# Тема 11. Множини

Тип множини використовують у задачах, у яких має значення тільки належність або неналежність елемента деякій множині. В основному реалізація множин у програмуванні повторює відомі операції та відношення для множин у математиці.

Множини, є такими, що змінюються (mutable).

Носій типу множина

Множина позначається включенням її елементів у фігурні дужки через кому.

{x1, …, xn}. Порожня множина позначається set(), щоб уникнути плутанини зі словниками.

Нехай множина *M* є носієм типу елементів t.

Тоді носієм типу множини елементів типу t буде

*Ms* = 2*M* – множина всіх підмножин множини *M*.

Основні операції для множин

|  |  |
| --- | --- |
| **Операція** | **Опис** |
| {x1, …, xn} | Створити множину з елементів x1, … , xn |
| set() | Порожня множина |
| set(x) | Перетворення *x* у множину (x повинно належати типу, що ітерується) |
| a | b | Об’єднання множин a U b |
| a & b | Перетин множин a ∩ b |
| a - b | Різниця множин a \ b |
| a ^ b | Симетрична різниця множин a та b |
| len(a) | Довжина *a –* кількість елементів у множині |
| min(a) | Найменший елемент множини *a* |
| max(a) | Найбільший елемент множини *a* |
| a.copy() | Повертає копію множини *a* |

Основні відношення для множин

Для множин визначено відношення з Rel = {==, !=, >, <, >=, <=} а також in, not in.

Відношення a == b означає рівність множин.

Відношення a != b ≡ not (a == b).

Відношення a < b означає включення множини a у множину b.

Відношення a <= b означає нестроге включення множини a у множину b.

Відношення a > b ≡ b < a.

Відношення a >= b ≡ b <= a.

x in a == True, коли x входить у a

x not in a == True, коли x не входить у a

Основні інструкції для множин

Для множин визначено присвоєння та виведення.

a **=** b**,** **print(**a**)**

Введення не визначено, тому треба вводити множину поелементно.

Визначено також цикл по всіх елементах множини

**for** x **in** a**:**

*P*

Приклад

Перевірити, чи складаються 2 рядки s1 та s2 з одних і тих же символів. Тобто, чи справедливе твердження, що кожний символ s1 входить у s2 та кожний символ s2 входить у s1.

Додаткові операції для множин

|  |  |
| --- | --- |
| **Операція** | **Опис** |
| a.union(b) | Об’єднання множин a U b, те ж саме, що й a | b |
| a.intersection(b) | Перетин множин a ∩ b, те ж саме, що й a & b |
| a.difference(b) | Різниця множин a \ b, те ж саме, що й a - b |
| a.symmetric\_difference(b) | Симетрична різниця множин a та b, те ж саме, що й a ^ b |

Додаткові відношення для множин

|  |  |
| --- | --- |
| **Відношення** | **Опис** |
| a.isdisjoint(b) | True, коли перетин a та b – порожня множина, те ж саме, що й a & b == set() |
| a.issubset(b) | Чи є a підмножиною b, те ж саме, що й a <= b |
| a.issuperset(b) | Чи є b підмножиною a, те ж саме, що й b <= a |

Додаткові інструкції для множин

|  |  |
| --- | --- |
| **Інструкція** | **Опис** |
| a.add(x) | Додає елемент x до множини a |
| a.remove(x) | Видаляє елемент x з множини a. Якщо елемента немає у множині, - дає помилку |
| a.discard(x) | Видаляє елемент x з множини a, якщо цей елемент є у множині |
| a.pop() | Видаляє довільний елемент з множини a. Якщо множина порожня та в ній немає жодного елемента, - дає помилку |
| a.clear() | Видаляє всі елементи з множини a |
| a.update(b) | Оновити a значенням об’єднання a та b. Те ж саме, що й a = a | b |
| a.intersection\_update(b) | Оновити a значенням перетину a та b. Те ж саме, що й a = a & b |
| a.difference\_update(b) | Оновити a значенням різниці a та b. Те ж саме, що й a = a - b |
| a.symmetric\_difference\_update(b) | Оновити a значенням симетричної різниці a та b. Те ж саме, що й a = a ^ b |

Незмінні множини frozenset

У Python окрім звичайних множин є також множини, що не змінюються (immutable), frozenset.

Для створення такої множини треба писати

frozenset(t), де t – вираз типу, що ітерується.

Для frozenset визначено всі ті ж основні операції, відношення та інструкції, що й для звичайних множин set. Також визначені додаткові операції та відношення, наведені вище. Не визначені тільки додаткові інструкції.

frozenset можуть фігурувати у виразах разом із звичайними множинами. При цьому результат виразу буде того типу, до якого належить перший операнд операції (set або frozenset).

frozenset використовують тоді, коли множина після створення не змінюється і потрібна більша швидкодія у порівнянні з використанням звичайних множин set.

Множиноутворення

Множиноутворення (set comprehension) – це вираз, результатом якого є множина. Множиноутворення схоже на спискоутворення та словникоутворення. Вираз має такий синтаксис:

**{***e***(**x**)** **for** x **in** *t* **if** *F***}**

де *e*(*x*) – вираз, *t* – вираз типу, що ітерується, *F* – умова.

Python вибирає всі *x* з *t*, які задовольняють умову *F*, додає у множину *e*(*x*) та повертає отриману множину.

Якщо умова *F* відсутня, то if *F* опускають.

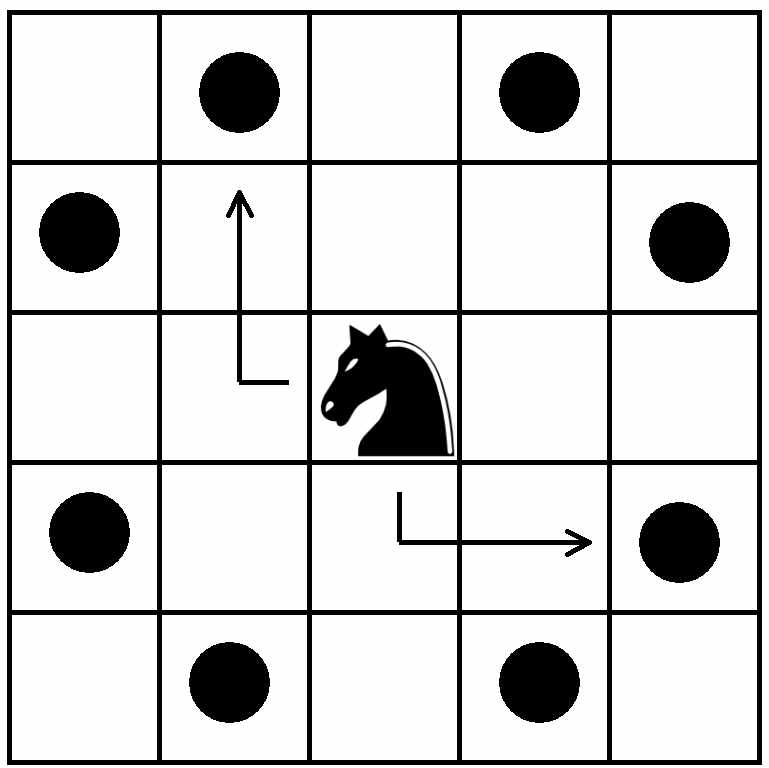
Наприклад, h = {1-x\*\*2 for x in range(-2,3)}

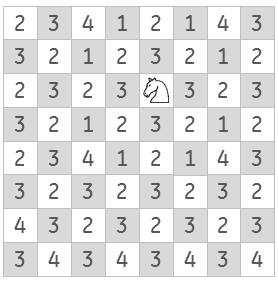
Приклад «Тур коня»

«Тур коня». Знайти шлях шахової фігури «кінь» з поля шахової дошки (*x*,*m*) на поле (*y*,*k*), де *x*, *y* - вертикалі (позначаються літерами від a до h), *m*, *k* - горизонталі (позначаються цифрами від 1 до 8).

Програма, яку ми розбирали у попередній темі, знаходила шлях коня, але цей шлях був неоптимальний через вибір наступного ходу, починаючи з початку списку ходів. Побудуємо програму, яка буде знаходити більш оптимальний шлях.

На *i*-му кроці будуємо граничну множину полів шахової дошки, яких кінь може досягти з початкової позиції за *i* кроків. Коли у множину потрапляє кінцеве поле, йде повернення до початкової позиції з побудовою шляху.





Резюме

Ми розглянули:

* 1. Множини: носій, основні операції, відношення та інструкції для множин.
  2. Додаткові операції, відношення та інструкції для множин.
  3. Множини, що не змінюються.
  4. Множиноутворення.

Де прочитати

1. A Byte of Python (Russian) Версия 2.01 Swaroop C H (Translated by Vladimir Smolyar), <http://wombat.org.ua/AByteOfPython/AByteofPythonRussian-2.01.pdf>
2. Марк Лутц, Изучаем Python, 4-е издание, 2010, Символ-Плюс
3. Python 3.4.3 documentation
4. Бублик В.В., Личман В.В., Обвінцев О.В.. Інформатика та програмування. Електронний конспект лекцій, 2003 р.,
5. <http://www.python-course.eu/python3_sets_frozensets.php>