Python. Laboratorium 9. Sety. Zestawy

Set to lista, zbiór danych, w których nie ma dwóch lub więcej identycznych elementów. Elementy są unikalne. Sety tworzymy używając dwóch nawiasów klamrowych {}. Set może przechowywać dane o różnych typach. Kolejność elementów nie ma znaczenia, dlatego też. nie można ich sortować, ani indeksować. Zbiór można przekształcić do listy.

Deklaracja seta :

```
A = {1, 2, 3}
A = set('Ala ma kota')
```

Do zestawu możemy dodawać elementy za pomocą metody add ().

```
A.add("ale") #dodanie do seta
```

Istnieją dwie metody usuwania elementu z zestawu: discard i remove .

Ich zachowanie zmienia się tylko w przypadku, gdy usunięty element nie był w zestawie. W takim przypadku metoda discard nic nie zwraca, a metoda remove KeyError zwróci wyjątek KeyError .

```
A.remove("ale") #usuniecie wartości z seta
print(A)
A.discard("ale") #usuniecie wartości z seta
print(A)
```

Iterowanie po zestawie:

```
for x in A:
        print(x)
```

Przykład:

```
Days=set(["Mon","Tue","Wed","Thu","Fri","Sat","Sun"])
for day in Days:
        print(d)
Days.add("Sun")
Days.discard("Sun")
print(Days)
```

Porównanie dwóch zbiorów:

```
A = {1, 2, 3}
B = {3, 2, 3, 1}
print(A == B)
```

Przykład:
Mamy dwa zbiory osób, które brały udział w dwóch naszych warsztatach.
A = set(["Adam", "Stanislaw", "Krzysztof"])
B = set(["Adam", "Stanislaw"])

Aby dowiedzieć się, które osoby brały udział w obu wydarzeniach, użyjemy metody intersection (część wspólna):
print (A.intersection(B))
print (B.intersection(A))

Aby ustalić osoby które brały udział tylko w jednym z wydarzeń, użyj metody symmetric_difference (różnica symetryczna):

print (A.symmetric_difference(B))
print (B.symmetric_difference(A))

Wyodrębnieniu osób, które brały udział w A i nie brały w B, służy różnica zbiorów difference:
print (A.difference(B))
print (B.difference(A))

Operacje na zestawach

1. A | b
   A.union (B)
   Zwraca zestaw będący połączeniem zbiorów A i B
2. A | = B
   A.update (B)
   Dodaje wszystkie elementy tablicy B do zbioru A
3. A i B
   Sekcja specjalna (B)
   Zwraca zestaw będący przecięciem zbiorów A i B
4. A & = B
   A.intersection_update (B)
   W zbiorze A tylko przedmioty należące do zbioru B
5. A - B
   A.difference (B)
   Zwraca ustawioną różnicę A i B (elementy zawarte w A , ale nieuwzględnione w B).
6. A - = B
   A.difference_update (B)
   Usuwa wszystkie elementy B z zestawu A
7. A ^ B
   A.symmetric_difference (B)

Zwraca symetryczną różnicę zbiorów A i B (elementy należące do A lub B , ale nie do obu zbiorów jednocześnie).

8. A ^ = B
   A.symmetric_difference_update (B)
   Zapisuje w A symetryczną różnicę zbiorów A i B
9. A <= B
   A.issubset (B)
   Zwraca wartość true jeśli A jest podzbiorem B
10. A>=B
    A.issuperset (B)
    Zwraca wartość true jeśli B jest podzbiorem A
11. A <B
    Odpowiednik A <= B and A != B
12. A>B
    Odpowiednik A >= B and A != B

Inne przykłady:

Sumowanie - Unia

Operacja sumowania na dwóch zestawach tworzy nowy zestaw zawierający wszystkie różne elementy z obu zestawów.

```
DaysA = set(["Mon","Tue","Wed"])
DaysB = set(["Wed","Thu","Fri","Sat","Sun"])
AllDays = DaysA|DaysB
print(AllDays)
```

Przecięcie zbiorów

Operacja przecięcia na dwóch zestawach tworzy nowy zestaw zawierający tylko wspólne elementy z obu zestawów. W poniższym przykładzie element „Wed” występuje w obu zestawach.

```
DaysA = set(["Mon","Tue","Wed"])
DaysB = set(["Wed","Thu","Fri","Sat","Sun"])
AllDays = DaysA & DaysB
print(AllDays)
```

Różnica zestawów

Operacja różnicowa na dwóch zestawach tworzy nowy zestaw zawierający tylko elementy z pierwszego zestawu i bez elementów z drugiego zestawu. W poniższym przykładzie element „Wed” występuje w obu zestawach, więc nie zostanie znaleziony w zestawie wyników.

```
DaysA = set(["Mon","Tue","Wed"])
```

```
DaysB = set(["Wed","Thu","Fri","Sat","Sun"])
AllDays = DaysA - DaysB
print(AllDays)
```

Porównanie zestawów

Możemy sprawdzić, czy dany zbiór jest podzbiorem lub nadzbiorem innego zbioru. Wynik jest True lub False w zależności od elementów obecnych w zestawach.

```
DaysA = set(["Mon","Tue","Wed"])
DaysB = set(["Mon","Tue","Wed","Thu","Fri","Sat","Sun"])
SubsetRes = DaysA <= DaysB
SupersetRes = DaysB >= DaysA
print(SubsetRes)
print(SupersetRes)
```

Lista zadań :

1. Utwórz 4 zestawy złożone z nazw drużyn sportowych min. 10 na każdy zestaw. Spraw by część nazw się powtarzała np. LigaMistrzow= set(["Real Madrid", "PSG", "Bayern Monachium”"]) Nazwy drużyn do zestawów powinny zostać wpisane losowo z puli 20 zespołów.
2. Sprawdź za pomocą metod intersection(), difference(), union(), issuperset() oraz issubset() jak zachowają się te 4 zbiory. Jakie można wyciągnąć z tych zestawów informacje?
3. Sprawdź długość zestawów. Usuń wybrane elementy z zestawów, które się pojawiają. Dokonaj wybranych porównań. Dokonaj konwersji z zestawu na listę. Co dzięki takiemu działaniu zyskujemy?