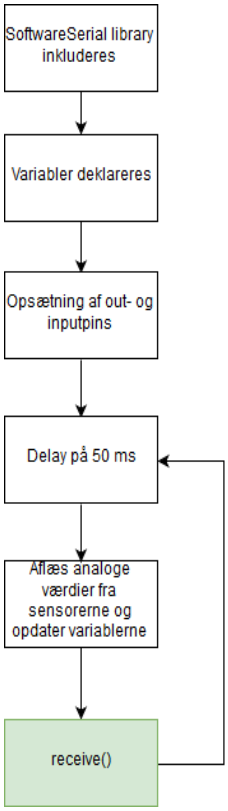
# Opgave 2 Elevarbejde

Der er blevet skrevet noget kode til at styrer en robot, disse elever har valgt at holde fokus på følgende 3 flowdiagrammer.

## 3. Flowcharts

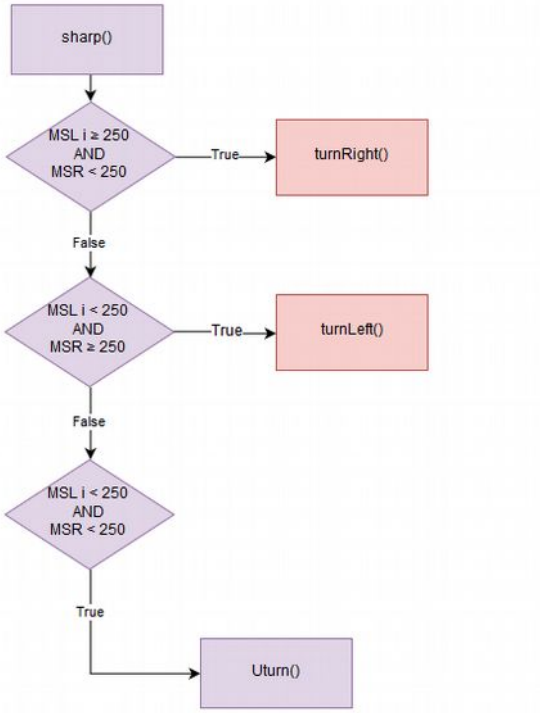
### Arduino loop-funktion

Nedenstående ses Arduinoens loopfunktion, som konstant kører funktionen receive og afventer signaler fra Bluetooth.



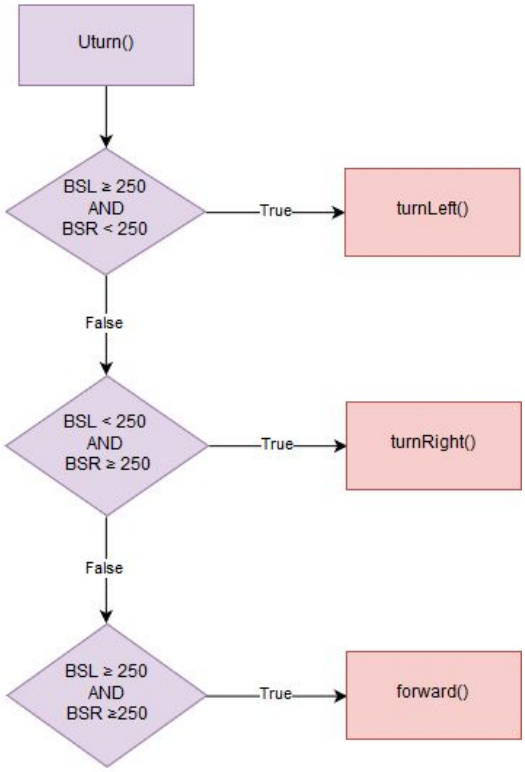

### Arduino sharp-funktion

Nedenstående ses Arduinoens sharp-funktion.



## Arduino Uturn-funktionen

Nedenstående ses Arduinoens Uturn-funktion




## Arduino kode

Nedenstående ses vores arduinokode. Vi har kommenteret koden på engelsk, da dette er det sprog, man benytter i industrien samtidig med, at mange udtryk ikke er at finde på dansk.

```
#include<SoftwareSerial.h> //Includes the software serial library

// Bluetooth Pins
int bluetoothTx = 12; //Bluetooth transmitter pin
int bluetoothRx = 9; //Bluetooth receiver pin

SoftwareSerial bluetooth(bluetoothTx, bluetoothRx); //Creates a bluetooth
software serial

/*Missile launcher pin*/
int launchPin = 4;

/*Declaration of motor pins*/
int left[2] = {5,6};
int right[2] = {7,8};
```

```
/*Declaration of enable pins which is used to control the speed*/
int enableRight = 11;
int enableLeft = 10;

/*Declaration of sensor pins*/
int sensorRight  = A0;
int sensorLeft = A1;
int midSensorLeft = A3;
int backSensorLeft = A5;
int midSensorRight = A2;
int backSensorRight = A4;

/*Declares the sensor value pins*/
int SL;
int MSL;
int BSL;
int SR;
int MSR;
int BSR;

/*Starting speed */
int SpeedLeft = 255; //Starting PWM value on the left wheel
int SpeedRight = 235; //Starting PWM value on the right wheel. Its value is
lower than the left wheel, because the motors are different.


int del=2000; //Missile launcher delay is set to 2000 milliseconds
/*Declaration of millis variables to ensure a delay when firing the missiles*/
int lastMillis;
int currentMillis;

/*declaration of message variables containing the received char from
bluetooth*/
char msg;
char msg1;


/*
 * ------------------------------------------------------
 * SETUP
 * ------------------------------------------------------
 */

void setup()  {
  Serial.begin(9600); //Serial baudrate
  bluetooth.begin(115200); //Bluetooth serial baudrate

  /*Missile launcher pin*/
  pinMode(launchPin, OUTPUT);

  /*Motor-pins*/
  pinMode(left[0], OUTPUT);
  pinMode(left[1], OUTPUT);
  pinMode(right[0], OUTPUT);
  pinMode(right[1], OUTPUT);

  /*H-bro pins*/
```

```
  pinMode(enableRight, OUTPUT);
  pinMode(enableLeft, OUTPUT);

  /*Sensor pins*/
  pinMode(sensorLeft, INPUT);
  pinMode(midSensorLeft, INPUT);
  pinMode(backSensorLeft, INPUT);
  pinMode(sensorRight, INPUT);
  pinMode(midSensorRight, INPUT);
  pinMode(backSensorRight, INPUT);


  /*Sets the starting speed by outputting a value on the enable pins. Uses
Arduino PWM*/
  analogWrite(enableRight, SpeedRight);
  analogWrite(enableLeft, SpeedLeft);


  lastMillis = 0;
}


/*
 * ------------------------------------------------------
 * LOOP
 * ------------------------------------------------------
 */

void loop() {
  delay(50);
  //Bluetoothfunktioner, der henter data via. bluetooth
  bluetooth.listen();
  receive();

 /*
 * Read the sensor pins and stores the value
 * sensorLeft and midSensorLeft is subtracted by 50 because the sensors return
a higher                    *value than the others.
 */
  SL = analogRead(sensorLeft)-50;
  MSL = analogRead(midSensorLeft)-50;
  BSL = analogRead(backSensorLeft);
  SR = analogRead(sensorRight);
  MSR = analogRead(midSensorRight);
  BSR = analogRead(backSensorRight);

  /*Prints the sensorvalues for debuggin purposes*/
  Serial.println(SL);
  Serial.println(SR);
  Serial.println(MSL);
  Serial.println(MSR);
  Serial.println();
  Serial.println(firstShot);

}
```

```
/*
* ------------------------------------------------------
* Car control functions
* ------------------------------------------------------
*/


/*
 * Car forward function
 */
void forward() {
  digitalWrite(left[0], LOW);
  digitalWrite(left[1], HIGH);
  digitalWrite(right[0], HIGH);
  digitalWrite(right[1], LOW);
}


/*
 * Car reverse function
 */
void reverse() {
  digitalWrite(left[0], HIGH);
  digitalWrite(left[1], LOW);
  digitalWrite(right[0], LOW);
  digitalWrite(right[1], HIGH);
 }

void brake() {
  digitalWrite(left[0], LOW);
  digitalWrite(left[1], LOW);
  digitalWrite(right[0], LOW);
  digitalWrite(right[1], LOW);
}

/*
 * Car turn left function
 */
void turnLeft() {
  digitalWrite(left[0], LOW);
  digitalWrite(left[1], LOW);
  digitalWrite(right[0], HIGH);
  digitalWrite(right[1], LOW);
}

/*
 * Car turn right function
 */
void turnRight() {
  digitalWrite(left[0], LOW);
  digitalWrite(left[1], HIGH);
  digitalWrite(right[0], LOW);
  digitalWrite(right[1], LOW);
}

/*
```

```
* This function decreases the motor speed
*/
void slowdown() {
  if(155<SpeedL){
    SpeedL=SpeedL-10;
    SpeedR=SpeedR-10;
  }
  Serial.println(SpeedL);
  Serial.println(SpeedR);
  analogWrite(enableRight, SpeedR);
  analogWrite(enableLeft, SpeedL);
  delay(300);

}

/*
* This function increases the motor speed
*/
void speedup(){
  if(SpeedL<255){
    SpeedL=SpeedL+10;
    SpeedR=SpeedR+10;
  }
  analogWrite(enableRight, SpeedR);
  analogWrite(enableLeft, SpeedL);
  delay(300);
  Serial.println(SpeedL);
  Serial.println(SpeedR);
}

/*
 * This function reads the data from the sensors, and uses it to determine
which way to turn.
 */
void sensings() {
  if(SR>=250 && SL>=250){
      forward();

      // If both the front and mid sensors senses black the missile launcher
fires
      if(MSR>=250 && MSL>=250){

        currentMillis = millis(); //Stores the current millis

        /*If the difference between current and last millis is bigger than the
delay (2000 milliseconds) the missile will fire. This ensures that the car
will only fire once when it meets a road intersection*/
        if(currentMillis - lastMillis >= del){
          fire();
          lastMillis = currentMillis; //Sets lastMillis to currentMillis
        }
      }
  }
  else if(SL<=250 && SR>=250){
    turnRight();
  }
  else if(SR<=250 && SL>=250){
```

```
    turnLeft();
  }
  else if(SR<=250 && SL<=250){
    sharp();
  }
}


/*
* This function makes the car take a sharp turn. It is only triggered when
both the left and right sensor senses white
*/
void sharp(){
  if(MSL>=250 && MSR<=250){
    turnLeft();
  }
  else if(MSR>=250 && MSL<=250){
    turnRight();
  }
  else if (MSR<=250 && MSL<=250){
    Uturn();
  }
}

/*
 * This function uses the data from the backmost sensors to determine whether
to turn or not. (Security measure)
 */
void Uturn(){
  if(BSL>=250 && BSR<=250){
    turnLeft();
  }
  else if(BSR>=250 && BSL<=250){
    turnRight();
  }
  else if (BSR>=250 && BSL>=250){
    forward();
  }
}

/*
* rapidFire fires all the missiles in quick succession
*/
Void rapidFire(){
  digitalWrite(launchPin, HIGH);
  delay(1000);
  digitalWrite(launchPin, LOW);
}

/*
* fire() only fires 1 missile at the time
*/
void fire(){
  digitalWrite(launchPin, HIGH);
  delay(100);
  digitalWrite(launchPin, LOW);
}
```

```
/*
 * Bluetooth function, recieves data from the bluetooth module, and uses it to
determine which action to take
 */
void receive(){

 /*
  * Reads the char received via bluetooth if bluetooth is available
  */
 if(bluetooth.available()){
    msg1=(char)bluetooth.read();
 }

//We ran into some problems with recieving a 'null' value from the BT device,
so we made it so the car would only change its course of action if the
character was one of the following characters:
 if(msg1 =='a' || msg1 =='s' || msg1 =='l' || msg1 =='r' || msg1 =='b' || msg1
=='+' || msg1 =='-' || msg1 =='c'){
    msg = msg1;
 }
 if(msg =='a') {
    forward();
 }else if(msg =='s') {
    brake();
 }else if(msg =='l') {
    turnLeft();
 }else if(msg =='r') {
    turnRight();
 }else if(msg =='b') {
    reverse();
 }else if(msg =='+'){
    speedup();
 }else if(msg =='-') {
    slowdown();
 }else if(msg =='c'){
    sensings(); //If the message is 'C' the car will go into autonomous mode
 }
 if(msg1 =='f'){
    fire();
    msg1 =' ';
  }else if(msg1 =='e'){
    rapidFire();
    msg1 =' ';
  }
}
```

Hvad med den dokumentation der er lavet?

Er det nok eller skal der mere til og hvad skal det mere være?

Udvælgelsen af dokumentation er den fin?

Kommentarer er de fyldestgørende?