Meno a priezvisko: ..... Trieda: ..... Dátum: .....

04 Cyklus s pevným počtom opakovaní

Pracovný list

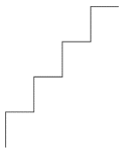
Zapojenie

Úloha 1 *Opíšte **stručne** a čo **najpresnejšie** (pre kamaráta na telefóne), čo vidíte na uvedených obrázkoch **A, B, C, D**:*

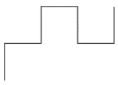
A

B

C



D



A .....

B .....

C .....

D .....

Uveďte, ktoré z obrázkov sa vám opisovali ľahšie ..... a ktoré ťažšie .....

Uveďte, čo majú spoločné vaše popisy ľahšie opísateľných obrázkov: .....

.....

Skúmanie

Úloha 2 *Preskúmajte obidva uvedené programy **A** a **B**. Najprv len na základe prečítania uvedeného programového kódu, potom po spustení ich kódov uložených v súboroch **04_02A.py** a **04_02B.py**.*

A

```
import turtle

tabula = turtle.Screen()
pero = turtle.Turtle()
pero.penup()

pero.dot(40, 'orange')
pero.forward(40)
pero.dot(40, 'orange')
pero.forward(40)
pero.dot(40, 'orange')
pero.forward(40)
pero.dot(40, 'orange')
pero.forward(40)
pero.dot(40, 'orange')
pero.forward(40)

tabula.mainloop()
```

B

```
import turtle

tabula = turtle.Screen()
pero = turtle.Turtle()
pero.penup()

for i in range(5):
    pero.dot(40, 'orange')
    pero.forward(40)

tabula.mainloop()
```

a) *Opíšte obrázky, ktoré vykreslia uvedené programy **A** a **B**:* .....

.....

b) *Ktorý zo zápisov pokladáte za lepší a prečo:* .....

c) *Ako by sa zmenil výsledok programu **B**, ak by sme v riadku 7 namiesto* `range(5)` *uviedli* `range(10)`*:*

.....

Meno a priezvisko: ..... Trieda: ..... Dátum: .....

VYSVETLENIE

Prediskutujte a vysvetlite:

1. Načo je dobrý príkaz cyklu `for`?
2. Z akých častí sa skladá zápis príkazu cyklu `for`?
3. Ako funguje príkazu cyklu `for` , aký význam majú hlavička a telo cyklu?
4. Musíme použiť v cykle `for` vlastné funkcie?

ROZPRACOVANIE

Úloha 3 *V uvedených programoch **04_03A.py** a **04_03B.py** vyznačte opakujúce sa časti a upravte tieto programy tak, aby ste pomocou príkazu `for` skrátili ich zápisy. (Poznámka: Príkaz `print` používame na výpis textu do konzoly)*

A (pôvodný program)

```
import turtle

def schod():
    pero.forward(20)
    pero.left(-90)
    pero.forward(50)
    pero.left(90)


tabula = turtle.Screen()
pero = turtle.Turtle()
pero.left(90)

schod()
schod()
schod()
schod()
schod()

tabula.mainloop()
```

B (pôvodný program)

```
def dvojriadok():
    print('XOXOXOXO')
    print('OXOXOXOX')


dvojriadok()
dvojriadok()
dvojriadok()
dvojriadok()
```

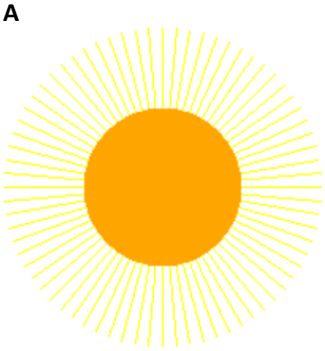
A (upravená časť programu)

B (upravená časť programu)

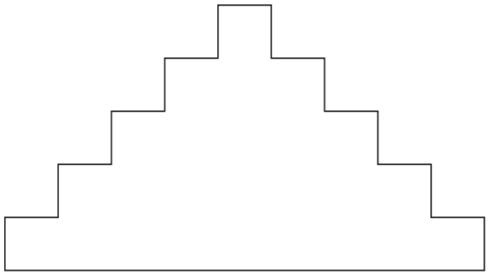
Úloha 4 *Vytvorte programy **04_04A_slnko.py** a **04_04B_schodiky.py** vykresľujúce uvedené obrázky **A** a **B** tak, aby bolo grafické pero na konci vykreslenia v počiatočnej pozícii a počiatočnom natočení.*

Riešte podľa pokynov učiteľa

A



B

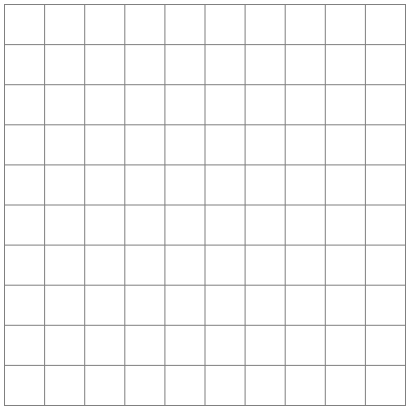


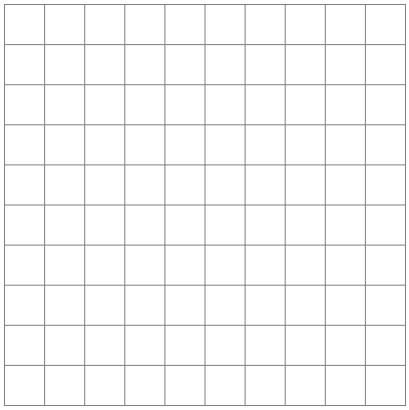
Meno a priezvisko: ..... Trieda: ..... Dátum: .....

Úloha 5

Riešte podľa pokynov učiteľa

Potrebujeme vytvoriť hrací plán pre hru Piškvorky v tvare štvorcovej mriežky 10 × 10. Do uvedených obrázkov načrtnite jeden, prípadne dva spôsoby vykreslenia tohto ***hracieho plánu****. Vo svojom návrhu vyznačte* ***počiatočný*** *a* ***koncový bod*** *vykresľovania* ***s natočením*** *grafického pera a* ***vzor****, ktorý sa pravidelne opakuje v obrázku.*





Napokon vytvorte program ***04_05_mriezka.py*** *na vykreslenie štvorcovej mriežky pomocou niektorého z navrhnutých spôsobov riešenia.*

HODNOTENIE

Sebahodnotiaci test

Úloha 6

Uveďte koľkokrát sa vykoná vlastná funkcia `prikaz()` *v programoch* ***A****,* ***B*** *a* ***C****.*

A

```
for i in range(3):
    prikaz()
    prikaz()
prikaz()
```

B

```
for i in range(3):
    prikaz()

    prikaz()
```

C

```
for i in range(3):
    prikaz()
prikaz()
prikaz()
```

Úloha 7

Upravte uvedený program ***04_07.py****, aby vykreslil obrázok časti notovej osnovy tak, aby na konci vykreslenia bolo grafické pero v* ***počiatočnej pozícii*** *a v* ***počiatočnom natočení****.* ***Dĺžka čiar*** *notovej osnovy má byť 200 bodov a* ***vzdialenosť prvej čiary od piatej*** *má byť 100 bodov.*

```
import turtle

tabula = turtle.Screen()
pero = turtle.Turtle()

for i in range(5):
    pero.pendown()
    pero.forward(200)
    pero.forward(-200)
    pero.penup()
    pero.left(90)
    pero.forward(100 / 5)
    pero.left(-90)

tabula.mainloop()
```

Meno a priezvisko: ..... Trieda: ..... Dátum: .....

VEDOMOSTI V KOCKE

*Príkaz `for` sa využíva pri riešení problémov, ktoré pozostávajú z viacerých rovnakých (alebo podobných) podproblémov. Dá sa prirovnať k maliarskemu valčeku, ktorý pri otáčaní opakovane odtláča farbu so zadaným vzorom na stenu. Príkaz `for` sa tiež označuje ako **príkaz s pevným (so známym) počtom opakovaní** alebo aj skrátene **cyklus for**.*

Poradie	Zápis kódu bez príkazu cyklu FOR	Zápis kódu s príkazom cyklu FOR
1 2 3 4	**príkazy()** príkazy() príkazy() príkazy()	`for i in range(4):` `  príkazy()`
1 2 ... počet	**príkazy()** príkazy() ... príkazy()	`for i in range(počet):` `  príkazy()`

1
2
3
4
5
6
7
	a()
	b()
	a()
	b()
	a()
	b()
	c()
		`for i in range(3):`
		`  a()`
		`  b()`
		`c()`

1
2
3
4
5
	a()
	a()
	a()
	b()
	c()
		`for i in range(3):`
		`  a()`
		`b()`
		`c()`

*Cyklus `for` pozostáva z **hlavičky cyklu** (prvého riadku začínajúcim slovom `for` a končiacim dvojbodkou, na ktorú často začiatočníci zabúdajú) a **tela cyklu** (riadkami s odsadenými príkazmi). Príkazy v tele cyklu sa vykonajú toľkokrát, ako je to uvedené v parametri funkcie `range()` v hlavičke cyklu. Telo cyklu sa neukončí zaradením voľného riadku, ale zrušením odsadenia príkazov v tele cyklu vzhľadom k hlavičke cyklu.*

*Pri kreslení obrázkov pomocou príkazu cyklu `for` je dôležité, aby v tele cyklu boli uvedené príkazy, po vykonaní ktorých sa dostane grafické pero **do určitej významnej pozície**, napr. počiatočný bod celého kreslenia, alebo počiatočný bod kreslenia nasledovnej časti obrázka.*

*Cyklus `for` aj vlastná funkcia umožňujú **skrátiť** a **sprehľadniť** programový kód. Pomocou vlastných funkcií sme riešenie problému mohli rozdeliť na riešenie rôznych podproblémov, napr. kreslenie obrázkov s rôznymi vzormi alebo s rovnakými vzormi aj na nepravidelných pozíciách. Pomocou cyklu `for` rozdelíme problém do rovnakých alebo podľa určitého pravidla podobných podproblémov, napr. kreslenie pravidelných obrázkov.*