

Množiny

Poznámky k téme

Množina (typ `set`) je skupina hodnôt **bez opakovania** a **bez poradia** – presne ako množina v matematike. Hodí sa vždy, keď nás zaujíma, či niečo v skupine je, a nie *koľkokrát* alebo *na ktorom mieste*: rôzne písmená v texte, unikátni návštevníci, spoločné prvky dvoch zoznamov. Zisťovanie, či je hodnota v množine, je navyše veľmi rýchle.

Čo sa naučíš

- vytvoriť množinu a pridávať či odoberať prvky,
- zistiť príslušnosť prvku operátorom `in`,
- počítať zjednotenie, prienik a rozdiel množín,
- použiť množinu na odstránenie duplikátov a porovnávanie skupín.

1 Vytvorenie množiny

Množinu zapíšeme do zložených zátvoriek. Opakujúce sa hodnoty sa uložia iba **raz**:

```
>>> farby = {"červená", "modrá", "červená", "zelená"}
>>> len(farby)
3
>>> cisla = {3, 1, 3, 2, 1}
>>> cisla
{1, 2, 3}
```

Z iných postupností vytvorí množinu funkcia `set`. To je najjednoduchší spôsob, ako sa zbaviť duplikátov:

```
>>> set([1, 2, 2, 3, 3, 3])
{1, 2, 3}
>>> len(set("abakadabra"))
5
>>> sorted(set("abakadabra"))
['a', 'b', 'd', 'k', 'r']
```

Pozor: Prázdna množina sa vytvorí iba ako `set()`. Zápis `{}` vytvorí prázdny **slovník** (téma Slovníky).

1.1 Množina nemá poradie

Prvky množiny nemajú pevné poradie ani indexy. Pri výpise sa môžu zobrazit' v inom poradí, než v akom sme ich zapísali – a na inom počítači aj v inom poradí ako tu. Ak potrebuješ poradie, použi `sorted(mnozina)`, ktorá vráti zoradený zoznam.

```
>>> m = {"jablko", "hruška", "slivka"}
>>> sorted(m)
['hruška', 'jablko', 'slivka']
>>> m[0]
TypeError: 'set' object is not subscriptable
```

1.2 Čo môže byť v množine

Prvkami množiny môžu byť iba **nemenné** hodnoty: čísla, reťazce, n-tice. Zoznam do množiny vložiť nemožno:

```
>>> {(1, 2), (3, 4)}
{(1, 2), (3, 4)}
>>> {[1, 2]}
TypeError: unhashable type: 'list'
```

2 Pridávanie a odoberanie

Metóda	Čo urobí
<code>m.add(x)</code>	pridá prvok <code>x</code> (ak už tam je, nič sa nestane)
<code>m.remove(x)</code>	odstráni <code>x</code> ; ak tam nie je, chyba <code>KeyError</code>
<code>m.discard(x)</code>	odstráni <code>x</code> , ak tam je; inak nič
<code>m.clear()</code>	vyprázdni množinu

```
navstevnici = set()
for meno in ["Ema", "Ján", "Ema", "Sofia", "Ján"]:
    navstevnici.add(meno)
print(len(navstevnici), "rôzni návštevníci:", sorted(navstevnici))
navstevnici.discard("Tomáš")      # nebol tam – nič sa nestane
navstevnici.remove("Ján")
print(sorted(navstevnici))
```

Výstup:

```
3 rôzni návštevníci: ['Ema', 'Ján', 'Sofia']
['Ema', 'Sofia']
```

3 Príslušnosť: in

Operátor `in` (a `not in`) zistí, či prvok v množine je:

```
samohlasky = {"a", "e", "i", "o", "u", "y"}
slovo = "programovanie"
for znak in slovo:
    if znak in samohlasky:
        print(znak, end="")
print()
```

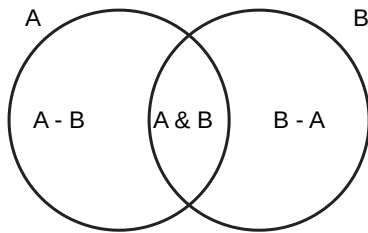
Výstup:

```
oaoaie
```

Pri zozname musí Python pri `in` prechádzať prvky jeden po druhom. Množina si prvky ukladá tak, že odpoveď nájde **takmer okamžite** bez ohľadu na veľkosť. Ak sa v programe veľakrát pýtaš, či je niečo v skupine, ulož skupinu do množiny. O rýchlosti programov viac v téme Algoritmické úlohy.

4 Množinové operácie

Množinové operácie poznáš z matematiky. V Pythone sa zapisujú operátormi (alebo metódami s rovnakým významom):



Operácia	Operátor	Metóda	Výsledok obsahuje prvky, ktoré sú...
zjednotenie	<code>a b</code>	<code>a.union(b)</code>	aspoň v jednej z množín
prienik	<code>a & b</code>	<code>a.intersection(b)</code>	v oboch množinách
rozdiel	<code>a - b</code>	<code>a.difference(b)</code>	v <code>a</code> , ale nie v <code>b</code>
symetrický rozdiel	<code>a ^ b</code>	<code>a.symmetric_difference(b)</code>	práve v jednej z nich

```
>>> a = {1, 2, 3, 4}
>>> b = {3, 4, 5, 6}
>>> a | b
{1, 2, 3, 4, 5, 6}
>>> a & b
{3, 4}
>>> a - b
{1, 2}
>>> b - a
{5, 6}
>>> a ^ b
{1, 2, 5, 6}
```

Porovnávať možno aj celé množiny: `a <= b` znamená „`a` je podmnožinou `b`“ (každý prvok `a` je aj v `b`).

```
>>> {1, 2} <= {1, 2, 3}
True
>>> {1, 4} <= {1, 2, 3}
False
>>> {3, 2, 1} == {1, 2, 3}
True
```

Dve množiny sa rovnajú, ak majú rovnaké prvky – na poradí zápisu nezáleží.

5 Prechádzanie množiny

Cyklus `for` prejde všetky prvky, ale v nepredvídateľnom poradí. Ak na poradí záleží (napríklad pri výpise), prechádzaj `sorted(m)`:

```
znamky = {3, 1, 2}
for z in sorted(znamky):
    print("známka", z)
```

Výstup:

```
známka 1
známka 2
známka 3
```

6 Riešené úlohy

6.1 Rôzne písmená v texte

Zadanie: Zisti, koľko rôznych písmen obsahuje veta, a vypíš ich abecedne. Zisti aj, ktoré písmená anglickej abecedy v nej chýbajú.

Riešenie: Množina písmen vznikne z textu po odstránení medzier a interpunkcie. Chýbajúce písmená sú rozdiel celej abecedy a písmen vety.

```
veta = "Kde bolo, tam bolo, bolo raz jedno malé mestečko."
pismena = set()
for znak in veta.lower():
    if znak.isalpha():
        pismena.add(znak)
print(len(pismena), "rôznych písmen:", "".join(sorted(pismena)))

abeceda = set("abcdefghijklmnopqrstuvwxyz")
print("chýbajú:", "".join(sorted(abeceda - pismena)))
```

Výstup:

```
16 rôznych písmen: abdeijklmnorstzč
chýbajú: cfghipquvwxy
```

Veta, ktorá obsahuje všetky písmená abecedy, sa volá *pangram* – pre ňu by rozdiel vyšiel prázdny.

6.2 Lotéria

Zadanie: Vyžrebuj 6 rôznych čísel od 1 do 49. Potom ich porovnaj so zadaným tipom a vypíš, koľko čísel tip uhádol.

Riešenie: Náhodné čísla pridávame do množiny, kým ich nie je 6 – opakované číslo sa do množiny nepridá, takže na konci máme istotu, že sú všetky rôzne. Počet uhádnutých čísel je veľkosť prieniku. (V ukážke je pevne nastavené `random.seed()`.)

```
import random

zrebovanie = set()
while len(zrebovanie) < 6:
    zrebovanie.add(random.randint(1, 49))
tip = {3, 11, 17, 25, 31, 42}
print("Vyžrebované:", sorted(zrebovanie))
print("Tip:", sorted(tip))
print("Uhádnuté:", sorted(zrebovanie & tip), "-", len(zrebovanie & tip), "čísla")
```

Výstup:

```
Vyžrebované: [9, 16, 24, 35, 38, 39]
Tip: [3, 11, 17, 25, 31, 42]
Uhádnuté: [] – 0 čísla
```

Modul `random` má na to aj hotovú funkciu: `random.sample(range(1, 50), 6)` vráti 6 rôznych čísel.

6.3 Spoločné záujmy

Zadanie: Dve osoby, A a B, vymenovali svoje záujmy. Vypíš spoločné záujmy, záujmy, ktoré má iba A, a všetky záujmy dohromady.

```
a = {"futbal", "hudba", "programovanie", "šach"}
b = {"hudba", "kreslenie", "programovanie", "plávanie"}
print("spoločné:", sorted(a & b))
print("iba A:", sorted(a - b))
print("všetky:", sorted(a | b))
```

Výstup:

```
spoločné: ['hudba', 'programovanie']
iba A: ['futbal', 'šach']
všetky: ['futbal', 'hudba', 'kreslenie', 'plávanie', 'programovanie', 'šach']
```

6.4 Ktoré hodnoty sa opakujú?

Zadanie: Vypíš hodnoty, ktoré sa v zozname vyskytujú viackrát.

Riešenie: Prejdeme zoznam a pamätáme si v množine hodnoty, ktoré sme už videli. Ak hodnotu už poznáme, je to opakovanie.

```
z = [4, 7, 1, 7, 9, 4, 4, 2]
videne = set()
opakovane = set()
for x in z:
    if x in videne:
        opakovane.add(x)
    else:
        videne.add(x)
print(sorted(opakovane))
```

Výstup:

```
[4, 7]
```

7 Časté chyby

- `{}` nie je prázdna množina, ale prázdny slovník – použi `set()`.
- Indexovanie množiny (`m[0]`) nie je možné – množina nemá poradie; preved ju na zoradený zoznam.
- Spoliehanie sa na poradie pri výpise – výsledok môže vyzerat inak; na výpis použi `sorted`.
- `remove` neexistujúceho prvku skončí chybou `KeyError`; použi `discard` alebo najprv `in`.
- Zoznam v množine – prvky musia byť nemenné (čísla, reťazce, n-tice).

Zhrnutie

- Množina `{a, b, c}` obsahuje každú hodnotu najviac raz a nemá poradie; prázdna je `set()`.
- `set(postupnosť)` odstráni duplikáty; `len` je počet rôznych hodnôt.
- `add`, `remove` (chyba pri chýbajúcom prvku), `discard` (bez chyby).
- `x in m` je veľmi rýchle – vhodné na časté zisťovanie príslušnosti.
- `a | b` zjednotenie, `a & b` prienik, `a - b` rozdiel, `a ^ b` symetrický rozdiel, `a <= b` podmnožina.
- Na výpis v poradí použi `sorted(m)`.

Úlohy na precvičenie

1. Načítaj vetu a vypíš počet rôznych slov (bez ohľadu na veľké a malé písmená).
2. Zisti, či je zadaná veta pangram (obsahuje všetky písmená anglickej abecedy).
3. Vygeneruj dve množiny po 10 náhodných čísel od 1 do 20 a vypíš ich prienik a zjednotenie.
4. Z dvoch zoznamov mien vypíš mená, ktoré sú iba v jednom z nich.
5. Načítaj čísla oddelené medzerou a zisti, či sú všetky rôzne (porovnaj dĺžku zoznamu a množiny).

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