

## Słowniki

Słowniki to nieuporządkowane, zmienne modyfikowalne oraz indeksowalne kolekcje danych. Każdy słownik składa się z zestawu par **klucz** : **wartość**. Słownik tworzymy za pomocą następującej składni:

**TO JEST PRZYKŁAD OGÓLNEGO ZAPISU NIE KOD**

```
moj_sownik = {  
    klucz1 : wartosc1,  
    klucz2 : wartosc2,  
    klucz3 : wartosc3,  
}
```

Jeśli chcemy uzyskać dostęp do jakiegoś elementu słownika, możemy to zrobić dokonując wskazania konkretnego klucza słownika w nawiasach kwadratowych lub za pomocą metody **get()**, (przykład 2):

```
pol_ang = {  
    "kwiat" : "flower",  
    "woda" : "water",  
    "gleba" : "soil"  
}  
  
ele1 = pol_ang["gleba"]      # przykład 1  
print(ele1)                 # wydrukuje: soil  
  
ele2 = pol_ang.get("woda")   # przykład 2  
print(ele2)                 # wydrukuje: water
```

Jeśli chcemy zmienić wartość elementu powiązaną z określonym kluczem, możemy tego dokonać odnosząc się do klucza elementu w następujący sposób:

```
pol_ang = {  
    "zamek" : "castle",  
    "woda" : "water",  
    "gleba" : "soil"  
}  
  
pol_ang["zamek"] = "lock"  
print(pol_ang["zamek"])     # zwraca: lock
```

Możemy skorzystać z pętli **for**, aby iterować po elementach słownika, np.:

```
pol_ang = {  
    "zamek" : "castle",  
    "woda" : "water",  
    "gleba" : "soil"  
}  
  
for element in pol_ang:  
    print(element)
```

Słowo kluczowe **del** może być użyte do usunięcia konkretnego elementu słownika lub całego słownika. Aby usunąć wszystkie elementy słownika za jednym zamachem, należy skorzystać z metody **clear()**:

```
print(len(pol_ang))          # na wyjściu: 3  
del pol_ang["zamek"]         # usuwa element  
print(len(pol_ang))          # na wyjściu: 2  
pol_ang.clear()              # usuwa wszystkie elementy  
print(len(pol_ang))          # na wyjściu: 0  
del pol_ang                  # usuwa cały słownik
```

Wreszcie, aby skopiować słownik, należy użyć metody `copy()`:

```
pol_ang = {
    "zamek" : "castle",
    "woda" : "water",
    "gleba" : "soil"
}

kopia_pol_ang = pol_ang.copy()
```

Aby utworzyć listę, której elementami są słowniki, możesz po prostu zainicjować listę i dodać do niej słowniki. Przykład:

```
lista_sownikow = [
    {'imie': 'Jan', 'nazwisko': 'Kowalski', 'wiek': 30},
    {'imie': 'Anna', 'nazwisko': 'Nowak', 'wiek': 25},
    {'imie': 'Piotr', 'nazwisko': 'Wiśniewski', 'wiek': 35}
]

print(lista_sownikow)
```

Możesz również utworzyć pustą listę i dodać słowniki przy użyciu metody `append()`.

```
lista_sownikow = []

lista_sownikow.append({'imie': 'Jan', 'nazwisko': 'Kowalski', 'wiek': 30})
lista_sownikow.append({'imie': 'Anna', 'nazwisko': 'Nowak', 'wiek': 25})
lista_sownikow.append({'imie': 'Ewa', 'nazwisko': 'Las', 'wiek': 35})

print(lista_sownikow)
```

Możesz również skorzystać z konstruktora listy `list()` z argumentami będącymi słownikami

```
lista_sownikow = list(
    {'imie': 'Jan', 'nazwisko': 'Kowalski', 'wiek': 30},
    {'imie': 'Anna', 'nazwisko': 'Nowak', 'wiek': 25},
    {'imie': 'Piotr', 'nazwisko': 'Wiśniewski', 'wiek': 35}
)

print(lista_sownikow)
```

W Pythonie, jeśli masz ciąg znaków, który reprezentuje klucz słownika, a chcesz uzyskać wartość pod tym kluczem, możesz użyć nawiasów kwadratowych `[]` lub operatora indeksowania `.get()`

```
sownik = {'klucz1': 'wartosc1', 'klucz2': 'wartosc2'}
klucz = 'klucz1'
print(sownik[klucz]) # Output: 'wartosc1'

klucz = 'klucz2'
print(sownik[klucz]) # Output: 'wartosc2'
```

Jeśli chcesz uzyskać wartość, ale chcesz przechwycić wyjątek, jeśli klucz nie istnieje w słowniku, można skorzystać z metody `get()`.

```
sloownik = {'klucz1': 'wartosc1', 'klucz2': 'wartosc2'}
klucz = 'klucz1'
print(sloownik.get(klucz)) # Output: 'wartosc1'

klucz = 'klucz2'
print(sloownik.get(klucz)) # Output: 'wartosc2'
```

Jeśli chcesz uzyskać wartość i podać wartość domyślną jeśli klucz nie istnieje:

```
sloownik = {'klucz1': 'wartosc1', 'klucz2': 'wartosc2'}
klucz = 'klucz3'
print(sloownik.get(klucz, 'brak wartosci')) # Output: 'brak wartosci'
```

Jeśli masz ciąg znaków, który reprezentuje wartość, a chcesz utworzyć klucz słownika na podstawie tej wartości, możesz użyć tego ciągu jako wartości, a jakiegoś innego ciągu jako klucza, przy użyciu operatora przypisania =

```
wartosc = "jakis_string"
sloownik = {wartosc: "wartosc_dla_klucza"}
print(sloownik)
# Output: {"jakis_string": "wartosc_dla_klucza"}
```

Jeśli chcesz utworzyć klucz słownika na podstawie wartości zmiennej string, możesz po prostu przypisać wartość zmiennej do klucza słownika.

```
wartosc = "jakis_string"
sloownik = {}
sloownik[wartosc] = "wartosc_dla_klucza"
print(sloownik)
# Output: {"jakis_string": "wartosc_dla_klucza"}
```

jeśli chcesz dodać więcej niż jedną parę klucz-wartość do słownika, możesz użyć metody update()

```
wartosc1 = "jakis_string1"
wartosc2 = "jakis_string2"
sloownik = {}
sloownik.update({wartosc1: "wartosc_dla_klucza1",
wartosc2: "wartosc_dla_klucza2"})

print(sloownik)
# Output: {"jakis_string1": "wartosc_dla_klucza1",
"jakis_string2": "wartosc_dla_klucza2"}
```

Aby sprawdzić, czy klucz istnieje w słowniku, możesz użyć operatora `in`. Przykład:

```
slovník = {'klucz1': 'wartosc1', 'klucz2': 'wartosc2'}
klucz = 'klucz1'
if klucz in slovník:
    print(f"klucz {klucz} istnieje i jego wartość to {slovník[klucz]}")
else:
    print(f"klucz {klucz} nie istnieje")
```

Możesz również użyć metody `keys()` zwracającej zbiór wszystkich kluczy słownika, i sprawdzić czy klucz jest w zbiorze

```
slovník = {'klucz1': 'wartosc1', 'klucz2': 'wartosc2'}
klucz = 'klucz1'
if klucz in slovník.keys():
    print(f"klucz {klucz} istnieje i jego wartość to {slovník[klucz]}")
else:
    print(f"klucz {klucz} nie istnieje")
```

Jeśli chcesz uzyskać wartość pod kluczem, ale nie chcesz przechwycić wyjątku, jeśli klucz nie istnieje, można skorzystać z metody `get()`.

```
slovník = {'klucz1': 'wartosc1', 'klucz2': 'wartosc2'}
klucz = 'klucz1'
value = slovník.get(klucz)
if value is not None:
    print(f"klucz {klucz} istnieje i jego wartość to {value}")
else:
    print(f"klucz {klucz} nie istnieje")
```

Aby znaleźć największą wartość liczbową w słowniku, możesz użyć metody `values()`, która zwraca kolekcję wartości z słownika, a następnie znaleźć największą wartość korzystając z funkcji `max()`. Przykład:

```
slovník = {'klucz1': 42, 'klucz2': 15, 'klucz3': 99}
najwieksza_wartosc = max(slovník.values())
print(najwieksza_wartosc)
# Output: 99

slovník = {'klucz1': 42, 'klucz2': 15, 'klucz3': 99}
klucz, najwieksza_wartosc = max(slovník.items(), key=lambda x: x[1])
print(f"największa wartość: {najwieksza_wartosc} jest pod kluczem {klucz}")
# Output: największa wartość: 99 jest pod kluczem klucz3
```