Lista 2: Elementy języka Python

Witold Dyrka

Marzec 2021

Rozwiązania zadań proszę przedstawić w formie kodu źródłowego w języku Python (plik tekstowy z rozszerzeniem .py). Pliki z kodem źródłowym należy zamieszczać w kontenerze odbiorczym bezpośrednio — pliki spakowane nie będą akceptowane. Każdy plik z kodem źródłowym proszę opatrzyć komentarzem informującym o autorstwie, np.

```
"""
@author: Witold Dyrka
"""
```

Powyższy zapis jest traktowany jako równoważny oświadczeniu, że kod został napisany samodzielnie. W przeciwnym przypadku, komentarz powinien określać rodzaj i zakres udziału zewnętrznego oraz precyzyjnie wskazywać jego źródła, np.

`Skorzystałem z odpowiedzi użytkownika frabjous na pytanie na:`
https://tex.stackexchange.com/questions/2291/how-do-i-change-the-enumerate-list-format-to-use-letters-instead-of-the-defaul.

Umiejętne *oraz* dobrze udokumentowane korzystanie ze źródeł zewnętrznych nie obniża wartości samego rozwiązania, jednak nadmierne lub, co gorsze, bezrefleksyjne użycie materiałów zewnętrznych może negatywnie wpłynąć na proces nauki programowania.

Przesyłany do oceny kod źródłowy powinien być opatrzony komentarzami — wymagania w tym zakresie zostały podane w treści zadań.

Zad. 1

Zaimplementuj algorytmy w formie skryptów języka Python:

a) algorytm z Zadania 1 z Listy 1,

b) algorytm z Zadania 2 z Listy 1,

c) algorytm z Zadania 4 z Listy 1,

d) algorytm obliczający elementy ciągu arytmetycznego opisanego wzorem:

$$a[n+1] = a[n] + r, \text{ gdzie } n \in \{0, 1, 2, \ldots\}$$

dla zadanych: wyrazu początkowego $a[0]$, różnicy r i liczby elementów do obliczenia N.

Sprawdź, czy działanie programów pokrywa się z wykonaną analizą stanów algorytmów.

Kod powinien być zredagowany zgodnie z zasadami przedstawiony na wykładzie. Dobre praktyki dotyczące redagowania kodu w języku Python opisuje dokument *PEP8 – Style Guide for Python Code*[1]. Uzupełnij kod o komentarze. Ogólne zasady dokumentowania kodu w języku Python zawiera dokument *PEP257 – Docstring Conventions*[2].

Rozwiązanie każdego podpunktu zadania powinno zostać zapisane w osobnym pliku źródłowym `.py` języka Python.

Zad. 2

Zapisz powyższe skrypty jako funkcje w języku Python. Każdą funkcję opatrz komentarzem dokumentacyjnym (dokumentującym), np.

```
def wzor_Herona(a, b, c):
    """Funkcja oblicza pole trójkąta, wykorzystując wzór Herona.

    Args:
        a, b, c (float): długości boków

    Returns:
        float: pole trójkąta, jeśli podano poprawne długości boków,
        None: w przeciwnym przypadku
    """
```

Polecamy trzymać się stylu komentarzy dokumentacyjnych Google[3] — jak w powyższym przykładzie — albo NumPy[4]. Zwięzłe porównanie obu stylów znajdziemy m.in. w dokumentacji generatora dokumentacji Sphinx[5].

[1] https://www.python.org/dev/peps/pep-0008/

[2] https://www.python.org/dev/peps/pep-0257/

[3] https://google.github.io/styleguide/pyguide.html

[4] https://numpydoc.readthedocs.io/en/latest/format.html#docstring-standard

[5] https://www.sphinx-doc.org/en/master/usage/extensions/napoleon.html

Ponadto dla każdej z funkcji implementujących algorytm napisz funkcję testującą, która na kilku przykładach zademonstruje poprawne działanie testowanej funkcji. Funkcja testująca powinna zwracać użytkownikowi czytelne komunikaty. Warto także wykorzystać asercje do automatycznej weryfikacji poprawności. Na przykład:

```
def test_wzor_Herona():
    """Funkcja testująca dla funkcji wzór_Herona(a, b, c).
    """

    pole = wzor_Herona(3, 4, 5)
    assert pole == 6, 'Niepoprawnie policzone pole'
    print('Pole trojkata o bokach 3, 4 i 5 wynosi ' + \
          str(pole) + '.')

    pole = wzor_Herona(3, 4, 10)
    assert pole is None, 'Niepoprawnie policzone pole'
    print('Trojkat o bokach 3, 4 i 10 nie istnieje.')
```

Zwróć uwagę na możliwość kontynuowania instrukcji w kolejnej linii dzięki znakowi \, który nazywamy w-tył-ciachem (ang. *backslash*).

Zad. 3 — nieobowiązkowe

W formie funkcji w języku Python zaimplementuj algorytm obliczający kolejne elementy ciągu:

$$c_{n+1} = \begin{cases} \frac{1}{2}c_n, & \text{gdy } c_n \text{ jest parzyste,} \\ 3c_n + 1, & \text{gdy } c_n \text{ jest nieparzyste,} \end{cases}$$

gdzie $n \in \{0, 1, 2, \ldots\}$, a c_0 jest dowolną liczbą naturalną. Ciąg ten jest związany z problemem Collatza[6]: *Czy niezależnie od wybranej wartości* c_0*, ciąg wpadnie ostatecznie w cykl* $(4, 2, 1)$*?* Pytanie to pozostaje jak dotąd bez odpowiedzi...

Kod źródłowy opatrz komentarzami oraz wykonaj i udokumentuj testy poprawności (napisz funkcję testującą). Zanotuj maksymalną wartość elementu ciągu oraz maksymalną długość ciągu — *przed* wpadnięciem ciągu w cykl, a także odpowiadające im wartości c_0.

Sposób oceny Zad. 3 zostanie określony przez prowadzącego laboratorium.

[6] https://pl.wikipedia.org/wiki/Problem_Collatza