Lab 05: Conditionals

(What’s up?)

# Objectives:

* Develop problem-solving skills
* Develop skills in using if statements in C
* Understand how to interpret the acceleration values in terms of gravity
* Understand tolerances and testing in relation to floating point values
* Understand testing multiple conditions

# Starting Point:

* [lab05.c](https://drive.google.com/open?id=1SpRgiO-W62XSHGJ6Pz0k2HAlHFF9JUpX)
* You will need magnitude function from lab03-1.c
* **Read** the *entire lab* **before** you start coding

# Turn-In:

* Your lab report including **answers to the questions** at the bottom of the lab manual and **your source code**. Check the rubric for grading details. **Remember to use the lab report format!**
* Demonstrate your working program to undergraduate TA, during lab or during office hours (before the start of next lab).
* **Upload one .zip** file of your **lab05** folder. Title this **firstname\_lastname\_lab05.zip**. Before zipping, delete all generated .exe files (not the **ds4rd.exe**).

# Process:

## Creating a New Folder

Create a new folder named **lab05** in your **se185**folder on the **U:** drive. You will want to copy over **ds4rd.exe** to the **lab05**folder.

## Basic Steps

1. Download **lab05.c** to your **lab05** folder.
2. The “Development Tips” section at the bottom of the lab manual should be read before continuing, as they will provide some insight into the lab requirements.
3. Once you’ve compiled your program. Run/test it by using the command:

**./ds4rd.exe -d 054c:05c4 -D DS4\_BT -a -g -b | ./lab05**

Remember to replace DS4\_BT with DS4\_USB if using the DS4 over USB, and lab05 is replaced with whatever you named your compiled program.

1. Hit **ctrl+c** to stop the program (you will add a feature later to end the program with a button).

## Problem

We will learn how to determine how the DualShock 4 is oriented and how to interpret the acceleration values in more detail. For this problem, we will calculate the magnitude of the acceleration from the DualShock 4. You will read from ds4rd.exe, but this time you will read the inputs over and over inside a “while” loop. Your code should be written in a modular fashion and **include 3 or more** of your own functions.

*When the DualShock 4 is not moving*, your program should output a line saying which side of the DualShock 4 is facing up. Consider using the magnitude of the acceleration (-a) for movement detection. You will need a tolerance value for this---experiment to find one that seems reasonable. You will need to use the gyroscope values (-g) to determine the orientation. Again, you may want a separate tolerance value. (see development tips below about the close\_to function and tolerance values)

For instance, when lying flat and still on the table with buttons up, it should output “**TOP**”. Flipping the DualShock 4 over, when still, it should output “**BOTTOM**.” When setting the left side of the DualShock 4 (i.e. side with the arrow buttons) down and the right side up, your program should output “**LEFT**”. When flipped (i.e. side with the shape buttons down), it should output “**RIGHT**.” When the LED light bar is pointed up at the ceiling, output “**FRONT**” and similarly when the LED light bar is down to the floor it should read “**BACK**.”

## Requirements

**Basics**

* Program outputs orientation of DualShock when *not moving*
* Program includes and uses **3 or more** of your own functions

**Feature 1**

Once your code does the above (displays which side is facing up), modify the while loop to stop and end the program when the user clicks the **TRIANGLE BUTTON** on the DualShock 4.

**Feature 2**

Modify your code so that the program only outputs a new line when its orientation changes. (e.g. “**TOP**” shouldn’t appear multiple times in a row)

**Demonstrate your working program (with the required features above) to a TA. Ensure the undergraduate TA has entered the appropriate points into the “Lab 05: Demo” column in “Grades” on Canvas.** This may be done during a TA's office hours if you do not complete the program in class. **It must be done before your next lab period**.

## Development Tips

* You will need to frequently compare two real numbers within some tolerance. Write such a function with the following prototype:

**int close\_to (double tolerance, double point, double value);**

* It should return **true** when **value** is within **tolerance** of **point**. For example, if **point** is **1.0** and **tolerance** is **0.25**, then close**\_**to should return true when value is from **0.75** to **1.25**, otherwise, it should **return false**.

Use **close\_to** to implement your real number comparisons in your code.

In C, **0** is evaluated as **false**, and any other number is **true** (Usually we use **1**)

* HINT: Write **close\_to** first and test it. Then implement the program. You will probably want a much smaller tolerance than this example uses.
* You can capture raw data from the controller by running the following command:

**./ds4rd.exe -d 054c:05c4 -D DS4\_BT -a -g -b > *output.csv***

* You can then send this data into your program using the following command:

**./yourprogram.exe < output.csv**

This is useful, as it allows you to record a sequence of movements of your controller and simulate running those movements on your program without physically having a controller. This is helpful if you want to run pre-recorded samples of inputs against your program without using the DS4 and the DS4RD program.

* Link to a document that explains a bit more about redirection: [Redirection](https://drive.google.com/open?id=13iq_MKNZDRAL6TeU3TSLxaIv_tVhClROVRaq0w1UNQQ)

*Note: When the captured data is run through your program, your program should detect that the controller is starting with the right side up, then the left side, and then terminate due to the triangle button press. The captured data is simply an example and does not guarantee that your program will work on an actual DS4.* ***Your program must work on an actual controller****, and the example data is simply another aid.*

## Questions and Experiments

1. How did you approach the design?
2. What data did you have to read in?
3. What functions did you choose to implement and why?
4. What tolerance values did you pick and how did you decide on them?

**Be sure to include your answers in the appropriate section of your lab report. If you don’t see where it should go, please put it in the comments section.**

## Ending the Lab Session:

1. Be sure you will be able to have access to your code and data when you work on your lab report.
2. Leave your source code on your U: drive.
3. **Plug in the DS4 into its charging station!**
4. Log off before leaving your workstation in the lab!