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.

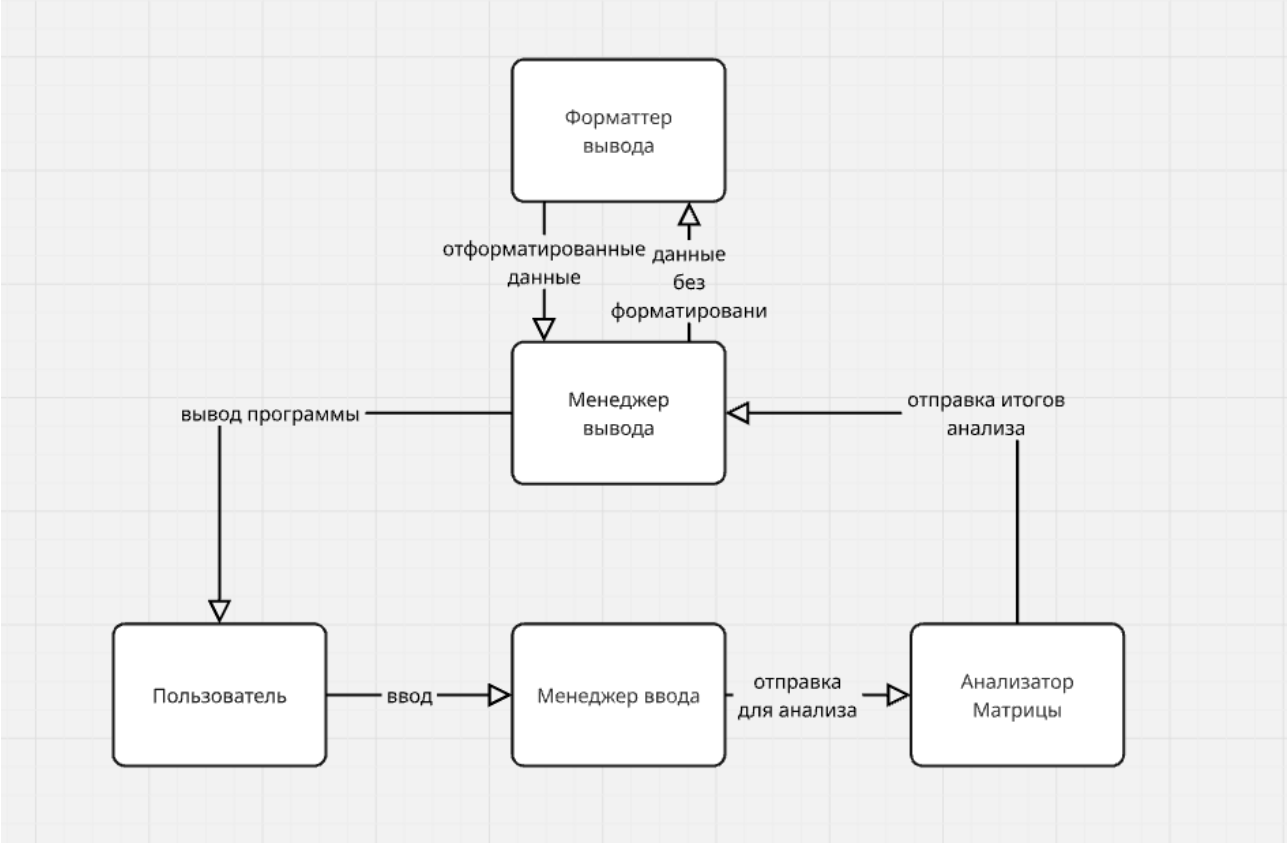
Белгород 2025

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: