

Programování 1 pro matematiky

1. cvičení, 29-9-2022

tags: Programování 1 2022, středa, čtvrtek

Obsah:

- co budeme dělat v tomto semestru
- co od sebe očekáváme a pravidla hry
- první kroky v Pythonu

Co budeme dělat

NMN111 Programování I, dotace 0/2, zakončení zápočtem

Náplní předmětu Programování 1 jsou základy programovacího jazyka Python, ovládnutí jeho příkazů a také praktické práce u počítače při psaní a ladění programů. Jde nám tedy pouze o technickou stránku věci, nebudeme se příliš věnovat nějakému výkladu algoritmů a už vůbec ne jejich efektivitě, tomu bude věnován až předmět Programování 2 v letním semestru.

V minulých letech jsme během zimního semestru učivo procházeli zhruba následovně:

1. Úvod, podmínky k zápočtu, ReCodEx. Instalace Pythonu, IDLE. Python jako kalkulačka. Výrazy, operace s čísly, relace, logické spojky. Programy - základní input a print, while, if, indentace, komentáře.
2. Pořádně print, if (zanořování, elif), ciferný součet, Euklidův algoritmus, test prvočíselnosti.
3. Zpracování posloupnosti dat. Seznamy, operace, indexování.
4. Seznamy - operace. Více čísel na řádku - split(). Fibonacciho čísla, vyhledávání v poli, třídění.
5. Formátovaný výstup. Ladění programu. Funkce - parametry, lokalita.
6. Řezy seznamů a řetězce.
7. N-tice (tuples). List comprehension. Gaussova eliminace.
8. Množiny a slovníky.
9. Základy objektového programování.
10. Objektový návrh programu.
11. Funkce jako objekt. Lambda-funkce.
12. Soubory. Výjimky.
13. Standardní knihovna (random, itertools, ...)

Tento přehled je jen rámcový, k jednotlivým tématům se budeme opakovaně vracet v kruzích.

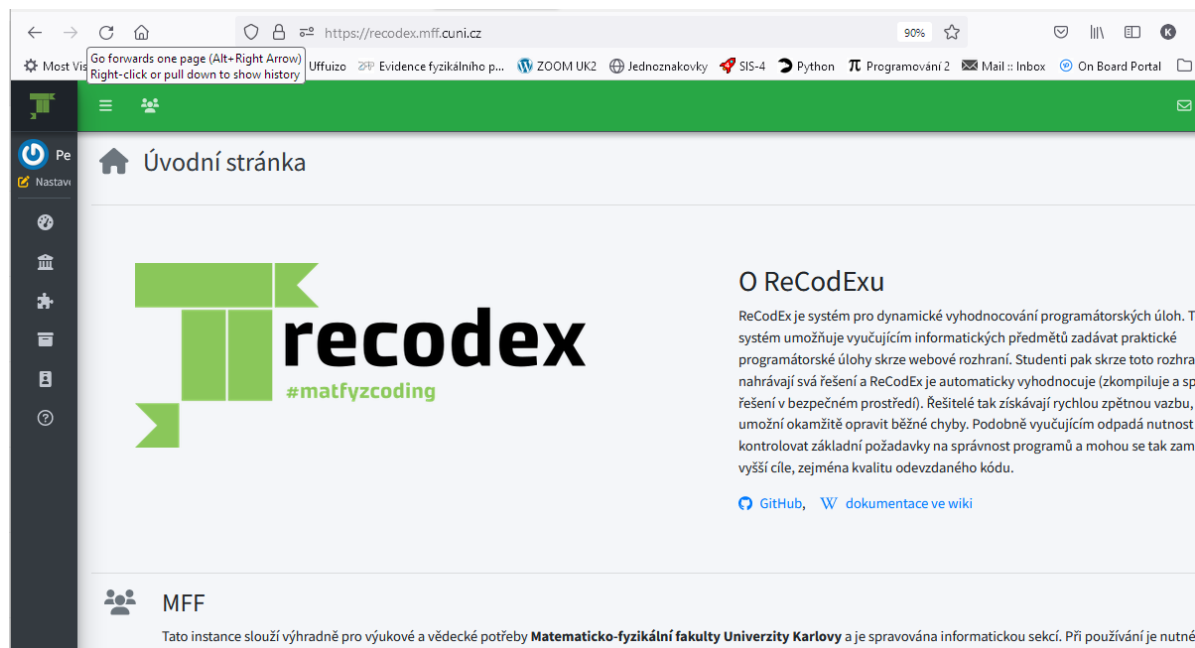
Pomoc

Budeme dělat jednoduché věci, ale programování je spojeno s častými pocity frustrace, když vám nebude fungovat něco, co by podle vás určitě fungovat mělo. Základní postup v takovémto případě je zeptat se lámanou angličtinou Googlu. Zpravidla najdete kvalifikovanou odpověď.

Klidně se ptejte i mě, v průběhu cvičení, nebo e-mailem. Nepodceňujte zejména problémy s instalací nebo během Pythonu na svém počítači, ty je potřeba vyřešit bezodkladně.

Podmínky zápočtu: domácí úkoly

Budete dostávat domácí úkoly a odevzdávat je přes **ReCodEx**, <https://recodex.mff.cuni.cz>.



Zaregistrujte se v ReCodExu. Použijte svoje přístupové údaje do SISu. Pak se prosím zaregistrujte do skupiny pro toto cvičení, můžete tak učinit volbou "SIS integration". Tento krok je nutný, abyste si v ReCodExu našli své domácí úkoly a mohli je i odevzdávat.

Pro zápočet budu požadovat **70% správných odevzdaných domácích úkolů**. Hodnocení bude tolerantní: váš kód nemusí být úplně správný, pokud bude jasně vykazovat autenticky vynaložené úsilí a adekvátní zvládnutí probraných témat.

Zpravidla dostanete na každém cvičení 2 úlohy + 1 "bonusovou". Z hlediska bodů jsou všechny tři úlohy ekvivalentní. Počet bodů pro získání zápočtu se ale odvíjí jenom od dvou úloh na cvičení.

Kdo jsou účastníci zájezdu a co z toho vyplývá

Někteří z vás už umí programovat v Pythonu a někteří jste úplní začátečníci.

Na tomto cvičení bychom měli poněkud srovnat hendikepy. Proto bude platit několik nespravedlivých pravidel:

- Pokud máte pocit, že Python umíte a jenom byste se tady nudili, nemusíte cvičení navštěvovat. Musíte ale nasbírat dostatečný počet bodů za domácí úkoly.
- Domácí úkoly jsou většinou hodnoceny 10 body bez ohledu na obtížnost. Bonusové úlohy bývají náročnější, aby si pokročilejší Pythonisté mohli vyzkoušet svaly.

Instalace Pythonu

Tady máme vícero možností a nechám na váš výběr, kterou si zvolíte.

1. Základní distribuce Pythonu

Stáhněte si instalátor pro svůj systém tady: <https://www.python.org/downloads/>.

Zvolte si nejnovější verzi 3.11. Součástí je vlastní interpret a jednoduché IDE *Idle*. S tímto klidně vystačíte pro celý kurz, ale pro následující semestr už budete chtít pracovat v nějakém vyspělejší prostředí.

2. Anaconda

Toto je velká distribuce, která obsahuje rozsáhlou podporu pro využití Pythonu ke zpracování dat, strojové učení a pod. Stáhnete si ji tady: <https://www.anaconda.com/products/individual> a zabere vám docela hodně místa na disku. Součástí je i vyspělé IDE pro vývoj v Pythonu - *Spyder*.

3. Google Colab notebooky

Nemusíte nic instalovat, stačí jít na [colab.google.com](https://colab.research.google.com) a začít psát kód do notebooku.

<https://colab.research.google.com>

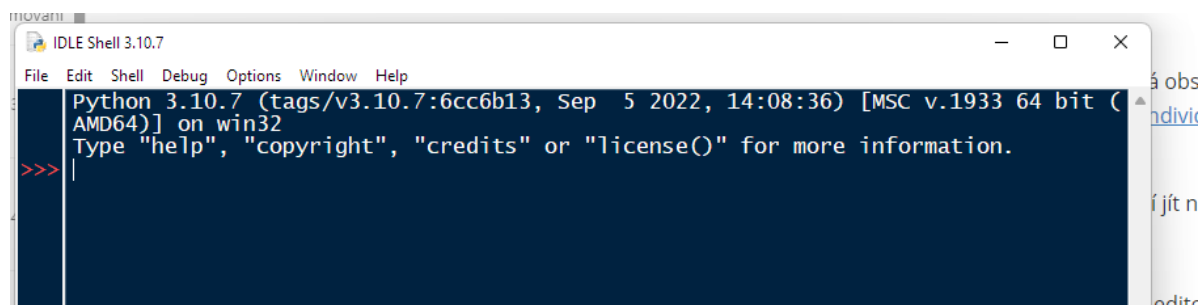
Potřebujete ale Google účet.

IDE pro Python

Existuje několik programovacích editorů a vývojových prostředí pro Python, například *PyCharm*, *Atom*, *Sublime Text*, atd. Klidně si na ně přejděte, když vám spartánské *Idle* přestane vyhovovat. U Windows je jednodušší pro tyto IDE použít distribuci Anaconda.

První kroky v Pythonu

Konečně se dostáváme k vlastnímu programování. Otevřeme si Idle konzoli:



Práce s konzolí - Python jako kalkulačka

```
1 >>> 1+1
2 2
3 >>> 2+3*4+5
4 19
5 >>> 2 + 3*4 + 5
6 19
7 >>> (2+3)*(4+5)
8 45
9 >>> 2**10
10 1024
11 >>> 2**100
12 1267650600228229401496703205376
```

Desetinná čísla

```
1 >>> 1/3
2 0.3333333333333333
3 >>> 1/3 * 3
4 1.0
5 >>> 1/6 + 1/6 + 1/6 + 1/6 + 1/6 + 1/6
6 0.9999999999999999
7 >>> 1 / (2**100)
8 7.888609052210118e-31
```

Celočíselné dělení a modulo

```
1 >>> 7 // 3
2 2
3 >>> 7 % 3
4 1
5 >>> -7 // 3
6 -3
7 >>> -(7//3)
8 -2
9 >>> (7 // 3) * 3 + 7 % 3
10 7
```

Proměnné

```
1 >>> a = 100
2 >>> b = 23
3 >>> a+b
4 123
5
6 >>> soucet = 0
7 >>> soucet = soucet + 10
8 >>> soucet = soucet + 3
9 >>> soucet
10 13
11 >>> soucet += 1
12 >>> soucet
13 14
```

Matematické funkce

```
1 >>> import math
2 >>> math.pi
3 3.141592653589793
4 >>> math.sin(math.pi / 3)
5 0.8660254037844386
```

Nápověda

```
1 >>> help(math.sin)
2 >>> help(math)
```

Logické výrazy

```
1 >>> 5**7 > 7**5
2 True
3 >>> math.cos(0) < 0
4 False
5 >>> 0.8 <= sin(pi/3) <= 0.9
6 True
7 >>> pi>3 and pi<4
8 True
9 >>> x>0 or not x>0
10 True
11 >>> 1 == 1
12 True
13 >>> 1 != 2
14 True
```

Seznamy, množiny, slovníky

```
1 >>> seznam = [1, 2, 3]
2 >>> seznam[0]
3 1
4 >>> seznam[1]
5 2
6 >>> seznam.append(4)
7 >>> seznam
8 [1, 2, 3, 4]
9 >>> seznam.pop()
10 4
11 >>> seznam
12 [1, 2, 3]
```

```
1 >>> ovoce = {"jablka", "hrušky", "pomeranče"}
2 >>> ovoce.add("švestky")
3 >>> ovoce
4 {"jablka", "hrušky", "švestky", "pomeranče"}
5 >>> ovoce.add("hrušky")
6 >>> ovoce
7 {"jablka", "hrušky", "švestky", "pomeranče"}
```

```
1 >>> čísllice = {"jedna" : 1, "dva" : 2, "tři" : 3}
2 >>> čísllice["tři"]
3 3
4 >>> čísllice["čtyři"] = 4
5 >>> čísllice
6 {"jedna" : 1, "dva" : 2, "tři" : 3, "čtyři" : 4}
```

Náš první program: počítáme od 1 do 10

```
1 i = 1
2 while i <= 10:
3     print(i)
4     i += 1
```

Odsazení je v Pythonu nekompromisně vyžadováno a musí být konzistentní. Tedy pokaždé pro stejnou úroveň stejné odsazení. Nahrazuje závorky kolem programových struktur.

```
1 i = 1
2 while i <= 10:
3     if i%2 == 0:
4         print(i)
5     i += 1
```

Teď se ještě zeptáme, do kolika se má počítat:

```
1 n = int(input("Do kolika chceš počítat? "))
2 i = 1
3 while i <= n:
4     if i%2 == 0:
5         print(i)
6     i += 1
```

Nakonec můžeme přidat do textu komentáře: Python ignoruje znaky za `#` až do konce řádku. Komentář s vykřičníkem v prvním řádku, `#!/usr/bin/env python3`, se nazývá *shebang* a v unixových systémech informuje, jak se má soubor spustit.

```
1 #!/usr/bin/env python3
2
3 # Nejprve zjistíme, do kolika počítat
4 n = int(input("Do kolika chceš počítat? "))
5
6 # Aktuální číslo
7 i = 1
8 while i <= n:          # Ještě pokračovat?
9     if i%2 == 0:       # Je číslo sudé?
10        print(i)
11    i += 1             # Další, prosím!
```

Radši nepoužívejte v zdrojovém kódu a v komentářích diakritiku, pokud to není nevyhnutné. Můžete občas narazit na nepříjemné problémy.