

# A joystick

Agócs Norbert és Nagy Dániel

2020. szeptember 8.

# Tartalom

- 1 Ismétlés
- 2 Ismétlő feladatok
- 3 Programozás: Új változó típusok
- 4 Arduino: analogRead és joystick
- 5 Feladat
- 6 Szótár

# Tartalom

- 1 Ismétlés
- 2 Ismétlő feladatok
- 3 Programozás: Új változó típusok
- 4 Arduino: analogRead és joystick
- 5 Feladat
- 6 Szótár

# Mit tanultunk eddig?

## Hardware:

- Arduino alaplap
- Breadboard
- LED
- Ellenállás
- Gomb
- Ultrahang szenzor
- Szervomotor
- Joystick ← Mai szakkör

## Software

- Változó létrehozása
- Alapműveletek ( $=$ ,  $+$ ,  $-$ ,  $/$ ,  $*$ )
- Alapvető parancsok
  - `Serial.begin(9600)`
  - `Serial.println(" ")`
  - `pinMode()`
  - `digitalWrite()`
  - `delay()`
  - `digitalRead()`
  - `pulseIn()`
  - `analogRead()` ← Ma
- Elágazás: `If()...else`
- Szervomotor irányítása

# Tartalom

- 1 Ismétlés
- 2 Ismétlő feladatok
- 3 Programozás: Új változó típusok
- 4 Arduino: analogRead és joystick
- 5 Feladat
- 6 Szótár

# Ismétlő teszt: Hány állítás igaz?

## Az állítások:

- A `for` ciklus mindig csak annyi alkalommal fut le, amennyit mi előre megmondtunk neki.
- A szervomotort a `digitalWrite()` függvénnyel irányíthatjuk.
- Az ultrahangos szenzor a hang útjának az idejét méri és mi ebből számolhatjuk ki a távolságot.
- Az `if` függvény eldönti egy állításról, hogy igaz-e és ez alapján végrehajt vagy kihagy egy kódrészletet.
- A `Serial.println("#MaradjOtthon")` függvény kiírja a soros kijelzőre, hogy `#MaradjOtthon`.

# Ismétlő teszt: Megoldások

## A javított állítások:

- ✓ A `for` ciklus mindig csak annyi alkalommal fut le, amennyit mi előre megmondtunk neki.
- X A szervomotort a `(digitalWrite())` `myservo.write()` függvénnyel irányíthatjuk.
- ✓ Az ultrahangos szenzor a hang útjának az idejét méri és mi ebből számolhatjuk ki a távolságot.
- ✓ Az `if` függvény eldönti egy állításról, hogy igaz-e és ez alapján végrehajt vagy kihagy egy kódrészletet.
- ✓ A `Serial.println("#MaradjOtthon")` függvény kiírja a soros kijelzőre, hogy `#MaradjOtthon`.

Megoldások: 4 igaz állítás volt.

# Feladat I: Mit csinál a kód?

A kód:

```
#include <Servo.h>
int servoPin = 9;
Servo azEnKicsiMotorom;

void setup() {
  azEnKicsiMotorom.attach(servoPin);
}

void loop() {
  azEnKicsiMotorom.write(0);
  delay(1000);
  azEnKicsiMotorom.write(90);
  delay(1000);
}
```



# Feladat I: Mit csinál a kód?

A kód:

```
#include <Servo.h>
int servoPin = 9;
Servo azEnKicsiMotorom;

void setup() {
  azEnKicsiMotorom.attach(servoPin);
}

void loop() {
  azEnKicsiMotorom.write(0);
  delay(1000);
  azEnKicsiMotorom.write(90);
  delay(1000);
}
```

**Megoldás:** A 9-es pinre kötött szervomotort elforgatja 90 fokkal, várakozik 1 másodpercet, majd visszaforgatja az alaphelyzetbe.

## Feladat II: Mit csinál a kód?

```
int osszeg = 0;

void setup() {
  Serial.begin(9600);
  pinMode(LED, OUTPUT);

  for(int i = 0; i<10; i++) {
    if (i%2 == 0) {
      osszeg = osszeg+i;
    }
  }
  Serial.println(osszeg);
}

void loop() {
}
```

## Feladat II: Mit csinál a kód?

```
int osszeg = 0;

void setup() {
  Serial.begin(9600);
  pinMode(LED, OUTPUT);

  for(int i = 0; i<10; i++) {
    if (i%2 == 0) {
      osszeg = osszeg+i;
    }
  }
  Serial.println(osszeg);
}

void loop() {
}
```

**Megoldás:** 30. A for ciklus összeadja az összes páros számot 0-tól 10-ig, majd kiírja a végeredményt. Mindez egyszer fut le, mert a void setup-ba írtuk meg az egészet.

# Tartalom

- 1 Ismétlés
- 2 Ismétlő feladatok
- 3 Programozás: Új változó típusok**
- 4 Arduino: analogRead és joystick
- 5 Feladat
- 6 Szótár

# A kiegészítő kulcsszavak

## Mik azok a specifierek?

- A létrehozott változók típusait tudjuk egy picit különböző kulcsszavakkal árnyalni.
  - Ezeket a kis kiegészítő kulcsszavakat hívjuk specifier-nek
  - Sok specifier létezik, nekünk ezek közül egyre van szükségünk
- A specifierek az adott változó típusának egy-egy tulajdonságát változtatják meg, ezzel kiegészítve az adott típust.

## A konstans specifier

- Az arduinózás közben legtöbbször a `const` specifier-re van szükségünk, ezért mi most ezt fogjuk nektek elmagyarázni.
- Ha a `const` szócskát a változó létrehozásánál a típus elé írjuk, akkor az adott változó értéket a program nem tudja felülírni.
- Mikor tudjuk mi ezt használni?

# A const specifier

## A const specifier használata:

- Amikor a különböző alkatrészek pinjeit tesszük bele egy-egy változóba, akkor azokat nem szeretnénk már változtatni.
- Azonban előfordulhat, hogy a program valamiért mégis felülírja őket, így a program nem fog jól működni. Megoldás:
  - Az alkatrészek pinjeit konstans változókba mentjük el
  - Ezeket a változókat a program nem tudja felírni

## Példa:

```
//A pi-t mindenki ismeri:  
const float pi = 3.1415926 //konstans változó  
  
//Mit tehetek meg vele?  
float sugar = 2;  
float kor_kerulet = 2*sugar*pi; //Olvasni tudom  
  
//De mi van, ha megpróbálom átírni?  
pi = 3; //Hupsz, ezt már nem tehetem meg...hiba
```

# A szöveg tárolása

Milyen változó típusokat ismertünk meg eddig?

- `int`: egész számokat tároló típus
- `float`: tört számokat tároló típus

## A karakterláncok:

Az eddig megismert változókkal csak számokat tudunk tárolni. Na de mi van akkor, ha mi szöveget szeretnénk elmenteni egy változóba?

- A szövegtárolásra létrehoztak egy külön változót: `String`
  - Ezzel a típussal csak szöveget tudunk eltárolni.
  - Ezen kívül ugyanúgy kell használni, mint az eddig tanult változókat.
- A szöveges változós (ún. karakterláncok, azaz stringek) kezelésére rengeteg függvény áll rendelkezésünkre, ezért ezeket csak akkor mutatjuk majd be, amikor szükségetek lesz rá.

# A stringek kezelése I.

## A stringek létrehozása:

```
String alma = "alma";  
korte = String("korte");
```

## Az alapvető műveletek stringekkel:

- Stringek összefűzése
- Számok stringgé konvertálása
- Stringek számmá konvertálása

```
// Stringek osszefuzese:  
String gyumolcsok;  
gyumolcsok = alma+korte; //string valtozok osszeadasa  
gyumolcsok = String("alma" + "korte") //szovegek megadasa  
  
//Szamok konvertalasa szovegge:  
int szam = 13;  
String tizenharom = String(szam);
```



## A stringek kezelése II.

### A stringesített számok:

- A stringgé alakított számokat már nem tudjuk számként használni, mivel ezek szöveg típusúak lesznek. Erre jó példa:

```
void setup {  
    Serial.begin(9600);  
  
    String szam1 = "13";  
    String szam2 = "7";  
    String osszeg = szam1 + szam2;  
    Serial.println(osszeg);  
}  
//Amit a program kiír: "137"
```

- Az egyszerű kis példából látszik, hogy a szöveggé alakított számokkal nem tudunk műveleteket végezni.
- Ha stringben tárolt számokkal mégis számolni szeretnénk, akkor ezeket vissza kell alakítani valamilyen szám típusúvá. Na de hogyan?

## A stringek kezelése III.

### A stringek átkonvertálása számmá:

- Egy stringet csak akkor lehet visszakonvertálni számmá, ha az számmal kezdődik vagy csak szám van benne.
- A visszakonvertáláshoz függvényeket használunk:
  - `ToInt()`
  - `ToFloat()`

```
void setup {  
    Serial.begin(9600);  
  
    String szam1 = "13";  
    String szam2 = "7";  
    int osszeg = ToInt(szam1)+ToInt(szam2);  
    Serial.println(osszeg);  
}  
//Amit a program kiír: "20"
```

- Fontos, hogy ilyenkor a számolási eredményeket, már valamilyen számot tároló típusba kell elmenteni!

# Tartalom

- 1 Ismétlés
- 2 Ismétlő feladatok
- 3 Programozás: Új változó típusok
- 4 Arduino: analogRead és joystick**
- 5 Feladat
- 6 Szótár

# Az analogRead() függvény

## A digitalValami függvények

- Eddig ezeket a függvényeket használtuk
  - `digitalRead()`
  - `digitalWrite()`
- Ezek `HIGH` és `LOW` értékekkel dolgoznak, ahol `HIGH` 5V és `LOW` 0V.

## Az analogValami függvények:

- Az analóg függvények nem csak 2 értéket különböztetnek meg, hanem 1024 különböző értéket.
  - `analogRead()`
  - `analogWrite()`
- Az arduino 0-5V közötti feszültségeket tud kezelni, ezért az analóg függvényekkel ilyen értékeket tudunk beállítani vagy kiolvasni.
- Az analóg függvények 0-1023 közötti értékeket adnak vissza a pinen lévő feszültség alapján.
- A két skála közötti aránnyal ki tudjuk számolni a visszaadott értékekből a feszültségeket.

# A read függvények

Nekünk most az `analogRead()` függvényre lesz szükségünk, ezért ezt tárgyaljuk részletesebben.

## Összehasonlítás

Szempont	<code>digitalRead(int pin)</code>	<code>analogRead(int pin)</code>
Lehetőségek száma	2	1024
Értékkészlet	HIGH és LOW	0 és 1023 között
Feszültség	HIGH = 5V LOW = 0V	0 = 0V és 1023 = 5V
Arduino PIN	D2-D13	A0-A5

## Feszültség meghatározása

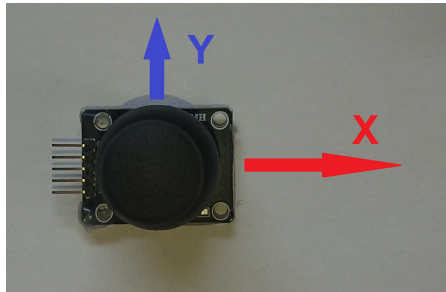
① Érték beolvasása: `int ertek = analogRead(pin);`

② Feszültség számítás:

```
float feszultseg = float(ertek) * 5.0 / 1023.0;
```

# Joystick

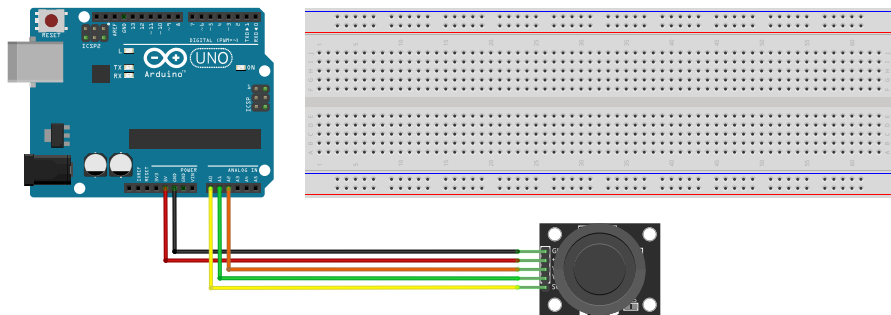
Mindenki tudja mit csinál...  
De hogy működik?



- GND pin és +5V pin  
feszültségellátás, Arduinora  
közvetlenül ráköthető
- VRx pin, az ezen olvasható  
feszültség változik az **X irányú**  
helyzet függvényében
- VRy pin, az ezen olvasható  
feszültség változik az **Y**  
**irányú** helyzet függvényében
- SW pin, 0 közeli feszültséget  
ad ki, ha benyomtuk a  
joystickot

# Joystick

## Bekötés



fritzing

- Joystick GND → Arduino GND
- Joystick +5V → Arduino 5V
- Joystick SW → Arduino A0
- Joystick VRx → Arduino A2
- Joystick VRy → Arduino A1

# Irassuk ki az olvasott értékeket!

```
//Joystick (Analóg PIN-ek! A0,A1,A2...)
const int VRx = 2;
const int VRy = 1;
const int SW = 0;

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

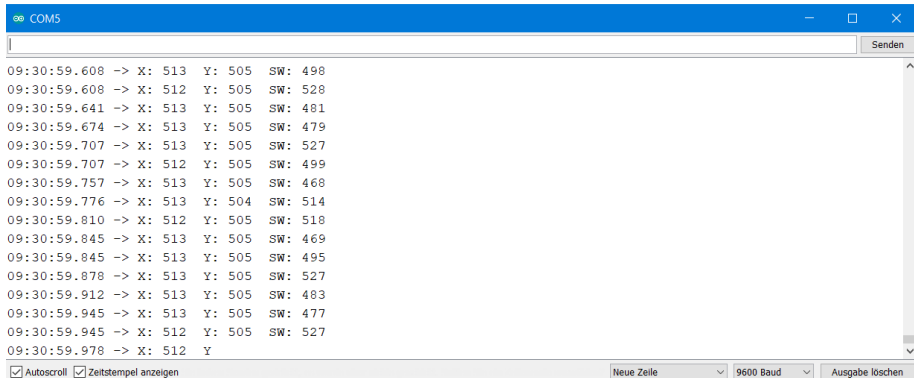
void loop()
{
  int X_value = analogRead(VRx);
  int Y_value = analogRead(VRy);
  int SW_value = analogRead(SW);

  Serial.println("X: "+String(X_value)+
    " Y: "+String(Y_value)+" SW: "+String(SW_value));
}
```



# Állapotok értelmezése

## Alap állapot



The screenshot shows the Arduino IDE Serial Monitor window titled "COM5". It displays a series of data points received from the Arduino, each representing a joystick state at a specific time. The data is formatted as "Time -> X: X Y: Y SW: SW". The X and Y coordinates are either 512 or 513, and the SW (switch) value is either 498, 481, 479, 527, 499, 468, 514, 518, 469, 495, 527, 483, 477, or 527. The time values range from 09:30:59.608 to 09:30:59.978. The window includes a "Senden" button at the top right and a status bar at the bottom with checkboxes for "Autoscroll" and "Zeitstempel anzeigen", and buttons for "Neue Zeile", "9600 Baud", and "Ausgabe löschen".

```
09:30:59.608 -> X: 513 Y: 505 SW: 498
09:30:59.608 -> X: 512 Y: 505 SW: 528
09:30:59.641 -> X: 513 Y: 505 SW: 481
09:30:59.674 -> X: 513 Y: 505 SW: 479
09:30:59.707 -> X: 513 Y: 505 SW: 527
09:30:59.707 -> X: 512 Y: 505 SW: 499
09:30:59.757 -> X: 513 Y: 505 SW: 468
09:30:59.776 -> X: 513 Y: 504 SW: 514
09:30:59.810 -> X: 512 Y: 505 SW: 518
09:30:59.845 -> X: 513 Y: 505 SW: 469
09:30:59.845 -> X: 513 Y: 505 SW: 495
09:30:59.878 -> X: 513 Y: 505 SW: 527
09:30:59.912 -> X: 513 Y: 505 SW: 483
09:30:59.945 -> X: 513 Y: 505 SW: 477
09:30:59.945 -> X: 512 Y: 505 SW: 527
09:30:59.978 -> X: 512 Y
```

Látható hogy ha a joystickot nem piszkáljuk mindegyik érték 500 körül van, ez nagyjából 2.5 V-nak felel meg.

# Állapotok értelmezése

## Balra állapot

```

COM5

09:30:00.603 -> X: 0  Y: 658  SW: 202
09:30:00.637 -> X: 2  Y: 658  SW: 466
09:30:00.672 -> X: 1  Y: 658  SW: 711
09:30:00.722 -> X: 1  Y: 659  SW: 617
09:30:00.722 -> X: 1  Y: 658  SW: 309
09:30:00.722 -> X: 1  Y: 658  SW: 195
09:30:00.741 -> X: 1  Y: 658  SW: 451
09:30:00.741 -> X: 1  Y: 658  SW: 715
09:30:00.776 -> X: 2  Y: 658  SW: 658
09:30:00.810 -> X: 2  Y: 658  SW: 351
09:30:00.845 -> X: 1  Y: 658  SW: 196
09:30:00.845 -> X: 2  Y: 658  SW: 417
09:30:00.879 -> X: 1  Y: 659  SW: 694
09:30:00.913 -> X: 2  Y: 659  SW: 684
09:30:00.913 -> X: 1  Y: 658  SW: 389
09:30:00.947 -> X: 2  Y: 6  SW: 6

☒ Autoscroll ☒ Zeitstempel anzeigen
Neue Zeile 9600 Baud Ausgabe löschen

```

Ha a joystickot balra húzzuk, akkor az X érték közel 0 lesz, a többi érték alig változik. A balra húzás X irányú mozgatus a korábbi ábra szerint.

# Állapotok értelmezése

## Lefele állapot

The screenshot shows the Serial Monitor window for COM5. It displays a series of joystick readings over time. The data shows that when the joystick is moved downwards, the Y-axis value is around 1023, while the X-axis value fluctuates slightly. The SW (switch) value also varies.

```

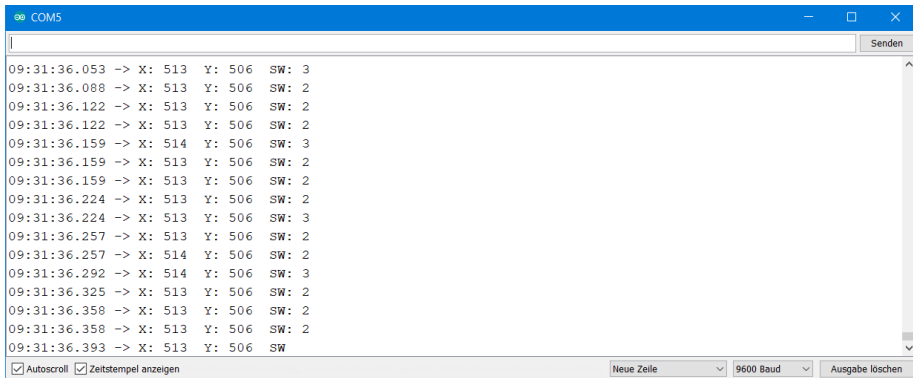
09:32:00.025 -> X: 748 Y: 1023 SW: 988
09:32:00.060 -> X: 748 Y: 1023 SW: 997
09:32:00.095 -> X: 749 Y: 1023 SW: 1023
09:32:00.128 -> X: 748 Y: 1023 SW: 978
09:32:00.162 -> X: 748 Y: 1023 SW: 1014
09:32:00.162 -> X: 748 Y: 1023 SW: 1011
09:32:00.197 -> X: 748 Y: 1023 SW: 983
09:32:00.231 -> X: 748 Y: 1023 SW: 1023
09:32:00.266 -> X: 749 Y: 1023 SW: 986
09:32:00.300 -> X: 748 Y: 1023 SW: 1001
09:32:00.333 -> X: 748 Y: 1023 SW: 1023
09:32:00.333 -> X: 747 Y: 1023 SW: 978
09:32:00.368 -> X: 749 Y: 1023 SW: 1023
09:32:00.402 -> X: 748 Y: 1023 SW: 1000
09:32:00.402 -> X: 748 Y: 1023 SW: 994
09:32:00.436 -> X: 748 Y: 1023 SW: 102
  
```

At the bottom of the window, there are checkboxes for 'Autoscroll' and 'Zeitstempel anzeigen' (both checked), a dropdown for 'Neue Zeile' (set to 'Neue Zeile'), a dropdown for '9600 Baud', and a button 'Ausgabe löschen'.

Ha a joystickot lefele húzzuk, akkor az Y érték 1023 körül lesz, a többi érték bár kicsit megváltozik, ezzel nem foglalkozunk. A lefelé húzás Y irányú mozgató.

# Állapotok értelmezése

## Gombnyomás



The screenshot shows the Arduino IDE Serial Monitor window for COM5. The window has a blue title bar with standard Windows window controls. Below the title bar is a text input field and a 'Senden' button. The main area displays a stream of data from the Arduino, consisting of timestamps followed by 'X', 'Y', and 'SW' values. The 'SW' value is consistently 2 or 3. At the bottom, there are checkboxes for 'Autoscroll' and 'Zeitstempel anzeigen', and buttons for 'Neue Zeile', '9600 Baud', and 'Ausgabe löschen'.

```
09:31:36.053 -> X: 513 Y: 506 SW: 3
09:31:36.088 -> X: 513 Y: 506 SW: 2
09:31:36.122 -> X: 513 Y: 506 SW: 2
09:31:36.122 -> X: 513 Y: 506 SW: 2
09:31:36.159 -> X: 514 Y: 506 SW: 3
09:31:36.159 -> X: 513 Y: 506 SW: 2
09:31:36.159 -> X: 513 Y: 506 SW: 2
09:31:36.224 -> X: 513 Y: 506 SW: 2
09:31:36.224 -> X: 513 Y: 506 SW: 3
09:31:36.257 -> X: 513 Y: 506 SW: 2
09:31:36.257 -> X: 514 Y: 506 SW: 2
09:31:36.292 -> X: 514 Y: 506 SW: 3
09:31:36.325 -> X: 513 Y: 506 SW: 2
09:31:36.358 -> X: 513 Y: 506 SW: 2
09:31:36.358 -> X: 513 Y: 506 SW: 2
09:31:36.393 -> X: 513 Y: 506 SW
```

☒ Autoscroll ☒ Zeitstempel anzeigen Neue Zeile 9600 Baud Ausgabe löschen

Ha a gombot lenyomjuk az SW érték közel 0 lesz. A gomblenyomás történhet bármilyen állapotban, SW mindig közel 0 lesz.

# Állapotok összefoglalása

## Balra állapot

$X < 100$  többi nem számít

## Jobbra állapot

$X > 900$  többi nem számít

## Lefele állapot

$Y > 900$  többi nem számít

## Felfele állapot

$Y < 100$  többi nem számít

## Gombnyomás

$SW < 100$  többi nem számít

# Állapotok összefoglalása

## Alap állapot

$100 < X < 900$  és  $100 < Y < 900$  és  $SW \approx 500$

**Megjegyzés:** Az állapotok definiálhatók szigorúbban, például megadható lenne az is hogy a gombnyomás csak akkor történik meg ha  $SW < 3$ , azonban a mérés pontatlansága elég nagy is lehet, ezért ha biztos minden gombnyomást érzékelni szeretnénk akkor egy bővebb tartományt célszerű megadni. Ugyanez igaz az X és Y értékekre is.

**Figyelem** ha a joystickot más irányból nézzük az irányok is megváltoznak, lehet nektek máshogy kell megadni az irányokat.

# Tartalom

- 1 Ismétlés
- 2 Ismétlő feladatok
- 3 Programozás: Új változó típusok
- 4 Arduino: analogRead és joystick
- 5 Feladat**
- 6 Szótár

# Állapotok kiírása

Írjunk egy programot ami az olvasott X, Y és SW értékek alapján megadja hogy a joystick hogy áll.

A programban minden állapotot egy `if` állítással tudunk megvizsgálni. Az alap állapotot nem kell vizsgálni.



# Állapotok kiiratása

```
const int VRx = 2;
const int VRy = 1;
const int SW = 0;

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

void loop() {
  int VRx_value = analogRead(VRx);
  int VRy_value = analogRead(VRy);
  int SW_value = analogRead(SW);
  .
  .
}
```

Első 3 sorban defináljuk a joystick pineket, majd elindítjuk a soros portot a **Serial.begin** függvénnyel, a **loop** elején beolvassuk a szükséges értékeket.

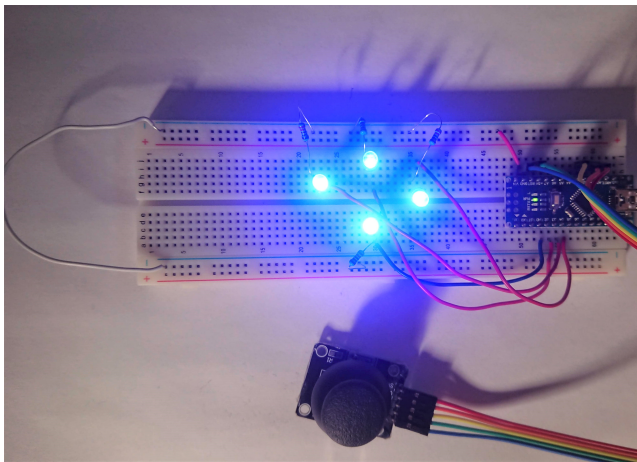
# Állapotok kiírása

```
if(SW_value < 10){  
    Serial.println("Gomb lenyomva");  
}  
if(VRx_value < 100){  
    Serial.println("Balra húzva");  
}  
if(VRx_value > 900){  
    Serial.println("Jobbra húzva");  
}  
if(VRy_value < 100){  
    Serial.println("Felfele húzva");  
}  
if(VRy_value > 900){  
    Serial.println("Lefele húzva");  
}  
} //void loop vége
```

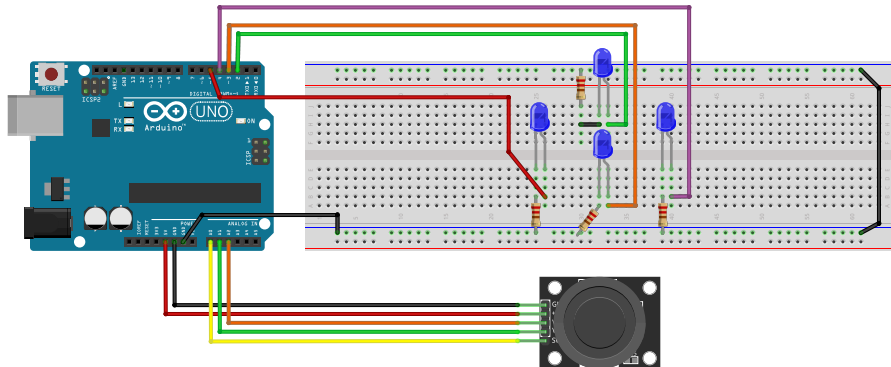
Az összes állapoton végigmegyünk.

# LED kiválasztása joystickkal

Helyezzünk el 4 LED-et egy + jel alakban, ahol a joystick megfelelő állapotba állításával lehet a megfelelő LED-et felvillantani, a csatolt videó alapján.



# Kapcsolás



fritzing

# Kód I

```
//LED-ek
const int FENT_LED = 2;
const int LENT_LED = 3;
const int BAL_LED = 5;
const int JOBB_LED = 4;

//Joystick (Analóg PIN-ek! A0,A1,A2...)
const int VRx = 2;
const int VRy = 1;
const int SW = 0;

void setup()
{
    //LED-ek beállítása kimenetnek
    pinMode(FENT_LED, OUTPUT);
    pinMode(LENT_LED, OUTPUT);
    pinMode(BAL_LED, OUTPUT);
    pinMode(JOBB_LED, OUTPUT);
} . . .
```

## Kód II folytatás

```
void loop() {  
  int VRx_value = analogRead(VRx);  
  int VRy_value = analogRead(VRy);  
  int SW_value = analogRead(SW);  
  
  if(SW_value < 100){ //Gomb lenyomva  
    digitalWrite(FENT_LED,HIGH);  
    digitalWrite(LENT_LED,HIGH);  
    digitalWrite(BAL_LED,HIGH);  
    digitalWrite(JOBB_LED,HIGH);  
  }  
  if(VRx_value < 100){ //Balra húzva  
    digitalWrite(FENT_LED,LOW);  
    digitalWrite(LENT_LED,LOW);  
    digitalWrite(BAL_LED,HIGH);  
    digitalWrite(JOBB_LED,LOW);  
  } . . .  
}
```

## Kód II folytatás

```
if(VRx_value > 900){ //Jobbra húzva
    digitalWrite(FENT_LED, LOW);
    digitalWrite(LENT_LED, LOW);
    digitalWrite(BAL_LED, LOW);
    digitalWrite(JOBB_LED, HIGH);
}
if(VRy_value < 100){ //Felfele húzva
    digitalWrite(FENT_LED, HIGH);
    digitalWrite(LENT_LED, LOW);
    digitalWrite(BAL_LED, LOW);
    digitalWrite(JOBB_LED, LOW);
}
if(VRy_value > 900){ //Lefele húzva
    digitalWrite(FENT_LED, LOW);
    digitalWrite(LENT_LED, HIGH);
    digitalWrite(BAL_LED, LOW);
    digitalWrite(JOBB_LED, LOW);
}
} //void loop vége
```

# Tartalom

- 1 Ismétlés
- 2 Ismétlő feladatok
- 3 Programozás: Új változó típusok
- 4 Arduino: analogRead és joystick
- 5 Feladat
- 6 Szótár**



## Szavak

r Joystick /-s	joystick
s Schlüsselwort /-wörter	kulcsszó
ergänzend	kiegészítő (mn.)
konstant	konstans
r Text /-e	szöveg
e ganze Zahl /-e	egész szám
r Bruch / e Brüche	törtszám
e Zeichenkette /-n (r String)	karakterlánc (string)
aneinanderhängen ( te, h. ge...t)	összefűz (stringeket)
e Behandlung /-en	kezelés
umwandeln von+D in+A	átváltani
auslesen +A.	kiolvasni vmit
einstellen +A.	beállítani vmit
s Verhältnis /-se	arány
e Richtung /-en	irány