

Lekce 4

While cyklus, for cyklus

Obsah

[Motivace](#)

[Prostředky I - While cyklus](#)

[Úloha 1 - Auto](#)

[Prostředky II - For cyklus](#)

[Úloha 2 - For cykly](#)

[Shrnutí](#)

[Poznámky pro učitele](#)

Motivace

Cykly obecně umožňují značně zkrátit kód. Vše, co bychom jinak museli psát několikrát lze cyklem zopakovat třeba milionkrát nebo pracovat se vstupy a situacemi různé, předem neznámé délky. Pokud bychom programy, které jsme již vytvořili neměli obalené v nekonečném cyklu, provedla by se pouze jedna iterace a program by skončil. Cykly umožňují programátorům snadno implementovat opakující se procesy a zpracovat velké množství dat pomocí malého množství kódu.

Prostředky I – While cyklus

While cyklus využívá podmínky, čímž plynule navazuje na předchozí lekci. Má také podobnou syntax. While cyklus se používá k řešení situací, kdy nevíme přesně, kolikrát bude třeba provést určitou akci. Smyslem cyklu je opakování zanořeného bloku kódu, dokud je podmínka splněna. Není-li podmínka splněna již v první iteraci, blok kódu se vůbec nevykoná. Zápis while cyklu vypadá následovně:

```
while podmínka:  
    # blok kódu, který se opakuje, dokud je podmínka pravdivá  
    # blok kódu, kde program pokračuje, když podmínka není pravdivá
```

I u cyklů je důležité dávat pozor na správné odsazení, bez toho nebude program fungovat správně. Přestože jsme dosud pro zjednodušení používali nekonečný cyklus `while True`, od této chvíle je to něco, čemu se budeme chtít spíš vyhnout. Každý program má nějaký konečný stav, v němž chceme, aby skončil.

Úloha 1 - Auto

Zadání

Sestavte ze Nezha sady vozidlo, které bude pohánět motor a v předu bude mít distance senzor a crash senzor. Poté ho naprogramujte tak, že pojede rychlostí `fast` dokud crash senzor nebude zmačknutý. Pokud bude vozidlo překážce blíže než 40 centimetrů zpomalí na rychlosť `slow`. Auto pojede až do chvíle, než sepne crash senzor, v tu chvíli se zastaví a program skončí.

Co budete potřebovat

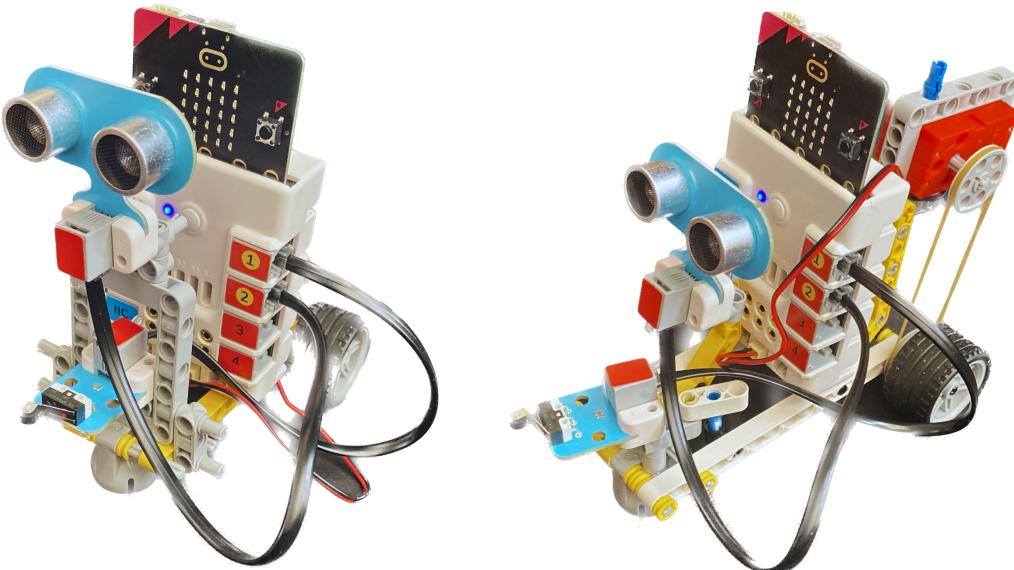
Pro tuto úlohu si připravte crash senzor a distance senzor. Oba jsou součástí Nezha sady.

Co se naučíte

Žáci si vyzkouší sestavit vlastního robota a naprogramovat ho s využitím while cyklu. Zároveň si vyzkouší, jaké je programovat ve dvojicích.

Jak postupovat

Prvním krokem v této úloze je sestavit vozítko, které bude schopné pohybu vpřed a bude mít umístěný distance tak, aby mohl snímat vzdálenost od překážek a crash senzor tak, aby to byla první součástka, která do překážky narazí. Ukažte žákům motor a jak ho zapojit. Vymezte žákům na stavbu přesný čas, jinak hrozí, že budou celou hodinu stavět, doporučujeme 30 minut.



Ukázka sestavených vozítek

V této lekci zkuste využít metodu párového programování, kdy jsou žáci rozděleni do dvojic a jeden z žáků ovládá klávesnici a myš, soustředí se na drobné kroky, které má před sebou a aktuálně neřeší větší problémy. Druhý žák je v roli pozorovatele a navigátora, kontroluje kód, v případě nejasností nebo pochybností je sdílí. Zároveň si udržuje přehled o struktuře a celistvosti kódu.

Nechte žáky ve dvojicích prozkoumat jaké metody jsou pro dané senzory dostupné a zamyslet se, jak by se mohly hodit při plnění zadání. Dohližejte na dodržování metody párového programování. V případě, že některá dvojice nebude vědět, jak postupovat dále, zkuste sestavit diagram.

Vzorová implementace

```
from microbit import *
from motor import *
from crash import *
from distance import *

motor = MOTOR(1)
```

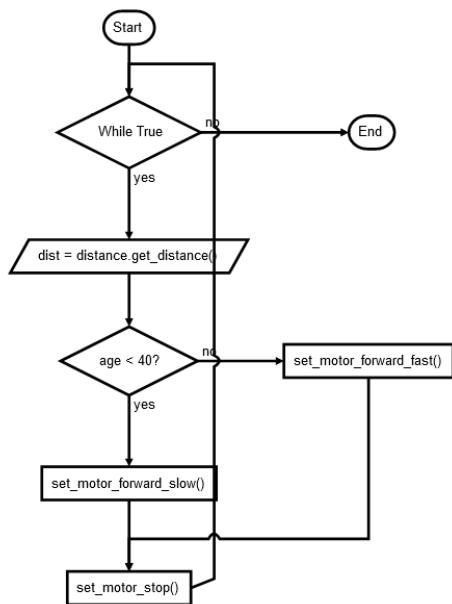
```

crash = CRASH(J1)
distance = DISTANCE(J2)

while not crash.crash_is_pressed():
    dist = distance.get_distance()
    if dist < 40:
        motor.set_motor_forward_slow()
    else:
        motor.set_motor_forward_fast()
motor.set_motor_stop()

```

Diagram



Popis řešení

Provedeme potřebné importy modulů a vytvoříme instance daných tříd. Na řádku 10 vytvoříme cyklus s podmínkou, že není stisknut crash senzor. Uvnitř cyklu měříme vzdálenost a dle její hodnoty necháme auto zrychlit nebo zpomalit.

Doplňující poznámky

Pokud vozidlo couvá, přestože je nastavena jízda dopředu, umístili jste motor obraceným směrem. Není třeba motor otáčet, stačí vozidlo nastavit na opačný směr.

Else větev program obsahuje z důvodu nepřesnosti senzoru. Někdy senzor zaznamená velmi malou vzdálenost, přestože je od překážky daleko. Else větev zajistí, že se opět rozjede rychle.

Pro opětovné spuštění programu, který je nahrán micro:bitu použijte tlačítko na jeho zadní straně.

Prostředky II – For cyklus

For cyklus je další základní typ cyklu v programování, který slouží k procházení prvků v určité sekvenci, např. v seznamu, řetězci nebo množině, a opakování určitého bloku kódu pro každý prvek v této sekvenci. Tento cyklus se používá tam, kde známe předem počet opakování, nebo je potřeba provádět operace s každým prvkem v dané sekvenci.

Syntaxe for cyklu s daným rozsahem v Pythonu vypadá následovně:

```
for prvek in range(0, 5):
    # blok kódu, který se opakuje, dokud je podmínka pravdivá
    # lze zde pracovat s promennou, která je aktuálně v promenne prvek, například:
    print(prvek) # do terminalu postupně vypíše hodnoty 0, 1, 2, 3, 4
```

Funkce `range()`, bere hodnoty typu `int`, první číslo značí ostrý počátek intervalu (tedy včetně), druhé konec intervalu (pro zadanou hodnotu se již cyklus nevykoná). Je možné zadat ještě třetí parametr typu `int`, který umožňuje definovat krok. Pokud s proměnnou označující aktuální hodnotu nepotřebujeme pracovat místo názvu dáme `_`.

Úloha 2 - For cykly

Zadání

Úloha má tři části, každá část je jeden drobný úkol, zadání proto bude očíslované 1, 2, 3.

1. Napište program, který rozsvítí na matrix modulu postupně všechny diody. Začněte na indexu nula a skončete na indexu 127.
2. Napište program, který rozsvítí na matrix modulu postupně každou třetí diodu (první bude svítit druhá a třetí ne, čtvrtá zase ano).
3. Napište program, který na matrix modulu vytvoří rozsvícením diod šachovnici.

Co budete potřebovat

Úloha je vytvořena pro maticový displej 8x16 bodů.

Co se naučíte

Žáci si na jednotlivých úlohách vyzkouší různé varianty for cyklu.

Jak postupovat

První dva úkoly jsou velmi podobné, nechte žáky se zamyslet v čem se liší a zjistit, co můžeme dát metodě `set_matrix_draw_index`. V třetím úkolu jde o zanoření dvou cyklů, pokud žáci zvládnou bez problému vyřešit první dva úkoly neměl by jim ani tento činit zásadní problémy. Vysvětlete operaci modulo, která se pro toto úlohu velmi hodí. Byť z matematiky žáci pravděpodobně neznají tento název, jedná se vlastně o zbytek po dělení. V informatice jde o velmi užitečnou a často používanou operaci. Celá část se zahodí a vrátí se pouze zbytek. Operaci modulo značíme symbolem procenta `%`. Hlídejte, aby si žáci při párovém programování ve svých dvojicích pravidelně střídali role.

Vzorová implementace

1.

```
from microbit import *
from matrix import *
```

```
matrix = MATRIX()

for i in range(0, 128):
    matrix.set_matrix_draw_index(i)
    sleep(100)
```

2.

```
from microbit import *
from matrix import *

matrix = MATRIX()

for i in range(0, 128, 3):
    matrix.set_matrix_draw_index(i)
    sleep(100)
```

3.

```
from microbit import *
from matrix import *

matrix = MATRIX()

for row in range(0, 16):
    for column in range(0, 8):
        if (row % 2 == 0) and (column % 2 == 0):
            matrix.set_matrix_draw(row, column)
        elif (row % 2 == 1) and (column % 2 == 1):
            matrix.set_matrix_draw(row, column)
```

Popis řešení

Najmírněji nainstalujeme modul pro maticový displej, stejně jako v předchozích lekcích. Následně

Doplňující poznámky

Zkuste experimentovat se složením dvojic. Je obvyklou praxí vytvářet páry nevyvážené, kdy jeden z dvojice je pokročilejší než druhý, tím se vzájemně obohatí.

Shrnutí

TODO

Poznámky pro učitele

Více informací o tom, jak funguje for cyklus si můžete přečíst v [dokumentaci Pythonu](#).

Inspiraci pro stavbu vozítka můžete čerpat například na webu [elecfra.ms](#). Je zde mnoho různých projektů, z nichž některé využívají pohybující se roboty.