PROJEKT C

24. 10.2018

TERMIN ROZLICZENIA **5.11.2018**

PYTHON: PODSTAWOWE INSTRUKCJE

- Example Using Sets
- input via the keyboard
- Conditional Statements
- Loops, while Loop
- For Loops
- Difference between interators und Iterables
- Output with Print
- Formatted output with string modulo and the format method
- Functions
- Recursion and Recursive Functions
- Parameter Passing in Functions

Wyrażenia warunkowe i pętla while

Ogólna składnia wyrażenia warunkowego:

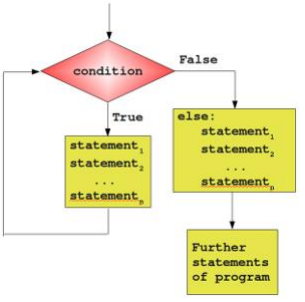
```
if warunek_1:
    blok_operacji_1
elif warunek_2:
    blok_operacji_2

...

elif warunek_ostatni:
    blok_operacji_ostatnich
else:
    blok_operacji_koncowych
```

c_0.py

Ogólna składnia pętli while:

```
while warunek:
        blok_operacji_1
else:
        blok_operacji_2
```

c_1.py



Pętla for

Ogólna składnia pętli for

```
for <variable> in <sequence>:
        blok_operacji_1
else:
        blok_operacji_2
```

Pętla *for* iteruje po zbiorze elementów.

Zatem jej użycie wymaga korzystania z **iteratora :** zmiennej (tutaj : wskaźnika) umożliwiającej przeglądanie zbioru.

Listy, krotki, słowniki i zbiory to obiekty złożone, które mogą być iterowane. Oznacza to, że posiadają iteratory : zmienne, umożliwiające dostęp do zawartości obiektów.

range(start, koniec, krok)
to popularny sposób budowania listy liczb z iteratorem je obsługującym.

Przykłady:

```
suma = 0                                  # inicjujemy wartość zmiennej suma
for licznik in range(1,n+1):              #wartości zmiennej licznik ustalone przez range() to 1,2,..., n
    suma = suma + licznik
```

```
for element in [4, "s", [2,3], 3.14159]:      # pętla po liście składającej się z elementów różnych typów.
    print ( element)
```

```
for (a, b) in [(2, "i"), (3.14, "f"), ("a", "s")]:  # pętla po liście składającej się z krotek.
    print( "pierwszy", a, "drugi", b)
```

```
D = {"a": 1, "b": 2, "c": 3}              # pętla for dla słownika.
for key in D:                             # pętla jest po kluczach słownika
    print( key, D[key])
for (key, value) in D.items():            # pętla jest po elementach słownika
    print( key, value)
```

c_2.py

Pętla for: zagadnienia zaawansowane

```
owoce = ["jabłko", "gruszka", "śliwka"]

for owoc in owoce:                        # Pętla for po liście stringów klasycznie.
    print("Bardzo lubię " + owoc + "!")

for indeks in range(len(owoce)):          # Jeśli potrzebujemy wykorzystać numer elementu.
    print( indeks, owoce[indeks])

for (indeks, zawartosc) in enumerate(owoce):   # Jest też naturalny iterator dostępny poprzez enumerate().
    print( indeks,zawartosc)

#szybkie generowanie listy
l1 = [c*i for (i, c) in enumerate("abcd")]     # l1=["", "b", "cc", "ddd"] : uwazaj, pierwsze i to 0
l2 = [2**i for i in range(5)]                  # l2=[1, 2, 4, 8, 16]
```

c_3.py

```
# Oczyszczanie linii z białych znaków po prawej stronie.
# zmienna lines to lista stringów.
lines = [line.rstrip() for line in lines]
```

c_4.py

itertools: biblioteka funkcji do tworzenia iteratorów

```
import itertools
for litery in itertools.product('01', repeat=3):   # produkuje krotke
    slowo = ''.join(litery)
    print(slowo)
```

c_5.py

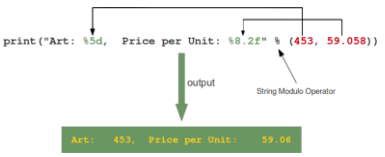
Metoda .join() elementy krotki o nazwie litery są łączone, a łącznikiem jest to co jest umieszczone w ' '

Formatowanie wydruków

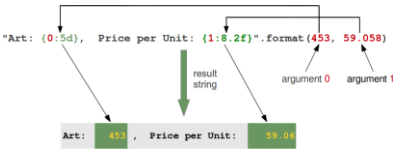
```
print(value1, ..., sep=' ', end='\n', file=sys.stdout, flush=False)
```

Funkcja drukuje dowolną liczbę wartości,
separowanych przez znak(i) określone przez *sep* ,
kończy wyświetlanie znakiem (znakami) określonym przez *end*
do strumienia określonego przez *file*

Drukowanie może być formatowane.
Schemat użycia formatowania może przypominać zasady z C, czyli
%[flagi][szerokosc][.precyzja]typ
I tak
%d –opis wyświetlanie liczby całkowitej
%f – opis wyświetlania liczby zmiennoprzecinkowej
%s – opis wyświetlania łańcucha



LUB własne z Pythona za pomocą metody format() dla dowolnego łańcucha



https://www.python-course.eu/python3_formatted_output.php

PROJEKT C 24. 10.2018

W oparciu o treści rozdziału 1 podręcznika przygotować rozwiązanie problemu:

Frequent Words Problem:
Find the most frequent k-mers in a string.

Input: A string *Text* and an integer *k*.
Output: All most frequent *k*-mers in *Text*.