Python (13) - Ważne algorytmy - porównywanie szybkości

(1) Napisz prostą funkcję zwracającą największy wspólny dzielnik liczb a,b, $pnwd(a, b)$. Zakładamy że nwd jest równy 1 i sprawdzamy czy kolejne liczby 2, 3, są dzielnikami a oraz b, aż do mniejszej z a i b. Przetestuj na parach (15,21), (91,39), (10,13).

(2) Napisz funkcję $nwd(a, b)$ służącą do znajdywania największego wspólnego dzielnika liczb a i b rekurencyjnie, stosując algorytm Euklidesa. Przetestuj czy funkcja działa.

(3) Porównaj szybkość działania funkcji *pnwd* oraz *nwd* na wielu parach dużych liczb losowych. Porównaj podobnie *nwd* z funkcją *gcd* z modułu *math*.

(4) Napisz funkcję $jest(lista, x)$ która zwraca 1 jeśli x jest w liście $lista$, a 0 jeśli nie. Użyj pętli po elementach listy.
Napisz funkcję $jestsort(lista, x)$ działającą podobnie jak poprzednia, ale korzystającą z założenia że $lista$ jest posortowana: porównujemy x ze środkowym elementem, potem szukamy w 1/4 lub 3/4, itd. Przetestuj obie funkcje na listach.

(5) Stwórz 1000 elementową listę liczb z zakresu od 1 do miliona. Porównaj szbykość funkcji z (4) na tej liście (szukaj wielu elementów x, np. też losowych z zakresu 1 do miliona).

(6) Napisz (lub znajdź w starych zadaniach) funkcję $piewsza(n)$ zwracająca 1 jeśli n jest pierwsza lub 0 jeśli jest złożona. Sprawdź przy jakich n funkcja zaczyna działać wolno. Zastanów się do jakiej liczby wystarczy sprawdzać potencjalne dzielniki n (można do n ale można do dużo mniejszej liczby). Popraw funkcję i sprawdź dla jak dużych liczb możemy jeszcze sprawdzać czy są pierwsze (w ciągu powiedzmy paru sekund).