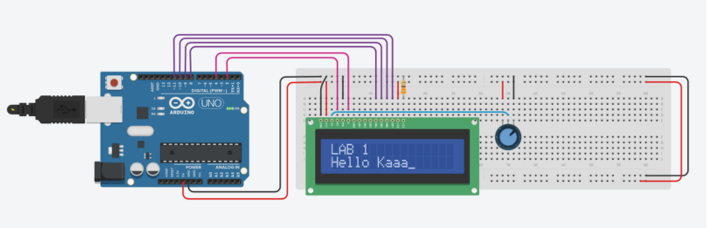
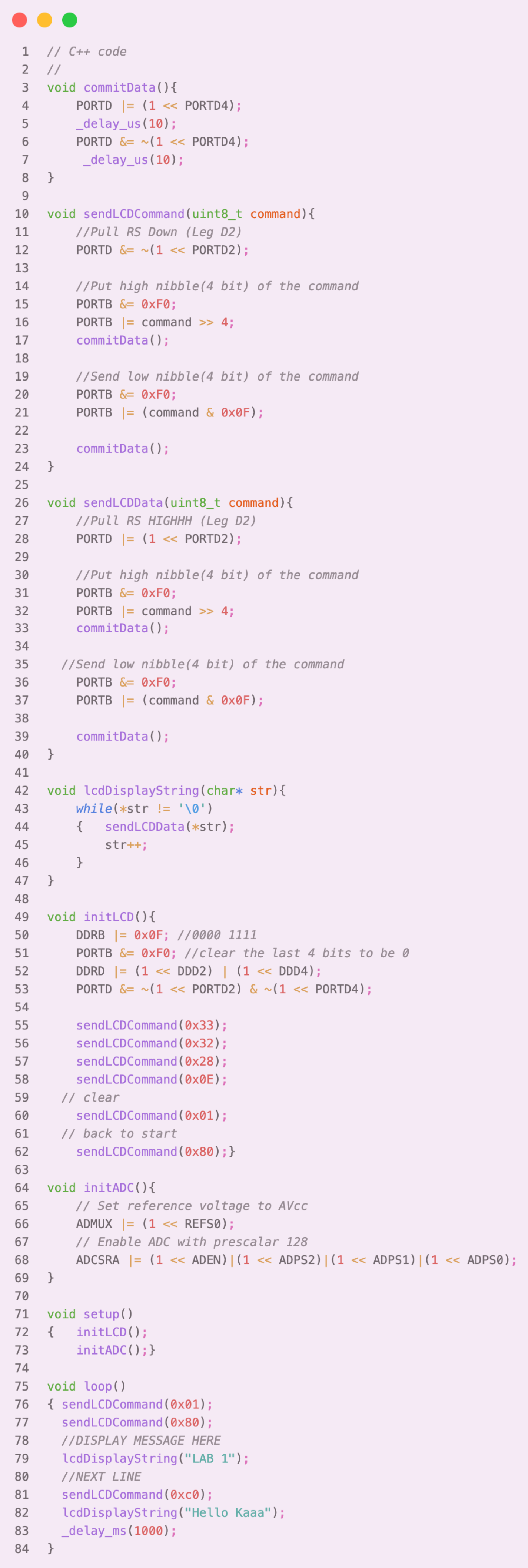
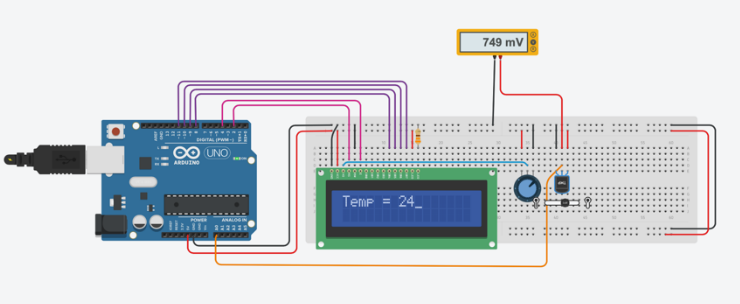
LAB1: LCD Display Interfacing

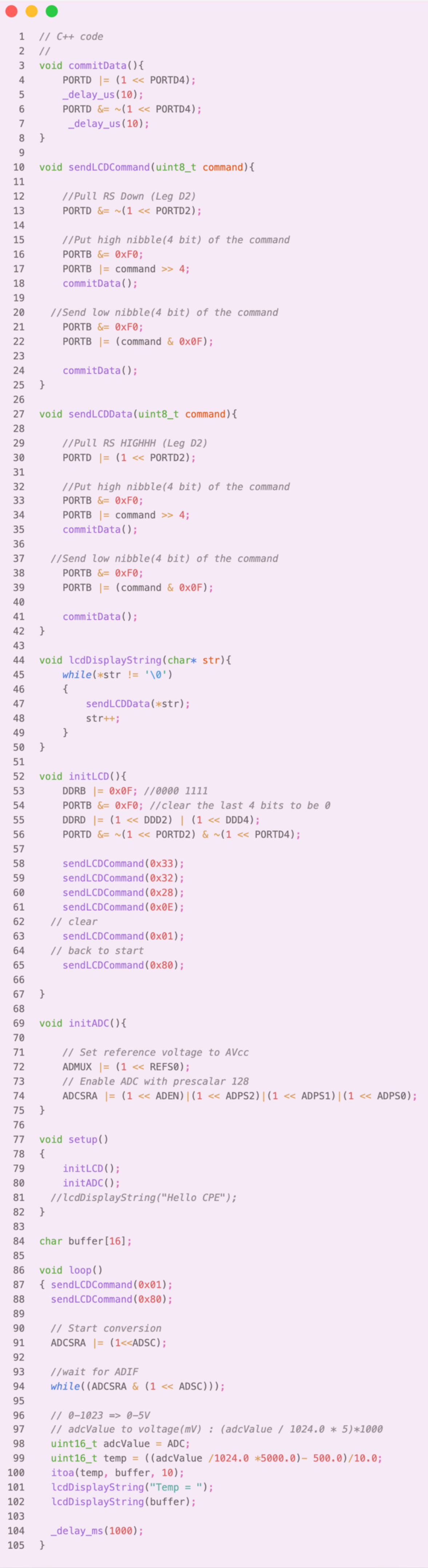




```
// C++ code
//
void commitData(){
    PORTD |= (1 << PORTD4);
    _delay_us(10);
    PORTD &= ~(1 << PORTD4);
     _delay_us(10);
}

void sendLCDCommand(uint8_t command){
    //Pull RS Down (Leg D2)
    PORTD &= ~(1 << PORTD2);

    //Put high nibble(4 bit) of the command
    PORTB &= 0xF0;
    PORTB |= command >> 4;
    commitData();

    //Send low nibble(4 bit) of the command
    PORTB &= 0xF0;
    PORTB |= (command & 0x0F);

    commitData();
}

void sendLCDData(uint8_t command){
    //Pull RS HIGHHH (Leg D2)
    PORTD |= (1 << PORTD2);

    //Put high nibble(4 bit) of the command
    PORTB &= 0xF0;
    PORTB |= command >> 4;
    commitData();

  //Send low nibble(4 bit) of the command
    PORTB &= 0xF0;
    PORTB |= (command & 0x0F);

    commitData();
}

void lcdDisplayString(char* str){
    while(*str != '\0')
    {   sendLCDData(*str);
        str++;
    }
}

void initLCD(){
    DDRB |= 0x0F; //0000 1111
    PORTB &= 0xF0; //clear the last 4 bits to be 0
    DDRD |= (1 << DDD2) | (1 << DDD4);
    PORTD &= ~(1 << PORTD2) & ~(1 << PORTD4);

    sendLCDCommand(0x33);
    sendLCDCommand(0x32);
    sendLCDCommand(0x28);
    sendLCDCommand(0x0E);
  // clear
    sendLCDCommand(0x01);
  // back to start
    sendLCDCommand(0x80);}

void initADC(){
    // Set reference voltage to AVcc
    ADMUX |= (1 << REFS0);
    // Enable ADC with prescalar 128
    ADCSRA |= (1 << ADEN)|(1 << ADPS2)|(1 << ADPS1)|(1 << ADPS0);
}

void setup()
{   initLCD();
    initADC();}

void loop()
{ sendLCDCommand(0x01);
  sendLCDCommand(0x80);
  //DISPLAY MESSAGE HERE
  lcdDisplayString("LAB 1");
  //NEXT LINE
  sendLCDCommand(0xc0);
  lcdDisplayString("Hello Kaaa");
  _delay_ms(1000);
}
```

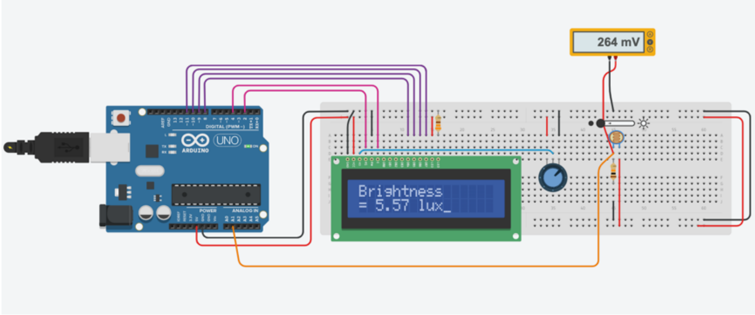
LAB2: AVR ADC Programming

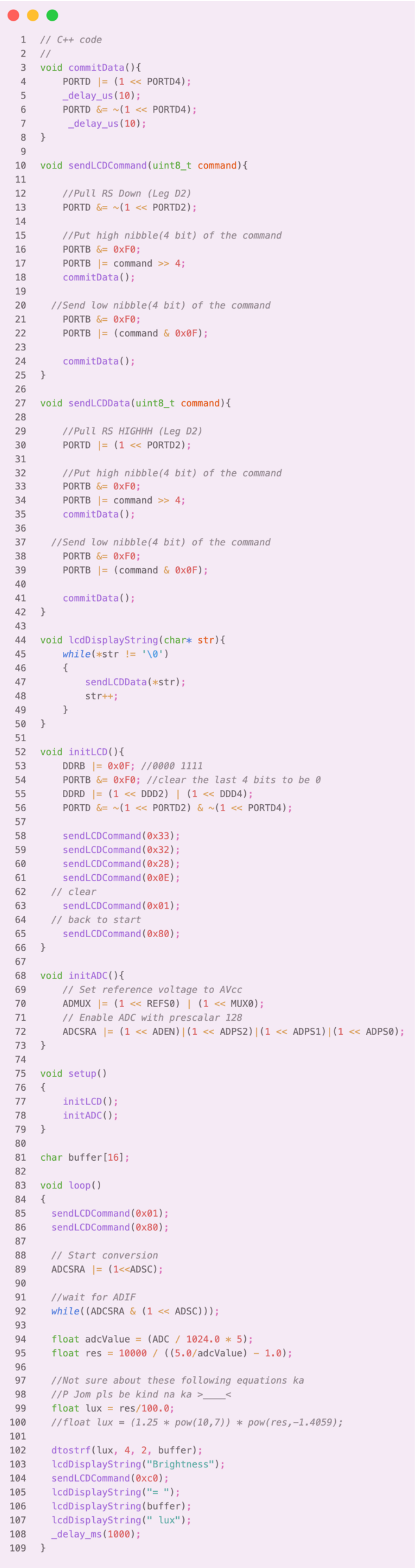




```
// C++ code
//
void commitData(){
    PORTD |= (1 << PORTD4);
    _delay_us(10);
    PORTD &= ~(1 << PORTD4);
     _delay_us(10);
}

void sendLCDCommand(uint8_t command){

    //Pull RS Down (Leg D2)
    PORTD &= ~(1 << PORTD2);

    //Put high nibble(4 bit) of the command
    PORTB &= 0xF0;
    PORTB |= command >> 4;
    commitData();

  //Send low nibble(4 bit) of the command
    PORTB &= 0xF0;
    PORTB |= (command & 0x0F);

    commitData();
}

void sendLCDData(uint8_t command){

    //Pull RS HIGHHH (Leg D2)
    PORTD |= (1 << PORTD2);

    //Put high nibble(4 bit) of the command
    PORTB &= 0xF0;
    PORTB |= command >> 4;
    commitData();

  //Send low nibble(4 bit) of the command
    PORTB &= 0xF0;
    PORTB |= (command & 0x0F);

    commitData();
}

void lcdDisplayString(char* str){
    while(*str != '\0')
    {
        sendLCDData(*str);
        str++;
    }
}

void initLCD(){
    DDRB |= 0x0F; //0000 1111
    PORTB &= 0xF0; //clear the last 4 bits to be 0
    DDRD |= (1 << DDD2) | (1 << DDD4);
    PORTD &= ~(1 << PORTD2) & ~(1 << PORTD4);

    sendLCDCommand(0x33);
    sendLCDCommand(0x32);
    sendLCDCommand(0x28);
    sendLCDCommand(0x0E);
  // clear
    sendLCDCommand(0x01);
  // back to start
    sendLCDCommand(0x80);

}

void initADC(){

    // Set reference voltage to AVcc
    ADMUX |= (1 << REFS0);
    // Enable ADC with prescalar 128
    ADCSRA |= (1 << ADEN)|(1 << ADPS2)|(1 << ADPS1)|(1 << ADPS0);
}

void setup()
{
    initLCD();
    initADC();
  //lcdDisplayString("Hello CPE");
}

char buffer[16];

void loop()
{ sendLCDCommand(0x01);
  sendLCDCommand(0x80);

  // Start conversion
  ADCSRA |= (1<<ADSC);

  //wait for ADIF
  while((ADCSRA & (1 << ADSC)));

  // 0-1023 => 0-5V
  // adcValue to voltage(mV) : (adcValue / 1024.0 * 5)*1000
  uint16_t adcValue = ADC;
  uint16_t temp = ((adcValue /1024.0 *5000.0)- 500.0)/10.0;
  itoa(temp, buffer, 10);
  lcdDisplayString("Temp = ");
  lcdDisplayString(buffer);

  _delay_ms(1000);
}
```

LAB3: More AVR ADC Programming





```
// C++ code
//
void commitData(){
    PORTD |= (1 << PORTD4);
    _delay_us(10);
    PORTD &= ~(1 << PORTD4);
     _delay_us(10);
}

void sendLCDCommand(uint8_t command){

    //Pull RS Down (Leg D2)
    PORTD &= ~(1 << PORTD2);

    //Put high nibble(4 bit) of the command
    PORTB &= 0xF0;
    PORTB |= command >> 4;
    commitData();

  //Send low nibble(4 bit) of the command
    PORTB &= 0xF0;
    PORTB |= (command & 0x0F);

    commitData();
}

void sendLCDData(uint8_t command){

    //Pull RS HIGHHH (Leg D2)
    PORTD |= (1 << PORTD2);

    //Put high nibble(4 bit) of the command
    PORTB &= 0xF0;
    PORTB |= command >> 4;
    commitData();

  //Send low nibble(4 bit) of the command
    PORTB &= 0xF0;
    PORTB |= (command & 0x0F);

    commitData();
}

void lcdDisplayString(char* str){
    while(*str != '\0')
    {
        sendLCDData(*str);
        str++;
    }
}

void initLCD(){
    DDRB |= 0x0F; //0000 1111
    PORTB &= 0xF0; //clear the last 4 bits to be 0
    DDRD |= (1 << DDD2) | (1 << DDD4);
    PORTD &= ~(1 << PORTD2) & ~(1 << PORTD4);

    sendLCDCommand(0x33);
    sendLCDCommand(0x32);
    sendLCDCommand(0x28);
    sendLCDCommand(0x0E);
  // clear
    sendLCDCommand(0x01);
  // back to start
    sendLCDCommand(0x80);
}

void initADC(){
    // Set reference voltage to AVcc
    ADMUX |= (1 << REFS0) | (1 << MUX0);
    // Enable ADC with prescalar 128
    ADCSRA |= (1 << ADEN)|(1 << ADPS2)|(1 << ADPS1)|(1 << ADPS0);
}

void setup()
{
    initLCD();
    initADC();
}

char buffer[16];

void loop()
{
  sendLCDCommand(0x01);
  sendLCDCommand(0x80);

  // Start conversion
  ADCSRA |= (1<<ADSC);

  //wait for ADIF
  while((ADCSRA & (1 << ADSC)));

  float adcValue = (ADC / 1024.0 * 5);
  float res = 10000 / ((5.0/adcValue) - 1.0);

  //Not sure about these following equations ka
  //P Jom pls be kind na ka >____<
  float lux = res/100.0;
  //float lux = (1.25 * pow(10,7)) * pow(res,-1.4059);

  dtostrf(lux, 4, 2, buffer);
  lcdDisplayString("Brightness");
  sendLCDCommand(0xc0);
  lcdDisplayString("= ");
  lcdDisplayString(buffer);
  lcdDisplayString(" lux");
  _delay_ms(1000);
}
```