Sprawy organizacyjne

Sprawy organizacyjne:

- Dostęp do Datacampa (https://www.datacamp.com/) – link przez maila.
- Reguły zaliczenia.
  1. Listy zadań 70%.
  2. Projekt 30% (+ obowiązek wykonania).
- Reguły zwolnień.
  1. Studenci informatyki: zaliczenie przedmiotu za PWI + znajomość Pythona.
  2. Zwolnienie z list zadań – indywidualnie na konsultacjach.

Pętle

Kontrola przepływu sterowania – pętle:
Pętla while (*dopóki*):

```
while condition:
    ...
```

```
s = ""
while s != "koniec":
    s = input("Napisz 'koniec' aby wyjść: ")
```

Pętla for (minimalistyczna wersja) – iteracja (przebieganie) po ciągu elementów.

```
for name in sequence:
    ...
```

```
for i in range(1, 10): # ciąg 1, 2, ..., 9
    print(i)
```

Użytkownicy C++: pętla for to odpowiednik *range-based for loops* (C++11).

Przykłady

Przykład 1. (Bardzo prymitywny) test pierwszości liczby naturalnej $n > 1$:

```
n = int(input("Wpisz n > 1:"))

found_divisor = False

for d in range(2, n):
    if n % d == 0:
        found_divisor = True

print("Liczba", n, "jest pierwsza:", not found_divisor) # and n > 1
```

(prime.py w materiałach)

(live coding: warianty range)

Przykłady

Przykład 2. Punkty siatkowe na płaszczyźnie:

```
n = int(input("Wpisz n naturalne:"))

for x in range(n):
    for y in range(n):
        print(x, y)
```

Modyfikacje: tylko punkty, gdzie $x < y$, punkty innych ćwiartek.
(pairs.py w materiałach; break i $\mathrm{continue}$ wyjaśnione na Liście 2)

Obiekty i typy

W Pythonie pracujemy z **obiektami** (liczby, napisy, macierze, obrazy, listy zakupów, gatunki książek, ...).

W Pythonie, każdy obiekt ma **typ**. Typ obiektu determinuje jakie wartości może przyjmować obiekt, i jakie operacje można na nim wykonać.

Niektóre elementarne, wbudowane typy w Pythonie:

Nazwa	**Znaczenie**	**Przykładowe wartości**
int	liczba całkowita	1337, -1
float	liczba zmiennoprzecinkowa	2.5, -7e100
bool	wartość logiczna	True, False (jedyne)
str	napis (string)	"abc123", "X Y Z"
NoneType	„brak wartości"	None (jedyna)

type(x) – typ obiektu x.

Obiekty i typy

Komentarze:

- int – liczby całkowite, w teorii dowolne (natywne *bignumy*).
- float – liczby zmiennoprzecinkowe (przybliżenia liczb rzeczywistych) w standardzie IEEE podwójnej precyzji (64 bity). Odpowiednik w C/C++: double.
  - Zakres $\pm 1.7 \cdot 10^{308}$, ale liczby reprezentowane są w przybliżeniu.
  - Dokładność do 15-16 znaczących cyfr dziesiętnych.
  - Obliczenia arytmetyczne wykonywane z (najlepszym możliwym) przybliżeniem.
  - Notacja naukowa: $5e6 == 5000000$, $3e\text{-}2 == 0.03$
- Liczby typu int, float (a także inne) można dowolnie mieszać w wyrażeniach arytmetycznych – wynik będzie najogólniejszego (w sensie matematycznym) typu.
  - $2 + 2 == 4$
  - $2.0 + 2.0 == 4.0$
  - $2 + 2.0 == 4.0$ # int, float, float

(numbers.py w materiałach do wykładu)

Obiekty i typy

Konstrukcja obiektów ustalonego typu z innych obiektów (a – obiekt, t – nazwa typu), tutaj od razu z przypisaniem:

```
b = t(a)
```

np.

```
n = int("20")   # 20
x = float("2.5") # 2.5
n = int(10.5) # 10
s = str(100) # "100"
```

(Na razie) typowe użycie:

```
n = int(input("Podaj liczbę: "))
```

Użytkownicy C/C++: to *nie jest* rzutowanie (casting), a konstrukcja nowego obiektu.

Krotki

Typ tuple: *krotka*, skończony ciąg dowolnych obiektów.
Krotki długości $2, 3, 4, \ldots$: para, trójka, czwórka, etc.

Konstrukcje krotek (wraz z przypisaniem):

```
three = (1, 'a', 1.0)
test = (0, 0, 0, 0, 0)

# a nawet prościej:
three = 1, 'a', 1.0
test = 0, 0, 0, 0, 0
```

„Rozpakowanie" krotki:

```
three = 1, 'a', 1.0
a, b, c = three
print(a) # 1
print(b) # 'a'
print(c) # 1.0
```

Ciągi

Krotki i napisy (oraz wiele innych obiektów) reprezentują ciągi obiektów i mogą być użyte w pętli for:

```
for x in (1, 4, 9, 16):
    print(x)
```

```
for c in "Jestem napisem":
    print(c)
```

Przykład 3 (bonus). Suma cyfr w liczbie (w stylu Pythona):

```
s = input("Podaj liczbę naturalną: ")
suma = 0
for c in s:
    suma = suma + int(c) # !
print(suma)
```

Użytkownicy C/C++: *w tym wypadku* można też $\mathrm{suma} \mathrel{+}= \mathrm{int(c)}$