Jazyk Python

**Python** je moderný programovací jazyk, ktorého popularita stále rastie.

* jeho autorom je [Guido van Rossum](http://sk.wikipedia.org/wiki/Guido_van_Rossum) (vymyslel ho v roku 1989)
* používajú ho napríklad v Google, YouTube, Dropbox, Mozilla, Quora, Facebook, Rasperry Pi, …
* na mnohých špičkových univerzitách sa učí ako úvodný jazyk, napríklad MIT, Carnegie Mellon, Berkeley, Cornell, Caltech, Illinois, …
* beží na rôznych platformách, napríklad Windows, Linux, Mac. Je to *freeware* a tiež *open source*.

Na rozdiel od mnohých iných jazykov, ktoré sú kompilačné (napríklad Pascal, C/C++, C#) je Python interpreter. To znamená, že

* interpreter nevytvára spustiteľný kód (napríklad .exe súbor vo Windows)
* na spustenie programu musí byť v počítači nainštalovaný Python
* interpreter umožňuje aj interaktívnu prácu s prostredím

Hlavné vlastnosti jazyka Python:

* veľmi jednoduchá a dobre čitateľná syntax a keďže Python je aj vysoko interaktívny, je veľmi vhodný aj pre vyučovanie programovania
* na rozdiel od staticky typovaných jazykov, pri ktorých je treba dopredu deklarovať typy všetkých dát, je Python dynamicky typovaný, čo znamená, že neexistujú žiadne deklarácie
* Python obsahuje pokročilé črty moderných programovacích jazykov, napríklad podpora práce s dátovými štruktúrami, objektovo-orientovaná tvorba softvéru, …
* je to univerzálny programovací jazyk, ktorý poskytuje prostriedky na tvorbu moderných aplikácií, takých ako analýza dát, spracovanie médií, sieťové aplikácie a pod.
* Python má obrovskú komunitu programátorov a expertov, ktorí sú ochotní svojimi radami pomôcť aj začiatočníkom

Naštartujeme Python

Ako ho získať

* zo stránky <https://www.python.org/> stiahnete najnovšiu verziu Pythonu - momentálne je to verzia **3.9.0**
* spustite inštalačný program (podľa typu operačného systému napríklad python-3.9.0.exe)
* POZOR! Nesťahujte verziu začínajúcu 2 (napríklad 2.7.16) - tá nie je kompatibilná s verziou 3.x
* pri inštalácii odporúčame nastaviť:
  + for all users
  + add Python 3.9 to PATH
  + inštalovať pip, tkinter aj IDLE

Alternatívne vývojové prostredia

* Inštalácia najnovšej verzie Pythonu obsahuje aj vývojové prostredie **IDLE**. Toto prostredie je pre úplného začiatočníka ideálne a budeme ho používať aj v tomto kurze.
* Pre skúsenejších odporúčame vybrať si jedno z týchto prostredí (mali by ste už mať nainštalovaný Python)
  + [PyCharm Edu - Easy and Professional Tool to Learn & Teach Programming with Python](https://www.jetbrains.com/pycharm-edu/)
  + [Wing Personal - A free Python IDE for students and hobbyists](https://wingware.com/downloads/wing-personal)

Spustíme IDLE

**IDLE (Python GUI)** je vývojové prostredie (Integrated Development Learning Environment), vidíme informáciu o verzii Pythonu a tiež riadok s tromi znakmi >>> (tzv. výzva, t.j. **prompt**). Za túto výzvu budeme písať príkazy pre Python.

#### Ako to funguje

* Python je interpreter a pracuje v niekoľkých možných režimoch
* teraz sme ho spustili v **interaktívnom režime** (niekedy hovoríme aj príkazový režim): očakáva zadávanie textových príkazov (do riadka za znaky >>>), každý zadaný príkaz vyhodnotí a vypíše prípadnú reakciu (alebo **chybovú správu**, ak sme zadali niečo nesprávne)
* po skončení vyhodnocovania riadka sa do ďalšieho riadka znovu vypíšu znaky >>> a očakáva sa opätovné zadávanie ďalšieho príkazu
* takémuto interaktívnemu oknu hovoríme **shell**
* niekedy sa môžete dočítať aj o tzv. REP cykle interpretera, znamená to **Read**, **Evaluate**, **Print**, teda prečítaj, potom tento zápis vyhodnoť a na koniec vypíš výsledok, toto celé stále opakuj

Celočíselné operácie

* oba operandy musia byť celočíselného typu

|  |  |  |
| --- | --- | --- |
| + | sčitovanie | 1 + 2 má hodnotu 3 |
| - | odčitovanie | 2 - 5 má hodnotu -3 |
| \* | násobenie | 3 \* 37 má hodnotu 111 |
| // | celočíselné delenie, | 22 // 7 má hodnotu 3 |
| % | zvyšok po delení | 22 % 7 má hodnotu 1 |
| \*\* | umocňovanie | 2 \*\* 8 má hodnotu 256 |

* zrejme nemôžeme deliť 0

Operácie s desatinnými číslami

* aspoň jeden z operandov musí byť desatinného typu (okrem delenia /)

|  |  |  |
| --- | --- | --- |
| + | sčitovanie | 1 + 0.2 má hodnotu 1.2 |
| - | odčitovanie | 6 - 2.86 má hodnotu 3.14 |
| \* | násobenie | 1.5 \* 2.5 má hodnotu 3.75 |
| / | delenie | 23 / 3 má hodnotu 7.666666666666667 |
| // | delenie zaokrúhlené nadol | 23.0 // 3 má hodnotu 7.0 |
| % | zvyšok po delení | 23.0 % 3 má hodnotu 2.0 |
| \*\* | umocňovanie | 3 \*\* 3. má hodnotu 27.0 |

* zrejme nemôžeme deliť 0

Operácie so znakovými reťazcami

|  |  |  |
| --- | --- | --- |
| + | zreťazenie (spojenie dvoch reťazcov) | 'a' + 'b' má hodnotu 'ab' |
| \* | viacnásobné zreťazenie reťazca | 3 \* 'x' má hodnotu 'xxx' |

**Prikazy**

**print()**

* uvidíme neskôr, že je to volanie špeciálnej funkcie
* táto funkcia vypisuje hodnoty výrazov, ktoré sú uvedené medzi zátvorkami
* hodnoty sú pri výpise oddelené medzerami
* print() bez parametrov spôsobí len zariadkovanie výpisu, teda vloží na momentálne miesto prázdny riadok

**input()**

* je tiež funkcia, ktorá najprv vypíše zadaný znakový reťazec (ak je zadaný) a potom čaká na vstupný reťazec ukončený ENTER
* funkcia vráti tento nami zadaný znakový reťazec

**Úprava pythonovského programu**

Programátori majú medzi sebou dohodu, ako správne zapisovať pythonovský kód (oficiálny dokument je [PEP 8](https://www.python.org/dev/peps/pep-0008/)). My zatiaľ píšeme len veľmi jednoduché zápisy, ale je dôležité si zvykať už od začiatku na správne zápisy. Takže niekoľko základných pravidiel:

* mená premenných obsahujú len malé písmená
* pre znak = v priraďovacom príkaze dávame medzeru pred aj za
* operácie v aritmetických výrazoch sú väčšinou tiež oddelené od operandov medzerami
* riadky programu by nemali byť dlhšie ako 79 znakov
* za čiarky, ktoré napríklad oddeľujú parametre v príkaze print(), dávame vždy medzeru

Postupne sa budeme zoznamovať aj s ďalšími takýmito odporúčaniami.

**pretypovanie hodnôt**

Mená typov int, float a str zároveň súžia ako mená pretypovacích funkcií, ktoré dokážu z jedného typu vyrobiť hodnotu iného typu:

* int(hodnota) z danej hodnoty vyrobí celé číslo, napríklad:
  + int(3.14) => 3
  + int('37') => 37
* float(hodnota) z danej hodnoty vyrobí desatinné číslo, napríklad:
  + float(333) => 333.0
  + float('3.14') => 3.14
* str(hodnota) z danej hodnoty vyrobí znakový reťazec, napríklad:
  + str(356) => '356'
  + str(3.14) => '3.14'

Veľa našich programov bude začínať načítaním niekoľkých vstupných hodnôt. Podľa typu požadovanej hodnoty môžeme prečítaný reťazec hneď prekonvertovať na správny typ, napríklad takto:

cele = int(input('zadaj celé číslo: ')) *# konverovanie na celé číslo*

desatinne = float(input('zadaj desatinné číslo: ')) *# konverovanie na desatinné číslo*

retazec = input('zadaj znakový reťazec: ') *# reťazec netreba konvertovať*

**Úprava pythonovského programu**

Programátori majú medzi sebou dohodu, ako správne zapisovať pythonovský kód (oficiálny dokument je [PEP 8](https://www.python.org/dev/peps/pep-0008/)). My zatiaľ píšeme len veľmi jednoduché zápisy, ale je dôležité si zvykať už od začiatku na správne zápisy. Takže niekoľko základných pravidiel:

* mená premenných obsahujú len malé písmená
* pre znak = v priraďovacom príkaze dávame medzeru pred aj za
* operácie v aritmetických výrazoch sú väčšinou tiež oddelené od operandov medzerami
* riadky programu by nemali byť dlhšie ako 79 znakov
* za čiarky, ktoré napríklad oddeľujú parametre v príkaze print(), dávame vždy medzeru

Postupne sa budeme zoznamovať aj s ďalšími takýmito odporúčaniami.

V Pythone je zadefinovaných niekoľko štandardných funkcií, ktoré pracujú s číslami. Ukážeme si dve z nich: výpočet absolútnej hodnoty a zaokrúhľovaciu funkciu:

**abs() absolútna hodnota**

**abs(*cislo*)**

**Parametre**

**cislo** – celé alebo desatinné číslo

Funkcia abs(cislo) vráti absolútnu hodnotu zadaného čísla, napríklad:

* abs(13) => 13
* abs(-3.14) => 3.14
* Ďalšia funkcia help() nám niekedy môže pomôcť v jednoduchej nápovedi k niektorým funkciám a tiež typom. Ako parameter pošleme buď meno funkcie, alebo hodnotu nejakého typu:
* **>>>** help(abs)
* Help on built-in function abs in module builtins:
* abs(...)
* abs(number) -> number
* Return the absolute value of the argument.
* **>>>** help(0)
* Help on int object:
* class int(object)
* | int(x=0) -> integer
* | int(x, base=10) -> integer
* **...**

**round() zaokrúhľovanie čísla**

**round(*cislo*)**

**round(*cislo*, *pocet*)**

**Parametre**

* **cislo** – celé alebo desatinné číslo
* **pocet** – celé číslo, ktoré vyjadruje na koľko desatinných miest sa bude zaokrúhľovať; ak je to záporné číslo, zaokrúhľuje sa na počet mocnín desiatky

Funkcia round(cislo) vráti zaokrúhlenú hodnotu zadaného čísla na celé číslo. Funkcia round(cislo, pocet) vráti zaokrúhlené číslo na príslušný počet desatinných miest, napríklad:

* round(3.14) => 3
* round(-0.74) => -1
* round(3.14, 1) => 3.1
* round(2563, -2) => 2600

### Znakové reťazce

Ak znakový reťazec obsahuje dvojicu znakov '\n', tieto označujú, že pri výpise funkciou print() sa namiesto nich prejde na nový riadok. Napríklad:

**>>>** a = 'prvý riadok**\n**stredný**\n**tretí riadok'

**>>>** a

'prvý riadok\nstredný\ntretí riadok'

**>>>** print(a)

prvý riadok

stredný

tretí riadok

Takáto dvojica znakov '\n' zaberá v reťazci len jeden znak.

Python umožňuje pohodlnejšie vytvárania takýchto „viacriadkových“ reťazcov. Ak reťazec začína tromi apostrofmi ''' (alebo úvodzovkami """), môže prechádzať aj cez viac riadkov, ale opäť musí byť ukončený rovnakou trojicou, ako začal. Prechody na nový riadok v takomto reťazci sa nahradia špeciálnym znakom '\n'. Napríklad:

**>>>** ab = '''prvý riadok

stredný

tretí riadok'''

**>>>** ab

'prvý riadok\nstredný\ntretí riadok'

**>>>** print(ab)

prvý riadok

stredný

tretí riadok

Takýto reťazec môže obsahovať aj apostrofy a úvodzovky.

Niekedy potrebujeme vytvárať znakový reťazec pomocou komplikovanejšieho zápisu, v ktorom ho budeme skladať (zreťaziť) z viacerých hodnôt, napríklad:

**>>>** meno, x, y = 'A', 180, 225

**>>>** r = 'bod ' + meno + ' na súradniciach (' + str(x) + ',' + str(y) + ')'

**>>>** r

'bod A na súradniciach (180,225)'

Python poskytuje špeciálny typ reťazca (tzv. formátovací znakový reťazec), pomocou ktorého môžeme vytvárať aj takto komplikované výrazy. Základom je formátovacia šablóna, do ktorej budeme vkladať ľubovoľné aj číselné hodnoty. V našom prípade bude šablónou reťazec f'bod {meno} na súradniciach ({x},{y})'. V tejto šablóne sa každá dvojica znakov {...} nahradí príslušnou hodnotou, v našom prípade týmito hodnotami sú postupne meno, x, y. Všimnite si, znak f pred začiatkom reťazca. Zápis takejto formátovacej metódy bude:

**>>>** meno, x, y = 'A', 180, 225

**>>>** r = f'bod *{*meno*}* na súradniciach (*{*x*}*,*{*y*}*)'

**>>>** r

'bod A na súradniciach (180,225)'

V Pythone existuje aj špeciálny typ funkcie (tzv. metódu znakového reťazca), aj nej môžeme vytvárať takto komplikované výrazy. Základom je opäť formátovacia šablóna, do ktorej budeme vkladať ľubovoľné aj číselné hodnoty. V našom prípade bude šablónou reťazec 'bod {} na súradniciach ({},{})'. V tejto šablóne sa každá dvojica znakov {} nahradí nejakou konkrétnou hodnotou, v našom prípade týmito hodnotami sú postupne meno, x, y. Zápis takejto formátovacej metódy bude:

**>>>** meno, x, y = 'A', 180, 225

**>>>** r = 'bod *{}* na súradniciach (*{}*,*{}*)'.format(meno, x, y)

**>>>** r

'bod A na súradniciach (180,225)'

To znamená, že za reťazec šablóny píšeme znak bodka a hneď za tým volanie funkcie format() s hodnotami, ktoré sa do šablóny dosadia (zrejme ich musí byť rovnaký počet ako dvojíc {}). Neskôr sa zoznámime aj s ďalšími veľmi užitočnými špecialitami takéhoto formátovania.