Algorytmy i struktury danych, Teleinformatyka, I rok

Raport z laboratorium nr: 1

Imię i nazwisko studenta: Filip Korus

nr indeksu: 413143

1. W pole poniżej wklej najważniejszy (według Ciebie) fragment kodu źródłowego z zajęć (maksymalnie 15 linii).

```
def insertion_sort(A: list) -> None:
  for i in range(1, len(A)):
    x = A[i]
    j = i - 1
    while j >= 0 and A[j] > x:
      A[j + 1] = A[j]
      j -= 1
    A[j + 1] = x

def merge_sort(A: list, start: int, end: int) -> None:
  if start < end:
    mid = (start + end) // 2
    merge_sort(A, start, mid)
    merge_sort(A, mid + 1, end)
    merge(A, start, mid, end)
```

Uzasadnij swój wybór.

Wybrałem ten fragment kodu, ponieważ na nim idealnie widać złożoności obliczeniowe obydwu algorytmów. W sortowaniu poprzez wstawianie z powodu pętli w pętli złożoność czasowa będzie o wiele większa niż w sortowaniu poprzez scalanie, gdzie żadna pętla nie występuje. Występują za to wywołania rekurencyjne.

2. Podsumuj wyniki uzyskane podczas wykonywania ćwiczenia. Co ciekawego zauważyłeś? Czego się nauczyłeś? Jeśli instrukcja zawierała pytania, odpowiedz na nie. Do sprawozdania możesz dodać wykresy jeśli jest taka potrzeba.

Dla 10^2 iteracji i losowo wygenerowanych ciągów długości 10^4 insertion sort potrzebował dużo więcej czasu na posortowanie tych samych zbiorów danych co merge sort oraz sortowanie wbudowane w język. Jest to spowodowane pętlą w pętli, więc jego złożoność to $O(N^2)$, a złożoność merge sorta to $O(N \cdot \log N)$.
Jednak merge sort, z powodu rekurencji, zużywa dużo więcej pamięci niż insertion sort. Sortowanie wbudowane w Pythona poradziło sobie najszybciej – co było do przewidzenia.
Poniżej załączam zrzut ekranu z tabelką z łącznymi, minimalnymi, maksymalnymi oraz średnimi czasami wykonywania poszczególnych algorytmów.

```
Amount of iterations = 100
Random list size = 10000
+-----------+----------+----------+----------+----------+
| SORT TYPE | SUM TIME | MIN TIME | MAX TIME | AVG TIME |
+===========+==========+==========+==========+==========+
| insertion | 243.491  | 2.356    | 2.673    | 2.435    |
+-----------+----------+----------+----------+----------+
| merge     | 2.864    | 0.027    | 0.044    | 0.029    |
+-----------+----------+----------+----------+----------+
| python    | 0.009    | 0        | 0.001    | 0.000    |
+-----------+----------+----------+----------+----------+
```