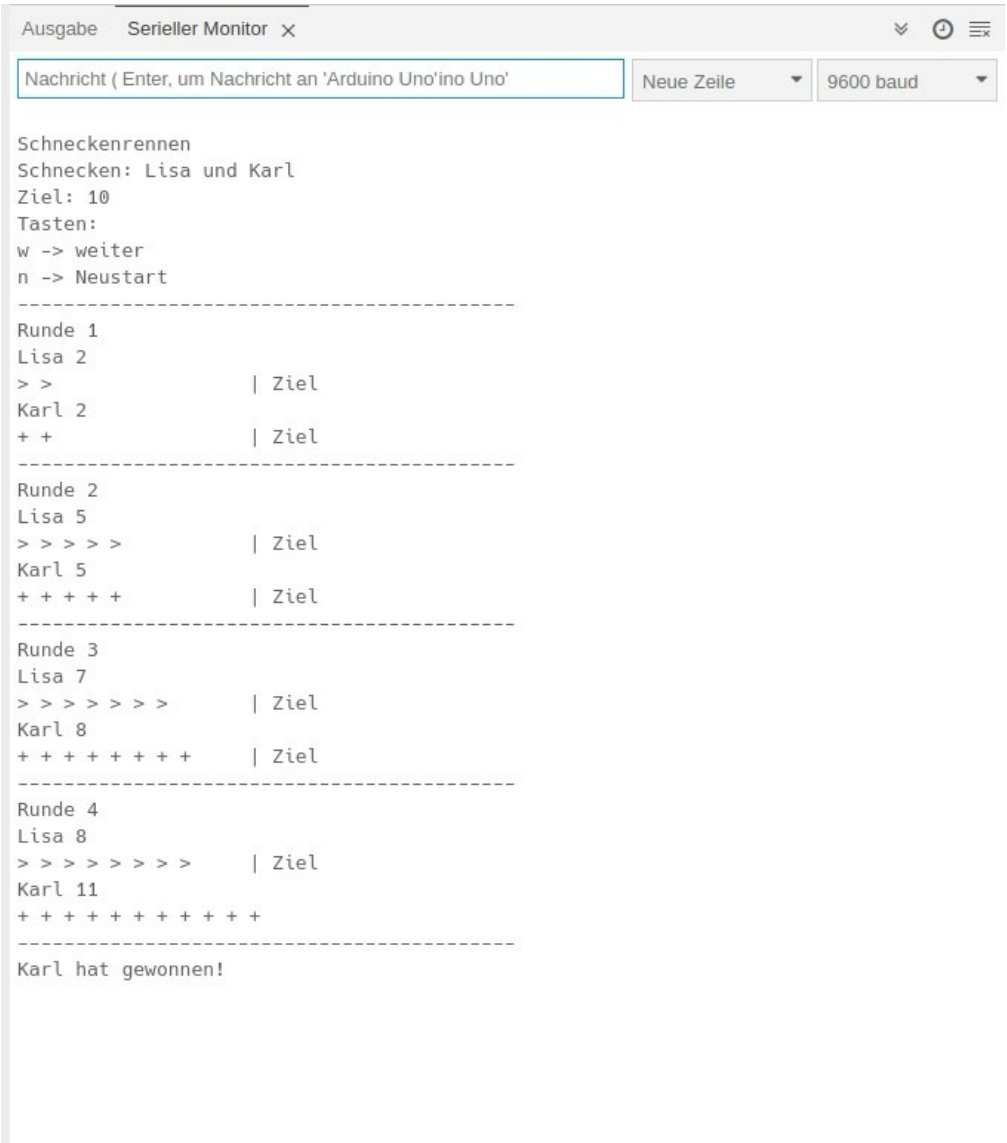
Zwei Schnecken liefern sich ein Rennen. Sie laufen eine zufällig ermittelte Anzahl von Schritten. Wer als Erstes mit dem letzten Zug das Ziel überschreitet, hat gewonnen.

Für diese Aufgabe brauchst du nur den Arduino.

So sieht es aus:



Im Kopf des Programms werden verschiedene Arrays verwendet, damit die Ausgabe möglichst flexibel gestaltet werden kann.

Beachte die Kommentare.

```
// Anzahl der Schritte Minimum/Maximum
int Minimum = 1;
int Maximum = 4;

// Schritte bis zum Ziel
int Ziel = 10;
int Schritte = 0;
int Runde = 1;

// wenn Neustart = true -> Variable zurücksetzen
bool Neustart = false;

/*
  Arrays für die Schnecken
  BisherigeSchritte[] -> Anzahl zurückgelegter Schritte
  Schnecke[] -> Namen der Schnecken
  SchrittZeichen[] -> Zeichen für die zurückgelegten Schritte
  es können (fast) beliebige Zeichen verwendet werden
*/
int BisherigeSchritte[2];
String Schnecke[2] = {"Lisa", "Karl"};
String SchrittZeichen[2] = {"> ", "+ "};
```

Im setup-Teil wird über das Ziel, die beteiligten Schnecken und über die Tastenbelegung informiert.

```
void setup()
{
  Serial.begin(9600);

  // Info über das Spiel
  Serial.println("Schneckenrennen");
  Serial.println("Schnecken: " + Schnecke[0] + " und " + Schnecke[1]);
  Serial.println("Ziel: " + String(Ziel));
  Serial.println("Tasten:\nw -> weiter\nn -> Neustart");
  Serial.println("-----------------------------------------");

  // Zufallsgenerator starten
  randomSeed(analogRead(0));
}
```

Der loop-Teil. Beachte die Kommentare.

```
void loop()
{
  // auf serielle Eingabe warten
  while (Serial.available() > 0)
  {
    // Eingabe im Seriellen Monitor lesen
    char Zeichen = Serial.read();

    // Enter/Senden gedrückt
    if (Zeichen == 'n')
    {
      Neustart = true;
    }
```

```
  if (Zeichen == 'w')
  {
    // beim Neustart -> alle Variable zurücksetzen
    if (Neustart)
    {
      BisherigeSchritte[0] = 0;
      BisherigeSchritte[1] = 0;
      Runde = 1;
      Neustart = false;
    }

    Serial.println("Runde " + String(Runde));

    // Schnecke 1
    // Anzahl der Schritte zufällig ermitteln
    Schritte = ZufallsZahl(Minimum, Maximum);

    // Summe der Schritte berechnen
    BisherigeSchritte[0] = BisherigeSchritte[0] + Schritte;

    /*
      Variabeln an die Methode Bildschirmausgabe übergeben
      Name der Schnecke, Summe der Schritte,
      Zeichen für die zurückgelegten Schritte
      für das erste Element [0] -> die erste Schnecke
    */
    Bildschirmausgabe(Schnecke[0], BisherigeSchritte[0], SchrittZeichen[0]);

    /*
      Variabeln an die Methode Bildschirmausgabe übergeben
      Name der Schnecke, Summe der Schritte,
      Zeichen für die zurückgelegten Schritte
      für das zweite Element [1] -> die zweite Schnecke
    */
    Schritte = ZufallsZahl(Minimum, Maximum);
    BisherigeSchritte[1] = BisherigeSchritte[1] + Schritte;
    Bildschirmausgabe(Schnecke[1], BisherigeSchritte[1], SchrittZeichen[1]);

    Serial.println("-------------------------------------------");

    // Runde hochzählen
    Runde ++;

    // Sieger Schnecke 1
    if (BisherigeSchritte[0] > Ziel)
    {
      if (BisherigeSchritte[0] > BisherigeSchritte[1])
      {
        Serial.println(Schnecke[0] + " hat gewonnen!");
        Neustart = true;
      }
    }
```

```
      // Sieger Schnecke 2
      if (BisherigeSchritte[1] > Ziel)
      {
        if (BisherigeSchritte[0] < BisherigeSchritte[1])
        {
          Serial.println(Schnecke[1] + " hat gewonnen!");
          Neustart = true;
        }
      }

      // unentschieden
      if (BisherigeSchritte[0] > Ziel && BisherigeSchritte[1] > Ziel)
      {
        if (BisherigeSchritte[0] == BisherigeSchritte[1])
        {
          Serial.println("Unentschieden!");
          Neustart = true;
        }
      }
    }
  }
}
```

Jetzt fehlen noch die Funktion `ZufallsZahl()` und die Methode `Bildschirmausgabe()`.

```
int ZufallsZahl(int Minimum, int Maximum)
{
  int Zahl =  random(Minimum, Maximum);
  return Zahl;
}
```

```
void Bildschirmausgabe(String Name, int Schritte, String SchrittZeichen)
{
  // Name der Schnecke und Anzahl der Schritte anzeigen
  Serial.println(Name + " " + String(Schritte));

  // entsprechend der Anzahl der Schritte Zeichen anzeigen
  for (int i = 0; i < Schritte; i ++)
  {
    Serial.print(SchrittZeichen);
  }

  // mit Leerzeichen auffüllen und Ziel anzeigen
  if (Schritte <= Ziel)
  {
    for (int i = Schritte; i < Ziel; i ++)
    {
      Serial.print("  ");
    }
    Serial.print("| Ziel");
  }
  Serial.println();
}
```