**Mod 8: Portfolio Milestone**

Alex Mehler

CSU Global

CSC450-1

Dr. Bindu George

July 7, 2024

**Execution:**

**A screenshot of a computer

Description automatically generated**

**Code:**

**Main:**

public class concurrency {

public static void main(String[] args) throws InterruptedException {

Count1 c1 = new Count1();

Count2 c2 = new Count2();

Thread t1 = new Thread(c1);

Thread t2 = new Thread(c2);

t1.start();

t1.join();

System.***out***.println("Thread 2 start: ");

t2.start();

t2.join();

}

}

**Count1:**

public class Count1 implements Runnable{

int c1 = 0;

public void run() {

while (c1 < 21) {

System.***out***.println(c1);

c1++;

}

}

}

**Count2:**

public class Count2 implements Runnable{

int c2 = 20;

public void run() {

while (c2 > 0) {

System.***out***.println(c2);

c2--;

}

System.***out***.println(c2);

}

}

**Analysis**

Java handles multithreading easily, and this project was a breeze. After making a few count modules to hold the code for the up and down counts, I created the main thread. The main thread handled the operation of designating the threads that handled the work and running the 2 count modules. With all of the classes being public, the program would be very vulnerable to exploitation. This could be easily mitigated by making the classes and variables private. Otherwise, the application is very light weight in operation. Increasing the thread usage and complexity of each module would increase the overhead, but again that’s not a problem with this specific problem.

**Git:**

**A screenshot of a computer

Description automatically generated**

<https://github.com/SevRnce/c_mod_7>