13 października 2023

Kurs rozszerzony języka Python

Lista 3.

Każde zadanie jest warte 4 punkty. Na pracowni do oceny należy przedstawić dwa zadania.

Tam, gdzie to ma sens, wykorzystaj implementację list `deque` z modułu `collections`, która efektywniej implementuje operacje dodawania i usuwania elementów z końców listy niż standardowe Pythonowe listy.

Zadanie 1.

Zaprogramuj funkcję implementującą algorytm wyszukujący *najdłuższy palindrom* w zadanym tekście podanym jako argument funkcji. Wynikiem jest lista krotek postaci (i, l), gdzie i to pozycja początku palindromu, a l to jego długość. Jeśli nie ma w tekście palindromów, funkcja zwraca listę pustą.

Przyjmujemy, że

- palindrom powinien mieć długość co najmniej 2;
- nie jest konieczne usuwanie znaków przestankowych, spacji czy unifikowanie wielkości liter tak jak to było w zadaniu na liście pierwszej;
- algorytm powinien mieć złożoność co najwyżej $O(n^2)$[1].

Zadanie 2.

Zaprogramuj dwie funkcje potrzebne do operowania na wyrażeniach arytmetycznych w postaci *odwrotnej notacji polskiej*:

- funkcję `konwersja(wyrażenie_infiksowe)`, która zwraca `wyrażenie_infiksowe` w postaci ONP. Przyjmujemy, że `wyrażenie_infiksowe` jest listą liczb, nawiasów i operatorów, np. `['(', 2, '+', 3, ')', '*', 4]`, a wynik jest również listą;
- funkcję `oblicz(wyrażenie_onp)`, które wylicza wyrażenie w postaci ONP, gdzie argument `wyrażenie_onp` jest w takiej postaci, jak wynik poprzedniej funkcji.

Zadanie 3.

Napisz algorytm wyszukujący drogę w labiryncie. Przyjmujemy, że labirynt jest zadany jako lista list, gdzie przeszkody oznaczamy znakiem `'X'`, a brak przeszkody spacją. Rozwiązanie powinno mieć postać funkcji, której argumentami są labirynt oraz współrzędne punktu początkowego, a wynikiem jest lista par współrzędnych miejsc tworząca ścieżkę do wyjścia.

Zadanie 4.

Zaprogramuj algorytm sortujący listę, który poprawnie posortuje listę liczb zapisanych słownie, np. listę

```
['sto dwadzieścia trzy', 'osiemset piętnaście',\
 'trzydzieści tysięcy dwieście']
```

W przykładzie podano polskie liczebniki, ale nie można też zaprogramować dla innych języków. Wystarczy, jeśli program będzie działał dla liczb sześciocyfrowych.

Zadanie 5.

Zaprogramuj funkcję `max_sublist_sum(lista)`, która dla listy liczb `lista` zwróci

[1] Chętni mogą poszukać algorytmu Manachera o złożoności $O(n)$

taką parę (i, j), że suma `sum(lista[i:j+1])` jest największa dla wszystkich par $0 \leq i \leq j < len(lista)$.
Zauważ, że w liście mogą być liczby ujemne, więc rozwiązanie

$$i = 0, j = len(lista) - 1$$

nie zawsze jest najlepsze.

Marcin Młotkowski