Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ №1**

Оценка сложности алгоритма

Вариант № 7

Преподаватель \_\_\_\_\_\_\_\_\_\_ И. В. Матковский

подпись, дата

Студент КИ20-08Б 032049025 \_\_\_\_\_\_\_\_\_\_ А. Н. Головань

номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2021

**1 Порядок выполнения работы**

Разработать для решения поставленной задачи два алгоритма с разными уровнями сложности.

**2 Задание**

Отсортировать элементы одномерного массива по возрастанию.

**3 Цель работы**

Научиться оценивать сложность алгоритмов.

**4 Код программы**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42 | from time import time  import os  from random import randint  os.system("cls")  def sort\_1(b):  a = b.copy()  for i in range(len(a)):  for j in range(i):  if a[i] < a[j]:  a[i],a[j] = a[j],a[i]  return a  def sort\_2(b):  a = b.copy()  swapped = True  while swapped:  swapped = False  for i in range(len(a) - 1):  if a[i] > a[i + 1]:  a[i], a[i + 1] = a[i + 1], a[i]  swapped = True  return a  a = []  for i in range(10000):  a.append(randint(0,10))    start\_time = time()  sort = sort\_1(a)  end\_time = time()  print("list sort by bubble\_2 for {} sec".format(end\_time-start\_time))  start\_time = time()  sort = sort\_2(a)  end\_time = time()  print("list sort by bubble\_1 for {} sec".format(end\_time-start\_time)) |

**5 Теоретические оценки сложности алгоритмов**

Данная программа использует два разных алгоритма для проверки числа N на простоту – проверка на остаток от деления в цикле и решето Эратосфена.

1. Очевидно, что для выполнения первого алгоритма приходится N\*N операций на каждый элемент списка. Таким образом, сложность этого фрагмента составит O(N2)
2. Для второго алгоритма действует та же самая сложность O(N2), но при этом цикл внутри цикла выполняется для всех элементов списка, поэтому количество итераций увеличиться в два раза, но сложность так и останется O(N2).

**6 Экспериментальные оценки сложности алгоритмов**

Таблица 1 - зависимость времени от объема входных данных

|  |  |  |  |
| --- | --- | --- | --- |
|  | Количество элементов списка | Количество итерация | Время затраченное на алгоритм (в секундах) |
| Алгоритм №1 | 1000 | 499500 | 0.0359с |
| Алгоритм №2 | 1000 | 904095 | 0.1032с |

Таблица 2 - зависимость времени от объема входных данных

|  |  |  |  |
| --- | --- | --- | --- |
|  | Количество элементов списка | Количество итерация | Время затраченное на алгоритм (в секундах) |
| Алгоритм №1 | 10000 | 49995000 | 3.556с |
| Алгоритм №2 | 10000 | 91010898 | 10.755с |