Cvičení z programování – Python

verze 1.1 Jakub Mazuch

Úvod do programování

Programové vybavení:

— Visual studio Code (https://code.visualstudio.com/)

— Python 3 (https://code.visualstudio.com/docs/python/python-tutorial)

— Rozšíření Visual studio Code – Linter

Plán

1. Proměnné a datové typy
2. Logický výraz, podmínka
3. Cyklus
4. Pole
5. Známé algoritmy - Euklidův algoritmus, test prvočíselnost, Eratosthenovo síto
6. Funkce, rekurze
7. Vícerozměrné pole
8. Prohledávání do šířky, prohledávání do hloubky
9. Třídící algoritmy, časová a prostorová složitost
10. Spojový seznam, fronta, zásobník

Teorie

— Algoritmus, algoritmizace

— Programování – programovací paradigmata, programovací jazyk

- Zdrojový kód
- Překladač mezi člověkem a cílovým (binárním) kódem
- Datové typy, dynamické datové typy
- Case sensitive `a = 5` ≠ `A = 5`
- Syntaktické vychytávky Pythonu

Cvičení

1. Vytvořte program, který vypíše `Hello world`

I. Proměnné a datové typy

— Příkaz

— Hodnota – int, string, ... — desetinná čárka, nebo tečka?

— Syntaxe, rozdíl mezi `=` a `==`

— Proměnná

— Parsování

Cvičení

2. **Papoušek.**

   Vytvořte program Papoušek, který bude papouškovat Váš vstup.

3. **Druhá mocnina.**

   Uživatel zadá číslo x. Program vypíše jeho druhou mocninu (x^2).

4. ***n*-tá mocnina.**

   Uživatel zadá číslo n a x. Program vypíše x^n.

5. **Kalkulačka.**

   Uživatel zadá čísla x a y. Program vypíše $x + y$, $x - y$ a xy.

6. **Obvod a obsah rovinného obrazce.**

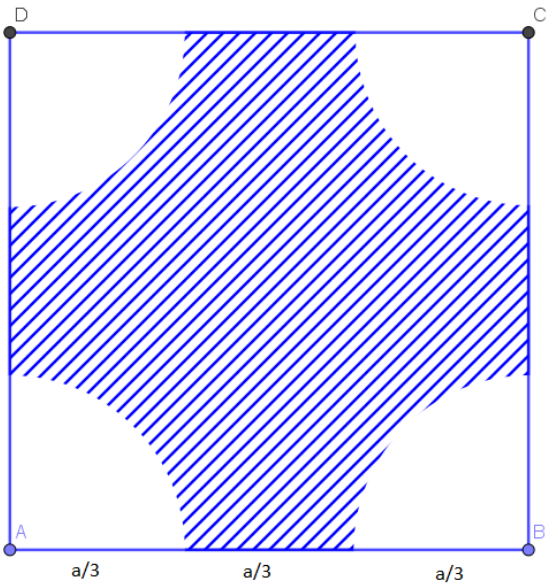
   (a) **čtverec**: Uživatel zadá délku strany a. Program vypočítá a vypíše obvod a obsah čtverce o straně a.

   (b) **obdélník**: Uživatel zadá délku stran a, b. Program vypočítá a vypíše obvod a obsah obdélníku o stranách a, b.

   (c) **kruh**: Uživatel zadá poloměr kruhu r. Program vypočítá a vypíše obvod a obsah kruhu.

(d) kombinace obrazců: Uživatel zadá číslo a. Program vypočítá obsah šrafované části obrazce níže znázorněného.

Nápověda: $|AB| = a$



7. **Povrch a objem prostorového obrazce**.

   (a) **krychle**: Uživatel zadá délku strany a. Program vypočítá a vypíše povrch a objem krychle o straně a.

   (b) **kvádr**: Uživatel zadá délky stran a, b, c. Program vypočítá a vypíše povrch a objem krychle o straně a.

   (c) **polokoule**: Uživatel zadá poloměr polokoule r. Program vypočítá a vypíše povrch a objem polokoule.

   (d) **komolý kužel**: Uživatel zadá výšku kuželu v a poloměry podstav v_1 a v_2. Program vypočítá a vypíše V a S.

8. **Zbytek po dělení.**

   Uživatel zadá čísla x a y. Program vypíše zbytek po dělení x číslem y.

$$d = (x \bmod y)$$
$$\frac{x}{y} = n + d$$

9. **Počet dnů, hodin, minut.**

   Uživatel zadá počet týdnů. Program vypočítá a vypíše počet dnů, hodin, minut obsahuje zadaný počet týdnů.

10. **Doba volného pádu.**

    Uživatel zadá výšku $h\,[m]$. Program vypíše dobu volného pádu.

    Pozn. 1: Tíhové zrychlení počítejte $g = 9{,}81\,kg \cdot s^{-2}$

II. Logický výraz, podmínka

Teorie

— Logické operátory AND, OR, $>$, $\geq$, $=$, $\geq$, $<$, $\neq$

— Podmínka úplná a neúplná

Cvičení

11. **Číslo větší nebo menší I**.

    Uživatel zadá číslo a. Program vypíše, zda je číslo a větší nebo menší, než 10.

12. **Číslo větší nebo menší II**.

    Uživatel zadá čísla a, b. Program vypíše, které číslo je větší.

13. **Číslo větší nebo menší III**.

    Upravte program z úlohy (11) tak, aby program vypisoval diferenci mezi čísly.

14. **Délka čísla**.

    Uživatel zadá číslo a. Program vypíše, zda je číslo a jednociferné, dvouciferné, tříciferné, čtyřciferné, pěticiferné, šesticiferné nebo více jak šesticiferné.

15. **BMI.**

    Uživatel zadá výšku $h\,[cm]$ a váhu $m\,[kg]$. Program vypíše hodnotu BMI $[kg/m^2]$ a příslušnou kategorii (např. *ideální váha*), viz (Pozn. 2).

$$BMI = \frac{m[kg]}{(h[m])^2}$$

    Pozn. 1: Nezapomeňte vyřešit problém, kdy uživatel zadává výšku v centimetrech, ale vzorec předpokládá výšku v metrech.

    Pozn. 2: Jednotlivé kategorie BMI:

podváha:	BMI $<$ 18,5
ideální váha:	18,5–25
nadváha:	25–30
mírná obezita:	30–35
střední obezita:	35–40
morbidní obezita:	BMI $>$ 40

16. **Zmrzlinový test I.**

    Uživatel zadá svoji sumu peněz. Program vypíše, zda si může dovolit koupit zmrzlinu, či nikoliv. Uvažujte pevnou částku zmrzliny (20,00 Kč).

17. **Zmrzlinový test II.**

    Uživatel zadá svoji sumu peněz a cenu zmrzliny. Program vypíše, zda si může dovolit koupit zmrzlinu či nikoliv.

18. **Kvadratická rovnice.**

    Uživatel zadá čísla a, b, c. Program vypočítá a vypíše kořeny kvadratické rovnice.

    Pozn.: Předpokládejte uživatelovu elementární neznalosti. Upravte program tak, aby dokázal efektivně reagovat na všechny případy, vč. těch speciálních:

    $$(a = 0) \vee (b = 0) \vee (c = 0) \vee (b^2 - 4ac) < 0$$

19. **Kvadratická rovnice II.**

    Upravte program z úlohy (17) tak, aby dokázal pracovat i na oboru $\mathbb{C}$, resp. aby pracoval i s problémem $(b^2 - 4ac) < 0$.

20. **Parita I.**

    Pro zadané číslo n zjistěte, zda se jedná o číslo sudé nebo liché.

21. **Pěkná čísla.**

    Uživatel zadá dvojici libovolných čísel a, b. Čísla jsou pěkná, pokud aspoň jedno z čísel je sudé. Program otestuje a vypíše, zda zadaná dvojice a, b je pěkná.

III. Cyklus

Teorie

— Cyklus se známým počtem opakování

— Cyklus s podmínkou na začátku

— Modulo

Cvičení

22. **Řady čísel I**

    Vytvořte program, který vypíše řadu čísel dle zadání pomocí vhodného cyklu.

    a) od 1 od 10 po 1
    b) od 0 do 8 po 1
    c) od −5 do 12 po 1
    d) od 1 do 20 po 3
    e) od −10 do +20 po 4
    f) od 10 do 0 po 1 (resp. −1)
    g) od 7 do −3 po 1
    h) od 20 do −20 po 4

23. **Řady čísel II**

    Uživatel zadá a, b a c. Program vypíše řadu čísel od a do b po c.

24. **Zmrzlinový test III.**

    Uživatel zadá svoji sumu peněz, cenu jednoho kopečku zmrzliny a počet kopečků. Program vypíše, zda si může zmrzlinu dovolit či nikoliv.

25. **Zmrzlinový test IV (pokladna).**

    Uživatel nejprve zadává počet jednotek a sortiment.

    Následně zadává hotovost, ve které zákazník platí (1, 2, 5, 10, 20, 50, 100, 200, 500, 1000, 2000, 5000).

    Stisknutím specifické klávesy (např. H) se transakce potvrdí a program vypíše kolik uživateli vrací peněz.

26. **Hádání čísel I.**

Uživatel zadá a a b. Program si myslí náhodné číslo od a do b. Uživatel postupně zadává čísla a v závislosti na zadaném číslu program odpovídá, zda zadané číslo je:

— Větší než číslo, které si program myslí, nebo

— Menší než číslo, které si program myslí, nebo

— Čísla jsou shodná – BINGO!

Hra končí tehdy, uhádne-li uživatel číslo.

Kód pro generování náhodného čísla:

```
from random import randint
nahodne_cislo = randint(a, b)
print(nahodne_cislo)
```

27. **Hádání čísel II.**

Upravte program z úlohy (26) tak, aby

a. dokázala v případě uživatelské chyby ($b > a$) obrátit hranice.

b. uživatel mohl po ukončení hry (uhádnutí čísla) nadále

(1) pokračovat se stejnými hranicemi

(2) pokračovat s jinými hranicemi

(3) ukončit hru

c. *počítala uživatelovi skóre (tj. počet nutných tahů pro uhádnutí čísla) v každém kole a po skončení určila nejvyšší skóre

28. **Parita II.**

Upravte program z úlohy (19) tak, aby uživatel mohl číslo zadávat opakovaně.

29. **Faktoriál.**

Uživatel zadá číslo n. Program vypočítá a vypíše $n!$ pomocí cyklu se známým počtem opakování.

IV. Seznamy (pole)

Teorie

— Pole, index

Cvičení

30. **Mocniny I.**

    Uživatel zadá číslo n. Program vypíše všechny mocniny dvojky, menší než n.

31. **Mocniny II**.

    Upravte program z úlohy (21) tak, aby program vypsal součet všech mocnin dvojky menších, než n.

32. **Převod z desítkové soustavy do binární.**

    Uživatel zadá dekadické číslo a. Program převede toto číslo do binární soustavy a vypíše.

33. **Převod z binární soustavy do desítkové.**

    Uživatel zadá binární číslo a. Program převede toto číslo do desítkové soustavy a vypíše.

    Pozn.: Zabezpečte (ošetřete) program tak, nevypisoval žádné chybné hlášky ani nepravdivé výsledky při chybném vstupu (tj. jiná čísla než 0 nebo 1).

V. Známé algoritmy (Euklidův algoritmus, test prvočíselnosti, Eratosthenovo síto)

Teorie

— Prvočíslo

— Euklidův algoritmus

Eratosthenovo[1] síto

https://cs.wikipedia.org/wiki/Eratosthenovo_s%C3%ADto

Cvičení

34. **Převod z desítkové soustavy do soustavy o libovolného základu**.

    Uživatel zadá dekadické číslo x a soustavu z. Program vypíše číslo $(x)_z$, resp. číslo x v soustavě o základu z.

    Nápověda: Využijte tzv. *Euklidův algoritmus*, ověřte zadání přirozeného základu z.

35. **Výpočet nsn a NSD**.

    Uživatel zadá čísla x a y. Program vypočítá a vypíše nejmenší společný násobek a největší společný dělitel čísel x, y.

36. **Test prvočíselnosti**.

    Uživatel zadá číslo x. Program vypíše, zda je číslo x prvočíslo, či nikoliv.

37. **Eratosthenovo síto.**

    Uživatel zadá číslo n. Program vypíše všechna prvočísla.

[1] **Eratosthenés z Kyrény** (řecky Ἐρατοσθένης, mezi 276-272 v Kyréně – 194 př. n. l. v Alexandrii) byl matematik, astronom a zřejmě největší geograf antického Řecka. Působil též jako správce alexandrijské knihovny. Věnoval se také literární činnosti jako básník. Bylo po něm pojmenováno tzv. Eratosthenovo síto a kráter Eratosthenes na Měsíci.

VI. Funkce, Rekurze

Teorie

— Funkce

— Rekurze

Cvičení

38. **Ověření rodného čísla.**

Uživatel zadá rodné číslo. Program vyhodnotí, zda je zadané rodné číslo validní.

Pozn.:

Pozor! Obecná pravidla jsou trochu složitější, než pouhá dělitelnost 11.

Do roku 1985 bylo vydáno asi 1 000 čísel, která dělitelnost 11 nesplňují. Váš skript by je ovšem měl vyhodnotit jako správná.

Tato pravidla jsou platná pro rodná čísla vydaná po 1. lednu 1954.

Pravidla:

1. spočti zbytek po dělení prvních devíti číslic a čísla 11
2. je-li zbytek 10, musí být poslední číslice 0
3. jinak poslední číslice musí být rovna zbytku

Úvahy:

- Co když bude číslo jen 9místné?
- Co může být na pozici měsíců?

Vyzkoušejte na číslech:

- 935405 / 0728 (ano)
- 935106 / 0410 (ne)
- 80123 / 3540 (ano)

39. **Ověření čísla účtu.**

Uživatel zadá číslo bankovního účtu. Program ověří, zda je číslo účtu validní.

Pozn.:

3.2. Zajištění čísel účtů modulo 11

Čísla účtů jsou zajištěna podle následujícího algoritmu v souladu s vyhláškou.

Algoritmus kontroly čísla **ABCDEFGHIJ** na modulo 11:

Číslice	A	B	C	D	E	F	G	H	I	J
Váhy*)	6	3	7	9	10	5	8	4	2	1

*)Pozn.: Váhy jsou získány jako rozdíl n-té mocniny 2 a nejbližšího nižšího násobku 11.

Váhy se k číslicím na jednotlivých pozicích čísla účtu přiřazují zprava. Číslo **ABCDEFGHIJ** je zajištěno modulo 11, pokud je součet **S** beze zbytku dělitelný 11, přičemž

$$S = J*1 + I*2 + H*4 + G*8 + F*5 + E*10 + D*9 + C*7 + B*3 + A*6$$

40. **Aritmetická posloupnost I.**

Uživatel zadá první člen (a_1) posloupnosti $(a_n)_{n=1}^{\infty}$, její diferenci d a počet členů n. Program vypíše všechny členy posloupnosti $a_1, a_2, a_3, \dots, a_n$.

$$a_n = a_{n-1} + d$$

41. **Aritmetická posloupnost II.**

Upravte program z úlohy () tak, aby program vypsal součet všech členů posloupnosti.

$$\sum_{n=1}^{n}(a_n) = a_1 + a_2 + \cdots + a_n$$

42. **Geometrická posloupnost I.**

Uživatel zadá první člen (a_1) posloupnosti $(a_n)_{n=1}^{\infty}$, její kvocient q a počet členů n. Program vypíše všechny členy posloupnosti $a_1, a_2, a_3, \dots, a_n$.

$$a_n = q \cdot a_{n-1}$$

43. **Geometrická posloupnost II.**

Upravte program z úlohy () tak, aby program vypsal součet všech členů posloupnosti.

$$\sum_{n=1}^{n}(a_n) = a_1 + a_2 + \cdots + a_n$$

44. ***Pascalův trojúhelník.**

Uživatel zadá číslo n. Program vypíše n řádků Pascalova trojúhleníku.

Př.: $n = 6$:

```
1
1  1
1  2  1
1  3  3  1
1  4  6  4  1
1  5  10  10  5  1
1  6  15  20  15  6  1
```

45. Fibonacciho posloupnost.

V program definujte funkce odpovídající následujícím podúlohám. Pro úvod programu vytvořte menu – možnosti pro odpovídající volbu možností.

i) Uživatel zadá číslo n. Program vypíše n členů Fibonacciho posloupnosti

j) Program vypíše 10 členů Fibonacciho posloupnosti.

k) Uživatel zadá číslo n. Program vypíše součet n členů Fibonacciho posloupnosti.

l) *Uživatel zadá číslo n a k. Program vypíše k členů Fibonacciho posloupnosti od n-tého členu dále.

Rekurze

46. **Faktoriál.**

Uživatel zadá číslo n. Program vypočítá a vypíše $n!$ pomocí rekurze.

47. **Výpočet a^b.**

Uživatel zadá a, b. Pomocí rekurze program vypočítá a^b.

Nápověda 1: $x^m \cdot x^n = x^{m+n}$

Nápověda 2: $b = b + 1 - 1 = 1 + b - 1$

$\Rightarrow a^b = a^{1+b-1} = a \cdot a^{b-1}$

48. **Fibonacciho posloupnost I**.

Uživatel zadá číslo n. Program vypíše n-tý člen Fibonacciho posloupnosti pomocí rekurze.

Nápověda:

$$F(n) = \begin{cases} 0, & \text{pro } n = 0; \\ 1, & \text{pro } n = 1; \\ F(n-1) + F(n-2) & \text{jinak.} \end{cases}$$

VII. Vícerozměrné pole

Teorie

— Pole – opakování

— Dvourozměrné pole

— Vícerozměrné pole

Cvičení

49. **Šachovnice.**

    Pomocí dvourozměrného pole reprezentujte šachovnici 8×8. Prázdné pole šachovnice bude označeno symbolem `*`. Dále bude moci uživatel na libovolné pole šachovnice umístit věž (pole bude označeno symbolem `V`). Program následně označí symbolem `O` všechna pole šachovnice, která tato věž ohrožuje.

50. ***Ověření symetrie matice.**

    Uživatel zadá velikost n čtvercové matice $n \times n$, Dále postupně naplňuje všechny členy matice. Program ověří, zda se jedná o symetrickou matici.

    Pozn.: Matice $A = (a_{ij})$ stupně n se nazývá symetrická právě tehdy, když pro všechna $i, j = 1,2, \dots, n$ platí:

    $$a_{ij} = a_{ji}.$$

    Např.: Nechť $B = (b_{ij})$ stupně 3

    $$B = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}.$$

    Pak B je symetrická, pokud

    $$a_{21} = a_{12} \wedge a_{31} = a_{13} \wedge a_{32} = a_{23}$$

51. ****Problém osmi dam na šachovnici.** (významný matematický problém)

    Umístěte na šachovnici 8 dam tak, aby se podle pravidel šachu vzájemně neohrožovali.

52. *****Problém n dam na šachovnici.**

    Zobecněte úlohu (49) na n dam na šachovnici. Číslo n zadá uživatel.

    Pozn.: Počet řešení se asymptoticky chová asi jako $\frac{n!}{2{,}54^n}$.

    Nápověda: Zamyslete se nad počtem řešení pro $n = 1, 2, 3$.

VIII. Prohledávání do šířky (BFS), prohledávání do hloubky (DFS)

Teorie

— Úvod do teorie grafů

— Typy grafů: cesta, kružnice, strom

— Úplný graf, podgraf, bipartitiní graf

— Orientovaný graf, neorientovaný graf

— Reprezentace grafů: Matice sousednosti, matice incidence, reprezentace polem

— Průchod stromem do hloubky (DFS) + časová složitost

— Průchod stromem do šířky (BFS) + časová složitost

— *Hledání nejkratší cesty (Dijkstrův algoritmus)

Cvičení

53.***Jezdcova procházka** (super multi hardcore!)

Jezdec se pohybuje v souladu s šachovými pravidly po prázdné šachovnici a jeho úkolem je, aby každé pole navštívil právě jednou.

Nápověda 1: Pro vizualizaci problému je vhodné sestrojit (orientovaný) graf.

Nápověda 2: Nejjednodušším algoritmem pro nalezení řešení problému jezdcovy procházky je tzv. *backtracking* (prohledávání do hloubky). Při něm jezdec skáče zcela libovolně na zatím neobsazená pole, dokud se neocitne v situaci, kde již nemůže táhnout dále. V takovém případě se vrátí o jeden tah zpět a pokračuje jinou cestou. Tento algoritmus je ale vzhledem k poměrně časté nutnosti vracet se ze „slepých uliček“ dosti pomalý.

IX. Řadící algoritmy, časová a prostorová složitost

Teorie

— Selection-sort

— Bubble-sort

— Insertion-sort

— Heapsort

— Merge sort

— Quicksort

— Časová složitost algoritmu

— Prostorová složitost algortimu

Cvičení

54. **Řazení algoritmů I.**

    Uživatel zadá velikost pole n. Následně pole naplní a seřadí sestupně algoritmem dle zadání.

    a. Selection-sort
    b. Bubble-sort
    c. Insertion-sort
    d. Heapsort
    e. Merge sort
    f. Quicksort

55. **Řazení algoritmů II.**

    Upravte úlohu (54) tak, aby si uživatel mohl vybrat z možnosti *sestupné řazení* a *vzestupné řazení*.

X. Lineární spojový seznam, fronta a zásobník

Teorie

— Lineární spojový seznam

— Fronta

— Zásobník

Cvičení

(přemýšlím)