**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информатика»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

по дисциплине **«Алгоритмы и структуры данных»**

на тему: **«Оценка трудоёмкости задачи»**

Выполнил: студент гр. ИП-21

М. С. Есис

Принял: преподаватель

В. Н. Шибеко

Дата сдачи отчета: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата допуска к защите: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата защиты: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Гомель 2024

**Цель работы**: ознакомиться с наиболее эффективным и оптимальным решения задач, связанных с обработкой и организацией данных.

**Задания к лабораторной работе:**

**Задание1.**

Дано: массив чисел (задать массив на основе списка)

1. Разработать и описать алгоритм решения задачи
2. Построить блок-схему и оценить по блок-схеме его трудоемкость
3. Реализовать алгоритм на Питоне
4. Используя модуль питона time – оценить длительность реализации для разных значений n – размера массива (вектора)
   * N=100
   * N=1000
   * N=10000
   * N=100000

5) Рассчитать асимптотическую оценку.

6) Построить табулированную функцию и визуально пояснить ее вид и сходство (различие) с асимптотической оценкой

**Индивидуальные задания**

1. Подсчитать утроенную сумму элементов вектора, по значению принадлежащих интервалу [A,B]

Текст программы:

from random import randint

import matplotlib.pyplot as plt

import time

def triple\_sum(vector, A, B):

    triple\_sum = 0

    for num in vector:

        if A <= num <= B:

            triple\_sum += num \* 3

    return triple\_sum

def generate(size):

    return list(randint(-100, 100) for i in range(size))

N = [100, 1000, 10000, 100000]

A = -50

B = 50

durations = []

for n in N:

    vector = generate(n)

    start\_time = time.perf\_counter\_ns()

    sum = triple\_sum(vector, A, B)

    end\_time = time.perf\_counter\_ns()

    duration = end\_time - start\_time

    durations.append(duration)

    print(f"Размер вектора: {n}, Длительность: {duration:.12f} секунд , Сумма: {sum}")

    sum = 0

plt.plot(N, durations)

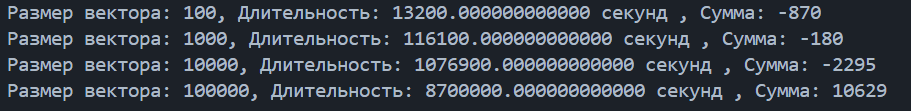
plt.xlabel("Размер вектора")

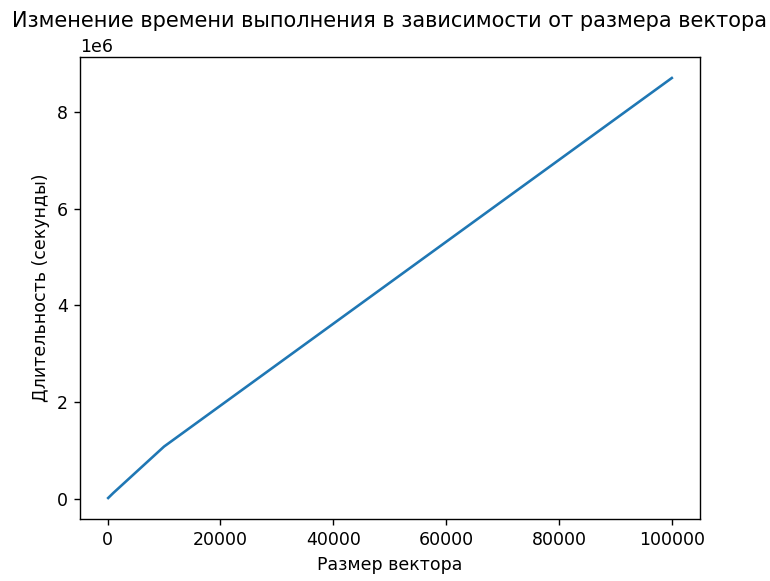
plt.ylabel("Длительность (секунды)")

plt.title("Изменение времени выполнения в зависимости от размера вектора")

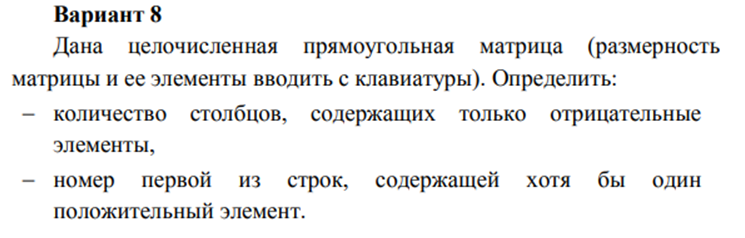
plt.show()

Результат:





**Задание 2.**



Текст программы:

from random import randint

import matplotlib.pyplot as plt

import time

def count\_negative\_cols(matrix):

    rows = len(matrix)

    cols = len(matrix[0])

    negative\_cols = 0

    for j in range(cols):

        has\_positive = False

        for i in range(rows):

            if matrix[i][j] >= 0:

                has\_positive = True

                break

        if not has\_positive:

            negative\_cols += 1

    return negative\_cols

def find\_positive\_row(matrix):

    rows = len(matrix)

    cols = len(matrix[0])

    positive\_row = -1

    for i in range(rows):

        has\_positive = False

        for j in range(cols):

            if matrix[i][j] > 0:

                has\_positive = True

                break

        if has\_positive:

            positive\_row = i

            break

    return positive\_row

rows = int(input("Введите количество строк: "))

cols = int(input("Введите количество столбцов: "))

matrix = [[0]\*cols for i in range(rows)]

for i in range(rows):

    for j in range(cols):

        matrix[i][j] = randint(-10, 1)

for row in matrix:

    print(row)

start\_time = time.time()

negative\_cols\_count = count\_negative\_cols(matrix)

pozitiv\_row\_index = find\_positive\_row(matrix)

end\_time = time.time()

print("Количество столбцов, содержащих только отрицательные элементы:", negative\_cols\_count)

if pozitiv\_row\_index != -1:

    print("Номер первой строки, содержащей хотя бы 1 положительный элемент:", pozitiv\_row\_index)

else:

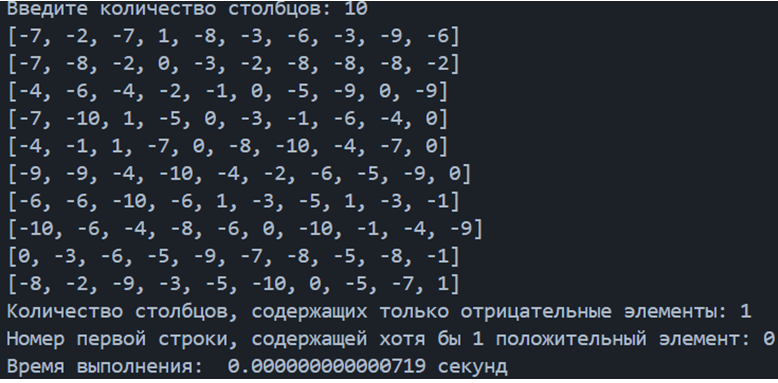
    print("Нет строк, содержащих положительный элемент")

execution\_time = end\_time - start\_time

formatted\_execution\_time = "{:.15f}".format(execution\_time)

print("Время выполнения: ", formatted\_execution\_time, "секунд")

Результат:



**Вывод**: ознакомиться с наиболее эффективным и оптимальным решения задач, связанных с обработкой и организацией данных.