

## CYKLUS FOR

Cyklus **for** se používá k opakování bloku příkazů, uzavřených do složených závorek. Využívá čítače inkrementů (přírůstků), který se používá pro ukončení průchodu cyklu. Cyklus **for** je vhodný pro jakékoliv opakující se operace a je často používán v kombinaci s **poli** pro jejich průchod.

### SYNTAXE

```
1  for (inicIALIZACE; PODMÍNKA; PŘÍRŮSTEK) {
2      // blok příkazů
3  }
4
5 // praktická ukázka
6 void loop()
7 {
8     for (int i=0; i <= 255; i++){
9         analogWrite(PWMpin, i);
10        delay(10);
11    }
12 }
```

Nejprve nastane **inicIALIZACE**, a to minimálně jednou. V každém průchodu je testována **PODMÍNKA**. Pokud podmínka nabude hodnoty **True**, provede se blok příkazů a zvětší se **PŘÍRŮSTEK**. Podmínka je opět testována. Když podmínka nabude hodnoty **False**, smyčka se ukončí.

## SÉRIOVÁ KOMUNIKACE

Pokud chceme od Arduina získávat nějaké hodnoty, nebo mu je posílat, přichází na řadu právě sériová komunikace. Ta je také důležitým nástrojem při ladění programů. Když něco nefunguje a nevíme proč, může být užitečné si nechat vypisovat hodnotu proměnné, nebo určitý text na libovolném místě programu a tím zjistit co přesně kód provádí. Dá se také využít při komunikaci Arduina a dalších zařízení (jiné Arduino, moduly atd.).

Pro čtení informací v textové podobě na počítači se používá tzv. **Serial monitor**. Ten se spustí kliknutím na ikonu  v IDE Arduina. Po spuštění Serial monitoru se musí ještě nastavit rychlosť komunikace (300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 a 115200).



### Základní funkce sériové komunikace

#### **Serial.begin(9600)**

Funkce pro zahájení sériové komunikace. Zpravidla ji stačí zavolat v části **setup()**. Parametrem je rychlosť komunikace, která odpovídá počtu přenosů za sekundu.

#### **Serial.println(hodnota)**

Funkce slouží k odeslání hodnoty z Arduina do počítače nebo jiného zařízení. Nám poslouží také k výpisu hodnot v sériovém monitoru.

## SYNTAXE

```
1 void setup() {  
2     Serial.begin(9600);  
3     for (count=0; count<255; count++) {  
4         Serial.println(count);  
5         delay(10);  
6     }  
7 }  
8  
9 void loop() {  
10}  
11 }
```

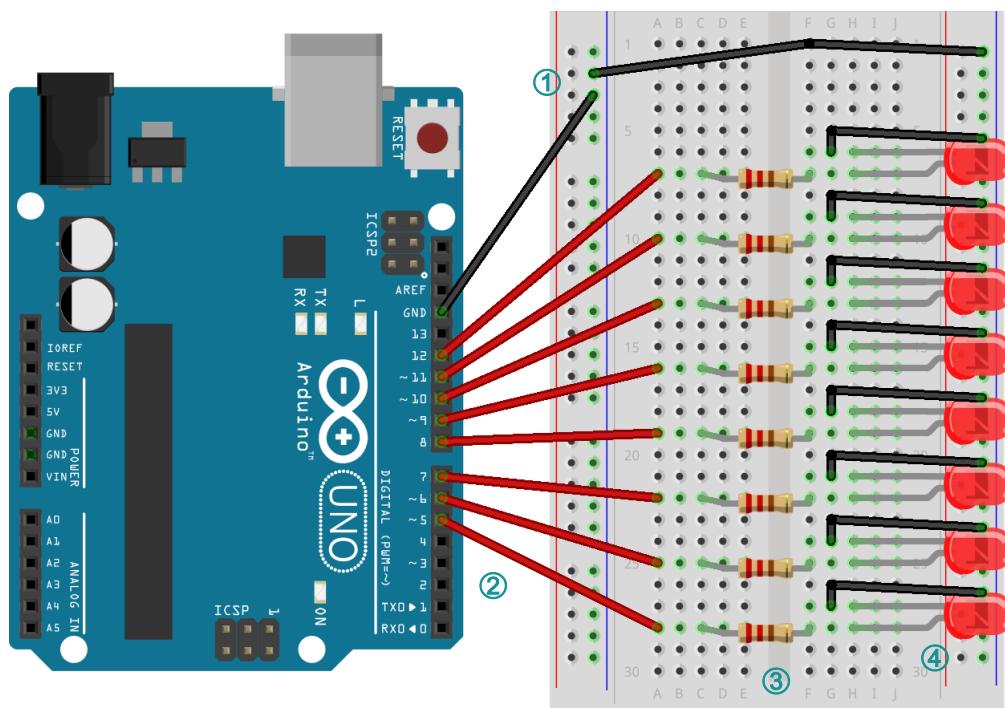
- ① Pokud se bude využívat sériové komunikace, je nutné nastavit rychlosť prenosu.
- ② Výpis hodnôt v sériovom monitoru. V tomto prípade se bude vypisovať hodnota premennej **i**. Každá hodnota bude na novom riadku, ak bychom chceli hodnoty všetkých vedľa seba v jednom riadku, použijeme funkciu **Serial.print(count)**.

## SVĚTELNÁ ANIMACE

Vytvoříme nástroj na světelnou animaci, který vás naučí řídit větší počet pinů desky Arduino. Nejprve využijeme kontaktní pole, kde si otestujeme zapojení všech diod. Následně si vytvoříte kartonovou konstrukci, do které LED diody umístíte a připojíte pomocí vodičů.

## ZAPOJENÍ OBVODU

Zapojení obvodu vychází z přechozího příkladu zapojení LED diody s tím rozdílem, že zde bude zapojeno osm diod osm. Každá z osmi diod bude ovládána nezávisle.



Obr. 1 - Zapojení osmi LED diod

- ① Vodič zemnění z desky Arduino zapojte do kontaktní desky k **modré** čáře. Tím se zpřehlední celé zapojení.
- ② Najděte 8 rezistorů o hodnotě  $220\Omega$  (dva oranžové proužky, hnědý a zlatý). Je důležité, aby tyto rezistory byly zapojeny do kontaktní desky tak, aby propojovaly její obě

poloviny. Pokud by tomu tak nebylo, došlo by ke zkratování. Rezistory zajistí snížení proudu, aby nedošlo ke zničení LED diod.

- ③ Zapojte 8 LED diod do kontaktní desky tak, jak je uvedeno na obrázku Obr. 1 - Zapojení osmi LED diod. Delší vývody každé LED diody jsou připojeny k rezistorům. Kratší vývod je propojen s **uzemněním**.
- ④ Druhá strana rezistorů je zapojena do digitálních pinů 5, 6, 7, 8, 9, 10, 11 a 12.

## PROGRAMOVÝ KÓD

Při psaní kódu pro všechny příklady bude použité jediné zapojení uvedené na obrázku Obr. 1 - Zapojení osmi LED diod.



Příklady na sebe navazují, takže pro jejich inovace stačí provádět pouze dílčí úpravy programového kódu. Není nutné vždy psát celý program od začátku.

## ZÁKLADNÍ PŘÍKLAD – POSTUPNÉ ROZSVĚCOVÁNÍ LED DIOD

Ukázka „hrubého“ programového kódu pro postupné rozsvěcování a zhasnání LED diod.

Tím se vytvoří efekt „běžícího světla“.

```
1 void setup() {
2     pinMode(5, OUTPUT);      // dioda 1
3     pinMode(6, OUTPUT);      // dioda 2
4     pinMode(7, OUTPUT);      // dioda 3
5     pinMode(8, OUTPUT);      // dioda 4
6     pinMode(9, OUTPUT);      // dioda 5
7     pinMode(10, OUTPUT);     // dioda 6
8     pinMode(11, OUTPUT);     // dioda 7
9     pinMode(12, OUTPUT);     // dioda 8
10 }
11
12 void loop() {
13     digitalWrite(5, HIGH);    // rozsvícení diody 1
14     delay(50);
15     digitalWrite(5, LOW);     // zhasnutí diod
16     delay(50);
17     digitalWrite(6, HIGH);    // rozsvícení diody 2
18     delay(50);
19     digitalWrite(6, LOW);     // zhasnutí diod
20     delay(50);
21     digitalWrite(7, HIGH);    // rozsvícení diody 3
22     delay(50);
23     digitalWrite(7, LOW);     // zhasnutí diod
24     delay(50);
25     digitalWrite(8, HIGH);    // rozsvícení diody 4
26     delay(50);
27     digitalWrite(8, LOW);     // zhasnutí diod
28     delay(50);
29     digitalWrite(9, HIGH);    // rozsvícení diody 5
30     delay(50);
31     digitalWrite(9, LOW);     // zhasnutí diod
32     delay(50);
33     digitalWrite(10, HIGH);   // rozsvícení diody 6
34     delay(50);
35     digitalWrite(10, LOW);    // zhasnutí diod
36     delay(50);
37     digitalWrite(11, HIGH);   // rozsvícení diody 7
38     delay(50);
39     digitalWrite(11, LOW);    // zhasnutí diod
40     delay(50);
41 }
```

①

②

```
42   digitalWrite(12, HIGH);      // rozsvícení diody 8
43   delay(50);
44   digitalWrite(12, LOW);       // zhasnutí diod
45   delay(50);
46 }
```

⑤ Pro každou LED diodu je definován výstupní pin pomocí funkce **pinMode()**.

V příkladu jsou definovány piny **5, 6, 7, 8, 9, 10, 11, 12**.

⑥ Pro rozsvícení diody se využívá funkce **digitalWrite()** s parametrem **číslo pinu** a hodnotu **HIGH**.

Pro zhasnutí diody se volá také funkce **digitalWrite()** s parametrem **číslo pinu**, ale s hodnotou **LOW**. Aby bylo zřetelné postupné rozsvěcování a zhasínání, je použita funkce pro pozastavení běhu programu **delay()**.



(Př. 1) Změňte předchozí příklad tak, aby se nejdříve všechny LED diody postupně rozsvítily a následně se od konce opět postupně zhasínaly.



### NESVÍTÍ DIODY

**Zapojení LED diody** – zkontrolujte zapojení LED diod v kontaktním poli. Ujistěte se, že jednotlivé diody mají anodu a katodu správně zapojenou. **Anoda** musí být zapojena k rezistoru a následně do digitálního pinu na desce Arduino. **Katoda** je zapojena na zemnění desky.

**Zapojení v desce** – zkontrolujte, zda jsou vývody zapojeny do odpovídajících pinů desky Arduino.

**Rezistory** – zkontrolujte, zda jste použili správné rezistory.

### NEJDE NAHRÁT KÓD DO DESKY

**USB kabel** – ujistěte se, že máte desku Arduino připojenou k počítači.

**Správný port** – ujistěte se, že máte vybraný správný port pro připojení k desce Arduino pomocí USB kabelu.

## ZÁKLADNÍ PŘÍKLAD – VLASTNÍ FUNKCE

Příklad pro rozsvícení a zhasnání LED diod je jednoduchý, ale příliš dlouhý a tím i málo přehledný. Provedeme tedy jeho inovaci s využitím vlastní funkce.

Nadefinujeme funkci **changeLED(int pin)**, která má jediný parametr. Tím je číslo pinu, na který je dioda připojena.

```
1 void setup() {
2     pinMode(5, OUTPUT);      // dioda 1
3     pinMode(6, OUTPUT);      // dioda 2
4     pinMode(7, OUTPUT);      // dioda 3
5     pinMode(8, OUTPUT);      // dioda 4
6     pinMode(9, OUTPUT);      // dioda 5
7     pinMode(10, OUTPUT);     // dioda 6
8     pinMode(11, OUTPUT);     // dioda 7
9     pinMode(12, OUTPUT);     // dioda 8
10 }
11
12 void loop() {
13     changeLED (5);
14     changeLED (6);
15     changeLED (7);
16     changeLED (8);
17     changeLED (9);
18     changeLED (10);
19     changeLED (11);
20     changeLED (12);
21 }
22
23 void changeLED(int pin) {
24     digitalWrite(pin, HIGH);    // rozsvícení diody 1
25     delay(50);
26     digitalWrite(pin, LOW);    // zhasnutí diod
27     delay(50);
28 }
```

②

①

- ① Deklarace funkce **changeLED()**, která rozsvěcí a zhasíná konkrétní diodu, definovanou číslem digitálního pinu jako jediného parametru funkce.
- ② Funkce **changeLED()** se volá ve smyčce funkce **loop()**.



(Př. 2) Změňte předchozí příklad s využitím funkce tak, aby se diody nejdříve postupně všechny rozsvítily a následně postupně zhasínaly. Návod vám může být, že se bude muset změnit funkce **changeLED()**.

## ZÁKLADNÍ PŘÍKLAD – POLE A CYKLUS FOR

Následující příklad využívá kombinaci programových struktur jako jsou **funkce**, **pole** a cyklus **for**. Program tak získá na přehlednosti a větší flexibilitě. Příklad ukazuje jezdící světlo z jedné strany na druhou a zpět.

```
1 int pinArray[] = {5, 6, 7, 8, 9, 10, 11, 12};          | ①
2 int count = 0;                                         | ②
3 int timer = 50;                                        | ③
4
5 void setup() {
6     for (count=0; count<8; count++) {
7         pinMode(pinArray[count], OUTPUT);                | ④
8     }
9 }
10
11 void loop() {
12     for (count=0; count<8; count++) {
13         changeLED(pinArray[count]);                   | ⑤
14     }
15     for (count=7; count>=0; count--) {
16         changeLED(pinArray[count]);                   | ⑥
17     }
18 }
19
20 void changeLED(int pin) {
21     digitalWrite(pin, HIGH);
22     delay(timer);
23     digitalWrite(pin, LOW);
24     delay(timer);
25 }
```

- ① Definice pole **pinArray[]**, které obsahuje čísla pinů, na které jsou připojeny LED diody. Výhodou této konstrukce je možnost jednoduchého rozšíření o další LED diody.
- ② Definice proměnné **count**, která je iterátorem v cyklech **for**.
- ③ Definice proměnné **timer**, která je parametrem funkce **delay**, která přerušuje běh programu proto, aby bylo vidět problíknutí LED diody.
- ④ Pomocí cyklu **for** jsou definovány piny jako výstupy desky Arduino. Všimněte si, že proměnná **count** začíná hodnotou **0** a cyklus bude probíhat, dokud její hodnota bude

menší než **8**. Proč? Připomeňme si, že prvky pole jsou indexovány od nuly a zapojených diod máme osm. Takže od 0 do 7 je to právě hodnota 8.

- ⑤ První cyklus **for** ve funkci **loop()** rozsvěcí LED diody ve vzestupném pořadí. V každém kroku se volá funkce **changeLED()** s jediným parametrem, kterým je číslo pinu, na který je konkrétní dioda připojena.
- ⑥ Druhý cyklus ve funkci **loop()** určuje sestupné pořadí rozsvěcení. Všimněte si proměnné **count**, která se v každém kroku zmenšuje díky dekrementačnímu operátoru **--**.
- ⑦ Funkce **changeLED()** rozsvěcí a zhasíná konkrétní diodu připojenou na pin desky Arduino, který je funkci předáván jako parametr.



(Př. 3) Vytvořte libovolnou světelnou animaci. Můžete využít předchozí příklad a změnit pořadí pinů v poli, přidat další kombinaci pořadí pinů.



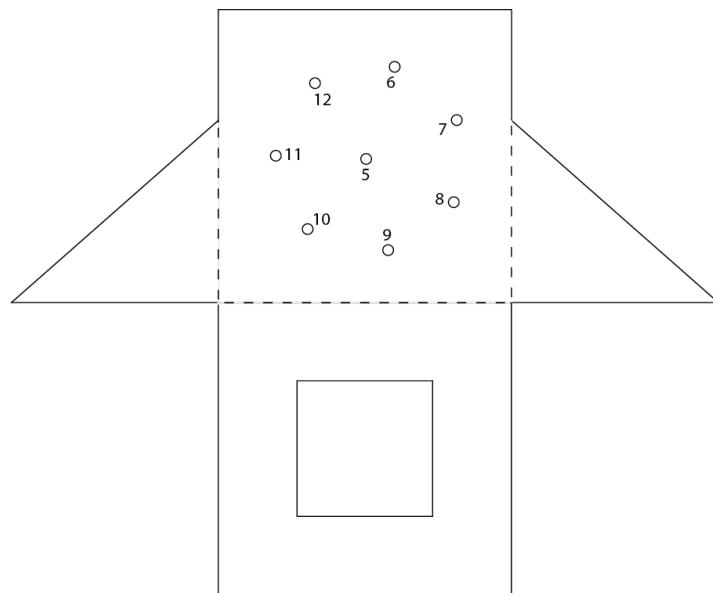
Pokud budete mít připravenou a funkční programovou část, můžete vytvořit stojánek pro LED diody. Tím animace bude efektnější. Návod, je uveden v následující kapitole.

## STOJAN PRO LED DIODY

Pro vytvoření stojanu budeme potřebovat: karton (tvrdší papír), nůžky a tavnou pistoli (lepící pásku).

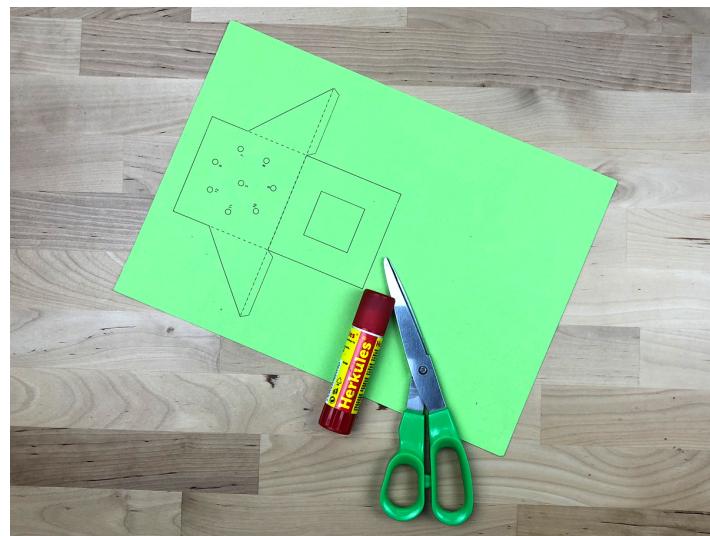
## PAPÍROVÁ KONSTRUKCE

Konstrukce stojánku je velice jednoduchá. Stačí si vytisknout na pevný papír přiloženou šablonu, která je na obrázku Obr. 2 - Šablona stojanu.

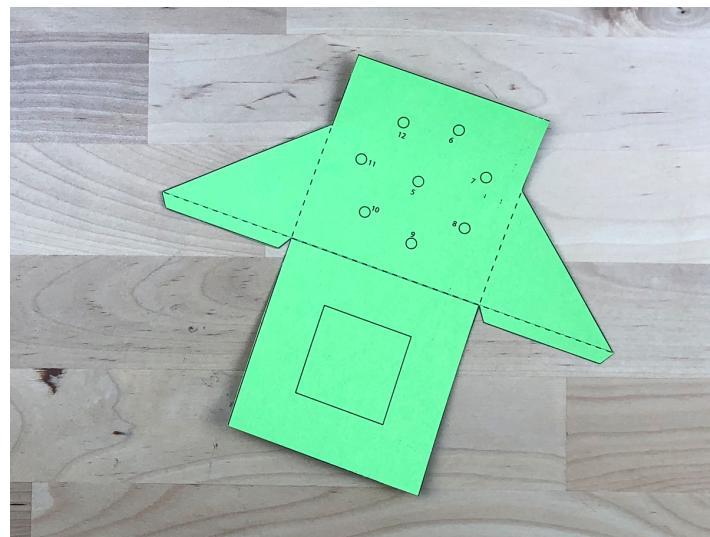


Obr. 2 - Šablona stojanu

Vytištěnou šablonu vystříhněte podle plných čar.



Obr. 3 - Vytištěná šablona



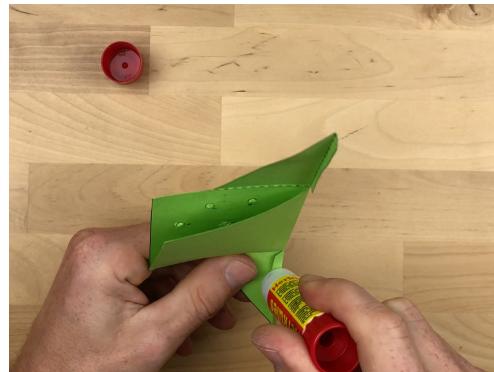
Obr. 4 - Vystřížená šablona

Do naznačených otvorů, které jsou očíslovány podle čísel pinů, do kterých jsou připojeny jednotlivé LED diody, udělejte otvory. Velikost otvorů odpovídá průměru LED diod.

Následně naohýbejte šablonu podle přerušovaných čar. Na lepící hrany naneste lepidlo.

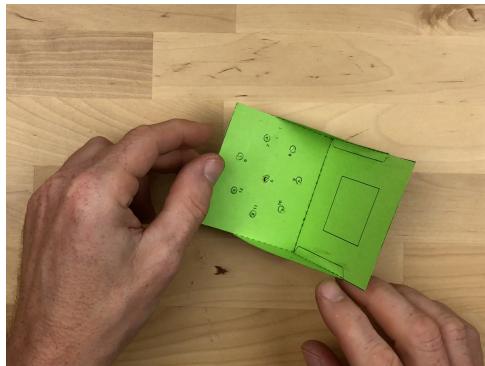


Obr. 5 – Ohýbání lepicích hran

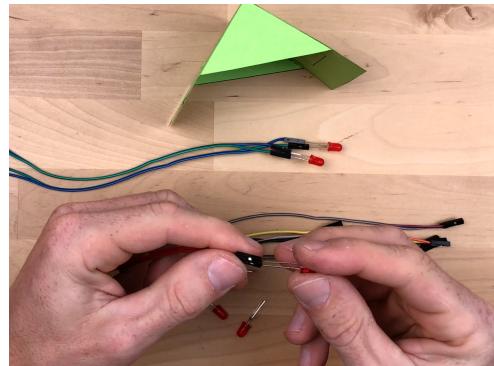


Obr. 6 – Lepení hran

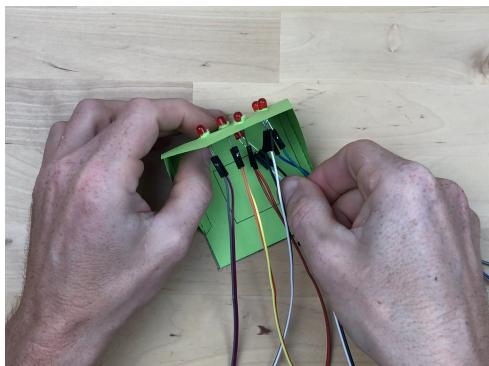
Lepící hrany přilepte k základně stojanu Obr. 7 – Přilepení hran k základně. Než zasunete LED diody do stojanu, připojte k nim vodiče Obr. 8 – Připojení LED diod.



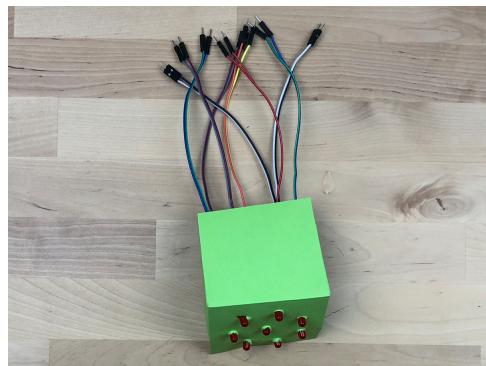
Obr. 7 – Přilepení hran k základně



Obr. 8 – Připojení LED diod



Obr. 9 – Vložení LED diod do stojanu



Obr. 10 – Složený stojan