

# Тепляков Владислав БПИ2310

## SET 3

### A2

ID ссылки: 293133679

Ссылка на GitHub

#### Реализация ArrayGenerator

```
1  from random import randint
2
3  n = int(input())
4  arr = []
5  for i in range(0, n):
6      arr.append(randint(a=0, b=6001))
7  print(n)
8  print(*arr)
9  with open('full_tests.txt', 'a') as destination_file:
10     print(n, file=destination_file)
11     print(*arr, file=destination_file)
```

Выбрал генерацию тестов, запуск файлов и построение графика делать в питоне, поэтому генератор решил сделать тоже в питоне.

#### Реализация SortTester

```
import os, sys
import numpy as np
import matplotlib.pyplot as plt

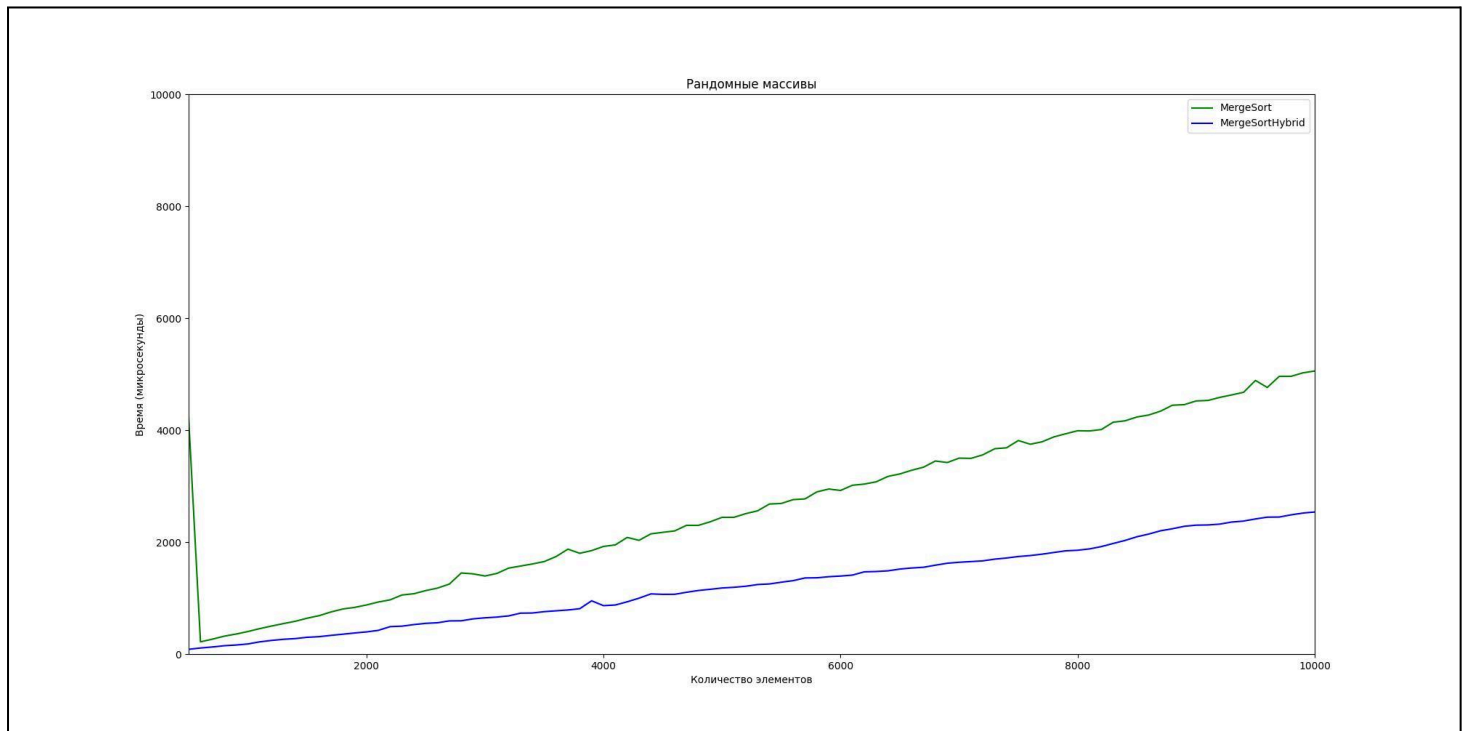
_, algo1, algo2, gen = sys.argv
arr_x = []
arr_algo1 = arr_algo2 = arr_algo3 = []
for i in range(500, 10001, 100):
    with open('for_gen.txt', 'w') as f:
        print(i, file=f)

    os.popen('python %s > test.txt %s < for_gen.txt' % (gen, gen))
    v1 = float(*os.popen('%s < test.txt' % algo1).readlines())
    v2 = float(*os.popen('%s < test.txt' % algo2).readlines())
    with open("result.txt", "a") as f:
        print(f"N = {i}", file=f)
        print(f"Time for MergeSort: ", "{:.8f}".format(v1), file=f)
        print(f"Time for MergeSortHybrid: ", "{:.8f}".format(v2), file=f)
        print("=====", file=f)

    arr_x.append(i)
    arr_algo1.append(v1)
    arr_algo2.append(v2)
```

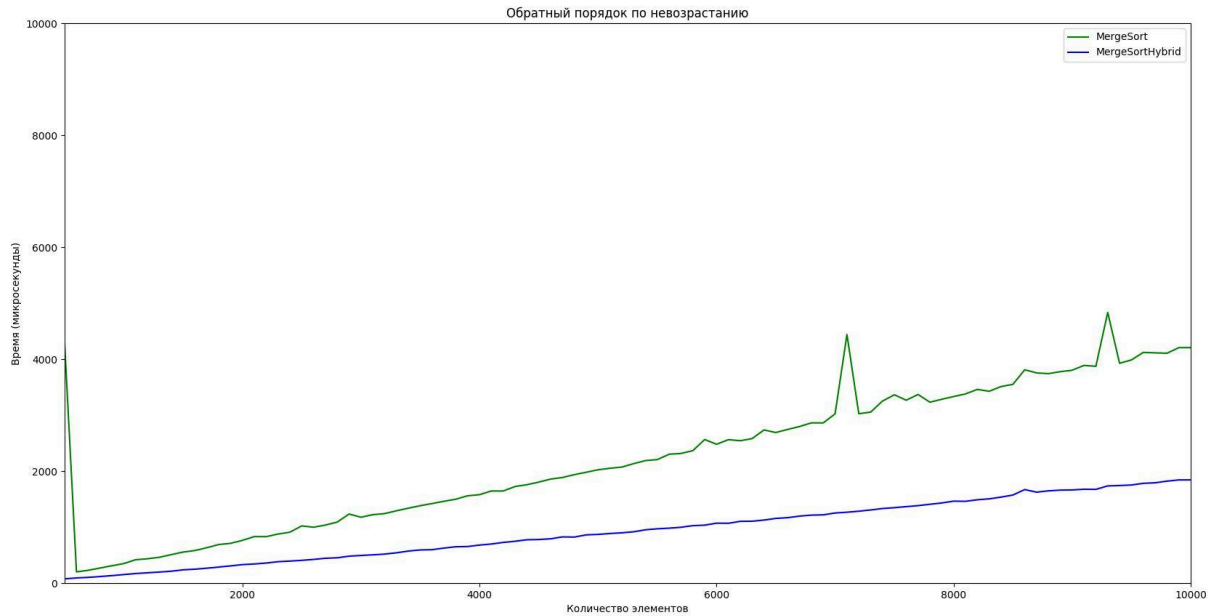
Запускаемый файл возвращает сколько он отработывал сортировку, соответственно скрипт на питоне потом обрабатывает данные и выводит их в файл.

**График 1 (рандомные массивы)**



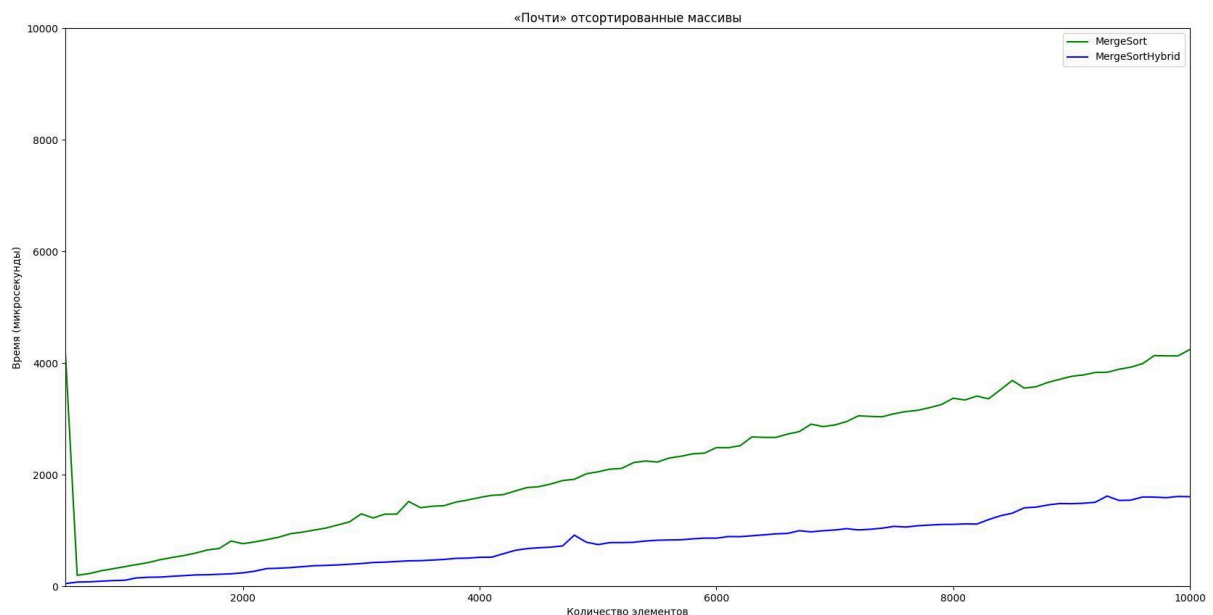
Можно заметить, что при маленьких  $n$  отличие в скорости работы минимально, но чем больше  $n$ , тем ярче прослеживается ускорение работы. Скорость работы MergeSort при  $n = 10000 \rightarrow 5056$  микросекунд, а алгоритма MergeSortHybrid в 2 раза меньше  $\rightarrow 2537$  микросекунд.

## График 2 (обратный порядок по невозрастанию)



Глобальных изменений не видно, кроме выбросов при отдельных  $n$ . Оба алгоритма в этом случае работают быстрее, например при  $n = 10000$ , MergeSort  $\rightarrow$  4205 микросекунд, а MergeSortHybrid  $\rightarrow$  1842 микросекунд

## График 3 («почти» отсортированные массивы)



Глобальный тренд сохраняется и в этом случае, только еще более заметный. При  $n = 10000$ , MergeSort работает за 4241 микросекунд, а MergeSortHybrid за 1604 микросекунд.

**Вывод:** MergeSortHybrid работает во всех представленных случаях в несколько раз быстрее, что означает полезность добавления сортировки вставками для маленьких  $n$