C2184 Úvod do programování v Pythonu

Nepovinné úkoly – vzorová řešení

Cvičení 4.1: Kvadratická rovnice

V této ukázce si napíšeme program pro řešení kvadratické rovnice. Obecně máme kvadratickou rovnici definovanou takto:

$$ax^2 + bx + c = 0$$

Kvadratická rovnice může mít v oboru reálných čísel dva, jeden nebo žádné kořeny. Počet kořenů můžeme určit podle hodnoty diskriminantu D:

$$D = b^2 - 4ac$$

Pokud je D kladné, rovnice má dva kořeny x_1, x_2. Ty spočítáme podle vzorců:

$$x_1 = \frac{-b - \sqrt{D}}{2a} \qquad x_2 = \frac{-b + \sqrt{D}}{2a}$$

Když je D rovno nule, rovnice má pouze jeden kořen x, který je:

$$x = \frac{-b}{2a}$$

Když je D záporné, rovnice nemá žádné kořeny.

Úkol:

Ze vstupu načtěte koeficienty kvadratické rovnice a, b, c. Rovnici vyřešte a její kořeny vypište na výstup.

Vypisujte s přesností na dvě desetinná místa. Když budou kořeny dva, vypište menší, pak větší. Když nebude žádný kořen, vypište `No solution`

Vzorový vstup 1:	**Vzorový vstup 2:**	**Vzorový vstup 3:**
`1 -5 6`	`2.25 15.0 25.0`	`1 2 2`
Vzorový výstup 1:	**Vzorový výstup 2:**	**Vzorový výstup 3:**

2.00 3.00	-3.33	No solution

[ ]:
```python
import math

a, b, c = input().split()
a = float(a)
b = float(b)
c = float(c)

discriminant = b**2 - 4*a*c

if discriminant > 0:
    x1 = (-b - math.sqrt(discriminant)) / (2*a)
    x2 = (-b + math.sqrt(discriminant)) / (2*a)
    print(f'{x1:.2f} {x2:.2f}')
elif discriminant == 0:
    x = -b / (2*a)
    print(f'{x:.2f}')
else:
    print('No solution')
```

Cvičení 4.2: Testování čísel

Úkol:

Ze vstupu načtěte celé číslo.

1. Otestujte, jestli jsme dostali číslo z intervalu 0–100 (včetně)
2. Pokud ano, otestujte, jestli je číslo dělitelné pěti, třemi nebo oběma čísly zároveň

Na výstup vypište nejvhodnější z těchto hlášek:

- `Out of range` (pokud je mimo intervalu 0 až 100)
- `Divisible by 3 and 5`
- `Divisible only by 3`
- `Divisible only by 5`
- `Not divisible by 3 or 5`

Vzorový vstup 1:	**Vzorový vstup 2:**
`100`	`101`
Vzorový výstup 1:	**Vzorový výstup 2:**
`Divisible only by 5`	`Out of range`

[ ]:
```python
n = int(input())

if 0 <= n <= 100:
    if n % 3 == 0 and n % 5 == 0:
        print('Divisible by 3 and 5')
    elif n % 3 == 0:
        print('Divisible only by 3')
    elif n % 5 == 0:
        print('Divisible only by 5')
    else:
        print('Not divisible by 3 or 5')
else:
    print('Out of range')
```

Cvičení 4.3: Sčítačka

Úkol:

Vytvořte program, který bude od uživatele načítat reálná čísla, vždy jedno číslo na každém řádku. Když pak uživatel zadá místo čísla znak =, vypíše se součet dosud zadaných čísel a program skončí. Součet vypisujte na 2 desetinná místa.

Vzorový vstup 1:	**Vzorový vstup 2:**	**Vzorový vstup 3:**
5 1.2 0.5 =	-1.12 =	=
Vzorový výstup 1:	**Vzorový výstup 2:**	**Vzorový výstup 3:**
6.70	-1.12	0.00

[ ]:
```python
# U tohoto úkolu nevíme předem, kolik řádků budeme načítat.
# Proto musíme načítání provádět v rámci cyklu while.
# Abychom ale mohli zkontrolovat, jestli je řádek '=' nebo číslo,
# musíme už nějaký řádek mít načtený.
# Tento problém lze vyřešit více způsoby:
```

[ ]:
```python
# Řešení 1
# Načteme první řádek ještě před cyklem:
total = 0
line = input()
while line != '=':
    total += float(line)
    line = input()
print(f'{total:.2f}')
```

```
[ ]: # Řešení 2
     # Abychom nemuseli funkci input volat dvakrát,
     # použijeme fake řádek '0', který neovlivní výsledek:
     total = 0
     line = '0'
     while line != '=':
         total += float(line)
         line = input()
     print(f'{total:.2f}')
```

```
[ ]: # Řešení 3
     # Řádky načítáme až v cyklu, cyklus pak ukončíme pomocí break:
     total = 0
     while True:  # Zde ještě neznáme hodnotu line, takže nemůžeme provést␣
       ↪kontrolu.
         line = input()
         if line == '=':
             break
         total += float(line)
     print(f'{total:.2f}')
```

```
[ ]: # Řešení 4:
     # Použijeme "walrus operator" :=, který umí uložit hodnotu
     # do proměnné a rovnou tuto hodnotu dál použít např. v podmínce.
     # Toto je novinka Pythonu 3.8:
     total = 0
     while (line := input()) != '=':
         total += float(line)
     print(f'{total:.2f}')
```

Cvičení 4.4: Pyramida

Úkol:

Napište kód, který načte ze vstupu přirozené číslo n a znak z a vykreslí na výstup n-patrovou pyramidu složenou ze znaku z.

(Volný prostor zleva pyramidy je vyplněn mezerami, pouze před posledním řádkem není žádná mezera. Zprava mezery nemusí být.)

Vzorový vstup 1:

```
3 i
```

Vzorový výstup 1:

```
  i
 iii
iiiii
```

Vzorový vstup 2:

```
5 *
```

Vzorový výstup 2:

```
    *
   ***
  *****
 *******
*********
```

[ ]:
```
# Nejdřív si musíme rozmyslet, kolik znaků bude na kterém řádku.
# Na 0. je jeden znak, pak na každém dalším řádku o dva víc.
# Matematickou úvahou zjistíme, že na i-tém řádku má být 2*i+1 znaků.
# Podobně spočítáme, že na začátku i-tého řádku má být n-i-1 mezer.
# Na posledním (n-1) řádku tedy bude
# 2*(n-1)+1 = 2*n-1 znaků a n-(n-1)-1 = 0 mezer.
```

[ ]:
```
# Řešení 1 (nízko-úrovňové)
# Vypisujeme znaky po jednom pomocí vnořeného cyklu:
n, z = input().split()
n = int(n)

for i in range(n):
    n_chars = 2*i + 1
    n_spaces = n - i - 1
    for j in range(n_spaces):
        print(' ', end='')
    for j in range(n_chars):
        print(z, end='')
    print()  # Nový řádek
```

[ ]:
```
# Řešení 2
# Využijeme násobení řetězců:
n, z = input().split()
n = int(n)

for i in range(n):
    n_chars = 2*i + 1
    n_spaces = n - i - 1
    print(n_spaces * ' ' + n_chars * z)
```

[ ]:
```
# Řešení 3
# Koho nebaví vymýšlet vzorce:
n, z = input().split()
n = int(n)

n_chars = 1
```

```
n_spaces = n-1
for i in range(n):
    print(n_spaces * ' ' + n_chars * z)
    n_chars += 2
    n_spaces -= 1
```

[ ]:
```
# Řešení 4
# Využijeme násobení řetězců + formátování se zarovnáním:
n, z = input().split()
n = int(n)

widest_line = 2*n - 1

for i in range(n):
    n_chars = 2*i + 1
    print(f'{n_chars*z:^{widest_line}}')
```

Cvičení 4.5: Výpočet mocniny pomocí cyklu

Umocňování čísla z na exponent e lze rozepsat pomocí násobení: $z^e = z \cdot z \cdot z \cdot$ ..., kde počet zetek je roven e.

Úkol:

Ze vstupu získejte dvě přirozená čísla z, e. Spočítejte a vypište mocninu z^e.

Úlohu řešte pomocí cyklu, bez použití operátoru ** nebo funkce `pow`.

Vzorový vstup 1:	**Vzorový vstup 2:**
2 3	3 10
Vzorový výstup 1:	**Vzorový výstup 2:**
8	59049

[ ]:
```
z, e = input().split()
z = int(z)
e = int(e)

result = 1
for i in range(e):
    result *= z

print(result)
```