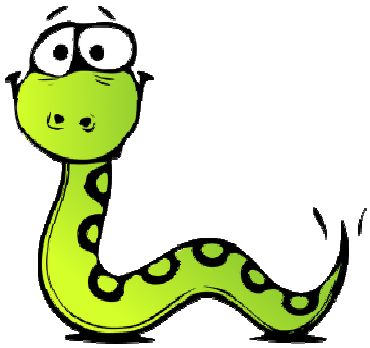
Programování v jazyce Python pro střední školy

Metodický list pro učitele

Lekce 15 – Kruhy a cykly



Andrej Blaho

Ľubomír Salanci

Václav Šimandl

Cíle lekce

- Trénovat určování a odvozování výrazů pro vzájemnou polohu kruhů a obdélníků
- Naučit se používat náhodný výběr prvků z určité množiny (příkaz `random.choice`)

Dovednosti

- Kreslení náčrtů na papír při odvozování vzorců se vzájemnou polohou kruhů

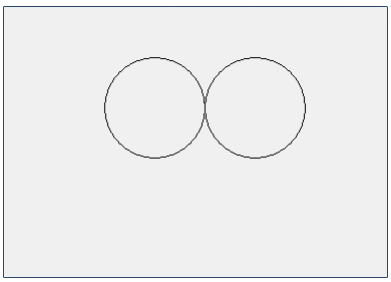
Osvojená syntaktická pravidla

- Zápis příkazu `random.choice` a jeho parametrů

Průběh výuky

Začínáme úlohou na opakování:

1. Vytvoř program `dve_kruznice.py`, který nakreslí dvě kružnice jako na obrázku níže. Do proměnných `x`, `y` přiřaď souřadnice bodu, ve kterém se kružnice dotýkají (například v bodě `[200, 100]`). Kružnice budou umístěné vedle sebe a jejich poloměr bude 50. Při kreslení kružnic používej proměnné `x`, `y` tak, aby bylo možné změnou jejich hodnot obě kružnice přemístit.

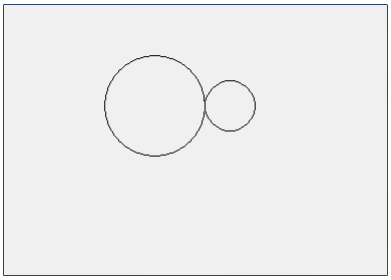


Řešení:

```
import tkinter

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

x = 200
y = 100
canvas.create_oval(x - 100, y - 50, x, y + 50)
canvas.create_oval(x, y - 50, x + 100, y + 50)
```

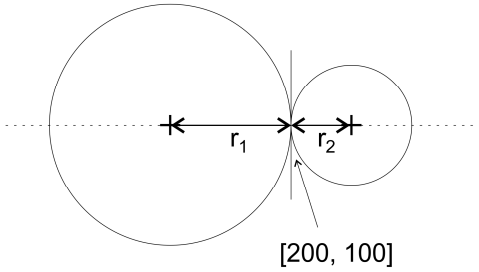
V případě, že žák vyřeší úlohu bez použití proměnných `x`, `y` jen s konstantami, lze mu například říci: „změň program tak, aby se kružnice dotýkaly v bodě `[243, 182]`“. Místo jednoduché změny hodnot proměnných `x` a `y` bude muset žák přepočítat všechny potřebné souřadnice. Žák by si tak měl znovu uvědomit, že je výhodnější použít vhodné proměnné (v tomto případě `x` a `y`) než měnit kód programu na několika různých místech.

2. Uprav předchozí program tak, že poloměry kružnic nejprve přiřadíš do proměnných `r1`, `r2`. Například pro `r1 = 50`, `r2 = 25` bude obrázek vypadat takto:



Bude program fungovat správně i v případě, že hodnotu proměnné `r1` zmenšíš o 10 a hodnotu proměnné `r2` zvětšíš o 5? Jestli ne, program oprav.

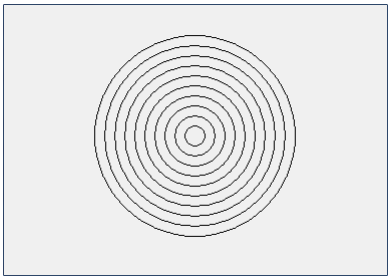
Některým žákům bude možná potřeba pomoci s výpočty souřadnic kružnic. V tom případě je vhodné nakreslit na tabuli obrázek, označit v něm poloměry kružnic, z nich odvodit pozice středů a následně hodnoty parametrů.



Řešení:

```
import tkinter

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

x = 200
y = 100
r1 = 50
r2 = 25
canvas.create_oval(x - 2 * r1, y - r1, x, y + r1)
canvas.create_oval(x, y - r2, x + 2 * r2, y + r2)
```

3. Napiš program `terc.py`, který pomocí cyklu a deseti soustředných kružnic nakreslí terč jako na obrázku níže. Nejmenší kružnice bude mít poloměr 10 a každá další bude mít poloměr o 10 větší než předchozí:



Řešení založené na postupném zvyšování hodnoty proměnné, která reprezentuje poloměr, v cyklu:

```
import tkinter

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

r = 10
for i in range(10):
    canvas.create_oval(190 - r, 130 - r, 190 + r, 130 + r)
    r = r + 10
```

Řešení založené na odvození poloměru z proměnné cyklu:

```
import tkinter

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

for i in range(10):
    r = i * 10 + 10
    canvas.create_oval(190 - r, 130 - r, 190 + r, 130 + r)
```

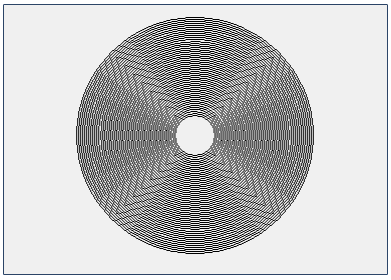
Někteří žáci mohou vymyslet řešení, ve kterém nepoužijí proměnnou `r` a které může vypadat například následovně (uvádíme pouze část kódu obsahující `for` cyklus):

```
for i in range(10):
    canvas.create_oval(190 - 10 * (i + 1), 130 - 10 * (i + 1),
        190 + 10 * (i + 1), 130 + 10 * (i + 1))
```

Ačkoliv je takové řešení také správné, je poměrně nepřehledné. Proto je vhodnější žáky vést k využívání pomocných proměnných, díky nimž je kód programu přehlednější a čitelnější.

V následující úloze očekáváme samostatné experimentování žáků:

4. Vytvoř nový program `gramofon.py` a zkopíruj si do něj kód z programu `terc.py`. Uprav v programu `gramofon.py` některé číselné hodnoty tak, aby se nakreslila gramofonová deska:



Řešení:

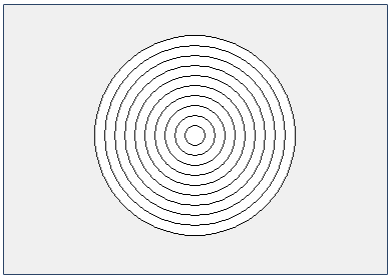
```
import tkinter

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

for i in range(50):
    r = i * 2 + 20
    canvas.create_oval(190 - r, 130 - r, 190 + r, 130 + r)
```

Námi uváděné hodnoty `50, 2, 20` mohou být v žákovských řešeních i jiné, přibližné. Je vhodné, aby žáci při řešení úlohy experimentovali, na základě čehož naleznou vhodné hodnoty. Úloha může být žáky vypracována s různou pečlivostí a je na našem uvážení, která řešení akceptujeme jako dostatečně kvalitní a u kterých budeme po žácích požadovat jejich úpravu.

Někteří žáci mohou místo odvozování vzorce pro výpočet poloměru od proměnné cyklu vymyslet řešení založené na postupném zvyšování hodnoty poloměru.

V následující úloze musí žáci přijít na to, že kruhy je potřeba kreslit od největšího po nejmenší:

5. Vrať se k programu `terc.py` a uprav kreslení kruhů tak, aby byl každý z nich vyplněný bílou barvou (tj. s parametrem `fill='white'`). Výsledek by měl vypadat podobně jako na obrázku níže:



Řešení:

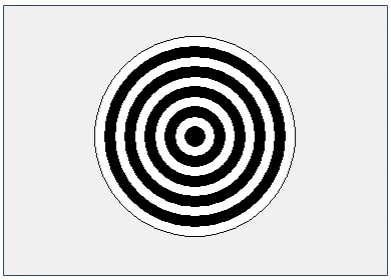
```
import tkinter

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

r = 100
for i in range(10):
    canvas.create_oval(190 - r, 130 - r, 190 + r, 130 + r,
        fill='white')
    r = r - 10
```

Ani v této úloze nemusíme poloměr vypočítávat postupným snižováním hodnoty dané proměnné v cyklu, ale můžeme jej odvodit přímo z proměnné cyklu:

```
r = 100 - 10 * i
```

Následují úlohy na trénování:

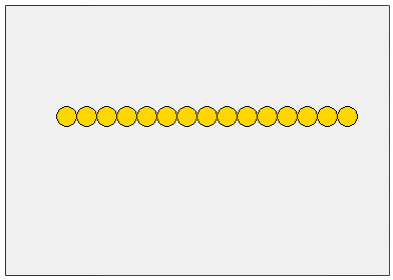
6. Uprav kreslení terče tak, aby se střídaly černé a bílé oblasti jako na obrázku níže. V cyklu se kreslí vždy dva kruhy – větší bílý a menší černý.



Řešení:

```
import tkinter

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

r = 100
for i in range(5):
    canvas.create_oval(190 - r, 130 - r, 190 + r, 130 + r,
        fill='white')
    r = r - 10
    canvas.create_oval(190 - r, 130 - r, 190 + r, 130 + r,
        fill='black')
    r = r - 10
```

7. Napiš program `retizek.py`, který pomocí cyklu nakreslí řetízek z 15 zlatých kroužků:



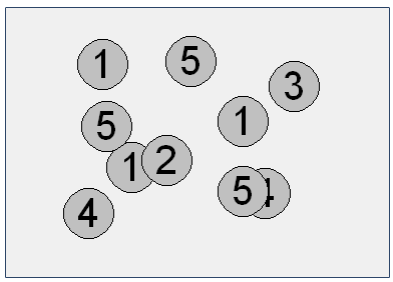
Řešení:

```
import tkinter

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

x = 50
for i in range(15):
    canvas.create_oval(x, 100, x + 20, 120, fill='gold')
    x = x + 20
```

Úlohu lze řešit i na základě odvozování proměnné `x` od proměnné cyklu:

```
import tkinter

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

for i in range(15):
    x = 50 + i * 20
    canvas.create_oval(x, 100, x + 20, 120, fill='gold')
```

8. Vytvoř nový program `mince.py` a v něm vytvoř podprogram `mince`. Podprogram bude generovat náhodnou pozici a náhodnou hodnotu mince od 1 do 5. Minci nakresli jako kruh s číslem (viz následující obrázek).



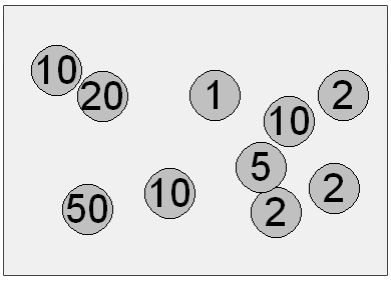
Podprogram `mince` zavolej pomocí cyklu desetkrát.

Řešení:

```
import tkinter
import random

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

def mince():
    x = random.randint(50, 340)
    y = random.randint(50, 210)
    h = random.randint(1, 5)
    canvas.create_oval(x - 25, y - 25, x + 25, y + 25,
        fill='silver')
    canvas.create_text(x, y, text=h, font='arial 30')

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

V následující úloze se žáci poprvé setkají s použitím příkazu `random.choice`:

9. Uprav svůj program tak, aby se generovaly jen mince s hodnotami `1, 2, 5, 10, 20, 50`.



Pro generování hodnot mincí použij místo `random.randint(1, 5)` zápis:

```
random.choice([1, 2, 5, 10, 20, 50])
```

Zápis `random.choice` čteme jako: náhodný výběr z vyjmenovaných hodnot.

Řešení:

```
import tkinter
import random

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

def mince():
    x = random.randint(50, 340)
    y = random.randint(50, 210)
    h = random.choice([1, 2, 5, 10, 20, 50])
    canvas.create_oval(x - 25, y - 25, x + 25, y + 25,
        fill='silver')
    canvas.create_text(x, y, text=h, font='arial 30')

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

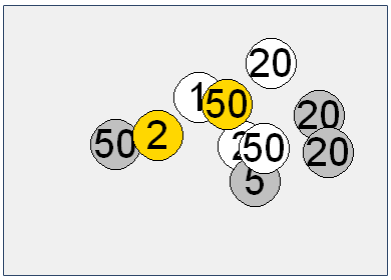
Zápis `random.choice([1, 2, 5, 10, 20, 50])` náhodně zvolí jednu z možností, které jsou uvedeny v hranatých závorkách `[ ... ]`. Samotný zápis `[1, 2, 5, 10, 20, 50]` znamená seznam prvků, které jsou v tomto zápisu odděleny čárkou. Fungují i následující varianty zápisu, které však žáky neučíme:

- `random.choice((1, 2, 5, 10, 20, 50))`
- `random.choice({1, 2, 5, 10, 20, 50})`

Někteří žáci se mohou ptát, proč jsou hranaté závorky zapsány uvnitř kulatých závorek, když jsou z matematiky zvyklí na zcela opačný zápis (tedy že při zápisu výrazů jsou vnitřní kulaté závorky a vnější závorky jsou hranaté). Můžeme jim vysvětlit, že v Pythonu má každý typ závorek svůj konkrétní význam, který je třeba dodržet.
Pokročilejším žákům můžeme prozradit, kulaté závorky se patří k příkazu `random.choice` (stejně jako se píší kulaté závorky za příkaz `random.randint`) a že hranaté závorky uvozují seznam prvků, ze kterých se náhodná hodnota vybírá.

Následují úlohy na experimentování s výběrem náhodné hodnoty:

10* Zápis `random.choice` můžeš použít i na výběr barvy. Uprav předchozí program tak, že do proměnné `barva` přiřadíš `random.choice(['silver', 'gold', 'white'])` a tuto proměnnou použiješ při kreslení oválu v parametru `fill=barva`.



Řešení:

```
import tkinter
import random

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

def mince():
    x = random.randint(50, 340)
    y = random.randint(50, 210)
    h = random.choice([1, 2, 5, 10, 20, 50])
    barva=random.choice(['silver', 'gold', 'white'])
    canvas.create_oval(x - 25, y - 25, x + 25, y + 25,
        fill=barva)
    canvas.create_text(x, y, text=h, font='arial 30')

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

V této úloze se žáci poprvé setkávají s tím, že jako hodnotu parametru `fill=` nezapisují název konkrétní barvy, ale uvádí zde název proměnné, která obsahuje název určité barvy. Zatímco název konkrétní barvy se zapisuje do apostrofů (např. `fill='red'`), název proměnné je nutné uvádět bez apostrofů (např. `fill=barva`).
Pokud by název proměnné některý žák uvedl v apostrofech (např. `fill='barva'`), vykonávání programu skončí chybou `tkinter.TclError: unknown color name "barva"`. Python tímto chybovým hlášením oznamuje, že nebyl zadán název žádné známé barvy.

11. Vyzkoušej, jako funguje `random.choice` – každý z příkazů nech pomocí cyklu vykonat několikrát:
   a) `print(random.choice(['Ahoj', 'Nazdar', 'Servus', 'Čau']))`
   b) `print(random.choice('POMERANČ'))`
   c) `print(random.choice([1 / 2, 1 / 3, 1 / 4, 1 / 5]))`

Řešení – pozdravy:

```
import random

for i in range(10):
    print(random.choice(['Ahoj', 'Nazdar', 'Servus', 'Čau']))
```

Ostatní podúlohy se řeší obdobně.

Pro `random.choice(['Ahoj', 'Nazdar', 'Servus', 'Čau'])` program vypíše náhodně zvolené pozdravy, například:

```
Ahoj
Ahoj
Servus
Nazdar
Servus
Ahoj
Nazdar
Nazdar
Čau
Ahoj
```

Speciálním případem je příkaz `random.choice('POMERANČ')`, v němž na rozdíl od ostatních podúloh nejsou použity hranaté závorky. Program v tomto případě vypíše náhodně zvolená písmena z řetězce `'POMERANČ'`, například:

```
O
M
Č
R
Č
E
O
R
E
A
```

Python v tomto případě chápe řetězec `'POMERANČ'` jako seznam znaků, a proto se zde hranaté závorky neuvádí. Kdybychom je použili, vznikl by příkaz `random.choice(['POMERANČ'])`. Tehdy by Python vybíral náhodný prvek ze seznamu obsahujícího jediný prvek – řetězec `'POMERANČ'` a pokaždé by tedy vypsal slovo `POMERANČ`. Takovéto podrobnosti však žákům neprozrazujeme.

Pro `random.choice([1 / 2, 1 / 3, 1 / 4, 1 / 5])` program vypíše náhodně zvolená čísla, například:

```
0.25
0.5
0.3333333333333333
0.2
0.2
0.2
0.2
0.3333333333333333
0.5
0.25
```

12. Napiš program `pocasi.py`, který zobrazuje zprávy ve tvaru:

```
Dnes je ... den
```

Místo ... se vypíše jedna z možností `'pěkný'`, `'ošklivý'`, `'deštivý'`, `'slunečný'`.

Očekávané řešení:

```
import random

pocasi = random.choice(['pěkný','ošklivý','deštivý',
  'slunečný'])
print('Dnes je', pocasi, 'den')
```

Možné řešení bez použití proměnné:

```
import random

print('Dnes je', random.choice(['pěkný','ošklivý','deštivý',
  'slunečný']), 'den')
```

Varianta řešení s náhodným výběrem celých vět:

```
import random

print(random.choice([
    'Dnes je pěkný den', 'Dnes je ošklivý den',
    'Dnes je deštivý den', 'Dnes je slunečný den']))
```