```
/*
  Griffin Mack
  11713813

  Lab5&6 - Introduction to Polling, Direct Port Manipulation, and Interrupt-Driven Systems

   For this lab, user input is taken from the serial monitor. If the user enters 'a', 'b', or 'c',
   30 analog readings are taken from a potentiometer connected to pin A0. Input 'a' utilizes the analogRead
   function. Input 'b' polls for an ADC conversion to finish. Input 'c' uses interrupts when an ADC conversion
   is finished. The analog values are converted to digital and displayed to the user, along with the time taken
   for each conversion, and an average conversion time.

   Additionally, this lab utilizes the Arduino watchdog timer. If user input is received within 4 seconds of
   prompting (valid or not), the watchdog timer is reset. If no input is found during the time, the board resets.

*/
#include <avr/wdt.h>

#define AnalogInputPin A0
#define MaxConversions 30

volatile boolean adcFinished = false;    //flag for ISR
volatile int adcReading = 0;             //stores ISR finished conversion
void setup() {
  /*program setup
  */
  Serial.begin(115200);                  //initialize serial monitor
  Serial.println("Board was reset...");  //notify of board reset

  //initial setup for ADC
  ADMUX |= B01000000;                         //set reference voltage to AVcc
  ADMUX |= B00000000;                         //select channel A0(last 4 bits: 0000)
  ADCSRA = bit(ADEN);                         //enable the ADC
  ADCSRA |= B00000111;                        //set the prescaler to 128
}


void loop() {
  /*main program loop.
  */
  bool restart_flag = false;                  //stores flag to restart program without a reboot
  unsigned long conversion_time = 0;          //stores time taken for ADC to convert input to digital
  unsigned long start_time = 0;               //stores beginning time before ADC begins conversion
  int time_array[MaxConversions];             //stores all conversion times for calculation average time

  int analog_input = 0;                       //stores analog input from AnalogInputPin
  String user_input = "";                     //stores user serial input

  boolean adcStarted  = false;                //starts a new ADC conversion(for ISR)
  int conversionsFinished = 0;                //stores number of conversions finished(for ISR)

  promptUser();

  while (restart_flag == false) {
    user_input = "";              //clear the user_input variable
    while (Serial.available()) {
      wdt_reset();              //reset the watchdog timer on valid or invalid input
      char c = Serial.read();   //get one byte from serial buffer
      user_input += c;          //add byte to the input string
      delay(2);                 //small delay to allow more accurate reading from serial
    }
    if (user_input == "a" || user_input == "b" || user_input == "c") {  //check if user input is 'a','b', or 'c'
      if (user_input == "a") {
        Serial.println("Starting a set of conversions using AnalogRead:");
        for (int i = 0; i < MaxConversions; i++) {      //take a set of 30 conversions
          start_time = micros();                        //store time conversion started in microseconds
          analog_input = analogRead(AnalogInputPin);    //read the analog input on A0 (value is 0-1023)
          conversion_time = micros() - start_time;      //calculate time taken to convert analog input
          time_array[i] = conversion_time;              //store the time for calculating the average
          printConversion(i, analog_input, conversion_time);
          delay(500);                                   //delay for user to change value
          wdt_reset();                                  //make sure watchdog does not time out
        }
      }
      if (user_input == "b") {
        Serial.println("Starting a set of conversions using polling and port manipulation:");
        for (int i = 0; i < MaxConversions; i++) {      //take a set of 30 conversions
```

```
          start_time = micros();                        //store time conversion started in microseconds
          ADCSRA |= B01000000;                          //start a conversion
          while (ADCSRA & B01000000);                   //wait for conversion to finish
          analog_input = ADC;                           //grab results from ADC data register
          conversion_time = micros() - start_time;      //calculate time taken to convert analog input
          time_array[i] = conversion_time;              //store the time for calculating the average
          printConversion(i, analog_input, conversion_time);
          delay(500);                                   //delay for user to change value
          wdt_reset();                                  //make sure watchdog does not time out
        }
      }
      if (user_input == "c") {
        Serial.println("Starting a set of conversions using interrupts:");
        while (conversionsFinished < MaxConversions) {
          if (adcFinished) {
            conversion_time = micros() - start_time;    //calculate time taken to convert analog input
            time_array[conversionsFinished] = conversion_time;  //store the time for calculating the average
            printConversion(conversionsFinished, adcReading, conversion_time);
            conversionsFinished ++;                     //increment amount of conversions completed
            delay(500);                                 //delay for user to change value
            adcStarted = false;                         //stop the ADC from running
            adcFinished = false;                        //stop conversions from happening
          }
          if (!adcStarted) { //start a new conversion
            adcStarted = true;
            // start the conversion
            start_time = micros();                      //store time conversion started in microseconds
            ADCSRA |= bit (ADSC) | bit (ADIE);          //starts conversion and enables ISR
          }
          wdt_reset();                                  //make sure watchdog does not time out
        }
      }

      printAverageTime(time_array);
      restart_flag = true;                              //restart the program but do not reboot the device
      while (Serial.available() > 0) {
        Serial.read();                                 //clear the serial buffer(ignore inputs during conversion)
      }
    }
    //check if user input is not 'a','b','c', or blank
    if (user_input != "a" && user_input != "b" && user_input != "c" && user_input != "") {
      Serial.println("Error: invalid user input - valid inputs are 'a','b', and 'c'");
      promptUser();                                    //re-prompt the user for input
    }
  }
}

void printAverageTime(int time_array[]) {
  /*calculates average conversion time for all 30 readings
     of the ADC. Then prints the value to the serial monitor
  */
  int total_time = 0;                    //stores sum of times in time_array
  for (int i = 0; i < MaxConversions; i++) {
    total_time += time_array[i];        //add time array entry to sum
  }
  float time_average = total_time / (double)MaxConversions; //calculate the average time (sum/entries)
  Serial.print("\navg conversion time = ");
  Serial.print(time_average);
  Serial.println(" usecs\n");
}

void printConversion(int measurement, int conversion_value, int conversion_time) {
  /*prints converted analog signal to the serial monitor
    along with time taken to convert the analog value
  */
  Serial.print("#");
  Serial.print(measurement + 1);         //measurement is 0 based
  if (measurement + 1 < 10) {            //conditional formatting if the measurement is two digits
    Serial.print(":    digital value = 0x");
  }
  else {
    Serial.print(":   digital value = 0x");

  }
  if(conversion_value < 0xFF){           //conditional formatting conversion value
    Serial.print("0");
  }
  if(conversion_value < 0xF){
    Serial.print("0");
  }
  Serial.print(conversion_value, HEX);   //display the integer in HEX
  Serial.print("    Time = ");
  Serial.print(conversion_time);
  Serial.println(" usecs");
}

void promptUser() {
  /*prints prompt message to serial monitor
  */
  Serial.println("Select a type of conversion to perform('a' for AnalogRead;'b' for polling;'c' for interrupts)");
  wdt_enable(WDTO_4S);                   //start the 4 second watchdog timer
}

ISR(ADC_vect) {
  adcFinished = true;                   //notify that ISR has been triggered
  adcReading = ADC;                     //grab results from ADC data register
}
```