**НЕГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ ЧАСТНОЕ** **УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ** **«МОСКОВСКИЙ ФИНАНСОВО-ПРОМЫШЛЕННЫЙ УНИВЕРСИТЕТ** **“СИНЕРГИЯ”»**

|  |  |  |
| --- | --- | --- |
| **Факультет/Институт** |  | Информационных технологий |
|  |  | (наименование факультета/ Института) |
| **Направление/специальность** |  | 09.02.07 ИСиП |
| **подготовки:** |  | (код и наименование направления /специальности подготовки) |
| **Форма обучения:** |  | очная |
|  |  | (очная, очно-заочная, заочная) |
|  |  |  |

**Отчет по лабораторной работе №4**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **на тему** |  | Массивы и строки | | |
|  |  | (наименование темы) | | |
|  |  |  | | |
| **по дисциплине** | | |  | **Разработка модулей программного обеспечения для компьютерных систем** |
|  | | |  | (наименование дисциплины) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Обучающийся** |  | Хакимжанов Салим Фарходович |  |  |
|  |  | (ФИО) |  | (подпись) |
| **Группа** |  | ДКИП-203прог |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Преподаватель** |  | Сибирев Иван Валерьевич |  |  |
|  |  | (ФИО) |  | (подпись) |

**Москва 2024 г.**

**Лабораторная работа №. 4 «Массивы и строки**»

**Лабораторная работа 5. Одномерные массивы.**

**Вариант 18**

Ниже представлен пример решения задачи на Python. Предполагается, что массив содержит вещественные числа.

import math

def analyze\_array(arr, C):

# 1) Количество элементов массива, меньших C

count\_less\_C = sum(1 for x in arr if x < C)

# 2) Сумма целых частей элементов массива, расположенных после последнего отрицательного элемента

last\_neg\_index = -1

for i in range(len(arr)-1, -1, -1):

if arr[i] < 0:

last\_neg\_index = i

break

if last\_neg\_index == -1:

# Отрицательных элементов нет, значит берем все элементы

after\_neg = arr

else:

after\_neg = arr[last\_neg\_index+1:]

sum\_int\_parts = sum(math.floor(x) for x in after\_neg)

# 3) Преобразование массива: сначала элементы, отличающиеся от максимального не более чем на 20%, потом остальные

max\_val = max(arr)

threshold = max\_val \* 0.8 # минимум для попадания в первую группу

group1 = [x for x in arr if x >= threshold]

group2 = [x for x in arr if x < threshold]

transformed\_arr = group1 + group2

return count\_less\_C, sum\_int\_parts, transformed\_arr

# Пример использования:

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

arr = [1.5, -2.3, 3.7, 4.0, 5.5, -1.0, 6.8, 4.5, 2.0]

C = 4.0

count\_less\_C, sum\_int\_parts, transformed\_arr = analyze\_array(arr, C)

print(f"Количество элементов меньше {C}: {count\_less\_C}")

print(f"Сумма целых частей элементов после последнего отрицательного: {sum\_int\_parts}")

print(f"Преобразованный массив: {transformed\_arr}")

Найти еще

### Объяснение:

* Считаем количество элементов, меньших C*C*.
* Находим индекс последнего отрицательного элемента, затем суммируем целые части элементов после него (если отрицательных нет — берем весь массив).
* Находим максимальный элемент, вычисляем порог 80% от максимума.
* Формируем новый массив: сначала элементы, которые не отличаются от максимума более чем на 20% (то есть x≥0.8×max⁡*x*≥0.8×max), потом остальные.

**Лабораторная работа 6. Двумерные массивы**

**Вариант 18**

1. Определяет количество строк, содержащих хотя бы один нулевой элемент.
2. Находит номер столбца, в котором находится самая длинная серия одинаковых элементов.

def analyze\_matrix(matrix):

# 1) Количество строк, содержащих хотя бы один нулевой элемент

count\_zero\_rows = sum(1 for row in matrix if 0 in row)

# 2) Номер столбца с самой длинной серией одинаковых элементов

max\_length = 0

column\_with\_max\_series = -1

num\_rows = len(matrix)

num\_cols = len(matrix[0]) if num\_rows > 0 else 0

for col in range(num\_cols):

current\_length = 1 # Длина текущей серии

for row in range(1, num\_rows):

if matrix[row][col] == matrix[row - 1][col]:

current\_length += 1

else:

if current\_length > max\_length:

max\_length = current\_length

column\_with\_max\_series = col

current\_length = 1 # Сброс длины серии

# Проверяем последнюю серию в столбце

if current\_length > max\_length:

max\_length = current\_length

column\_with\_max\_series = col

return count\_zero\_rows, column\_with\_max\_series

# Пример использования:

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

matrix = [

[1, 2, 0, 4],

[5, 6, 7, 8],

[9, 0, 1, 2],

[3, 3, 3, 4],

[5, 5, 5, 5]

]

count\_zero\_rows, column\_with\_max\_series = analyze\_matrix(matrix)

print(f"Количество строк, содержащих хотя бы один нулевой элемент: {count\_zero\_rows}")

print(f"Номер столбца с самой длинной серией одинаковых элементов: {column\_with\_max\_series + 1}") # +1 для 1-индекса

Найти еще

### Объяснение:

1. **Количество строк с нулем**: Мы используем генераторное выражение для подсчета строк, в которых присутствует хотя бы один нулевой элемент.
2. **Поиск столбца с самой длинной серией одинаковых элементов**:
   * Для каждого столбца мы проходим по всем строкам и считаем длину текущей серии одинаковых элементов.
   * Если текущая серия заканчивается (элементы различаются), мы проверяем, является ли она самой длинной, и обновляем соответствующий столбец.
   * В конце проверяем последнюю серию, чтобы убедиться, что она также учтена.

**Лабораторная работа 7. Строки**

**Вариант 18**

Вот пример программы на Python, которая считывает текст из файла, разбивает его на предложения и выводит их в обратном порядке:

def read\_file\_and\_reverse\_sentences(file\_path):

try:

with open(file\_path, 'r', encoding='utf-8') as file:

# Чтение содержимого файла

text = file.read()

# Разделение текста на предложения

# Здесь используем '. ' как разделитель предложений

sentences = text.split('. ')

# Удаляем возможные пробелы и пустые строки

sentences = [sentence.strip() for sentence in sentences if sentence]

# Выводим предложения в обратном порядке

for sentence in reversed(sentences):

print(sentence.strip() + '.') # Добавляем точку в конце для завершения предложения

except FileNotFoundError:

print(f"Файл '{file\_path}' не найден.")

except Exception as e:

print(f"Произошла ошибка: {e}")

# Пример использования:

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

file\_path = 'text.txt' # Замените на путь к вашему файлу

read\_file\_and\_reverse\_sentences(file\_path)

Найти еще

### Объяснение:

1. **Чтение файла**: Программа открывает файл с указанным путем и считывает его содержимое.
2. **Разделение на предложения**: Текст разбивается на предложения с использованием разделителя .  (точка и пробел). Это может быть изменено в зависимости от формата текста.
3. **Удаление пробелов**: Убираем лишние пробелы и пустые строки.
4. **Вывод в обратном порядке**: Используя reversed(), выводим предложения в обратном порядке, добавляя точку в конце каждого предложения для корректного завершения.