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: Объектно-ориентированное программирование

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

Выполнил: ст. группы ПВ-233

Ситников Алексей Павлович

Проверил:

Белгород 2025 г.

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

Вариант 13 (6)

В соответствии с вариантом задания требуется выполнить объектную декомпозицию задачи. В качестве одного из обязательных объектов выделить «матрицу». Для реализации соблюдения условия задачи требуется использовать возможности перегрузки операторов. При выводе также требуется выполнить перегрузку соответствующего оператора.

Задание варианта: 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: сумма соответствующих столбцов равна. Форма матрицы может быть не полной. Формат вывода требуется соблюсти.

Класс Matrix:

```
from numpy.matrixlib.defmatrix import matrix


class Matrix:

    def __init__(self):
        self.matrix = []

    def add_row(self, row):
        self.matrix.append(row)


    def get_sumCol(self):
        max_col = 0

        for i in self.matrix:
            max_col = max(max_col, len(i))

        col_sum = [0] * max_col

        for row in range(0, len(self.matrix)):
            for col in range(0, len(self.matrix[row])):
                col_sum[col] += self.matrix[row][col]
        return col_sum

    def print_row(self, index):
        if index < len(self.matrix):
            return ' '.join(map(str, self.matrix[index]))
        return ' '

    def get_count_rows(self):
```

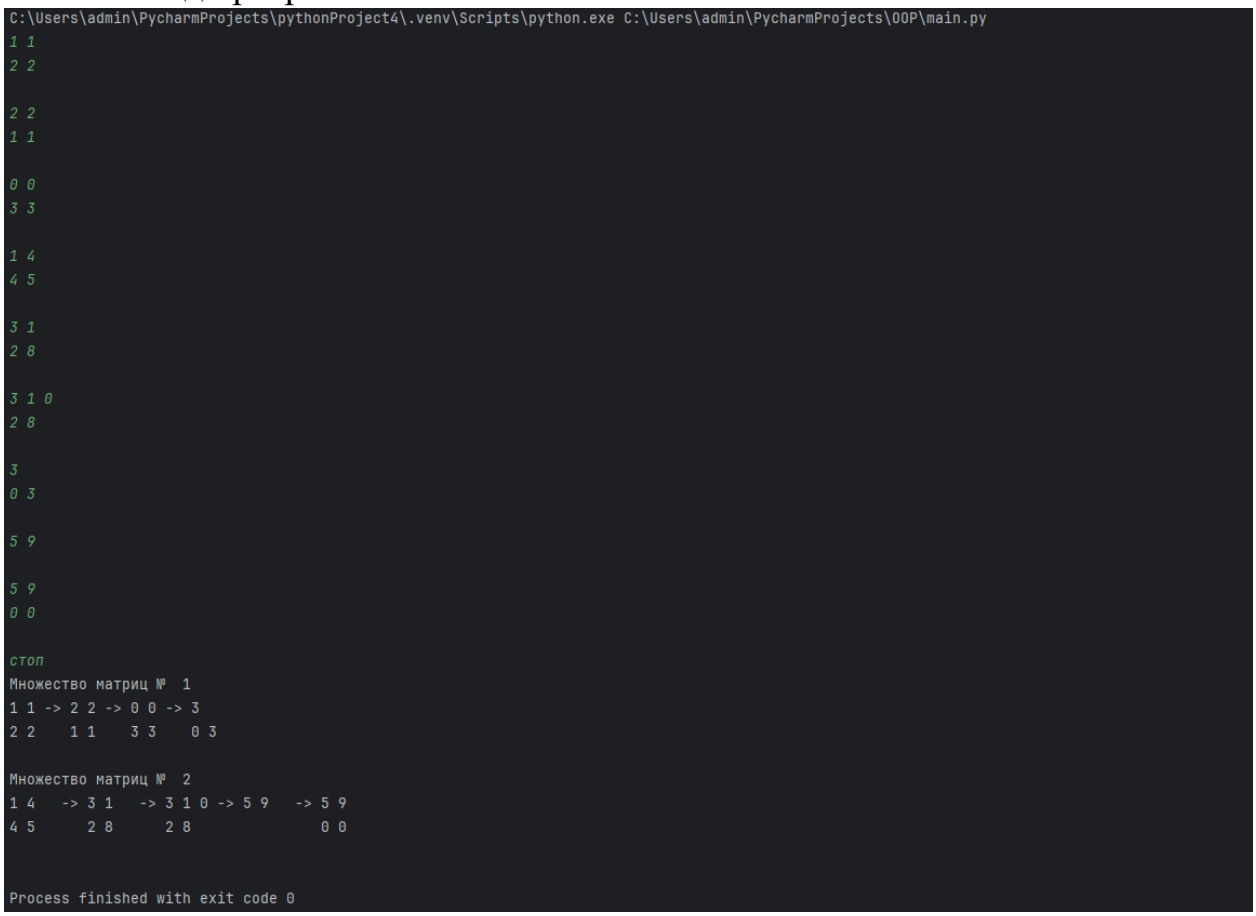
```
        return len(self.matrix)

    def get_max_cols(self):
        max_cols = 0
        for i in self.matrix:
            max_cols = max(max_cols, len(i))
        return max_cols
```

Модуль main:

```
import matrix

def modification_list(array):
    while len(array) > 0:
        if array[-1] == 0:
            array.pop(-1)
        else:
            return array
    return array

def print_matrix(array, list_matrix_temp):
    max_row = 0
    max_col = 0
    str_ = ""

    for i in array:
        max_row = max(list_matrix_temp[i].get_count_rows(), max_row)

        temp = list_matrix_temp[i].get_max_cols()

        max_col = max (temp + temp - 1 , max_col)

    for i in range(0, max_row):
        for j in array:
            temp = list_matrix_temp[j].print_row(i)
            if len(temp) != max_col:
                temp += ' ' * (max_col - len(temp))

            str_+= temp

            if i == 0 and j != array[-1]:
                str_+=" -> "
            else:
                str_+="    "
        str_+="\n"
    print(str_)

list_matrix = []
first_matrix = matrix.Matrix()
list_matrix.append(first_matrix)

flag_delete_last = 0

while True:
    line = input()
    if line == "стоп":
        if flag_delete_last:
            list_matrix.pop(-1)
        break
    elif line == "":
        flag_delete_last = 1
        list_matrix.append(matrix.Matrix())
```

```
    else:
        row_elements = [int(x) for x in line.split()]
        list_matrix[-1].add_row(row_elements)
        flag_delete_last = 0


list_sum = []

for i in list_matrix:
    list_sum.append(i.get_sumCol())

unique_sum = []
answer = []

for i in range(0, len(list_sum) - 1):

    list_sum[i] = modification_list(list_sum[i])

    if list_sum[i] in unique_sum:
        continue
    answer.append([])
    answer[-1].append(i)
    for j in range(i+1, len(list_sum)):
        list_sum[j] = modification_list(list_sum[j])
        if list_sum[i] == list_sum[j]:
            answer[-1].append(j)
    if len(answer[-1]) > 1:
        unique_sum.append(list_sum[i])


count = 1
print("\n")
for i in answer:
    if len(i) > 1:
        print("Множество матриц № ", count)
        print_matrix(i, list_matrix)
        count += 1
```

Вывод программы:



Вывод: в ходе проделанной работы я познакомился с базовыми конструкциями языка python, получил навыки создания простых приложений, изучил базовые типы.