Messungen mit dem Lichtsensor

Für die Programmierung eines Roboters, der einer Linie folgt, brauchen wir für die drei Farben die Helligkeitswerte, die dein Lichtsensor jeweils ausgibt. Arbeite dazu folgende Arbeitsanweisungen ab:

- **Tippe** das unten stehende **Programm** zur Lichtmessung **ab** und speichere es unter dem Namen n2a_10_licht_name.ino.
- **Übersetze** das Programm und **übertrage** das **Programm** auf deinen MiniQ-Roboter.
- Führe das Programm aus und **ermittle** damit die **Sensorwerte** für den **schwarzen**, den **weißen** und den **silbernen** Untergrund.
- **Trage** die **Werte** in die unten stehende Tabelle **ein**.

```
// Roboterprogrammierung - Stadtteilschule Eppendorf
// n2a_10_licht_name.ino - Sensorwert abfragen,
// auf den PC übertragen und über den seriellen Monitor
// (Strg-Shift-m) ansehen
// Jens Stolze - 2012-09-15

#include <nxc2arduino.h>

nxc2arduino nxc;

int licht;

void setup() {
  Serial.begin(9600);
}
/* - - - Sensorwert abfragen und auf PC übertragen - - - */;

void loop() {

  licht = nxc.Sensor(IN_3);
  Serial.println(licht);
  nxc.Wait(10);
}
```

Sensorwerte

Farbe	schwarz	weiß	silber
Wert			

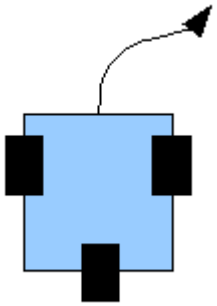
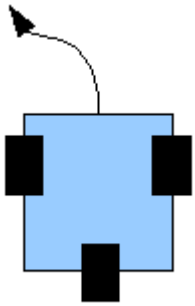
Merke: Nicht jeder Sensor gibt an gleicher Stelle die gleichen Werte aus. Schreibe dir deshalb hier zur Sicherheit die Bezeichnung von deinem Roboter auf.

Bezeichnung vom Roboter: _____

Datum: ***Name:***

Steuerung von Robotern mit zwei Motoren

Damit man gezielt einen Roboter nach links oder nach rechts steuern kann, muss man sich bewusst machen, wie jeweils die beiden Motoren an- bzw. ausgeschaltet werden müssen, damit der Roboter in die gewünschte Richtung fährt.

nach **rechts...**	nach **links...**
	

Roboter haben i.d.R. einen **linken** und einen **rechten Motor**, die jeweils direkt mit den linken und rechten Antriebsrädern verbunden sind.

Drei mögliche **Zustände** kann ein Motor haben:

- **vorwärts** drehend (↑)
- **ausgeschaltet (--)**
- **rückwärts** drehend (↓)

Aufgabe: Bestimme in der unten stehenden **Tabelle**, auf welche Arten ein Roboter jeweils nach links und rechts gesteuert werden kann. **Trage** dazu die passenden **Zeichen (↑, --, ↓) ein.**

Steuerung nach **rechts...**			**Steuerung** nach **links...**		
Nr.	**Motor links (Ausgang C)**	**Motor rechts (Ausgang B)**	**Motor links (Ausgang C)**	**Motor rechts (Ausgang B)**	**Nr.**
1					1
2					2
3					3

Merke: Eine **schnelle Drehung** haben wir bei der Vorgehensweise unter ____.

Langsame Drehungen werden bei den Wegen ____ und ____ durchgeführt.

Nur bei den **Wegen** ____ und ____ bewegt sich der Roboter **vorwärts.**

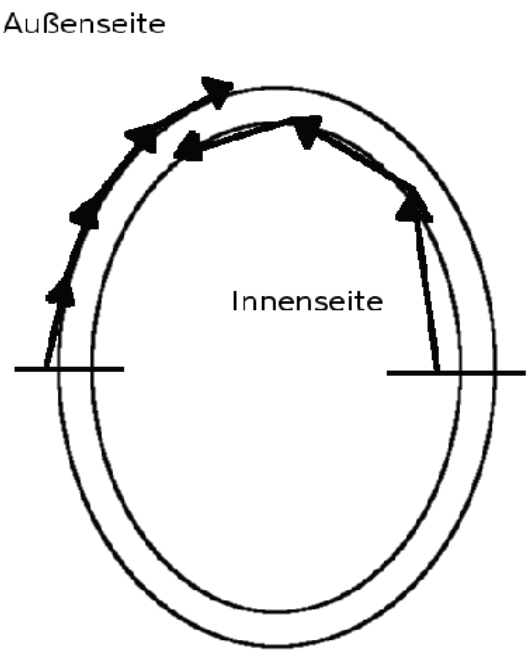
Datum: ***Name:***

Linienfolger für einen Rundkurs programmieren

Ein einfacher Linienfolger hat einige Besonderheiten, die ohne genauere Betrachtung nicht sofort offensichtlich sind:

- Wenn man nur **einen** Sensor hat, dann **fährt** man auf einen der **beiden Übergänge** zwischen der **Linie** und **Umgebungsfläche** und somit
  - entweder auf der **Innenseite** oder
  - auf der **Außenseite** des Kreises

  und nicht direkt auf der Linie.
- Als **Grenzwert** muss ein Wert **Zwischenwert** zwischen dem für Schwarz und Weiß gewählt werden.
- Hätte man zwei Sensoren im optimalen Abstand, wäre es auch anders möglich. Der Aufwand an Hardware ist dabei aber ungleich größer (... und weitere Sensoren sind nicht vorhanden... ;-)).



Merke: Der Weg auf der Innenseite des Kreise ist _____, folglich sind _____ Rundenzeiten zu erwarten. Der Außenkurs ist _____, da keine Linie überfahren werden kann.

Aufgaben:

- **Erstelle** ein **Programm** für einen **Linienfolger** nach der nebenstehenden Beschreibung.
- Setze dazu die umgangssprachlichen Beschreibungen in **NXC-Programmtext** um.
- **Hilfestellung** bekommst du durch die nxc2arduino-Anweisungen von **Seite 15.**
- Speichere dein Programm unter dem Dateinamen `11_linie_`*`name`*`.nxc` ab.

Umgangssprachliche Beschreibung:

Definiere Grenzwert (Wert zwischen Schwarz- und Weißwert)

Eingang 3 als Lichtsensor schalten

Dauerschleife

wenn weiß...

geradeaus fahren

sonst

nach links fahren

Optimierung eines Linienfolgers für einen Rundkurs

Ein **Linienfolger** für einen Rundkurs soll

- beim **Überfahren** eines **silbernen Feldes** stoppen und
- die berechnete **Fahrtzeit** soll nach dem **Anschließen an den PC** über die serielle Konsole (Strg-Shift-m) zu lesen sein und **im Programmkopf einzutragen.**

Hierzu sind

- die **Befehle** zum **Drehen** und **Geradeausfahren** des Roboters in die Datei nxclib_ticks_buzz_name.ino aus „Datei – Beispiele – nxs2arduino“ an der richtigen Stelle **einzufügen** und
- die **Datei** unter angepasstem Dateinamen n2a_12_rund_name.nxc **abzuspeichern.**

```
// Roboterprogrammierung - Stadtteilschule Eppendorf
// n2a_12_rund_name.ino - Linienfolger fuer Rundkurs mit Zeitchecker
// Jens Stolze – 2012-05-05

#include <nxc2arduino.h>

// Konstanten
#define FAHREN 1
#define GRENZE 600
#define ZUSILBER 986
#define SPEED_LOW 40
#define SPEED_FAST 120

nxc2arduino nxc;

/* - - - einfacher Linienfolger fuer Rundkurs mit Zeitchecker - - -  */

void setup (){
  // initialize serial communications at 9600 bps:
  Serial.begin(9600);
  int licht = 0;
  long startzeit = nxc.CurrentTicks();
  long dauer;

  licht = nxc.Sensor(IN_3);

  while (licht < ZUSILBER)
    {
    licht = nxc.Sensor(IN_3);
    if (licht > GRENZE)
    {
      // drehen
    }
   else
    {
        // geradeaus fahren
    }
```

```
  }
  dauer = nxc.CurrentTicks() - startzeit;
  nxc.buzzer();
  for (int i=0; i <= 20; i++)
  {
    nxc.Wait(1000);
    Serial.print("Fahrzeit: ");
    Serial.println(dauer);
  }
  nxc.buzzer();
}

void loop ()
{

}
```

Merke: Die benötigte Zeit kann ermittelt werden, wenn man den Roboter nach der Fahrt innerhalb von 20 Sekunden an den PC anschließt und die serielle Konsole (Tools – Serieller Monitor - Strg-Shift-m) startet.

Auftrag: Wer schafft den Rundkurs in der kürzesten Zeit? Optimiere dein Programm!

Veränderung eines Linienfolgers für einen Zickzackkurs

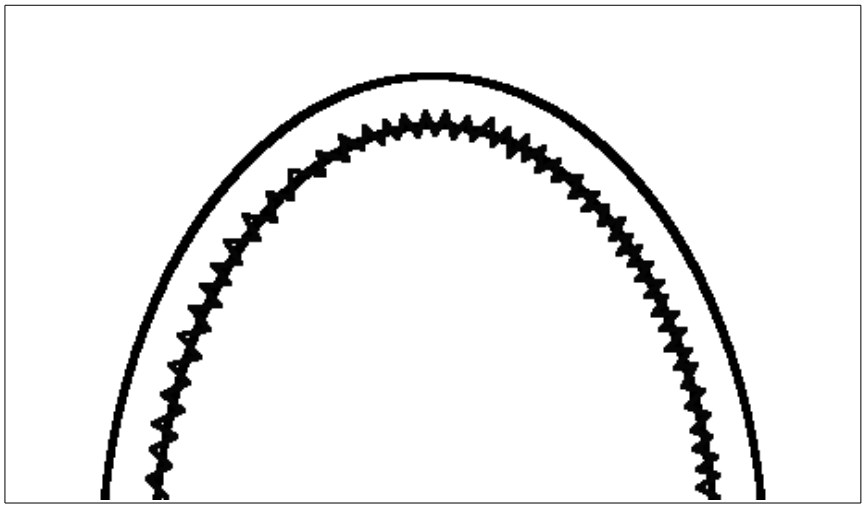
Der Linienfolger für den Rundkurs ist in der while-Schleife so abzuändern, dass er auch auf einem Zick-Zackkurs optimal läuft. Dabei ist folgendes zu bedenken:

Beim **Rundkurs** muss man nur dann die Richtung eines Roboters ändern, wenn beim Befahren

- des Innenkreises in die schwarze Fläche kommt und
- beim Außenkreis auf die weiße Fläche kommt (siehe Zeichnung Seite 18)

Eine **einfache Programmierung** wird möglich, **weil** die **Außen**- bzw. **Innenseite** der **Kurve** immer die **gleiche Seite** ist.

Beim **Zickzackkurs** ist dies nicht mehr der Fall, dort **ändert sich** diese **Zuordnung immer** wieder, die Programmierung wird aufwändiger. Folgendes Bild soll es veranschaulichen:



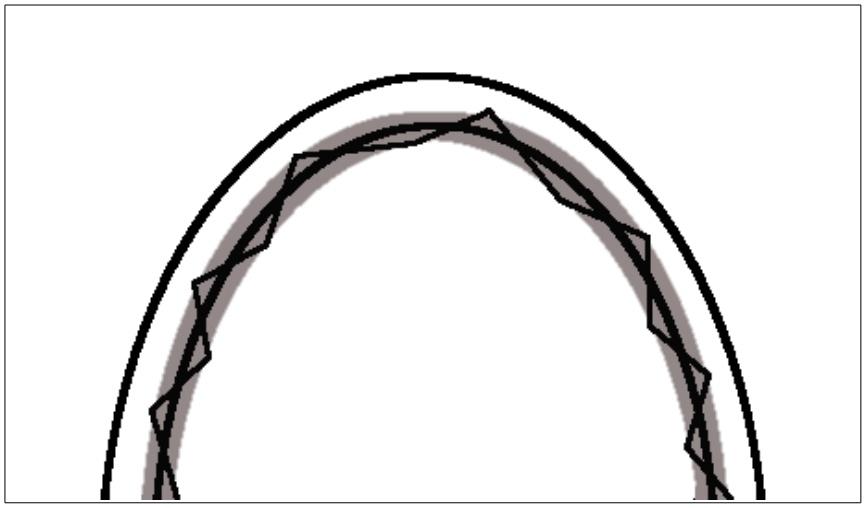
Aufgabe:

- Die Vorgehensweise in der Schleife ist in deutscher Sprache wie unten zu sehen. Setze sie in nxc2arduino-Programmcode um.
- Speichere das Ergebnis als n2a_13_zick_*name*.nxc.
- Wer schafft den Zickzackkurs in der kürzesten Zeit? Optimiere dein Programm!

Schleife – wenn nicht silber
wenn schwarz
  nach rechts fahren
sonst
  nach links fahren
ende-wenn

Optimierung eines Linienfolgers für einen Zickzackkurs

Der **Linienfolger** für den **Zickzackkurs** fährt ungleich langsamer als der für den Rundkurs. Er **„schwänzelt“**, d.h. er ändert seine Richtung immer sehr schnell. Dies gilt es zu optimieren, indem man eine „**Grauzone“** einführt, in der die Regelung nicht aktiv wird. Dadurch fährt er schon ruhiger. Folgendes Bild soll es veranschaulichen:



Aufgabe:

- Die Vorgehensweise in der Schleife ist in deutscher Sprache wie unten zu sehen. Setze sie in nxc2arduino-Programmcode um.
- Speichere das Ergebnis als n2a_14_zack_*name*.nxc.
- Wer schafft den Zickzackkurs in der kürzesten Zeit? Optimiere dein Programm!

Schleife – wenn nicht silber
wenn schwarz - GRAUZONE
  nach rechts fahren
sonst
  wenn weiß + GRAUZONE
    nach links fahren
  sonst
    nach vorne fahren