

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ПРАКТИЧЕСКОЙ РАБОТЕ №1.5
дисциплины «Алгоритмизация»

Выполнил:
Степанов Леонид Викторович
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение
средств вычислительной
техники и автоматизирование
систем», очная форма обучения

(подпись)

Руководитель практики:
Воронкин Р.А., канд. техн. наук,
доцент, доцент кафедры
инфокоммуникаций

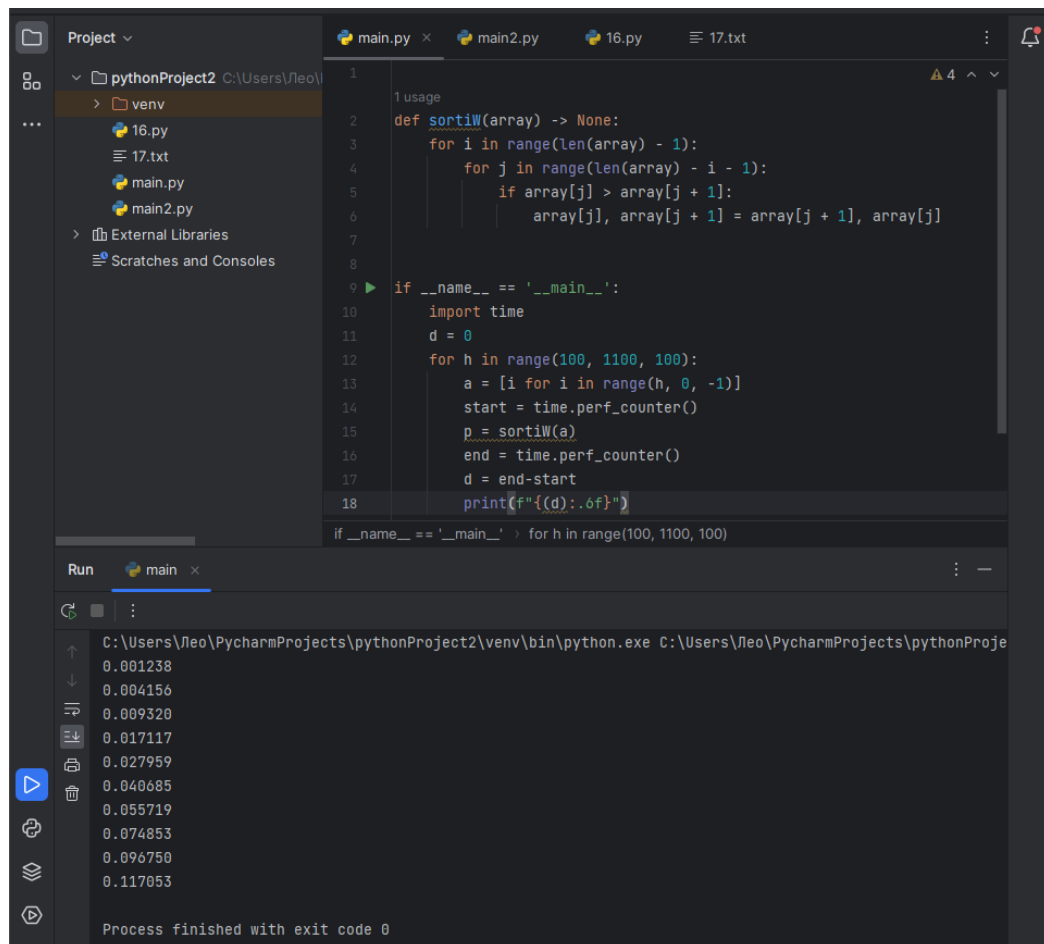
(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Порядок выполнения работы:

1. Написал программу (puzW.py) в которой измеряется время выполнения пузырьковой сортировки, метод пузырька заключается в том, что любые два подряд идущие элементы сравниваются и меньшее ставит левее большего и так с каждым элементом массива. Худший случай заключается в том, что элементы расположены в убывании.



```
1 usage
2 def sortiW(array) -> None:
3     for i in range(len(array) - 1):
4         for j in range(len(array) - i - 1):
5             if array[j] > array[j + 1]:
6                 array[j], array[j + 1] = array[j + 1], array[j]
7
8
9 if __name__ == '__main__':
10     import time
11     d = 0
12     for h in range(100, 1100, 100):
13         a = [i for i in range(h, 0, -1)]
14         start = time.perf_counter()
15         p = sortiW(a)
16         end = time.perf_counter()
17         d = end-start
18         print(f"{(d):.6f}")
19
20 if __name__ == '__main__':
21     for h in range(100, 1100, 100):
```

Run console output:

```
C:\Users\Leo\PycharmProjects\pythonProject2\venv\bin\python.exe C:\Users\Leo\PycharmProjects\pythonProje
0.001238
0.004156
0.009320
0.017117
0.027959
0.040685
0.055719
0.074853
0.096750
0.117053
Process finished with exit code 0
```

Рисунок 1 – Результат выполнения puzW.py

n	100	200	300	400	500	600	700	800	900	1000	5500
t	0,001032	0,00411	0,009329	0,01697	0,027323	0,040453	0,0572	0,07999	0,094113	0,121992	0,452512
n^2	10000	40000	90000	160000	250000	360000	490000	640000	810000	1000000	3850000
n*t	0,1032	0,822	2,7987	6,788	13,6615	24,2718	40,04	63,992	84,7017	121,992	359,1709

Рисунок 2 – Таблица данных puzW.py в Exel

При помощи метода наименьших квадратов вывели систему уравнений:
 $3850000a + 5500b = 359,1709$ и $5500a + 10b = 0,4525$, решив которую мы нашли график функции: $y = 0,0001337x - 0,028275$



Рисунок 3 – График функции $y = 0,0001337x - 0,028275$

2. Написал программу (ruzSR.py), в которой измеряется время использования метода пузырьковой сортировки в среднем случае, когда элементы массива вводятся случайно.

```

1 usage
2 def sortiSR(array) -> None:
3     for i in range(len(array) - 1):
4         for j in range(len(array) - i - 1):
5             if array[j] > array[j + 1]:
6                 array[j], array[j + 1] = array[j + 1], array[j]
7
8 if __name__ == '__main__':
9     from random import randint
10    from math import sqrt
11    import time
12    for i in range(100, 1000, 100):
13        print("---", i, "---")
14        for l in range(1, 31):
15            a = [0 * t for t in range(i)]
16            for k in range(i):
17                a.append(randint(100, 1000))
18            start = time.perf_counter()
19            p = sortiSR(a)
20            end = time.perf_counter()
21            d = end - start
22            print(f"{d:.8f}")
23        print("-----")

```

Run main2

```

C:\Users\Лео\PycharmProjects\pythonProject2\venv\bin\python.exe C:\Users\Лео\PycharmProjects\pythonProje
--- 100 ---
0.00161410
0.00162930
0.00161520
0.00162170
0.00161230
0.00162570
0.00164010
0.00164510
0.00164950
0.00165520
0.00163940

```

Рисунок 4 – Результат выполнения программы ruzSR.py

n	100	200	300	400	500	600	700	800	900
	0,0016141	0,0070556	0,0166721	0,0335798	0,047847	0,0703422	0,0981309	0,13973	0,1696325
	0,0016293	0,006963	0,016713	0,035812	0,0481516	0,0713674	0,1017204	0,1274296	0,1647315
	0,0016152	0,0070505	0,0165369	0,0304582	0,0481478	0,0788926	0,1058029	0,1367485	0,1683737
	0,0016217	0,007023	0,0168242	0,0303629	0,048743	0,0695978	0,0966826	0,13602	0,1676411
	0,0016123	0,0070886	0,0168519	0,0304549	0,0500205	0,0710445	0,1063378	0,1291939	0,1664938
	0,0016257	0,0068946	0,0168204	0,0305457	0,049207	0,0734726	0,0976117	0,135247	0,1661329
	0,0016401	0,0068292	0,01681	0,0303175	0,0494037	0,0712334	0,1015459	0,1464179	0,1643153
	0,0016451	0,0069563	0,0167496	0,0308199	0,0484793	0,0755413	0,1035408	0,1276685	0,163609
	0,0016495	0,0070057	0,0163951	0,0304588	0,0485671	0,074394	0,1061722	0,1292541	0,1711061
	0,0016552	0,0071621	0,0167161	0,0302149	0,0489196	0,0707114	0,1010941	0,1295243	0,1666429
	0,0016394	0,0070169	0,0169026	0,0303667	0,0503027	0,0712773	0,101301	0,1270424	0,1662101
	0,0016378	0,0071341	0,0166992	0,0300688	0,0475082	0,0766566	0,109882	0,1321342	0,168157
	0,0016388	0,0071507	0,0168972	0,0306287	0,0481928	0,0703933	0,1037267	0,1277854	0,1830035
	0,0016494	0,0069513	0,0167026	0,0300594	0,0482824	0,072634	0,1128532	0,1391388	0,1730032
	0,0016961	0,0070364	0,0167101	0,0305533	0,0481641	0,0698309	0,1023905	0,1282037	0,1764042
	0,0016382	0,0071523	0,0171016	0,0310132	0,0493812	0,0707936	0,0974712	0,141619	0,187317
	0,0016211	0,0070436	0,0167694	0,0307575	0,0511118	0,0706914	0,0963862	0,1408924	0,1750642
	0,001642	0,007077	0,0169304	0,0309426	0,0534479	0,0726533	0,0994582	0,1275609	0,1633265
	0,001652	0,0069651	0,0167111	0,0309627	0,0520364	0,0716467	0,1025295	0,1293299	0,1646562
	0,001593	0,0071294	0,016704	0,0312265	0,0480865	0,0725365	0,0956416	0,1294398	0,163816
	0,0016335	0,0068338	0,0165798	0,0312527	0,0481963	0,0715493	0,0976093	0,1275516	0,1633378
	0,0016648	0,0069322	0,0166747	0,0358171	0,0486387	0,0766211	0,0968522	0,1286041	0,1631952
	0,0017124	0,0069433	0,0208689	0,0311436	0,048114	0,0701432	0,0977767	0,1283644	0,1662405
	0,0017224	0,0069163	0,0180982	0,0343488	0,0484749	0,0728225	0,1003572	0,134426	0,1747539
	0,0016521	0,0071232	0,0167648	0,0341031	0,0482167	0,0814257	0,1030421	0,1280855	0,1684499
	0,0016485	0,0069787	0,0164611	0,0305393	0,0519823	0,0710421	0,0985859	0,1319764	0,1649624
	0,0016434	0,0070646	0,0166734	0,0308639	0,0527587	0,0711914	0,0994726	0,1401368	0,1710481
	0,0017003	0,0071036	0,0167426	0,0301837	0,0500886	0,0707696	0,0960209	0,1276778	0,164613
	0,0016505	0,0069964	0,0168554	0,0301703	0,0580577	0,0710532	0,1051615	0,12865	0,1644874
	0,0016313	0,0068007	0,0168242	0,0307942	0,0503931	0,0707965	0,0982601	0,1280709	0,1643714
e[n]	0,00164584	0,00701261	0,01692535	0,03129402	0,04963072	0,07243751	0,10111393	0,13213079	0,16850321

Рисунок 5 – Таблица данных ruзSR.py

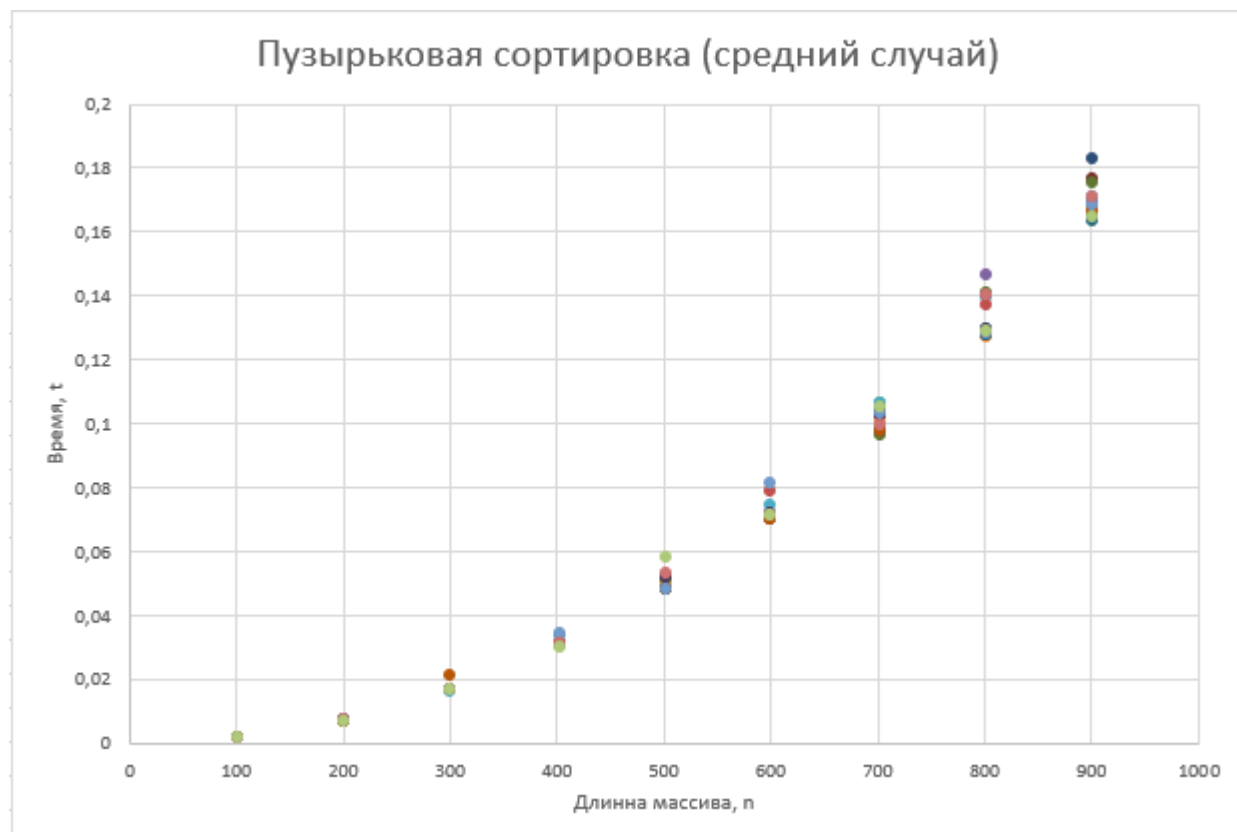


Рисунок 6 – График по данным ruзSR.py

При помощи метода наименьших квадратов вывели систему уравнений:

$$1533399900000000a + 2025000000b + 285000c = 325023,8751709, 2025000000a + 2850000b + 4500c = 427,5405628, 2850000a + 4500b + 9c = 0,597668217,$$

решив которую мы нашли график функции: $y = -1,36E - 10x^2 + 0,00015b - 0,0005$



Рисунок 7 – График функции $y = -1,36E - 10x^2 + 0,00015b - 0,0005$

Далее наложили на график пределы погрешностей:



Рисунок 8 – График с пределами погрешности

Вывод: в результате проделанной работы было выяснено, что количество элементов в массиве влияет на время сортировки и О-большое пузырьковой сортировки – $O(n^2)$.