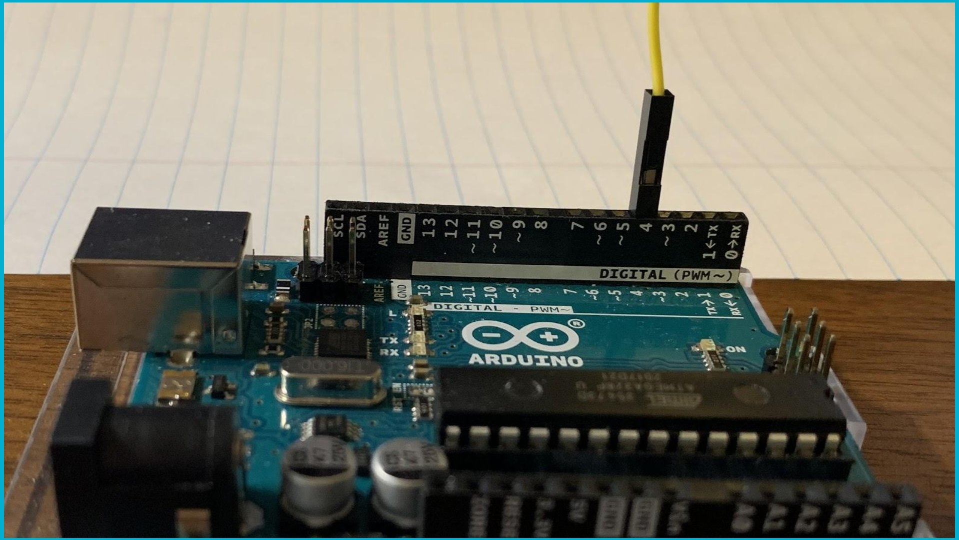
So many channels!



What Does channelNumber Mean?

On a physical system...

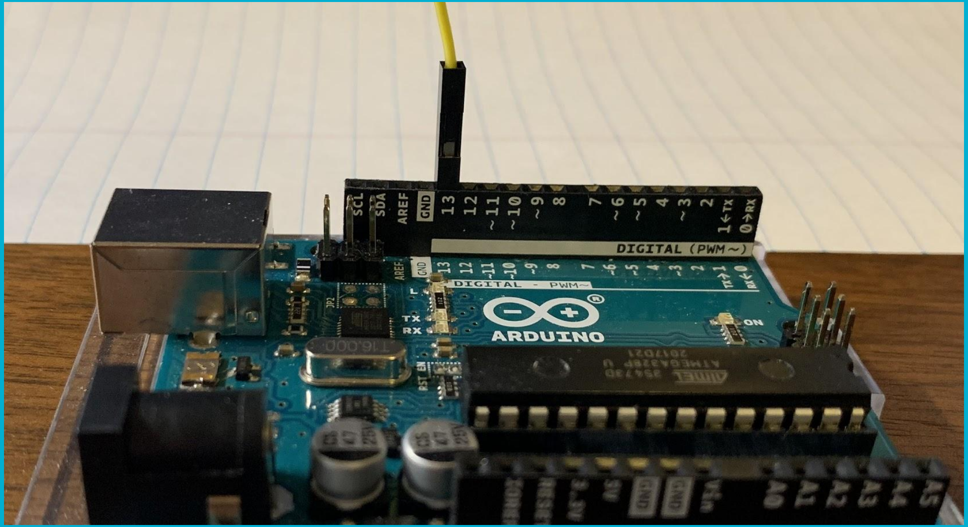
digitalWrite(4, ON)



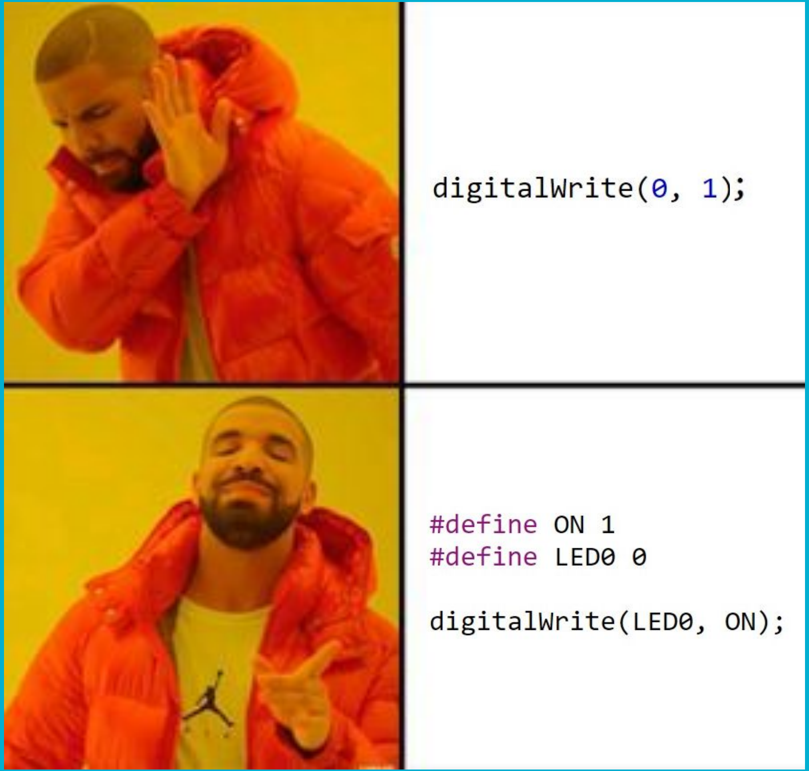
What Does channelNumber Mean?

On a physical system...

digitalWrite(13, ON)

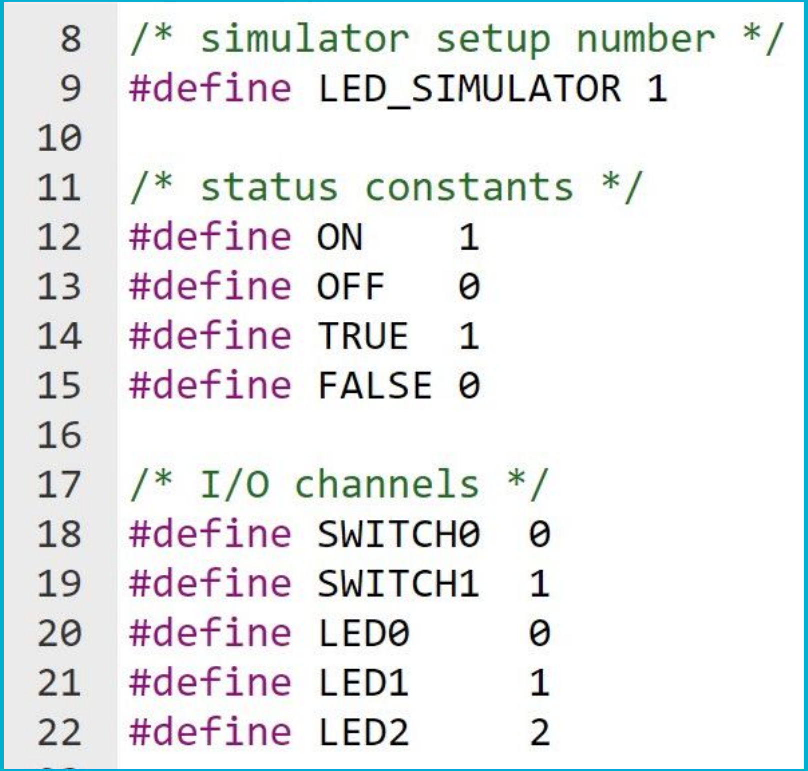


Good Code Style: Symbolic Constants



```
digitalWrite(0, 1);
```

```
#define ON 1
#define LED0 0

digitalWrite(LED0, ON);
```



```
/* simulator setup number */
#define LED_SIMULATOR 1

/* status constants */
#define ON     1
#define OFF    0
#define TRUE   1
#define FALSE 0

/* I/O channels */
#define SWITCH0  0
#define SWITCH1  1
#define LED0     0
#define LED1     1
#define LED2     2
```

Good Code Style: Good Commenting Habits

Not too many, not too few...

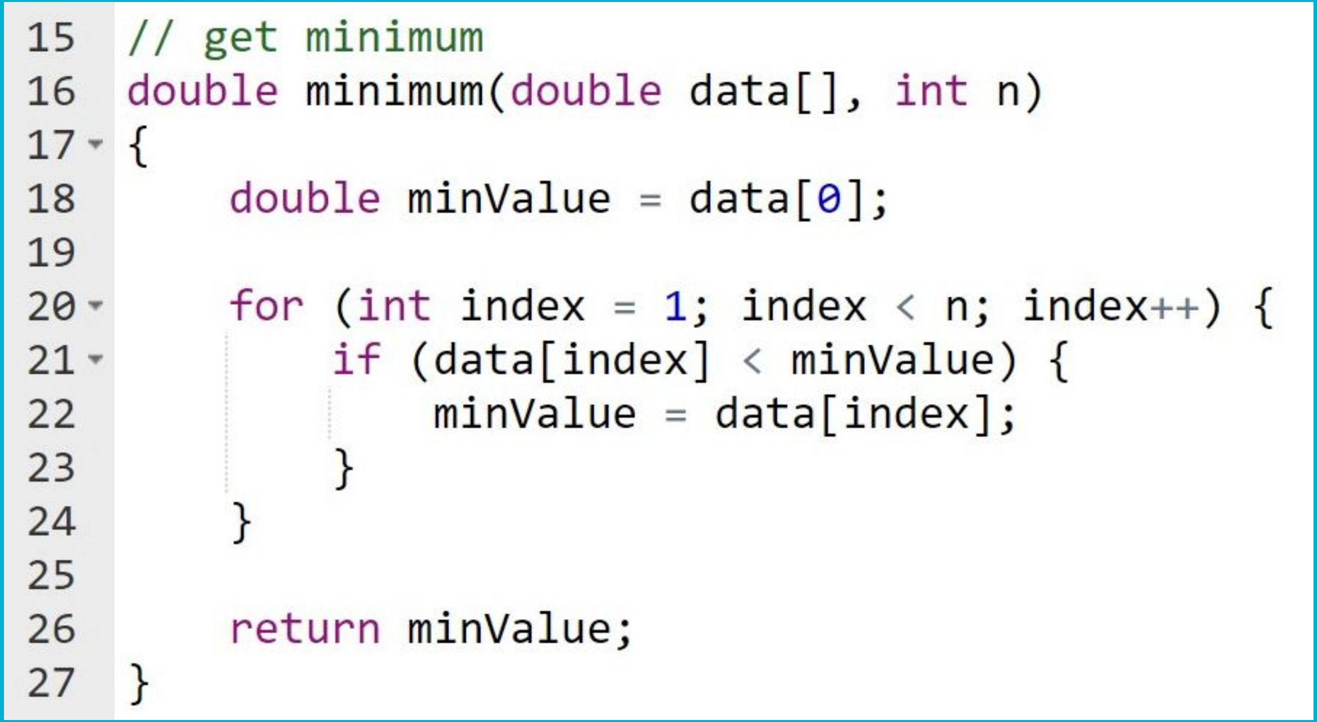
- Opening documentation
- Function documentation
- Blocks of code

Why?

- Help your teammates/co-workers/future self understand your code
- Maximize your marks on the manual grading!

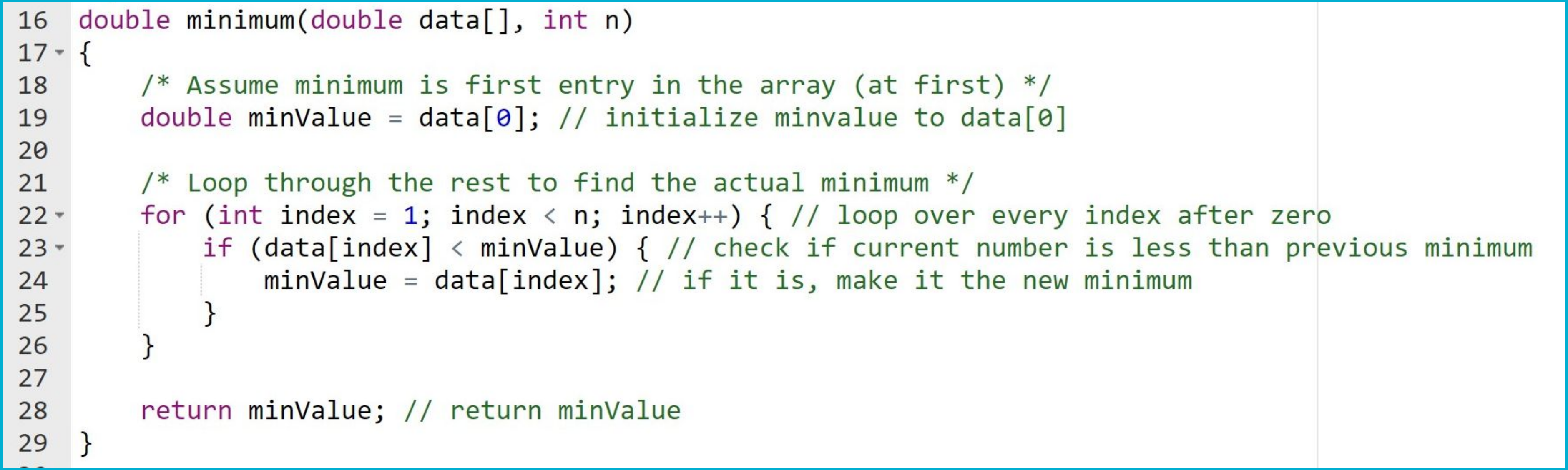
Too Few

(But better than none!)



```
// get minimum
double minimum(double data[], int n)
{
    double minValue = data[0];

    for (int index = 1; index < n; index++) {
        if (data[index] < minValue) {
            minValue = data[index];
        }
    }

    return minValue;
}
```

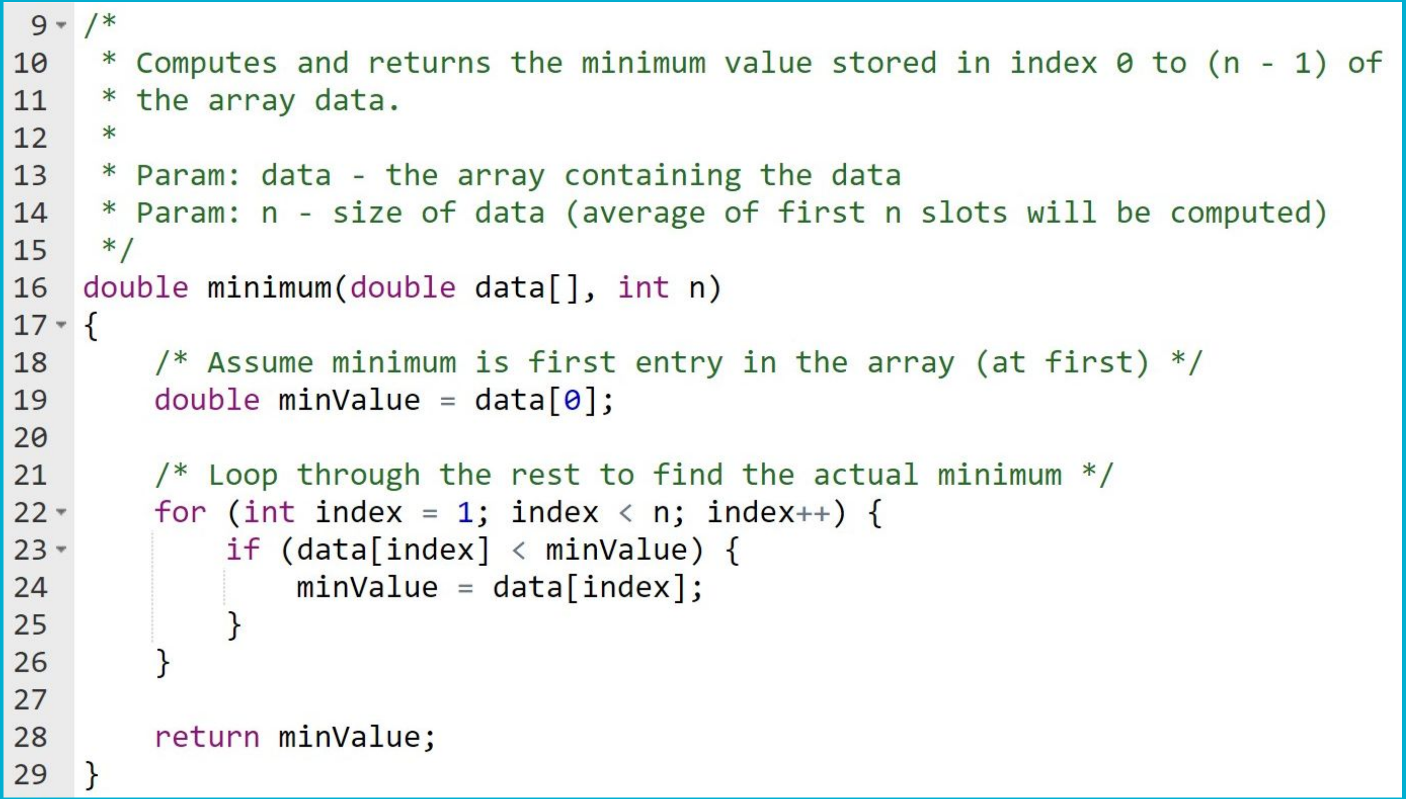
Too Many



```
double minimum(double data[], int n)
{
    /* Assume minimum is first entry in the array (at first) */
    double minValue = data[0]; // initialize minvalue to data[0]

    /* Loop through the rest to find the actual minimum */
    for (int index = 1; index < n; index++) { // loop over every index after zero
        if (data[index] < minValue) { // check if current number is less than previous minimum
            minValue = data[index]; // if it is, make it the new minimum
        }
    }

    return minValue; // return minValue
}
```

Just Right

(And function documentation!)

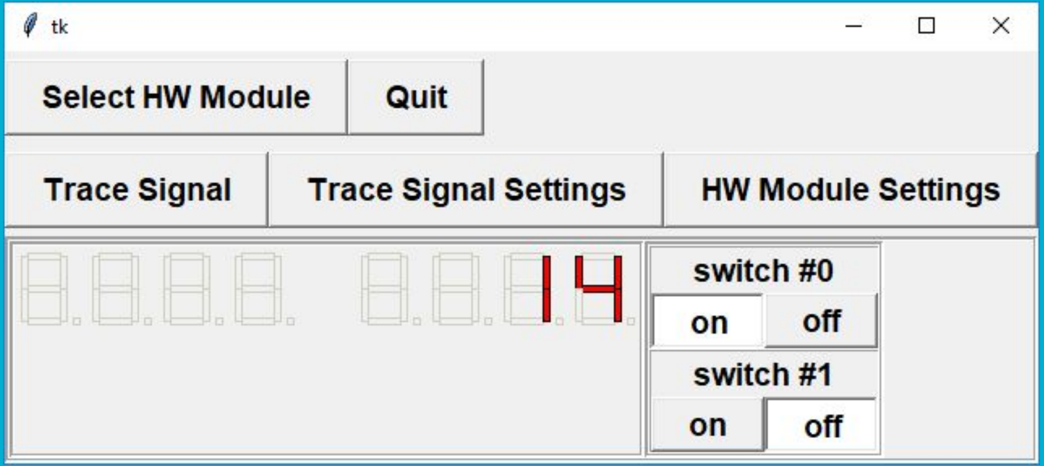


```
/*
 * Computes and returns the minimum value stored in index 0 to (n - 1) of
 * the array data.
 *
 * Param: data - the array containing the data
 * Param: n - size of data (average of first n slots will be computed)
 */
double minimum(double data[], int n)
{
    /* Assume minimum is first entry in the array (at first) */
    double minValue = data[0];

    /* Loop through the rest to find the actual minimum */
    for (int index = 1; index < n; index++) {
        if (data[index] < minValue) {
            minValue = data[index];
        }
    }

    return minValue;
}
```

Back to Pre-Lab 7...

Learning Goals

1. To write a C program that uses switches and LED displays of the DAQ module (data acquisition, I/O)
2. To implement a simple counter



Steps for Writing Programs

1. Understand the problem
2. Think through your algorithm
3. Come up with a test suite (both valid inputs and edge cases)
4. Code your algorithm
5. Test your algorithm

General DAQ Tips

- Define more functions! Break the problem down into smaller steps
- Write comments throughout your code
- Use example programs to help get started

- Don’t be afraid to ask questions about previous topics in the course

Lastly, let’s check out the DAQlib.h header file…

Thanks for listening! 😊

Feel free to ask any Pre-Lab/Lab #7 related questions, or really any other questions you might have!

Relevant lab prep materials:

- Relevant lecture notes (such as on the Display, Digital IO, and Sleep()) and class activities
- Review of the programming concepts such as modular programming with functions, repetition, branching, and arrays