Grzegorz Szulik 7 7 7 7

..... ----------

(Imię i nazwisko) (A, B, C, D)

Parametry:
liczba_prob = 7
liczba_dlugosci = 33

Przykład raportu z Pracowni nr 1

Zadanie 1.

1. Cel zadania

Celem zadania było zbadanie złożoności obliczeniowej algorytmów sortowania przez wstawianie i przez wybieranie.

2. Metody

W doświadczeniu wykorzystano kilka klas stworzonych w języku Python. Odpowiedni projekt stworzono i kompilowano w środowisku Thonny na komputerze przenośnym o procesorze Intel Celeron CPU 1007U.

3. Przebieg doświadczenia i wyniki

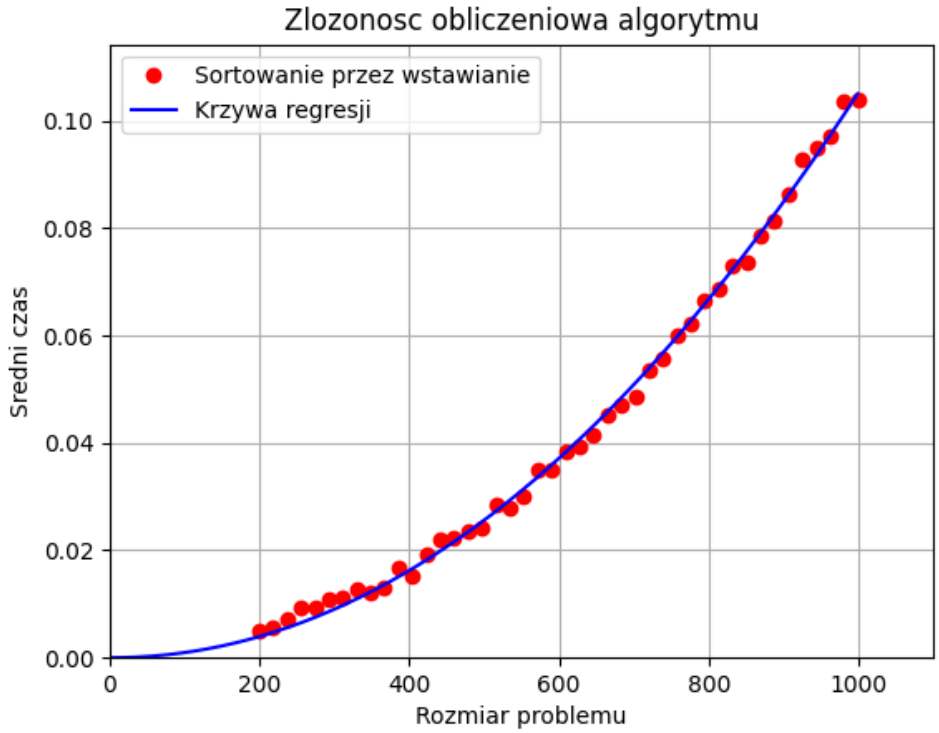
Doświadczenie rozpoczęto od ustalenia minimalnego i maksymalnego rozmiaru listy. Przyjęto, że będą to:

- *min_dlugosc* = 200, dla której czasy sortowania wyniosły powyżej 0,002 sekundy,
- *n* = 1000, dla której czas sortowania metodą wybierania wyniósł ok. 0,2 sekundy.

Opracowano metodę *mierz_czas*, która pozwala obliczyć czas sortowania listy o zadanej długości jedną z dwóch badanych metod. Poniżej zamieszczono kod tej metody:

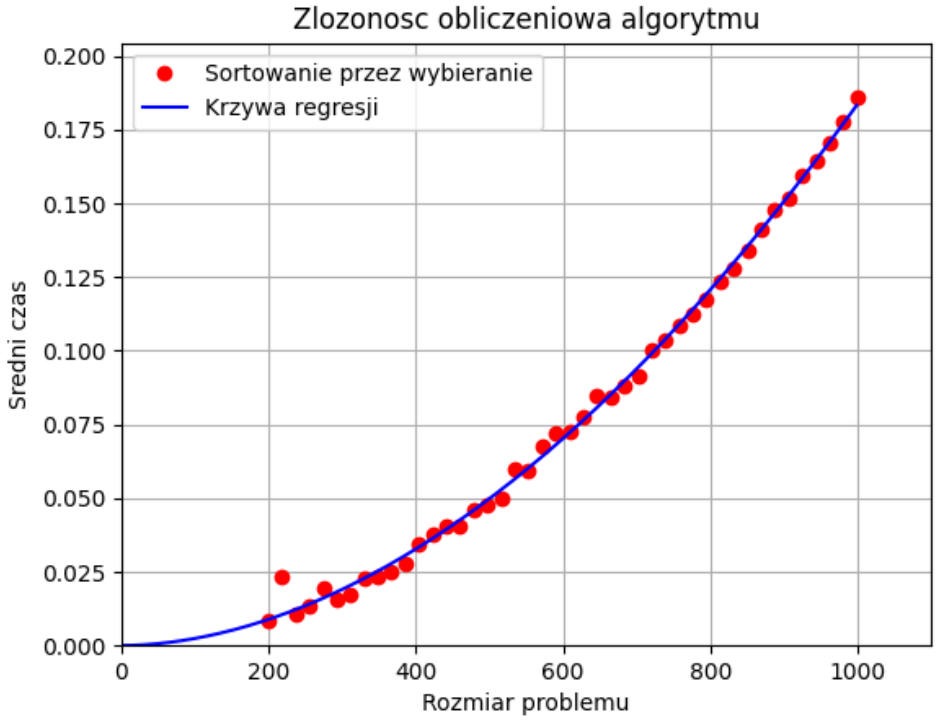
```
def mierz_czas(self, metoda, k=None):
    """Metoda mierzaca czas sortowan losowych list
       o dlugosci k"""
    if k is None:
        k = self.dlugosc
    self.lista = []
    czas = 0.0
    for _ in range(self.liczba_prob):
        self.losuj(k)
        if metoda == 1:
            stoper = time.time()
            self.sortuj_przez_wstawianie(k)
            stoper = time.time() - stoper
        else:
            stoper = time.time()
            self.sortuj_przez_wybieranie(k)
            stoper = time.time() - stoper
        czas = czas + stoper
    return czas/self.liczba_prob
```

Następnie wywołano metodę *badaj_zlozonosc* podając jako parametry klasy *Sortowania* wartości *n = 1000*, *lprob = 7*, *ldugosci = 44* oraz *najkrotsza* = 200.



Wykres 1. Zależność czasu sortowania listy metodą wstawiania od długości listy

Empiryczna złożoność obliczeniowa wyniosła: $n^{2,039}$, co bliskie jest teoretycznej wartości n^2. Podobny eksperyment przeprowadzono dla metody sortowania przez wybieranie.



Wykres 2. Zależność czasu sortowania listy metodą wybierania od długości listy

Tym razem uzyskano złożoność obliczeniową rzędu $n^{1.881}$, co zaskoczyło autora opracowania ze względu na brak czasu optymistycznego dla sortowania przez wybieranie (w przypadku sortowania przez wstawianie czas optymistyczny jest rzędu n). Niemniej jednak można zauważyć, że średnie czasy wykonywania się sortowania metodą wstawiania są prawie dwukrotnie mniejsze niż te uzyskane dla metody sortowania przez wybieranie.

4. Wnioski

W wyniku przeprowadzonego eksperymentu udało się oszacować złożoność obliczeniową algorytmów sortowania przez wstawianie i sortowania przez wybieranie. Otrzymana eksperymentalna złożoność jest bliska teoretycznej wartości $O(n^2)$.

Zadanie 2.

1. Cel zadania

Celem zadania było porównanie dwóch metod sortowania list – metodą wstawiania oraz metodą wybierania.

2. Metody

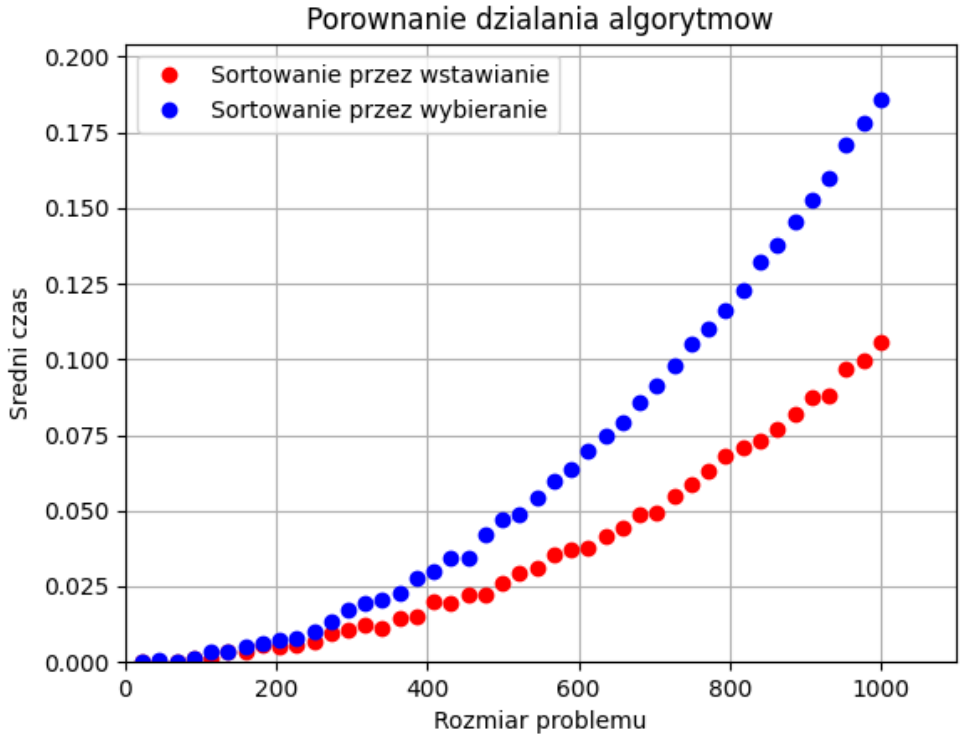
W doświadczeniu wykorzystano kilka klas stworzonych w języku Python. Odpowiedni projekt stworzono i skompilowano w środowisku Thonny na komputerze przenośnym o procesorze Intel Celeron CPU 1007U.

3. Przebieg doświadczenia i wyniki

Wykorzystano maksymalną długość listy ustaloną w zadaniu 1. Ze względu na to, że w zadaniu 1 opracowano metodę *mierz_czas* w ten sposób, by umożliwiała zastosowanie obu metod sortowania można ją było wykorzystać w tym zadaniu. Po wywołaniu metody *porownaj_metody* dla obiektu klasy *Sortowanie* powołanego do istnienia z parametrami *n* = 1000, *lprob* = 7, *ldlugosci* = 44 udało się uzyskać wykres zamieszczony na następnej stronie.

4. Wnioski

W wyniku przeprowadzonego doświadczania okazało się, że metoda sortowania przez wstawianie jest szybsza niż metoda sortowania przez wybieranie – różnice czasów dla krótkich list są niewielkie, ale rosną wraz ze wzrostem długości listy.



Wykres 3. Porównanie metod sortowania
(czerwone punkty – sortowanie przez wstawianie,
niebieskie – sortowanie przez wybieranie)