```
const int numOfReadings = 10;               // number of readings to take/ items in the array
int readings[numOfReadings];                // stores the distance readings in an array
int arrayIndex = 0;                       // arrayIndex of the current item in the array
int total = 0;                          // stores the cumlative total
int averageDistance = 0;                  // stores the average value

// setup pins and variables for SRF05 sonar device

int echoPin = 2;                        // SRF05 echo pin (digital 2)
int initPin = 3;                        // SRF05 trigger pin (digital 3)
unsigned long pulseTime = 0;              // stores the pulse in Micro Seconds
unsigned long distance = 0;               // variable for storing the distance (cm)
unsigned long soundDelay = 0;
int speakerPin = 5;
int motor1Pin1 = 8;                     // pin 2 on L293D
int motor1Pin2 = 7;                     // pin 7 on L293D
int motor2Pin1 = 10;                      // pin 10 on L293D
int motor2Pin2 = 9;                     // pin  15 on L293D
int led1 = 12;
int led2 = 4;
int button1 = 0; //
int button1Pin = 11; //
int button2 = 0; //
int button2Pin = 6; //


void setup() {
  // set the motor pins as outputs:
  pinMode(motor1Pin1, OUTPUT);
  pinMode(motor1Pin2, OUTPUT);
  pinMode(motor2Pin1, OUTPUT);
  pinMode(motor2Pin2, OUTPUT);
  pinMode(initPin, OUTPUT);               // set init pin 3 as output
  pinMode(echoPin, INPUT);              // set echo pin 2 as input
  pinMode(speakerPin, OUTPUT);              // sets pin 6 as outpu
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(button1Pin, INPUT);
  pinMode(button2Pin, INPUT); // create array loop to iterate over every item in the array

  for (int thisReading = 0; thisReading < numOfReadings; thisReading++) {
    readings[thisReading] = 0;
  }
}

void goLeft(){
  digitalWrite(motor1Pin1, LOW);
  digitalWrite(motor1Pin2, HIGH);
  digitalWrite(motor2Pin1, HIGH);
  digitalWrite(motor2Pin2, LOW);
  digitalWrite(led1, HIGH);
  digitalWrite(led2, LOW);
```

```
  delay(500);
}
void goRight(){
  digitalWrite(motor1Pin1, HIGH);
  digitalWrite(motor1Pin2, LOW);
  digitalWrite(motor2Pin1, LOW);
  digitalWrite(motor2Pin2, HIGH);
  digitalWrite(led1, LOW);
  digitalWrite(led2, HIGH);
  delay(500);
}
void goBackwards(){
  digitalWrite(motor1Pin1, HIGH);
  digitalWrite(motor1Pin2, LOW);
  digitalWrite(motor2Pin1, HIGH);
  digitalWrite(motor2Pin2, LOW);
  digitalWrite(led1, LOW);
  digitalWrite(led2, LOW);
  delay(500);
}

void loop() {
  digitalWrite(initPin, HIGH);               // send 10 microsecond pulse
  delayMicroseconds(10);                          // wait 10 microseconds before turning off
  digitalWrite(initPin, LOW);               // stop sending the pulse
  pulseTime = pulseIn(echoPin, HIGH);          // Look for a return pulse, it should be high as the
pulse goes low-high-low
  distance = pulseTime/58;                  // Distance = pulse time / 58 to convert to cm.
  total= total - readings[arrayIndex];         // subtract the last distance
  readings[arrayIndex] = distance;             // add distance reading to array
  total= total + readings[arrayIndex];         // add the reading to the total
  arrayIndex = arrayIndex + 1;               // go to the next item in the array
  soundDelay = pulseTime/3;
  // At the end of the array (10 items) then start again
  if (arrayIndex >= numOfReadings)  {
    arrayIndex = 0;
  }

  averageDistance = total / numOfReadings;      // calculate the average distance
  delay(10);
  button1 = digitalRead(button1Pin);
  button2 = digitalRead(button2Pin);
  if(button1 == 1 && button2 == 0){
    goLeft();
  }

  if(button2 == 1 && button1 == 0){
    //go right
    goRight();
  }

  if(button1 == 1 && button2 == 1){
```

```
    goBackwards();
  }

  // check the average distance and move accordingly
  if (distance < 25) {
    //turn left
    digitalWrite(speakerPin, HIGH);
    delayMicroseconds(soundDelay);
    digitalWrite(speakerPin, LOW);
    delayMicroseconds(soundDelay);
  }
  if (averageDistance <= 10){
    // go backwards
    digitalWrite(motor1Pin1, HIGH);
    digitalWrite(motor1Pin2, LOW);
    digitalWrite(motor2Pin1, HIGH);
    digitalWrite(motor2Pin2, LOW);
    digitalWrite(led1, HIGH);
    digitalWrite(led2, HIGH);
  }

  if (averageDistance <= 25 && averageDistance > 10) {
    // turn
    digitalWrite(motor1Pin1, HIGH);
    digitalWrite(motor1Pin2, LOW);
    digitalWrite(motor2Pin1, LOW);
    digitalWrite(motor2Pin2, HIGH);
    digitalWrite(led1, LOW);
    digitalWrite(led2, HIGH);
  }
  if (averageDistance > 25)   {
    // go forward
    digitalWrite(motor1Pin1, LOW);
    digitalWrite(motor1Pin2, HIGH);
    digitalWrite(motor2Pin1, LOW);
    digitalWrite(motor2Pin2, HIGH);
    digitalWrite(led1, LOW);
    digitalWrite(led2, LOW);
  }
}
```