

```

***** Name: Makya Stell *****  

Name: Makya Stell  

Date: 3/26/2021 @23:59  

Assignment #3 - Digital Thermometer  

*****  

#include "mbed.h"

//Takes in the VOUT from the MCP9701A
AnalogIn ain(p19);
//Changes the output to Celcius if GND and Fahrenheit if VOUT
DigitalIn degree(p21);
//Allows it to print values to the TeraTerm
Serial pc(USBTX, USBRX);

//Segment Code
BusOut display(p5,p6,p7,p8,p9,p10,p11,p12); // A,B,C,D,E,F,G,DP
//Function for segment characters
char SegConvert(char SegValue);

int main() {
    pc.baud(9600);
    while (1) {
        float v_temp, volts, temp, s;
        s = 0;
        for(int i = 0; i <= 1000; i++)
        {
            v_temp = ain;
            s = s + v_temp;
        }

    **** FOR CELCIUS *****
    //Calculating celcius, so that I can interchange it between F and C
    v_temp = s/1000;
    volts = v_temp * 3.3;
    temp = (volts - 0.4)/0.0195;
    //Will return the temperature in degrees celcius
    int C = temp + 0.5;

    //Printing the values in degrees C
    if (degree == 0)
    {
        pc.printf("Original Temp = %f | Volts = %f | Temp = %i Celcius \n\r",
v_temp, volts, C);
        //Printing to the Segment 1 Character at a time
        //Gets # in the hundreds place
        char h = C/100;
        //Gets # in the tens place
        char t = ((C%100)/10);
        //Gets # in the ones place
    }
}

```

```

char o = C%10;

//Will only happen if C is a negative number
if (C < 0)
{
    //Takes the negative out of F if it's negative
    C = abs(C);
    //Gets # in the hundreds place
    h = C/100;
    //Gets # in the tens place
    t = ((C%100)/10);
    //Gets # in the ones place
    o = C%10;
    //Displays a negative sign on the segment
    display = 0b01000000;
    wait(0.5);
    display = 0b00000000;
    wait(0.1);
}

//Displays # in the hundreds place
display = SegConvert (h);
wait(0.5);
display = 0b00000000;
wait(0.1);
//Displays # in the tens place
display = SegConvert(t);
wait(0.5);
display = 0b00000000;
wait(0.1);
//Displays # in the ones place
display = SegConvert(o);
wait(0.5);
display = 0b00000000;
wait(0.1);
//Displays a C for Celcius
display = SegConvert('C');
wait(0.5);
display = 0b00000000;
wait(0.1);
//Turns off the display for a restart
display = 0b00000000;
wait(0.5);
}

***** FOR FARHRENHEIT *****

//Printing the values in degrees F
else if (degree == 1)

```

```

{
    int F = ((C * 9)/5)+32;
    pc.printf("Original Temp = %f | Volts = %f | Temp = %i Fahrenheit \n\r",
v_temp, volts, F);

    //Gets # in the hundreds place
    char h = F/100;
    //Gets # in the tens place
    char t = ((F%100)/10);
    //Gets # in the ones place
    char o = F%10;

    //Will only happen if F is negative
    if (F < 0)
    {
        //Takes the negative out of F if it's negative
        F = abs(F);
        //Gets # in the hundreds place
        h = F/100;
        //Gets # in the tens place
        t = ((F%100)/10);
        //Gets # in the ones place
        o = F%10;
        //Displays a negative sign on the segment
        display = 0b01000000;
        wait(0.5);
        display = 0b00000000;
        wait(0.1);
    }

    //Displays # in the hundreds place
    display = SegConvert (h);
    wait(0.5);
    display = 0b00000000;
    wait(0.1);
    //Displays # in the tens place
    display = SegConvert(t);
    wait(0.5);
    display = 0b00000000;
    wait(0.1);
    //Displays # in the ones place
    display = SegConvert(o);
    wait(0.5);
    display = 0b00000000;
    wait(0.1);
    //Displays an F for Fahrenheit
    display = SegConvert('F');
    wait(0.5);
    display = 0b00000000;
    wait(0.1);
}

```

```

        //Turns off the display for a restart
        display = 0b00000000;
        wait(0.5);
    }
}
}

//***** SEGMENT VALUES *****/
//Got help from the book on Binary Declarations and function
//Function 'SegConvert'
char SegConvert(char SegValue) {
    //Begins the character at null or blank
    char SegByte=0x00;
    switch (SegValue)
    {
        //DP G F E D C B A
        case 0 : SegByte = 0x3F;break; // 0 0 1 1 1 1 1 1 binary
        case 1 : SegByte = 0x06;break; // 0 0 0 0 0 1 1 0 binary
        case 2 : SegByte = 0x5B;break; // 0 1 0 1 1 0 1 1 binary
        case 3 : SegByte = 0x4F;break; // 0 1 0 0 1 1 1 1 binary
        case 4 : SegByte = 0x66;break; // 0 1 1 0 0 1 1 0 binary
        case 5 : SegByte = 0x6D;break; // 0 1 1 0 1 1 0 1 binary
        case 6 : SegByte = 0x7D;break; // 0 1 1 1 1 1 0 1 binary
        case 7 : SegByte = 0x07;break; // 0 0 0 0 0 1 1 1 binary
        case 8 : SegByte = 0x7F;break; // 0 1 1 1 1 1 1 1 binary
        case 9 : SegByte = 0x6F;break; // 0 1 1 0 1 1 1 1 binary
        case 'F': SegByte = 0x71;break;// 0 1 1 1 0 0 0 1 binary
        case 'C': SegByte = 0x39;break;// 0 0 1 1 1 0 0 1 binary
    }
    return SegByte;
}

```