IB111

Základy programování

František Lachman lachmanfrantisek@mail.muni.cz

cvičení 3

2. říjen 2018

Osnova

- poznámka k domácí úloze
- kontrolní otázky
- pokročilejší výpočty, return

Docházka

Kontrolní otázky

Jaký je rozdíl mezi číselnými typy `int` a `float`?

Jak zapisujeme operace „celočíselné dělení“ a „dělení se zbytkem“?

```
type(3)
```

int

```
type(3.0)
```

float

```
type(3 // 3)
```

int

```
type(3 / 3)
```

float

```
type(3.0 / 3)
```

float

```
type(3.0 // 3)
```

float

Co dělají funkce `round`, `math.floor`, `math.ceil`?

```
from math import floor, ceil
```

```
print(round(1.5), round(2.5))
```

```
2 2
```

Jaká je základní myšlenka algoritmu pro výpočet ciferného součtu?

Jaká je základní myšlenka algoritmu pro výpočet odmocniny?

Jaká je základní myšlenka algoritmu pro převod na binární zápis?

Jaký je význam příkazu `return`?

Jaký je rozdíl mezi tím, když na konci funkce zavoláme `return` a `print`?

Jaký je rozdíl mezi `return` a `return x` ?

```
def five():
    return 5
```

```
a = five()
b = five()
```

```
type(a)
```

int

```
print(a)
print(a + b)
```

5
10

```python
def double(a):
    b = a * 2
    return b # zkráceně return a * 2
```

```python
b = double(5)
print("something else")
print(b)
```

something else
10

```
print(double(five()))
```

10

```
print(double(double(double(double(five())))))
```

80

```
x = five()
y = double(x)
print(x + y)
```

```
def print_double(x):
    print(x*2)
```

10

```
x = five()
y = print_double(x)
print(x + y)
```

TypeError: unsupported operand type(s) for +: 'int' and 'NoneType'

```
def nothing():
    return
```

```
nothing()
a = nothing()
print(a)
type(a)
```

None
NoneType

```
def my_function():
    print("jsem tu")
    return
    print("sem se vypocet nikdy nedostane")
```

```
my_function()
```

jsem tu

Návrat z funkce pomocí `return`

```
def my_max(number1, number2):
    if number1 == number2:
        print("jsou stejna")
        return number1
    if number1 > number2:
        print("prvni je vetsi")
        return number1
    print("druhe je vetsi")
    return number2
```

```
a = my_max(2, 2)
b = my_max(2, 3)
c = my_max(3, 2)
```

```
print(a, b, c) # 2 3 3
```

Jaký význam má `pass`?

Prázdné tělo - `pass`

```
def without_return():
    pass
```

- spustitelný kód bez implementace
- explicitně vyjádřené "nedělání ničeho"

Příklady - 3.1. Pokročilé počítání

Napište funkci `series_sum(n)`, která vrátí součet čísel od `1` do `n`.

```
def series_sum(n):
    result = 0
    for i in range(1, n+1):
        result += i
    return result

my_sum  = series_sum(6)

print("hello, my sum:", my_sum)
```

hello, my sum: 21

3.1.1. Faktoriál pomocí `for`

Napište funkci `factorial(n)`, která vrací faktoriál čísla `n` a využívá cyklus `for`.
(Připomeňme, že `n! = 1·2·3· ... ·n` a že `0! = 1`.)

```
def factorial(n):
    result = 1
    for i in range(1, n+1):
        result *= i
    return result

fact_of_six  = factorial(6)

print(fact_of_six)
```

720

3.1.3. Ciferný součet

Napište funkci `digit_sum(n)`, která vrátí ciferný součet čísla `n`.

```
def digit_sum(n):
    result = 0
    while n!=0:
        result += n%10
        n = n // 10
    return result

print(digit_sum(567890))
```

35

Příklady - 3.2. Dělitelnost a prvočísla

3.2.1. Dělitelé

Napište funkci `divisors(n)`, která vypíše všechny dělitele čísla `n`.

```
def divisors(n):
    for i in range(1, n//2 + 1):
        if n % i == 0:
            print(i, end=" ")
    print(n)

divisors(12)
```

1 2 3 4 6 12

3.2.2. Počet dělitelů

Napište funkci `divisors_count(n)`, která vrátí počet dělitelů čísla `n`.

```
def divisors_count(n):
    if n < 1:
        return None
    result = 1
    for i in range(1, n//2 + 1):
        if n % i == 0:
            result +=1
    return result

print(divisors_count(12))
```

6

3.2.3. Je prvočíslo

Napište funkci `is_prime(n)`, která vrátí `True` pokud je číslo `n` prvočíslo, jinak `False`.

```
def is_prime(n):
    return divisors_count(n) == 2

def is_prime(n):
    if n < 2:
        return False
    for i in range(2, int(math.sqrt(n)) + 1):
        if n % i == 0:
            return False
    return True
```

3.2.4. Prvočísla menší než n

Napište funkci `primes_less_than(limit)`, která vypíše všechna prvočísla menší než `limit`.

```
def primes_less_than(limit):
    for candidate in range(limit):
        if is_prime(candidate):
            print(candidate, end=" ")
    print()
```

3.2.12. Euklidův algoritmus

Implementujte Euklidův algoritmus pomocí funkce `euklid(a, b)`.

```
def euclid(a, b):
    while a:
        a, b = b % a, a
    return b
```

Příklady - 3.3. Aproximace

- 3.3.1. Eulerovo číslo
  Napište funkci `e()` pro přibližný výpočet Eulerova čísla pomocí nekonečného součtu s přesností na miliontiny.

$$\mathrm{e} = \sum_{n=0}^{\infty} \frac{1}{n!} = \frac{1}{0!} + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \cdots$$

```
def factorial(n):
    value = 1
    for i in range(1, n + 1):
        value *= i
    return value

def e():
    result = 0
    for i in range(15):
        result += 1 / factorial(i)
    return round(result, 6)
```

Příklady - 3.4. Převody mezi číselnými soustavami

3.4.1. Převod z desítkové soustavy do dvojkové

Napište funkci `convert_10_to_2(n)`, která vrátí zadané číslo `n` ve dvojkové soustavě jako řetězec.

```
def convert_10_to_2(n):
    if n==0:
        return ''
    else:
        return convert_10_to_2(n // 2) + str(n % 2)
```

3.4.2. Převod z dvojkové soustavy do desítkové

Napište funkci `convert_2_to_10(n)`, která vrátí zadaný řetězec `n` reprezentující binární číslo v desítkové soustavě jako int.

```
def convert_2_to_10(n):
    result = 0
    power = 0
    for b in n[::-1]:
        if b == "1":
            result += 2 ** power
        power += 1
    return result
```

Osnova

- poznámka k domácí úloze
- kontrolní otázky
- pokročilejší výpočty, return