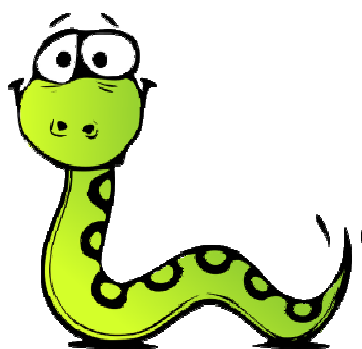




Programování v jazyce Python pro střední školy

Lekce 19

Podprogram s parametrem



Andrej Blaho

Ľubomír Salanci

Václav Šimandl

1. Vytvoř nový program `cifry_cisla.py`, ve kterém přiřadíš do proměnné `cislo` číslo od 0 do 999. Použij větvení na to, aby program rozhodl a správně vypsal hlášení o tom, zda je číslo jedno-, dvou- nebo trojciferné. Například pro `cislo = 128` program vypíše:

Číslo 128 je trojciferné.

2. Vytvoř nový program `muj_vek.py`. Přepiš do něj následující kód a dokonči jednotlivé podprogramy, aby vypisovaly správný věk:

```
def jemi10():
    vek = 10
    print('Je mi', vek, 'let')

def jemi20():
    vek = .....
    print('Je mi', vek, 'let')

def jemi30():
    vek = 30
    print(.....)

jemi10()
jemi20()
jemi30()
```

Udělej to tak, aby se všechny tři podprogramy navzájem co nejvíc podobaly.

3. Předchozí řešení se dá zapsat pomocí jediného podprogramu:

```
def jemi(vek):
    print('Je mi', vek, 'let')

jemi(10)
jemi(20)
jemi(30)
```

Vyzkoušej jej.

Jak program funguje?

v závorce je **název parametru**
 ↓
 zde se parametr **používá** – parametr funguje jako proměnná

```
def jemi(vek):
    print('Je mi', vek, 'let')
```

jemi(10)
 jemi(20)
 jemi(30)

↓
hodnota, která se přiřadí do parametru **vek**

4. Vytvoř nový program `druha_mocnina_parametr.py`. Přepiš do něj následující kód a dokonči podprogram `vypis`, který používá parametr `x` na to, aby vypsál hodnotu parametru `x` a jeho druhou mocninu:

```
def vypis(x):  
    print('Číslo', ...)  
    print('Umocněné na druhou se rovná', .....)  
  
vypis(1)  
vypis(2)  
vypis(3)
```

Program by měl po spuštění vypsát:

```
Číslo 1  
Umocněné na druhou se rovná 1  
Číslo 2  
Umocněné na druhou se rovná 4  
Číslo 3  
Umocněné na druhou se rovná 9
```

5. Doplně do předchozího podprogramu příkaz, kterým se vypíše i převrácená hodnota `x`.
Připomeňme, že převrácená hodnota čísla x je rovna $\frac{1}{x}$. Program by měl po spuštění vypsát:

```
Číslo 1  
Umocněné na druhou se rovná 1  
Převrácená hodnota se rovná 1.0  
Číslo 2  
Umocněné na druhou se rovná 4  
Převrácená hodnota se rovná 0.5  
Číslo 3  
Umocněné na druhou se rovná 9  
Převrácená hodnota se rovná 0.3333333333333333
```

6. Vytvoř nový program `kruh_parametr.py`. Přepiš do něj následující kód a dokonči podprogram `kruh` tak, aby kreslil kruhy se středem 200, 150 a poloměrem `r`, který bude parametrem podprogramu:

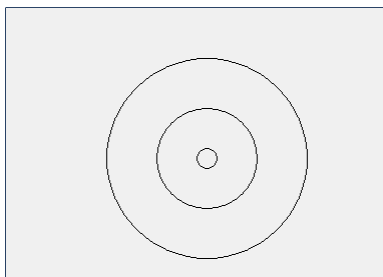
```
import tkinter

canvas = tkinter.Canvas()
canvas.pack()

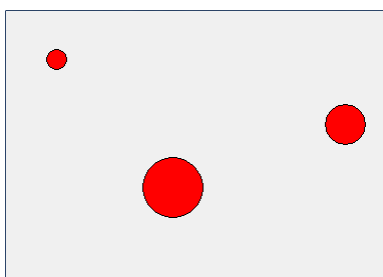
def kruh(r):
    canvas.create_oval(....., ....., ....., .....)

kruh(10)
kruh(100)
kruh(50)
```

Jestli jsi postupoval správně, program by měl nakreslit takovýto obrázek:



7. Vytvoř nový program `nahodny_kruh_parametr.py` a v něm vytvoř podprogram `nahodny_kruh` s parametrem `r`, který nakreslí na náhodných souřadnicích červený kruh o poloměru `r`. Zavolej tento podprogram pro různé hodnoty parametru. Výsledek může vypadat například jako na následujícím obrázku:



8. Vyzkoušej, co předchozí program nakreslí, když zavoláš podprogram `nahodny_kruh` následujícím způsobem:

```
for i in range(10):
    nahodny_kruh(i)
```

Diskutuj se svým spolužákem, jak program funguje.

9. Poloměr kruhu můžeme určit i takto:

```
for i in range(10):
    nahodny_kruh(i + 5)
```

Spusť program, abys viděl, co udělá, a vyplň následující tabulku:

hodnota v proměnné <code>i</code>	hodnota parametru <code>r</code>
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	

10. Vytvoř nový program `oblibene_cislo.py` a v něm definuj podprogram `obliba` s parametrem `cislo`. Podprogram podle následujících pravidel vypíše, zda má číslo v oblíbě:

- když je číslo menší než 7, vypíše `Mám rád číslo ...`
- jinak vypíše `Číslo ... se mi nelíbí`

Podprogram zavolej z příkazového řádku a ověř, že vypíše:

```
>>> obliba(1)
Mám rád číslo 1
>>> obliba(5)
Mám rád číslo 5
>>> obliba(10)
Číslo 10 se mi nelíbí
```

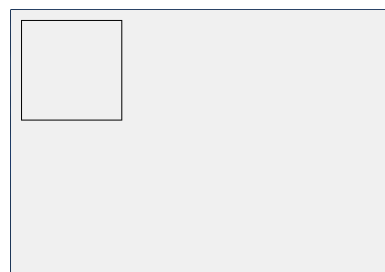
11. Uprav předchozí program tak, abys pomocí cyklu zavolal podprogram `obliba` pro čísla od 0 do 10. Výsledek by měl vypadat následovně:

```
Mám rád číslo 0
Mám rád číslo 1
...
Mám rád číslo 6
Číslo 7 se mi nelíbí
...
Číslo 10 se mi nelíbí
```

- 12* Vytvoř nový program `ctverec_parametr.py` a v něm definuj podprogram `ctverec` s parametrem `a`, který udává délku strany čtverce. Podprogram by měl fungovat tak, že čtverec kreslí jen pro kladné hodnoty parametru `a`, ale pro záporné hodnoty vypíše zprávu „Nedá se“. Levý horní roh kresleného čtverce bude mít souřadnice `[10, 10]`. Zprávu vypiš přibližně do středu grafické plochy. Otestuj, že podprogram pracuje správně.



ctverec(-50)



ctverec(100)

- 13* Uprav podprogram `ctverec` z předchozí úlohy tak, aby se pro záporné hodnoty parametru `a` nikde nic nevypsalo. Stačí, když smažeš větev `else: i` s příkazem pro výpis. Takovýto příkaz se nazývá `if` bez větve `else`:

```
if podmínka:
    příkaz
    příkaz
    ...
```

} větev if

14. Znáš hru *Myslím si číslo*, ve které je potřeba uhádnout neznámé číslo? Vytvoř takovou hru na počítači – počítač si vymyslí číslo od 1 do 5 a my ho musíme uhádnout. Vytvoř nový program `uhadni_cislo.py`, který bude fungovat následujícím způsobem:

- po spuštění programu počítač přiřadí do proměnné `cislo` náhodně vygenerované číslo,
- potom vypíše zprávu „Myslím si číslo od 1 do 5. Zkus ho uhádnout...“
- ve svém programu budeš mít definovaný podprogram `zkus` s parametrem `n`, který porovná `cislo` s hodnotou `n` a vypíše: buď „Hurá, uhádl jsi!“, nebo „Ne, moje číslo je jiné...“.

Hra může probíhat následovně – spustíme program a v příkazovém řádku odpovídáme tím, že voláme podprogram `zkus`:

```
===== RESTART =====
Myslím si číslo od 1 do 5. Zkus ho uhádnout...
>>> zkus(3)
Ne, moje číslo je jiné...
>>> zkus(2)
Ne, moje číslo je jiné...
>>> zkus(5)
Hurá, uhádl jsi!
>>>
```

15* Vylepši předchozí program tak, aby nám podprogram `zkus` poradil, zda je hádané číslo větší nebo menší, než jsme tipnuli:

```
===== RESTART =====
Myslím si číslo od 1 do 5. Zkus ho uhádnout...
>>> zkus(3)
Ne, moje číslo je menší...
>>> zkus(1)
Ne, moje číslo je větší...
>>> zkus(2)
Hurá, uhádl jsi!
>>>
```

16. Vytvoř nový program `kviz.py`, který bude fungovat jako jednoduchý kvíz na sčítání čísel. Počítač na začátku vygeneruje dvě náhodná čísla z rozsahu od 1 do 10, vypíše je a my musíme odpovědět tím, že z příkazového řádku zavoláme podprogram `over`. Počítač poté zkontroluje, zda byla naše odpověď správná, nebo ne:

```
===== RESTART =====
Kolik je 10 + 7 ?
>>> over(5)
Nesprávně, mělo to být 17
>>>
===== RESTART =====
Kolik je 4 + 9 ?
>>> over(13)
Správně
>>>
```

17* Je potřeba prozkoumat, jak často padne 6, když mnohokrát házíme hrací kostkou. Vytvoř nový program `stesti.py` a v něm podprogram `stesti` s parametrem `n`, který nasimuluje `n` hodů běžnou hrací kostkou. Podprogram `n`-krát vygeneruje náhodné číslo od 1 do 6, a když padne šestka, zvýší počítadlo o 1. Podprogram na závěr vypíše zprávu ve tvaru:

```
Pravděpodobnost výhry při 10 hodech: 0.2
```

Nech podprogram vypsát, jaké budou pravděpodobnosti výhry při 10, 100, 1000, 10000, 100000 hodech.

18* Dlaždič měl rovnoměrně poskládat dlažební kostky do jedné řady. Měl však dobrou náladu a mezi kostkami nechával náhodné mezery. Vytvoř nový program `dlazebni_kostky.py` a v něm podprogram `rada` s parametrem `y`, který nakreslí do grafické plochy vedle sebe 20 kostek. Kostky kreslí jako šedé čtverečky velikosti 10 x 10. První kostka má x-ovou souřadnici levého horního rohu 10 a každá další ji má větší o náhodné číslo z rozsahu od 12 do 20. Y-ovou souřadnici levého horního rohu mají všechny kostky stejnou; tato souřadnice je dána parametrem `y`.

Zavolej podprogram `rada` následujícím způsobem:

```
for i in range(15):  
    rada(5 + i * 15)
```

Jestli jsi postupoval správně, výsledek by měl vypadat podobně jako na následujícím obrázku:

