МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ

УНИВЕРСИТЕТ им. В. Г. ШУХОВА» (БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения

вычислительной техники и автоматизированных

систем

**Лабораторная работа №11**

по дисциплине: ООП

тема: «Знакомство с языком программирования Python. Базовые структуры данных»

Выполнил: студент группы ВТ-231

Масленников Д. А.

Проверили:  
Буханов Д. Г.

Морозов Д. А.

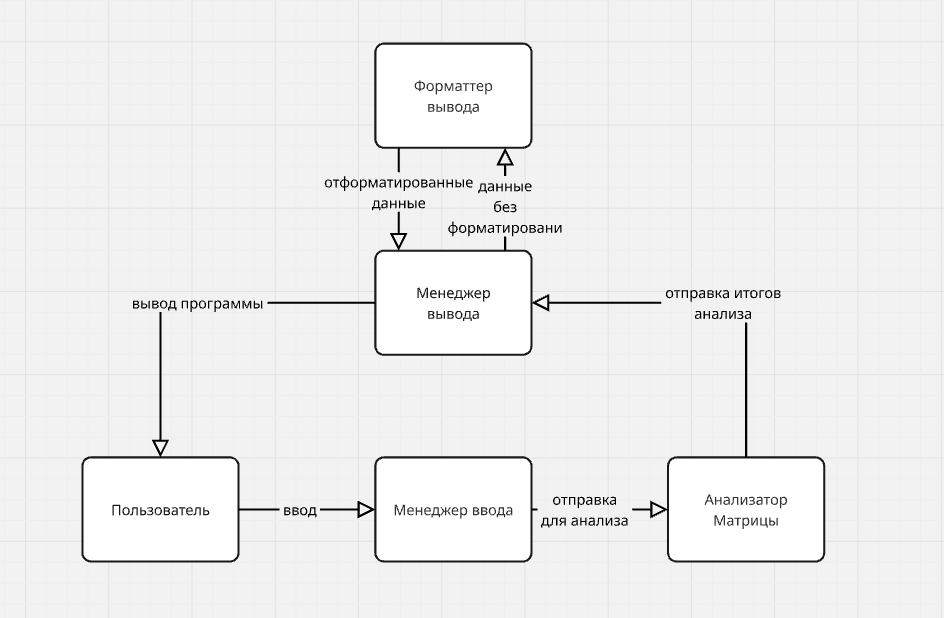
Белгород 2025

**Цель работы**: Познакомится с базовыми конструкциями языка. Получить навык создания простых приложений. Изучить базовые типы.

**Вариант 14**

На вход подаются данные в форме двумерных «матриц», количество матриц заранее не определено, разделителем между матрицами являются строки. Для каждой матрицы найти все, которые удовлетворяют следующему условию: суммы четных и нечетных элементов каждой строки равны. Форма матрицы может быть не полной. Формат вывода требуется соблюсти

**Обьектная декомпозиция:**

****

**Код программы:**

**class** **Matrix**:

**def** **\_\_init\_\_**(self, data):

self.data = data

self.rows = len(data)

self.cols = len(data[**0**])

**def** **\_\_eq\_\_**(self, other):

**return** self.sum\_even\_odd\_per\_row() == other.sum\_even\_odd\_per\_row()

**def** **\_\_len\_\_**(self):

**return** len(self.data)

**def** **sum\_even\_odd\_per\_row**(self):

result = []

**for** row **in** self.data:

sum\_even = sum(x **for** x **in** row **if** x % **2** == **0**)

sum\_odd = sum(x **for** x **in** row **if** x % **2** != **0**)

result.append((sum\_even, sum\_odd))

**return** result

**class** **MatrixComparator**:

**@staticmethod**

**def** **compare\_matrices**(matrices):

result = []

**for** i **in** range(len(matrices)):

**for** j **in** range(i+**1**, len(matrices)):

**if** matrices[i] == matrices[j]:

result.append((matrices[i], matrices[j]))

**return** result

**@staticmethod**

**def** **format\_comparison\_result**(pair):

m1, m2 = pair

result = []

line = ''

**for** row\_idx **in** range(len(m1)):

row1 = m1.data[row\_idx]

row2 = m2.data[row\_idx]

line = f"{' '.join(map(str, row1))}**\t**{' '.join(map(str, row2))}"

result.append(line)

**return** '**\n**'.join(result)

**class** **MatrixReader**:

**@staticmethod**

**def** **read\_from\_string**(input\_data):

matrices = []

current\_matrix = []

**for** line **in** input\_data.split('**\n**'):

**if** line:

row = list(map(int, line.split()))

current\_matrix.append(row)

**else**:

matrices.append(Matrix(current\_matrix))

current\_matrix = []

matrices.append(Matrix(current\_matrix))

**return** matrices

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

input\_text = """1 2

3 4

5 6

1 3

4 3

4 6

2 1

2 1 1 1 2

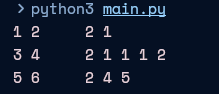
2 4 5"""

matrices = MatrixReader.read\_from\_string(input\_text)

matching\_pairs = MatrixComparator.compare\_matrices(matrices)

**for** pair **in** matching\_pairs:

print(MatrixComparator.format\_comparison\_result(pair))

Вывод программы: