МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
 РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский государственный университет геодезии и картографии»  
(МИИГАиК)  
Факультет геоинформатики и информационной безопасности  
Кафедра геоинформационных систем и технологий

**Лабораторная работа №1**  
**«Библиотечные реализации алгоритмов и их скорость»**

Проверил: Выполнил:

Лебедев Е.Д. Студент группы: 2024-ФГиИБ-ПИ-1б

Николаев М.А.

Москва 2025

**Цель работы**

Выполнить предоставленные задачи на языке **Python.**

**Задание (вариант №19)**

Оценка вычислительной сложности и потребления памяти: проверка элемента на вхождение в множество

**Анализ задания**

Алгоритм должен корректно работать и считать время своей работы.

Моё предположение, что алгоритм будет вести себя нелинейно, так как поиск элемента должен происходить быстрее остальных потенциальных функций.

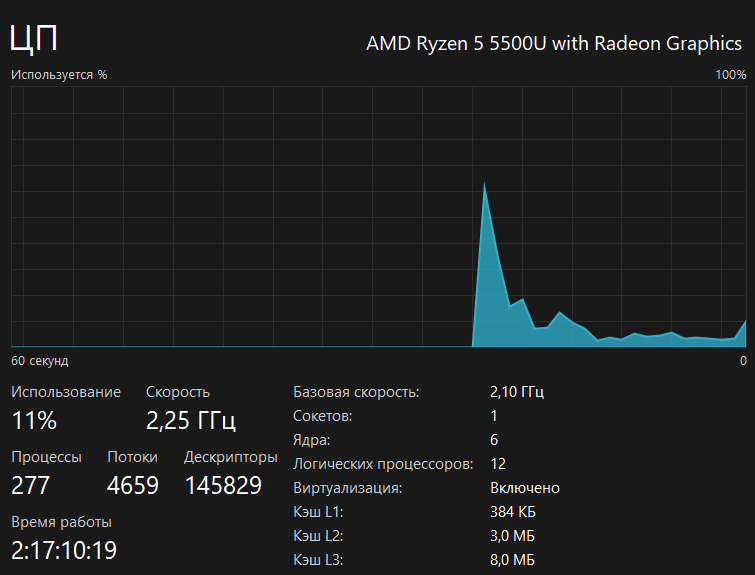
*Входные данные:*

На вход программа получает количество элементов, которые надо будет найти и сами элементы.

*Выходные данные:*

На выход программа выводит найден ли элемент и сколько заняло на его поиск.

Используемый процессор:



**Листинг программы**Ссылка на **GitHub**: <https://github.com/Ep1cMax/Second_Semester/tree/main/Works>

import time

def checkElementInSet(element, exmplSet):

startingTime = time.perf\_counter()

isPresent = element in exmplSet

endingTime = time.perf\_counter()

timeFormul = endingTime - startingTime

return (isPresent, timeFormul)

if \_\_name\_\_ == "\_\_main\_\_":

M = int(input('Введите кол-во нулей для заданного множества: '))

exmplSet = set(range(1, 10\*\*M+1))

N = int(input('Введите количество элементов для поиска: '))

elementsForTest = []

c = 1

for i in range(N):

inpElem = int(input(f'Элемент {c}: '))

elementsForTest.append(inpElem)

c += 1

print()

for elem in elementsForTest:

result, timeTaken = checkElementInSet(elem, exmplSet)

print(f"Элемент {elem}: {'найден' if result else 'не найден'}")

print(f"Время поиска: {timeTaken:.10f} секунд\n")

**Результаты работы программы:**

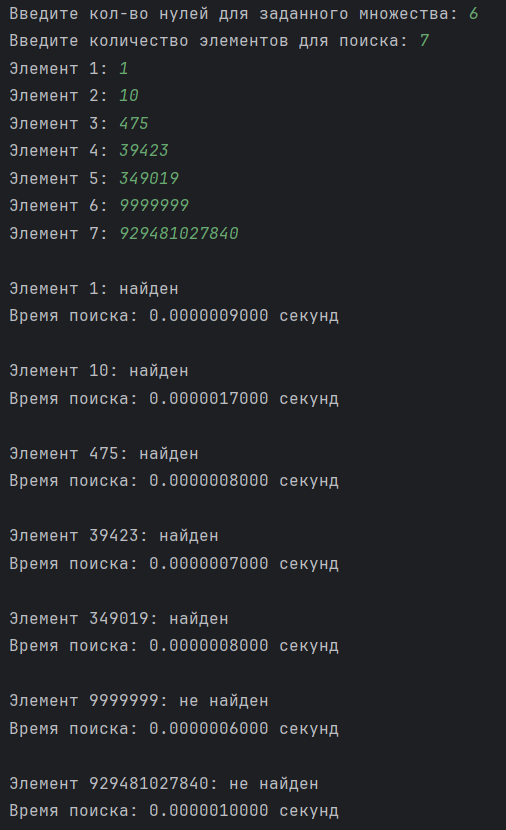


Рис. 1. – Результат работы программы

**Блок-схема**

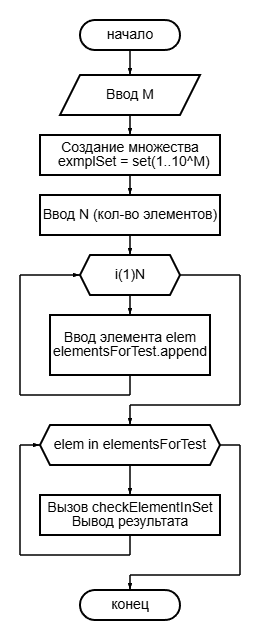
****

Рис. 2. – Блок-схема

Граф 1. – Зависимость времени поиска от размерности

|  |  |
| --- | --- |
| Размерность | Время |
| 10 | 13 |
| 10 | 3 |
| 100 | 3 |
| 10000 | 10 |
| 20000 | 7 |
| 30000 | 8 |
| 60000 | 7 |
| 100000 | 8 |
| 1000000 | 11 |
| 50000000 | 5 |
| 250000000 | 14 |
| 3000000000 | 6 |

Табл. 1. – Замеры.

**Вывод**

Алгоритм ведёт себя нелинейно и зависит от искомого элемента и размера массива. Сложность функции создания массива O(10^M). Сложность поиска элементов O(N). Итоговая оценка алгоритма: O(10^M + N).