Złożone typy danych

Krotki (tuple), listy, sety, słowniki

Krotki (tzw. tupples). Poniżej najważniejsze rzeczy, które powinieneś wiedzieć korzystając z krotek w Python:

- Krotki mogą przechowywać dane o różnych typach
- Krotka nie może być sortowana
- Każdy element krotki posiada swój indeks
- Elementy (składowe) krotki nie mogą być modyfikowane
- Krotkę tworzymy wykorzystując dwa zwykłe nawiasy

Poniżej kilka przykładów:

```
x = 12, "Python", 0.123, (12+5j), (1,2,3,4,5,6)
type(x)

# Odnosząc się do elementów krotki używaj indeksów. Indeks
rozpoczyna się od liczby zero
x[2]
x[0]

# Możesz także zacząć od prawej strony
x[-1]
x[-1][3]

# Dodając nowe elementy do krotki używaj znaku przecinka
y = 'Hello World !',
z = x + y
z

# Zwielokrotnianie elementów
print(z * 3)

# Sprawdzanie numeru indeksu elementu krotki
z.index("Python")

# index zwróci numer pierwszego wystąpienia elementu
(z*3).index("Python")

# Sprawdzanie liczby wystąpień elementu w krotce
z.count("Python")

# lub
(z*3).count("Python")

# Liczba elementów w krotce
len(z)

# Sum, min, max dla elementów. Używaj tylko, gdy w krotce są
wartości liczbowe
a = (1, 2, 3.14, 8)
```

```
min(a)
max(a)
sum(a)
```

Kolejnym ważnym złożonym typem danych w Python są listy (lists). Poniżej kilka ważnych rzeczy związanych z listami oraz przykłady.

- Listy tworzymy za pomocą dwóch nawiasów kwadratowych []
- Listy mogą przechowywać dane o różnych typach
- Listy w odróżnieniu od krotek w Python mogą być sortowane

```
l = [1, 2, 'Python', "Learn more"]
type(l)

# Odwołując się do elementu listu używaj indeksu. Indeks rozpoczyna
się od zera
l[2]
l[0]

# Możesz zacząć od prawej strony
l[-1]

# Możesz zmodyfikować elementy listy
l[2] = 'abc'
l


# Dodawanie nowych elementów do listy w Python
m = l + [1,2,3,4]
m
# lub
l.append("New string value")
l

# Wybór elementu z listy
l[-2]

# Wybor wielu elementów z listy
l[0:3]

# Zmiana więcej niż jednego elementu w liście
l[0:2] = [1,2,3,4,5,6]
l

# Usuwanie elementów z listy Python
l.remove('Learn more')
l

#  Usuwanie elementów z listy Python z użyciem indeksu
del(l[1:2])
l

# Sortowanie elementów w liście Python
```

```
l.sort
l

# Sortowanie listy Python w kolejności odwrotnej
n = [1,5,2,6,8,3.14]
n.sort(reverse=True)
n

# Generowanie elementów w liście. Przykład: liczby od 0 do 99
o = list(range(1,100))
o

# Proste działania arytmetyczne na listach Python
a = [1,2,3]
b = [4,5,6]
c = a + b
print(c)

# sposób na generowanie elementów w liście
L = list("jestem stringiem")
L
# wynik powinien być jak niżej
['j', 'e', 's', 't', 'e', 'm', ' ', 's', 't', 'r', 'i', 'n', 'g',
'i', 'e', 'm']
L[2]
```

Sety to kolejny złożony typ danych w Python.

- Sety tworzysz używając dwóch nawiasów klamrowych {}
- Set może przechowywać dane o różnych typach
- Elementy setów nie mogą się powtarzać. Są unikalne
- Kolejność elementów nie ma znaczenia
- Setu nie można sortować, ani indeksować

Przykłady poniżej:

```
# Tworzenie setu w Python
x = {1,2,3,4,1,2,3,4,"Hello World", "Python", "Python",1,2,3,4}
x

# Dodawanie nowych elementów do setu
x.add(987)
x

# Usuwanie elementów setu w Python
x.remove(987)
x
```

Słowniki w Python to złożony typ danych klucz-wartość. Chcąc pobrać wartość odwołujesz się do wcześniej określonego klucza, który w odróżnieniu od indeksu nie musi być wartością liczbową. Najważniejsze informacje:

- Typ danych klucz-wartość
- Klucz jest unikalny
- Słowniki mają klucze, nie indeksy
- Możesz tworzyć klucze także za pomocą wartości znakowych

Przykłady:

```
# Tworzenie słownika
d ={1:"Python", 2:"is", 3:"the", 4:"best", 7: "programming",
9:"language"}
d

dic = {'k1': 1, 'k2': 2, 'k3': 3}
dic
del dic['k2']
dic['kn'] = 'python'

# Wybór wartości słownika za pomocą klucza
d[1]
d[9]

# Modyfikowanie wartości w słowniku
d[1] = "Python 3"
d

# Usuwanie elementów słownika
del d[7]
d

# Listowanie kluczy w słowniku
d.keys()


# Listowanie wartości słownika
d.values()
```

Zadania

1. Napisz program, który wstawi do listy k – elementowej wartości podawane przez użytkownika, a następnie wyświetli wypełnioną listę
2. Napisz program, który dla zadanego dnia wyświetli jego dzień tygodnia (np. 1 - niedziela). (należy zbadać, czy liczba mieści się w zakresie i wykorzystać słownik)
3. Napisz program, który zwróci nazwę miesiąca dla przekazanej liczby (należy zbadać, czy liczba mieści się w zakresie i wykorzystać jeden z rodzajów tablicowych)