

Základy programovania

Poznámky k téme

V tejto téme napíšeš prvé programy v Pythone. Zistíš, ako počítač program vykonáva, ako niečo vypísať na obrazovku, ako si do programu písať poznámky a hlavne – ako čítať chybové hlásenia, ktoré ti Python ukáže, keď sa niečo pokazí. Chyby sú pri programovaní úplne bežné; dôležité je rozumieť, čo ti hovoria.

Čo sa naučíš

- čo je program a v akom poradí ho počítač vykonáva,
- vypisovať text aj čísla príkazom `print`,
- písať komentáre a vypínať nimi riadky programu,
- prečítať chybové hlásenie a podľa neho chybu nájsť a opraviť.

1 Program a jeho spustenie

Program je postupnosť príkazov, ktoré počítač vykoná. Príkazy musia byť zapísané presne – počítač nedomýšľa, čo sme „asi chceli“. Postup riešenia úlohy, ktorý je presný a jednoznačný, sa volá **algoritmus**; program je algoritmus zapísaný v programovacím jazyku.

Python je programovací jazyk, ktorý sa dobre číta a má jednoduchý zápis. Program v Pythone vykonáva **interpret**: číta program a vykonáva príkazy jeden po druhom. Program je obyčajný textový súbor s príponou `.py` (napríklad `prvy_program.py`). Môžeš ho písať v prostredí IDLE, Thonny alebo VS Code; na Mente aj priamo v prehliadači v časti Pieskovisko.

1.1 Poradie príkazov

Príkazy sa vykonávajú **zhora nadol**, v poradí, v akom sú napísané. Každý príkaz píšeme na samostatný riadok.

```
print("Dobré ráno!")  
print("Dnes sa učíme Python.")  
print("Koniec programu.")
```

Výstup:

```
Dobré ráno!  
Dnes sa učíme Python.  
Koniec programu.
```

Ak riadky prehodíš, zmení sa aj výstup. Počítač robí presne to, čo je napísané, v presne tom poradí.

1.2 Interaktívny režim a program v súbore

Python sa dá používať dvoma spôsobmi:

- **Interaktívny režim (shell)** – za výzvu `>>>` napíšeš jeden príkaz alebo výraz a Python hneď ukáže výsledok. Hodí sa na rýchle vyskúšanie.
- **Program v súbore** – napíšeš celý program, uložíš ho a spustíš. Takto vznikajú skutočné programy a takto riešiš úlohy.

Ukážka interaktívneho režimu (riadky s `>>>` píšeš ty, pod nimi je odpoveď Pythonu):

```
>>> 2 + 3
5
>>> 10 * 4 - 2
38
>>> "Ahoj"
'Ahoj'
```

Text shell zobrazí aj s úvodzovkami (`'Ahoj'`), aby bolo vidno, že ide o text; `print` ho vypíše bez nich.

Pozor: V interaktívnom režime sa výsledok výrazu ukáže sám. V programe zo súboru sa nevypíše nič, kým nepoužiješ `print`. Riadok `2 + 3` v programe síce vypočíta 5, ale výsledok sa nikde neobjaví.

2 Výpis pomocou print()

Príkaz `print` vypíše na obrazovku to, čo mu dáš do zátvoriek. Po vypísaní prejde na nový riadok.

2.1 Text a čísla

Text musí byť v úvodzovkách – v programovaní sa mu hovorí **ret'azec** (anglicky *string*). Čísla píšeme bez úvodzoviek.

```
print("Ahoj, svet!")
print(42)
print(3.5)
print(2 + 3 * 4)
print("2 + 3 * 4")
```

Výstup:

```
Ahoj, svet!
42
3.5
14
2 + 3 * 4
```

Všimni si posledné dva riadky: bez úvodzoviek Python výraz **vypočíta** (násobenie má prednosť, takže `2 + 12 = 14`), v úvodzovkách je to iba **text**, ktorý sa vypíše presne tak, ako je napísaný.

Desatinné čísla sa v Pythone píšu s **bodkou**, nie s čiarkou: `3.5`, nie `3,5`.

2.2 Úvodzovky

Text môže byť v dvojitéch `"..."` aj v jednoduchých `'...'` úvodzovkách – na začiatku a na konci však musia byť rovnaké. Druhý druh úvodzoviek sa hodí, keď potrebuješ úvodzovky vo vnútri textu:

```
print('Povedala: "Dnes píšeme test."')
print("Rock'n'roll")
```

Výstup:

```
Povedala: "Dnes píšeme test."
Rock'n'roll
```

2.3 Viac hodnôt v jednom print

Do `print` môžeš dať viac hodnôt oddelených čiarkou. Vypíšu sa na jeden riadok a medzi každé dve sa automaticky vloží **medzera**:

```
print("Mám", 16, "rokov.")  
print("Výsledok:", 7 * 6)  
print(1, 2, 3)
```

Výstup:

```
Mám 16 rokov.  
Výsledok: 42  
1 2 3
```

Prázdný `print()` vypíše prázdny riadok – hodí sa na oddelenie častí výstupu.

2.4 Nový riadok vo vnútri textu

Dvojica znakov `\n` znamená v texte **nový riadok**. Takýmto dvojiciam so spätnou lomkou sa hovorí *escape sekvencie*. Ďalšia užitočná je `\t` (tabulátor):

```
print("Prvý riadok\nDruhý riadok")  
print("Meno:\tEma")
```

Výstup:

```
Prvý riadok  
Druhý riadok  
Meno:   Ema
```

Dlhší text na viac riadkov sa dá zapísať aj medzi **trojité úvodzovky** – riadky sa vypíšu presne tak, ako sú napísané. To sa hodí napríklad na jednoduché obrázky zo znakov:

```
print("""  
  /\_/\   
 (  o.o  )  
  > ^ <  
""")
```

Výstup:

```
 /\_/\   
(  o.o  )  
> ^ <
```

Tip: Chceš vypísať spätnú lomku `\` ? V texte ju musíš napísať dvakrát: `\\` . Preto je v obrázku mačky zapísané `/\_\/\` .

2.5 Ovplyvnenie medzier a koncov riadkov

`print` má dve nastavenia, ktoré sa často hodia:

- `sep` – čo sa vloží **medzi** hodnoty (predvolene medzera),
- `end` – čo sa vypíše **na konci** (predvolene nový riadok).

```
print(1, 2, 3, sep=" ")
print("Na", "jeden", "riadok", end=" ")
print("pokračujem tu.")
```

Výstup:

```
1, 2, 3
Na jeden riadok pokračujem tu.
```

Podrobnejšie sa výstupu venuje téma [Vstup a výstup](#).

3 Komentáre

Komentár je poznámka pre človeka, ktorý program číta. Začína znakom `#` a pokračuje až do konca riadku. Python komentáre úplne ignoruje.

```
# Program vypíše pozdrav a výpočet.
print("Ahoj!")          # komentár môže byť aj za príkazom
# print("Tento riadok je vypnutý.")
print("Súčet:", 3 + 4)
```

Výstup:

```
Ahoj!
Súčet: 7
```

Na čo sa komentáre používajú:

- **vysvetlenie**, prečo je niečo v programe tak, ako je (najmä pri zložitejších miestach),
- **štruktúra** – nadpisy častí programu (načítanie, výpočet, výpis),
- **dočasné vypnutie riadku** („zakomentovanie“) pri hľadaní chyby alebo skúšaní.

Dobrý komentár vysvetľuje *prečo* niečo robíme, nie čo je zrejmé zo samotného kódu. Komentár `# vypíše Ahoj` nad riadkom `print("Ahoj")` nič nové nepovie.

Pozor: Znak `#` vo vnútri úvodzoviek nie je komentár, ale obyčajný znak textu: `print("# nie je komentár")` vypíše aj mriežku.

4 Chybové hlásenia

Každý, kto programuje, robí chyby. Python pri chybe program zastaví a vypíše **chybové hlásenie** – povie, kde chyba nastala a akého druhu je. Rozlišujeme tri druhy chýb:

Druh chyby	Kedy sa prejaví	Príklad
syntaktická chyba (<code>SyntaxError</code>)	ešte pred spustením – program sa vôbec nespustí	chýba úvodzovka alebo zátvorka
chyba počas behu	program beží, kým nepríde na chybný riadok	neznáme meno, delenie nulou
logická chyba	nikdy – program beží, ale robí niečo iné, než chceme	<code>+</code> namiesto <code>*</code> vo vzorci

Logickú chybu Python nenájde, lebo program je „správne napísaný“ – len nerobí to, čo sme chceli. Tú musíš nájsť ty, napríklad porovnaním výstupu so správnym výsledkom.

4.1 Ako čítať chybové hlásenie

Pozri si program s chybou na treťom riadku:

```
print("Začiatok programu")
print("Výpočet:", 5 * 3)
print(vysledok)
print("Koniec programu")
```

Výstup (program skončí chybou):

```
Začiatok programu
Výpočet: 15
Traceback (most recent call last):
  File "program.py", line 3, in <module>
    print(vysledok)
          ^^^^^^^
NameError: name 'vysledok' is not defined
```

Hlásenie sa číta takto:

1. `Traceback (most recent call last):` – úvodný riadok hlásenia.
2. `File "program.py", line 3` – chyba je v súbore `program.py` na **riadku 3**. Pod tým je vypísaný samotný chybný riadok.
3. Posledný riadok je najdôležitejší: **druh chyby** (`NameError`) a **správa** – meno `vysledok` nie je definované. Python nevie, čo je `vysledok`, lebo také meno sme nikde nevytvorili. Keby to mal byť text, patril by do úvodzoviek.

Všimni si aj to, že prvé dva riadky sa vypísali. Chyba počas behu zastaví program až na chybnom riadku; posledný `print` sa už nevykoná.

4.2 Najčastejšie chyby na začiatku

Chýbajúca úvodzovka. Text nie je ukončený:

```
print("Ahoj)
```

Výstup (program skončí chybou):

```
File "program.py", line 1
  print("Ahoj)
    ^
SyntaxError: unterminated string literal (detected at line 1)
```

Neuzavretá zátvorka. Python si to všimne až na ďalšom riadku – chyba preto môže byť aj o riadok vyššie, než hovorí hlásenie:

```
print("Ahoj"
print("svet")
```

Výstup (program skončí chybou):

```
File "program.py", line 1
  print("Ahoj"
    ^
SyntaxError: '(' was never closed
```

Veľké písmeno alebo preklep v mene príkazu. Python rozlišuje veľké a malé písmená: `Print` nie je to isté ako `print`. Novšie verzie Pythonu dokonca poradia, čo sme asi mysleli:

```
Print("Ahoj")
```

Výstup (program skončí chybou):

```
Traceback (most recent call last):
  File "program.py", line 1, in <module>
    Print("Ahoj")
    ^^^^^
NameError: name 'Print' is not defined
```

Text bez úvodzoviek. Python ho považuje za meno premennej, ktorá neexistuje:

```
print(Ahoj)
```

Výstup (program skončí chybou):

```
Traceback (most recent call last):
  File "program.py", line 1, in <module>
    print(Ahoj)
    ^^^^^
NameError: name 'Ahoj' is not defined
```

Medzera na začiatku riadku. Odsadenie (medzery na začiatku riadku) má v Pythone význam – smie byť iba tam, kde to pravidlá jazyka vyžadujú (napríklad v podmienkach a cykloch):

```
print("Ahoj")
  print("svet")
```

Výstup (program skončí chybou):

```
File "program.py", line 2
  print("svet")
IndentationError: unexpected indent
```

Spájanie textu s číslom. Operátor `+` vie spojiť dva texty alebo sčítať dve čísla, ale nie text s číslom:

```
print("Mám " + 16 + " rokov")
```

Výstup (program skončí chybou):

```
Traceback (most recent call last):
  File "program.py", line 1, in <module>
    print("Mám " + 16 + " rokov")
              ~~~~~^~
TypeError: can only concatenate str (not "int") to str
```

Oprava: použiť čiarky (`print("Mám", 16, "rokov")`) alebo číslo zapísať ako text (`"16"`).

Delenie nulou.

```
print(10 / 0)
```

Výstup (program skončí chybou):

```
Traceback (most recent call last):
  File "program.py", line 1, in <module>
    print(10 / 0)
    ~~~~^~~
ZeroDivisionError: division by zero
```

Prehľad najčastejších hlásení:

Hlásenie	Čo zvyčajne znamená
<code>SyntaxError: unterminated string literal</code>	chýba úvodzovka na konci textu
<code>SyntaxError: '(' was never closed</code>	chýba zatváracia zátvorka
<code>NameError: name '...' is not defined</code>	preklep v mene, chýbajúce úvodzovky alebo premenná, ktorá ešte neexistuje
<code>IndentationError: unexpected indent</code>	medzera alebo tabulátor na začiatku riadku navyše
<code>TypeError: can only concatenate str (not "int") to str</code>	spájanie textu s číslom cez <code>+</code>
<code>ZeroDivisionError: division by zero</code>	delenie nulou

4.3 Postup pri hľadaní chyby

1. Prečítaj **posledný riadok** hlásenia – druh chyby a správu.
2. Pozri sa na **číslo riadku**. Ak je na tom riadku všetko v poriadku, skontroluj riadok nad ním (často chýba zátvorka alebo úvodzovka).
3. Oprav **jednu** chybu a program spusti znova. Ďalšie hlásenia môžu byť len dôsledkom prvej chyby.
4. Pri logickej chybe si vypisuj **medzivýsledky** (`print`) a porovnaj ich s tým, čo si vypočítaš na papieri.

5 Riešené úlohy

5.1 Vizitka

Zadanie: Vypíš vizitku s menom, triedou a školou. Vizitka má byť ohraničená rámčekom zo znakov.

Riešenie: Každý riadok rámčeka je jeden `print`. Aby bol rámček rovný, musia mať riadky rovnakú dĺžku – medzery doplníme ručne.

```
print("+-----+")
print("| Ema Nováková      |")
print("| trieda: 3.A         |")
print("| Gymnázium Mente     |")
print("+-----+")
```

Výstup:

```
+-----+
| Ema Nováková      |
| trieda: 3.A         |
| Gymnázium Mente     |
+-----+
```

5.2 Stromček z hviezdíček

Zadanie: Vypíš stromček z hviezdíček s kmeňom.

Riešenie: Medzery na začiatku riadkov posúvajú hviezdčky do stredu.

```
print("    *")
print("    ***")
print("    *****")
print("    *****")
print("    |")
```

Výstup:

```
    *
    ***
    *****
    *****
    |
```

Písanie si môžeš skrátiť: text sa dá „vynásobiť“ číslom – `"*" * 5` je päť hviezdčiek. Podrobne o tom v téme Premenné a dátové typy.

```
print(" " * 3 + "*" * 3)
print(" " * 2 + "*" * 5)
```

Výstup:

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

5.3 Koľko sekúnd má deň?

Zadanie: Vypočítaj, koľko sekúnd má jeden deň a koľko jeden týždeň.

Riešenie: Deň má 24 hodín, hodina 60 minút, minúta 60 sekúnd. Python vypočíta výraz za nás; text pred výsledkom vysvetľuje, čo číslo znamená.

```
print("Deň má", 24 * 60 * 60, "sekúnd.")
print("Týždeň má", 7 * 24 * 60 * 60, "sekúnd.")
```

Výstup:

```
Deň má 86400 sekúnd.
Týždeň má 604800 sekúnd.
```

5.4 Oprav program

Zadanie: Program má vypísať pozdrav a súčet čísel 15 a 27, ale obsahuje tri chyby. Nájdi ich.

```
print("Ahoj, ideme počítať!")
Print("Súčet je:", 15 + 27)
print("Hotovo")
```

Riešenie: Spustenie ukáže najprv prvú chybu – na riadku 1 chýba ukončovacia úvodzovka. Po jej oprave Python nájde chýbajúcu zátvorku na riadku 3 (syntaktické chyby sa hlásia pred spustením) a po ďalšej oprave `NameError` pre `Print` s veľkým P. Opravený program:

```
print("Ahoj, ideme počítať!")  
print("Súčet je:", 15 + 27)  
print("Hotovo")
```

Výstup:

```
Ahoj, ideme počítať!  
Súčet je: 42  
Hotovo
```

Zhrnutie

- Program je postupnosť príkazov; Python ich vykonáva **zhora nadol**.
- `print(...)` vypíše hodnoty; viac hodnôt oddeľ čiarkou (vloží sa medzera), `print()` vypíše prázdny riadok.
- Text patrí do úvodzoviek `"..."` alebo `'...'`, čísla nie; desatinné čísla píš s bodkou.
- `\n` je nový riadok v texte; `sep` a `end` menia oddeľovač a koniec výpisu.
- Komentár začína `#` – Python ho ignoruje.
- Chybové hlásenie čítaj od **posledného riadku** (druh chyby a správa), potom pozri **číslo riadku**.
- Python rozlišuje veľké a malé písmená a medzery na začiatku riadku majú význam.

Úlohy na precvičenie

1. Vypíš svoje meno, vek a obľúbený predmet – každý údaj na samostatný riadok.
2. Vypíš jedným príkazom `print` čísla 1 až 5 tak, aby boli oddelené pomlčkou: `1-2-3-4-5`.
3. Nakresli zo znakov domček (strecha z `/` a `\`, steny z `|`).
4. Vypočítaj, koľko minút má rok (365 dní), a výsledok vypíš aj s vysvetľujúcim textom.
5. Napíš program so zámernou chybou každého druhu z tabuľky a prečítaj si hlásenia.

Úlohy k tejto téme nájdeš na learnwithmente.com v časti Informatika → Python.