Python: sortowanie, krotki, czytanie z pliku

Plan zajęć

1. Omówienie i dokończenie implementacji algorytmów sortujących z poprzednich zajęć:

- bąbelkowe (bubble sort)
- przez wybieranie (selection sort)
- przez wstawianie (insertion sort)
- przez zliczanie (counting sort)

Konspekt: http://student.agh.edu.pl/~lopiola/resources/2018-12-06/instrukcja.pdf

2. Mierzenie czasu działania algorytmów sortowania

- Dobrać odpowiedniej wielkości listę, aby sortowanie trwało w sekundach (np. 500000)
- Sprawdzić czas sortowania dla różnych algorytmów i dla trzech typów listy wejściowej:
    - nieposortowana (wygenerowana losowo)
    - posortowana
    - posortowana odwrotnie

3. Wspólne omówienie wyników pomiarów i podstaw złożoności obliczeniowej

4. Złożony typ danych w pythonie - krotka

Dotąd poznaliśmy kilka podstawowych typów danych:

```
calkowita = 100
zmiennoprzecinkowa = 3.75
napis = "napis"
logiczna = True
lista = [1, 2, 3]
```

W pythonie można łatwo wyrazić strukturę zawierającą kilka (potencjalnie różnych) typów danych - za pomocą **krotki** (ang. **tuple**). Odwoływanie się do elementów krotki działa tak samo jak w przypadku list. Należy pamiętać, że krotki są niezmienne (immutable) - nie można zmienić zawartości raz stworzonej krotki - należy stworzyć kolejną.

Przykład - wykonać i przeanalizować:

```
krotka = (1, 15, 60)
print(krotka)
print(len(krotka))
print(krotka[1])
# ponizsze spowoduje blad - krotek nie mozna modyfikowac
krotka[0] = 19
```

Nawiasy w zapisie krotki nie są obowiązkowe.
Jeśli znamy długość krotki, możemy w kodzie łatwo przypisać jej zawartość do zmiennych używając przyrównania do lewej strony.

Przykład - wykonać i przeanalizować:

```
krotka = ("Jan", "Nowak")
print(krotka)
(imie, nazwisko) = krotka
print(imie)
print(nazwisko)


krotka = "Jan", "Nowak"
print(krotka)
imie, nazwisko = krotka
print(imie)
print(nazwisko)
```

Nie ma przeszkód, aby przechowywać w krotce różne typy danych:

```
krotka = (1, "Hello world", 60.7, False, [1, 2, 3, 4])
print(krotka)
```

5. Czytanie z pliku
Pobierz plik z danymi do zadania - **ludzie-probka.csv**. Plik można dodać do projektu w repl.it, a następnie przeczytać linijka po linijce jak poniżej. Równie łatwo stworzyć własny plik i zapisać do niego tekst:

```
# czytanie z pliku
with open("ludzie-probka.csv", "r") as f:
for line in f:
  print(line)

f.close()



# pisanie do pliku
with open('wyjscie.csv', 'a') as f:
   # 'a' dopisuje na koniec pliku
   # 'w' nadpisuje caly plik
   f.write('linijka tekstu\n')

f.close()
```

Wykonaj następujące zadania:

- Wczytaj plik do listy przechowującej krotki o następującej zawartości:
  `(imie, nazwisko, miasto, zonaty_zamezna, telefon, wzrost, waga)`
- Twój program powinien dawać wybór: sortuj, szukaj, zakończ - aby pobrać wejście z klawiatury, użyj: `input('Twoj wybor:')`
    - Sortowanie - program powinien pozwolić na wybór kryterium (np. imię, wiek), czy ma być rosnące czy malejące, oraz umożliwić ograniczenie wyświetlanych wyników do pierwszych N. Następnie powinien wyświetlić wyniki sortowania. Użyj swoich algorytmów sortowania.
    - Szukanie - program powinien pozwolić na wybór kryterium (np. imię, wiek), oraz najlepiej wspierać szukanie po podciągach i bez rozróżnienia wielkości liter (czyli np. “ra” pasuje do “Rak” i “Baran”), i wyświetlać wszystkie pasujące rekordy.
    - Zakończenie - program się kończy (w innym przypadku powinien w kółko pytać o wybór kolejnej akcji)
- Gdy program będzie działał poprawnie, wytestuj go na dużym pliku - **ludzie-pelny.csv**.
- Zaimplementuj możliwość zapisania wyników sortowania/wyszukiwania do pliku csv. Program powinien zapytać, czy wyniki mają pojawić się na ekranie, czy w pliku, i odpowiednio się zachować.

Zadania dodatkowe: https://www.practicepython.org/