PODSTAWY PROGRAMOWANIA W JĘZYKU PYTHON

Dzień 4

Agenda

- Pętla for i zmienianie przebiegu pętli
- Krotka
- Inne kolekcje: zbiór i słownik
- Kolejka i stos

Pętla for
i zmienianie przebiegu pętli

Pętla for

```
for element in kolekcja:
    blok kodu
```

- Pętla `for` wykona się dla każdego elementu w kolekcji*.
- Przestrzeń nazw bloku kodu jest rozszerzana o zmienną `element` (nazwa zmiennej może być dowolna).
- Kolekcja może być pusta.
- Pętle można zagnieżdżać (dotyczy również pętli `while`).

** chyba, że pętla zostanie przerwana lub zmodyfikowana*

Zmienianie przebiegu pętli

`continue` – program pomija pozostałe instrukcje w bloku i wraca do sprawdzenia warunku (`while`) lub do kolejnego elementu (`for`).

`break` – działanie pętli jest przerywane, program przechodzi do kolejnej instrukcji po całym bloku pętli.

Krotka

Krotka

```
tuple(), (,)
```

Niezmienna (zarówno pod względem długości jak i zawartości) wersja listy.

```
>>> tuple1 = ("raz", "dwa", "trzy")
>>> tuple1[0] = "jeden"  # TypeError
>>> x = "raz",  # tuple
>>> x1 = (“raz”,)  # tuple
>>> x2 = ("raz")  # string
```

enumerate()

```
for (indeks, element) in enumerate(iterable):
    #  kod pętli for
```

- **`enumerate`** daje nam dwie wartości:
  - indeks bieżącego elementu,
  - oraz ten element.

zip()

```
for (element_a, element_b) in zip(iterable_a, iterable_b):
    # kod pętli
```

- **`zip`** daje nam elementy z tej samej pozycji w kilku kolekcjach.
- Gdy kolekcje są różnej długości, długość najkrótszej kolekcji będzie określała liczbę powtórzeń pętli.

- Inne funkcje wbudowane: https://docs.python.org/3/library/functions.html

Kolekcje c.d.

Zbiór

```
set(), {wartosc1, wartosc2 ...}
```

- Kolekcja wartości nie zawierająca powtórzeń.
- Każda wartość w zbiorze musi być obiektem typu niezmiennego (*immutable*) np. int, float, string, ale list już nie.
- Wartości w zbiorze nie mają kolejności - nie ma operacji indeksowania i wycinania (*slicing*).

Słownik

```
dict(), {klucz1: wartosc1, ...}
```

- Zbiór par klucz - wartość, mogących różnić się typem wartości (zarówno klucz - wartość jaki i klucze między sobą).
- Klucz musi być typem niezmiennym (np. string, int, tuple) i być unikatowy (tylko jeden wewnątrz słownika).
- Klucze nie zachowują kolejności (alfabetycznej, podania).

Kolekcje

	Lista (list)	Krotka (tuple)	Zbiór (set)	Słownik (dict)
Czy edytowalne (mutable)?	Tak	Nie	Tak	Tak
Indeksowanie [ ]	Tak	Tak	Nie	Tak
Indeks	Liczba od 0	Liczba od 0	-	Obiekt niezmiennego typu.
Wymaga unikatowych wartości	Nie	Nie	Tak	Tak (tylko klucze)
Zachowuje kolejność wprowadzania	Tak	Tak	Nie	Nie

Kolejka i stos

Kolejka i stos

- Dwie struktury danych wrażliwe na kolejność podawania danych.
- Kolejka:
  - kolejność dodawania do kolejki będzie zachowana przy wyciąganiu elementów. (*FIFO - FirstIn First Out*)
  - Operacja dodania (*insert*) i wyciągnięcia (*pop*)

Stos

- Stos:
  - wartość ostatnia dodana do stosu będzie pierwszą z niej wyciągniętą.
    (*LIFO - Last In First Out*)
  - Operacja dodania (*push*) i wyciągnięcia (*pop*)

Podsumowanie

- Pętla for i zmienianie przebiegu pętli
- Krotka
- Inne kolekcje: zbiór i słownik
- Kolejka i stos

Thanks!!