

PRACOVNÍ LIST – CYKLUS FOR

V TÉTO ČÁSTI BUDETE POKRAČOVAT V PROGRAMOVÁNÍ SOUSTAVY LED DIOD.
NAUČÍTE SE OPTIMALIZOVAT PROGRAMOVÝ KÓD POMOCÍ CYKLU FOR.

CO SE NAUČÍTE

- ① Programovat cyklus **for**.
- ② Procházení pole pomocí cyklu **for**.
- ③ Pracovat se sériovým monitorem.



CO BUDETE POTŘEBOVAT

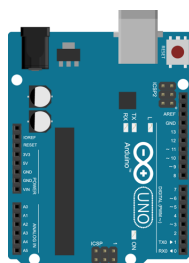
- ① LED diodu - 8x.
- ② Rezistor 220Ω – 8x.
- ③ Desku Arduino.
- ④ Kontaktní pole.
- ⑤ Vodiče typu zástrčka-zástrčka.



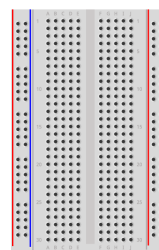
RGB led – 8 kusů



Rezistor 220Ω – 8 kusů



Deska Arduino



Kontaktní pole

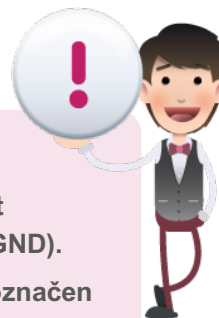
POUŽITÉ SOUČÁSTKY

A JDĚTE NA TO ...

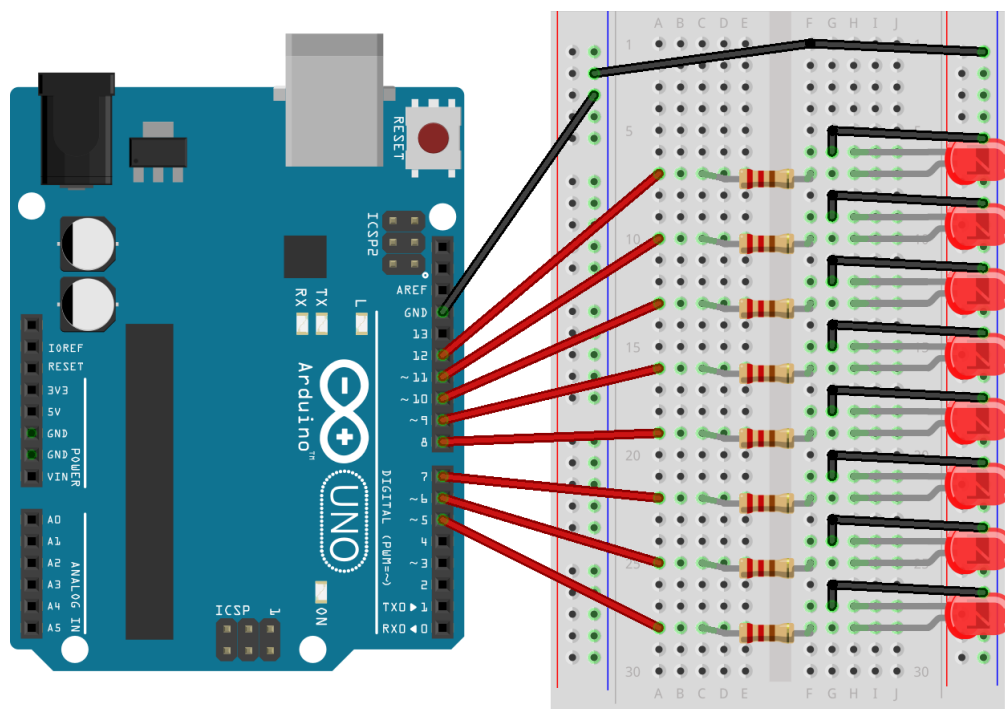
- ① Pokud máte složený elektronický obvod z minulé hodiny, můžete se pustit rovnou do programování. V opačném případě obvod musíte opět složit podle přiloženého schématu.

DEJTE SI POZOR

- ➔ Pozor si dejte na to, jak zapojujete LED diody. Delší vývod musí být připojen přes rezistor k pinu. Kratší vývod je připojen na zem (pin GND).
- ➔ Dejte si pozor na hodnotu rezistoru. Zkontrolujte si, že je barevně označen v pořadí červená, červená, modrá, černá, zlatá.
- ➔ Všimněte si, jak je zapojen vodič zemnění. Pro přehlednost je veden na druhou stranu kontaktního pole. Následně je zemnění vedeno ke každé LED diodě zvlášť (černý vodič).



- ② Otevřete programový kód z minulé hodiny, kde jste pracovali s polem.



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 setup()  
7  {  
8      for (int i=0; i < 255; i++){  
9          analogWrite(PWMPin, i);  
10         delay(10);  
11     }  
12 }
```

V první řadě 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.



Uvedená položka **přírůstek** se také nazývá složeným operátorem. Zkracuje nám zápis operací a v případě cyklu **for** hodnotu proměnné zvyšují **i++** a nebo snižují **i--**.

SÉRIOVÝ MONITOR

Sériový monitor se využívá při čtení informací v textové podobě. Souvisí se sériovou komunikací, kterou můžeme využít například pokud chceme získávat nějaké hodnoty z Arduina nebo naopak je do něj posílat.




Serial.begin(9600)

Funkce pro zahájení sériové komunikace. Zpravidla ji stačí zavolat v části **setup()**. Parametrem je rychlost 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.

Sériový monitor si spustíte kliknutím v IDE Arduino na ikonu 



ÚKOL PRO VÁS

A) Výše uvedený příklad v popisu syntaxe (praktická ukáзка), upravte tak, aby se v sériovém monitoru vypisovala hodnota proměnné `i`.

OTÁZKA PRO VÁS

- ➔ Podařilo se vám vypisovat hodnotu proměnné `i`?
- ➔ V jakém rozmezí se zobrazovaly proměnné `i`?
- ➔ Ve které části programového kódu s poli by se hodilo využít cyklu `for`, aby se kód zjednodušil?



ÚKOLY PRO VÁS

- ➔ B) Předchozí úkol, ve kterém jste čísla pinů nahradili prvky pole, upravte tak, abyste použili příkaz cyklu `for` a světlo diod probíhalo z jedné strany na druhou, neustále dokola.
- ➔ C) Upravte programový kód tak, aby se běžící světlo pohybovalo z jedné strany na druhou a zpět.

