

# IB111

## Základy programování

František Lachman [lachmanfrantisek@mail.muni.cz](mailto:lachmanfrantisek@mail.muni.cz)

cvičení 1

17. září 2018

# Osnova

- organizace cvičení
- studijní materiály
- seznámení s vývojovým prostředím
- základy Pythonu
- želví grafika

# Organizace cvičení

- cvičení jsou povinná
  - na začátku zapisována účast
  - povolené maximálně 2 neomluvené absence
- přednášky nepovinné, **ale**:
  - předpokládá se znalost ([kontrolní otázky](#) na začátku cvičení)
  - videa nemusí být spolehlivá/včas

# Kontrolní otázky

- Jaké jsou podmínky zakončení předmětu?
- Je nezbytné vyřešit všechny domácí úlohy?
- Proč je dobré umět programovat?
- Co je to algoritmus?
- Je důležité rozlišovat Python verze 2 a verze 3?  
Kterou verzi používáme v tomto kurzu?

# Přihlášení na počítače

- fakultní login ([seznam](#))
- fakultní heslo (defaultně sekundární)
  - nastavitelné [zde](#)
- vlastní notebook je možný  
(nainstalovat [Python 3.x](#) a IDLE)

# Hodnocení předmětu - vnitra

<https://www.fi.muni.cz/IB111/?p=hodnoceni>

- “ • Dvě vnitrosemestrální zkoušky u počítače (140 bodů): programování v Pythonu v rámci dvou cvičení (40+100 bodů), zadané úlohy jsou variace na příklady ze cvičení, je povoleno používat podklady.
- Alespoň 70 bodů za vnitrosemestrální zkoušky. „

# Hodnocení předmětu - úkoly

<https://www.fi.muni.cz/IB111/?p=hodnoceni>

- “
- Domácí úkoly (160 bodů): celkem 5 úloh v průběhu semestru (25, 25, 30, 40, 40 bodů). odevzdání všech 5 domácích úloh (alespoň 5 bodů za každou).
  - Alespoň 100 bodů za domácí úlohy.
- ”

# Studijní materiály

- Studijní materiály pro skupinu:
  - [https://is.muni.cz/auth/el/1433/podzim2018/IB111/um/skupina\\_03\\_lachman/](https://is.muni.cz/auth/el/1433/podzim2018/IB111/um/skupina_03_lachman/)
  - <https://gitlab.fi.muni.cz/xlachma1/ib111/>
- Harmonogram cvičení:  
<https://www.fi.muni.cz/IB111/?p=cviceni>
- Sbírka příkladů:  
<https://www.fi.muni.cz/IB111/sbirka/>
- Další zdroje:  
<https://www.fi.muni.cz/IB111/?p=zdroje>



**Základy programování ukázané na Pythonu**



~~**Základy programování v Pythonu**~~

# Python

- Používáme Python 3.x (nekompatibilní s 2.7)
- Test:

```
Python 2.7.15 (default, May 16 2018, 17:50:09)
[GCC 8.1.1 20180502 (Red Hat 8.1.1-1)] on linux2
Type "help", "copyright", "credits" or "license" for more
>>> 2/3
0
```

```
Python 3.6.6 (default, Jul 19 2018, 14:25:17)
[GCC 8.1.1 20180712 (Red Hat 8.1.1-5)] on linux
Type "help", "copyright", "credits" or "license" for more
>>> 2/3
0.6666666666666666
```

# IDLE

- jednoduché vývojové prostředí pro Python
- shell
  - restart shellu (`CTRL+F6`)
  - ukončení dlouhého/nekonečného běhu (`CTRL+C`)
- editor souborů (`CTRL+N`; spuštění pomocí `F5`)
  - nepojmenovávat soubory stejně jako názvy modulů (např. `turtle.py`) => chyby v importech

# Formátování

- bloky kódu odsazujeme mezerami (4)  
(editory mají většinou nastavené nahrazení tabulátoru patřičným množstvím mezer)

```
def hello():  
    for i in range(5):  
        print("Hello!")  
    print("Bye!")
```

# Dokumentace

- pojmenování proměnných je anglicky a smysluplné (ne jednopísmenné a matoucí názvy)
- kód je rozumně komentovaný

```
# comment
print("Hello") # comment

def hello(person):
    """
    Function description.

    :param person: description of parameter
    """
    print("Hello", person)
```

# Základní operace

```
print(2 + 3)           # scitani
print(2 * 5)          # nasobeni
print(10 / 3)         # desetinne deleni
print(10 // 3)       # celociselne deleni
print(10 % 3)        # modulo
print((5 - 3) * (7 - 4)) # uzavorkovani
```

```
5
10
3.333333333333333
3
1
6
```

# Proměnné

```
x = 2           # do promenne x prirad cislo 2  
y = 5           # do promenne y prirad cislo 5  
print(x + y)    # spocitej a vypis soucet hodnot promennych  
print(type(x))
```

```
7  
<class 'int'>
```

# Opakování

(podrobněji později)

```
for i in range(5):  
    print('Hello!', i)
```

```
Hello! 0  
Hello! 1  
Hello! 2  
Hello! 3  
Hello! 4
```



# Funkce

```
# definice funkce pro vypis obvodu ctverce,  
# jehoz delka strany je 'side'  
def print_square_perimeter(side):  
    perimeter = 4 * side  
    print(perimeter)  
  
# volani funkce pro delky strany 50 a 100  
print_square_perimeter(50)  
  
# urceni parametru pres jmeno  
print_square_perimeter(side=100)
```

```
200
```

```
400
```

# Knihovny

```
# z knihovny math importujeme funkci sqrt (odmocnina) a konstantu pi  
from math import sqrt, pi  
  
print(sqrt(4))  
print(pi)
```

Případně import modulu:

```
# importujeme modul math  
import math  
  
# k členům se odkazujeme pomocí "tečkove notace"  
print(math.sqrt(4))  
print(math.pi)
```

# Dobré praktiky

- Psát čitelný, přehledný kód.
- Používání `CTRL+C` / `CTRL+V` zavání špatným návrhem.
- Nevytvářet "dlouhé" metody.
- Podrobněji bude zmíněno na některé z přednášek.

# Želví grafika

```
# import tridy Turtle pro zelvi grafiku  
from turtle import Turtle, done
```

```
# vytvoreni nove zelvy jmenem julie  
julie = Turtle()
```

```
# kresleni  
julie.forward(100) # popojdi dopredu o 100 px  
julie.left(90) # zatoc doleva o 90 stupnu  
julie.forward(50) # popojdi dopredu o 50 px  
julie.right(135) # zatoc doprava o 135 stupnu  
julie.forward(25) # popojdi dopredu o 25 px  
  
# ukonci kresleni, lze zavrit okno  
done()
```

# Želví grafika - doplňující metody

```
speed(10)           # urychli vykreslovani
tracer(False)      # vypne postupne vykreslovani
clear()            # vymaze kreslici plochu
reset()            # vymaze plochu a vrati zelvu doprostred
bye()              # zavre okno
exitonclick()     # zavre okno pri kliknuti na nej
done()            # ukonci vykreslovani a okno staci zavrit
```

Dokumentace:

<https://docs.python.org/3.6/library/turtle.html>

# Příklady:

Předlohy obrázků:

[cviko01-zelvi-grafika.pdf](#)

# Příklady (1):

- [sbírka 1.1 Rozcvička](#)

## 1.1.2. Čtverec:

Využijte for cyklus pro nakreslení čtverce o délce strany 100 pixelů.

```
from turtle import Turtle
julie = Turtle()

for i in range(4):
    julie.forward(100)
    julie.left(90)
```



## 1.1.3. Obecný čtverec

Napište funkci pro vykreslení čtverce s danou délkou strany.

```
from turtle import Turtle

clearscreen()
julie = Turtle()

def square(side):
    for i in range(4):
        julie.forward(side)
        julie.left(90)

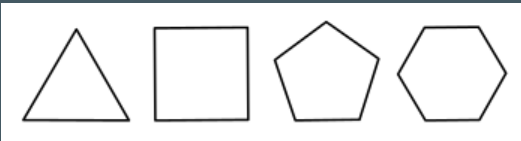
square(100)
```

# Příklady (2):

- [1.2. Pokročilé kreslení](#)

## 1.2.1. Mnohoúhelníky

Napište obecnou funkci pro vykreslení libovolného pravidelného n-úhelníku.



```
from turtle import Turtle, clearscreen
clearscreen()
julie = Turtle()
def polygon(n, side):
    angle = 360.0 / n
    for i in range(n):
        julie.forward(side)
        julie.left(angle)
```

```
polygon(3, 80)
```

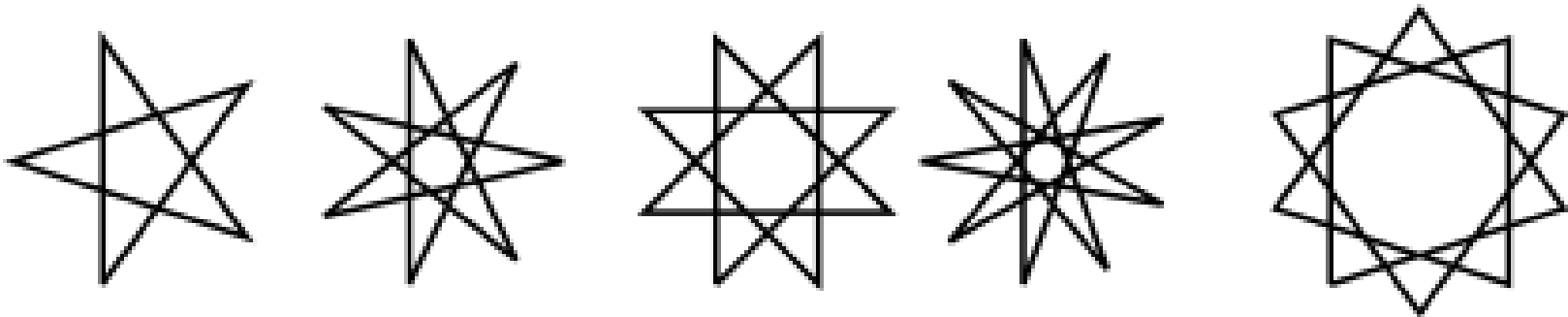
# Příklady (3):

- [1.2. Pokročilé kreslení](#)

## 1.2.2. Hvězdy

Napište obecnou funkci pro vykreslení hvězdy. Hvězda je zobecněním pravidelného  $n$ -úhelníka, kde želva nenavštívuje bezprostředně sousední vrcholy, ale “přeskakuje”. Délka skoku je daná parametrem  $step$ , ten je např. pro první, pěticípou hvězdu roven 2 a pro druhou, sedmicípou hvězdu roven 3. Při  $step = 1$  půjde o  $n$ -úhelník.

...



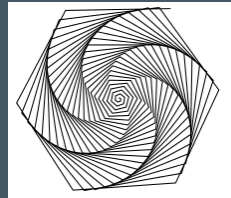
```
from turtle import Turtle, clearscreen  
clearscreen()  
julie = Turtle()
```

```
def star(n, step, side):  
    angle = step * 360.0 / n  
    for i in range(n):  
        julie.forward(side)  
        julie.right(angle)
```

```
star(5, 2, 100)
```

# Příklady (procvičení):

## 1.2.4. Spirála

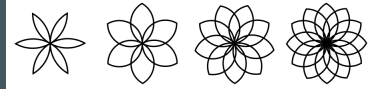


```
from turtle import Turtle
julie = Turtle()

def spiral(n, angle, step):
    for i in range(n):
        julie.forward(i * step)
        julie.right(angle)

spiral(100, 61, 1)
```

## 1.2.6. Kytky



```
from math import pi
from turtle import Turtle
julie = Turtle()
julie.speed(10)
def arc(r, angle):
    for i in range(angle):
        julie.forward(2 * pi * r / 360)
        julie.right(1)
def flower(radius, angle, leaves):
    for i in range(leaves):
        arc(radius, angle)
        julie.right(180 - angle)
        arc(radius, angle)
        julie.right(180 - angle + 360.0 / leaves)

flower(100, 80, 9)
```



# Další procvičení na



## Umíme programovat

- [Programování v Pythonu](#)
- [Python želva](#)

# Osnova

- organizace cvičení
- studijní materiály
- seznámení s vývojovým prostředím
- základy Pythonu
- želví grafika