Lista 2

Uwaga. W Zadaniach 3 i 4 możesz użyć funkcji `perf_counter` z modułu `time`, lub zapoznać się z modułem `timeit`[1].

Zadanie 1 (0,5 punktu). Opisz optymistyczną i pesymistyczną złożoność czasową algorytmu weryfikacji poprawności nawiasowania (na podstawie implementacji w `parens.py` z materiałów do wykładu).

Zadanie 2 (1 punkt). Opisz kontekst w którym zdanie „złożoność usuwania elementu listy w przypadku średnim ma złożoność $\Theta(n)$" ma sens oraz jest prawdziwe. Uzasadnij jego prawdziwość.

Zadanie 3 (0,75 punktu). Zbadaj czasy (w zależności od długości `n` danej listy) usuwania elementu spod indeksu:
(a) Na początku listy.
(b) Na końcu listy listy.
(c) W $\frac{2}{3}$ długości listy.
Zaprezentuj wszystkie wyniki na pojedynczym wykresie. Przetestuj listy długości co najmniej kilku tysięcy (niekoniecznie dla wszystkich kolejnych wartości `n`).

Zadanie 4 (1 punkt). Niech `s` będzie zbiorem (klasa `set`). Zbadaj średni czas wykonania następujących operacji w zależności od rozmiaru zbioru:
(a) `s = s | {0}`
(b) `s |= {0}`
Narysuj odpowiedni wykres. Testami obejmij zbiory rozmiaru dziesiątek tysięcy elementów.

Zadanie 5 (0,75 punktu). Uzupełnij klasę `Stack` z wykładu (patrz: materiały na stronie wykładu):
1) Dodaj metodę `__str__(self)`, zwracającą reprezentację stosu jako napis.
2) Zmodyfikuj metody `pop` i `peek` aby rzucały stosowne wyjątki, gdy stos jest pusty.
3) Dodaj do inicjalizatora opcjonalny, iterowalny parametr `values` (niekoniecznie będący listą). Jeśli został on podany, odłóż występujące w nim elementy (zaczynając od pierwszego) na stosie.

[1] `https://docs.python.org/3/library/timeit.html`