

# Wstęp do programowania 2024

## Pracownia 5

### Termin publikacji: 4.11.2024

**Uwaga:** Na tej liście będą wprawki i na kolejnych też (już nieogłaszane). Bonus (0.5) dla tej listy jest zdobycie ponad jednego punktu.

**Zadanie 1.(1pkt)** Będziemy rozważać problem Collatza. Funkcja  $F$  zdefiniowana jest następująco:

```
def F(n):
    if n % 2 == 0:
        return n / 2
    else:
        return 3 * n + 1
```

Będziemy rozważać ciąg:  $a, F(a), FF(a), FFF(a), FFFF(a) \dots$ , gdzie  $a$  jest całkowite dodatnie. Istnieje hipoteza, że ciąg ten zawsze kończy się powtarzającą w nieskończoność sekwencją 1, 4, 2, 1, 4, 2, 1, ... Dla danej liczby jej energią nazwiemy pozycję, na której w tym ciągu po raz pierwszy pojawia się jedynka. Przykładowo, energią liczby 7 jest 16, ponieważ ciąg Collatza rozpoczynający się od wartości 7 wygląda tak (z jedynką na 16-tej pozycji, licząc od zera):

7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1

Napisz funkcję, która wylicza energię liczby. Napisz funkcję `analiza_collatza(a,b)`, która oblicza wszystkie energie liczb od  $a$  do  $b$  oraz wypisuje:

- Średnią energię
- Medianę energii
- Maksymalną i minimalną energię

**Uwaga:** Zadanie będzie miało dalszy ciąg. **Uwaga 2:** Oczywiście możesz założyć, że hipoteza Collatza jest prawdziwa i zawsze kiedyś dojdziemy do jedynki.

**Zadanie 2.(1pkt)** Wybierz jeszcze jeden rysunek: albo czarno-biały ale nieco trudniejszy od tego, który robiłeś, albo kolorowy i napisz korzystając z modułu `turtle` program, który ten rysunek wykonuje.

**Zadanie 3.(1pkt)** W zadaniu wracamy do rysowania liczb. Napisz funkcję, która za pomocą modułu `turtle` rysuje liczbę  $n$ , korzystając z „definicji czcionek” (w module `duze_cyfry`). Oczywiście zamiast znaków `*` lub `#` teraz powinieneś używać wypełnionych kwadracików. Żeby wyglądało to ładniej, każda cyfra powinna być w losowo wybranym kolorze.

**Zadanie 4.(1pkt)**★ Na wykładzie pisaliśmy funkcję `usun_duplikaty(L)`, która dla listy  $L$  przeglądała jej elementy i kopiowała do wyniku, pomijając te, które już wcześniej w tym wyniku się pojawiły. Przykładowo `usun_duplikaty([1,2,3,1,2,3,8,2,2,2,9,9,4])` powinno zwrócić `[1,2,3,8,9,4]`.

Napisz tę funkcję efektywnie, korzystając z sortowania oraz tego, że listę par<sup>1</sup> Python sortuje patrząc w pierwszej kolejności na pierwszy element, potem na drugi. **Nie wolno** korzystać ze zbiorów!

Wskazówka 1: zobacz, co dzieje się, gdy napiszemy:

```
L1 = sorted([5,4,3,1,2,6])
L2 = [5,3,1,4,6,4,9]
L2.sort()
```

---

<sup>1</sup>O parach (krotkach) będziemy jeszcze mówić. Na potrzeby tego zadania możesz przyjąć, że para to po prostu lista dwuelementowa

(więcej o sortowaniu będzie na najbliższym wykładzie) Wskazówka 2: ([www.rot13.com](http://www.rot13.com)): Zbman  
mnzvravp yvfgr an yvfgr cne mnjvrenwnplpu jnegbfp j beltvanyarw yvfpr v cbmlpwr, anfgcavr gr  
yvfgr cbfbegbjnp, hfhanp qhcyvxngl jnegbfp v cbfbegbjnp wrfmpmr enm mr jmtyrqh an cbmlpwr.