В качестве программ для сравнения использования памяти я выбрал две программы для решения одной задачи: поиск простых чисел. Первый алгоритм — решето Эратосфена, второй — поиск простых чисел путём деления каждого очередного числа на уже найденные простые числа (если не нашли нулевого остатка от деления — простое число — добавляем в список найденных простых чисел).

Второй алгоритм, как показала практика в ДЗ-4, оказался 13 раз медленнее первого.

В работе используется компьютер с 64-разрядной ОС Windows и интерпретатором Python версии 3.8.3.

Алгоритм с использованием «решета Эратосфена» (файл task01.py) использует в работе 2 целочисленных переменных и 2 списка. Общий объём занимаемой памяти равен 902320 байт.

```
Переменная i: 28 байт
Переменная n: 28 байт
Переменная nmbrs: 824440 байт
Переменная primes: 77824 байт
Общий объём занимаемой памяти: 902320 байт
```

Второй алгоритм (файл task02.py) использует в работе 2 целочисленных переменных и 1 список. Общий объём занимаемой памяти равен 77880 байт.

```
Переменная i: 28 байт
Переменная n: 28 байт
Переменная primes: 77824 байт
Общий объём занимаемой памяти: 77880 байт
```

Выводы.

Первый алгоритм занимает памяти в 11,5 раз больше, чем второй, так как использует для работы массив чисел от 1 до n. Но, одновременно с такой большой разницей в использовании памяти, имеем такую же большую разницу в производительности, только наоборот: первый алгоритм в 13 раз быстрее второго.

Данный пример хорошо иллюстрирует сам принцип оценки алгоритмов для выполнения одной и той же задачи: либо по памяти, либо по производительности. В каждом конкретном случае, исходя из существующих ограничений и требований, разработчику нужно будет выбирать оптимальный алгоритм.