Выполнение практического задания по теме
«Производительность алгоритмов»

Студент группы М80-209М-23: Пешков Максим Юрьевич

Результаты выполнения сортировок (п. 1 – 4)

```
PS D:\Program Files\My_Desktop\python_projects\itl17> & C:/Users/Maksim/AppD
y"
1. Добавление в конец списка (O(1)):
    1000 элементов:     0 ms
   10000 элементов:  10.6 ms
  100000 элементов:   100 ms
 1000000 элементов: 1040.6 ms

2. Добавление в начало списка (O(n)):
    1000 элементов:     0 ms
   10000 элементов:     1 ms
  100000 элементов:  60.4 ms
 1000000 элементов:  5949 ms

3. Сортировка пузырьком (O(n^2)):
    1000 элементов:  33.4 ms
   10000 элементов: 2577.8 ms
  100000 элементов: 267019.4 ms

4. Стандартная сортировка (O(n log n)):
    1000 элементов:     0 ms
   10000 элементов:     0 ms
  100000 элементов:   5.2 ms
 1000000 элементов:  83.6 ms
PS D:\Program Files\My_Desktop\python_projects\itl17>
```

Ответы на вопросы

Временная сложность — это мера количества операций, которые выполняет алгоритм в зависимости от размера входных данных (обычно обозначается как n). Она описывает, как быстро растёт время выполнения алгоритма при увеличении n.

Временная сложность определяется с помощью анализа кода (подсчет количества вложенных циклов; определение того, как операции зависят от n) и экспериментального метода (замерка времени выполнения при различных n; постройка графиков и определение зависимостей).

В Python встроенный метод **list.sort() использует** алгоритм Timsort.

- Его сложность O(n log(n))
- Особенности данного алгоритма: он является гибридным (слияние + вставки), он устойчив (сохранение порядка равных элементов) и оптимизирован для реальных задач (частично отсортированных массивов данных).

Оптимизация добавления элементов в начало списка: использование двухсторонней очереди, хранение списка в обратном порядке, использование обратного списка.