**ФГАОУ ВО «МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**Лабораторная работа №5**

Двумерные массивы и функции

**Вариант № 28**

По дисциплине:

Основы программирования

Выполнил студент 1-го курса группы 243-323

Онищенко А. А.

Проверил

_____Никишина И. Н.

**Москва, 2025**

## Постановка задачи

Произведением двух матриц Amn на Bnl называется такая матрица Cml, для которой:

$$c_{ik} = a_{i1} * b_{1k} + a_{21} * b_{2k} + \cdots + a_{in} * b_{nk} = \sum_{j=1}^{n} a_{ij} * b_{jk}$$

То есть элемент $c_{ik}$ матрицы C равен сумме произведений элементов i-й строки матрицы A на соответствующие элементы k-го столбца матрицы B.

Написать программу вычисления произведения двух матриц.

Программа должна по заданным размерностям матриц сообщать о возможности получения такого произведения.

## Теоретическая часть

Двумерный массив (матрица) — это структура данных, представляющая собой массив массивов, где каждый элемент имеет два индекса: номер строки и номер столбца. В Python двумерные массивы чаще всего реализуются как:

- Списки списков (list of lists)
- Массивы из библиотеки NumPy
- Матрицы из специализированных библиотек (SciPy, Pandas)

## Описание программы

Программа написана на алгоритмическом языке Python 3.6, реализована в среде ОС Windows 10 и состоит из частей, отвечающих за ввод данных, вычисление и представление данных на экране монитора.

## Описание алгоритма

1. Запросить у пользователя количество строк и столбцов для матрицы A и преобразовать их к целым числам
2. Запросить у пользователя элементы матрицы A построчно и преобразовать их к вещественным числам
3. Запросить у пользователя количество строк и столбцов для матрицы B и преобразовать их к целым числам
4. Запросить у пользователя элементы матрицы B построчно и преобразовать их к вещественным числам
5. Проверить возможность умножения матриц:
6. Создать результирующую матрицу C с размерами: строк как у A, столбцов как у B
7. Вычислить произведение матриц:
8. Вывести на экран исходную матрицу A с указанием размеров
9. Вывести на экран исходную матрицу B с указанием размеров
10. Вывести на экран результирующую матрицу C (A×B) с указанием размеров

## Описание входных и выходных данных

Входные данные – размеры матриц и заполнение строк и столбцов матриц числами, поступают с клавиатуры от пользователя и имеют тип `int и float` соответственно. Выходные данные: информация об ошибках и вывод другой информации имеют тип `string`, исходные и результирующая матрица имеют тип двумерных массивов с вещественными числами.

## Описание подпрограмм

1. **Функция matrix_multiply(A, B)**:
    1. Принять на вход две матрицы A и B
    2. Проверить, что матрицы не пустые
    3. Сравнить количество столбцов A с количеством строк B
        - Если не равны - вывести ошибку и вернуть None
    4. Проверить согласованность размеров всех строк в матрицах
    5. Создать результирующую матрицу C с размерами: строк как у A, столбцов как у B
    6. Транспонировать матрицу B для оптимизации доступа к столбцам
    7. Для каждой строки i в A и каждого столбца j в B:
        - Вычислить скалярное произведение строки i A и столбца j B
        - Записать результат в C[i][j]
    8. Вернуть полученную матрицу C
2. **Функция input_matrix(name)**:
    1. Принять параметр name - название матрицы
    2. Запросить у пользователя количество строк и столбцов
        - Проверить, что введены положительные числа
    3. Для каждой строки от 1 до rows:
        - Запросить элементы строки через пробел
        - Проверить количество элементов
        - Преобразовать элементы к вещественным числам
        - При ошибке - повторить ввод строки
    4. Вернуть заполненную матрицу
3. **Функция print_matrix(matrix, name, width=10)**:
    1. Принять матрицу, её имя и ширину столбца
    2. Если матрица пустая - вывести сообщение
    3. Вывести заголовок с именем и размерами матрицы
    4. Для каждой строки матрицы:
        - Отформатировать все элементы:
            - Числа вывести с 4 знаками после запятой
            - Другие типы вывести по центру
        - Вывести строку элементов через пробел
4. **Функция main()**:
    1. Вывести приветственное сообщение
    2. Вызвать input_matrix для матриц A и B
    3. Вызвать matrix_multiply для умножения матриц
    4. Если результат не None:
        - Вывести исходные матрицы через print_matrix
        - Вывести результат умножения
    5. Обработать возможные ошибки:
        - Прерывание пользователем
        - Другие исключения
5. **Блок if name == "main":**
    1. Проверить, что программа запущена напрямую
    2. Вызвать функцию main()

## Листинг программы

```
def matrix_multiply(A, B):
    """
    Умножение матриц A и B
    Возвращает матрицу-произведение или None, если умножение невозможно
    """
    # Проверка на пустые матрицы
    if not A or not B:
        print("Ошибка: одна из матриц пустая")
        return None

    # Проверка возможности умножения
    cols_A = len(A[0])
    rows_B = len(B)

    if cols_A != rows_B:
        print(f"Ошибка: невозможно умножить ({len(A)}×{cols_A}) на
({rows_B}×{len(B[0])})")
        return None

    # Проверка согласованности размеров
    for row in A:
        if len(row) != cols_A:
            print("Ошибка: несоответствие размеров в матрице A")
            return None

    for row in B:
        if len(row) != len(B[0]):
            print("Ошибка: несоответствие размеров в матрице B")
            return None

    # Создание результирующей матрицы
    m = len(A)
    n = len(B[0])
    C = [[0.0 for _ in range(n)] for _ in range(m)]

    # Транспонируем B для более эффективного доступа по столбцам
    B_transposed = list(zip(*B))

    # Вычисление произведения с оптимизированным доступом
    for i in range(m):
        for j in range(n):
            # Используем sum и генератор для вычисления скалярного произведения
            C[i][j] = sum(A[i][k] * B_transposed[j][k] for k in range(cols_A))

    return C


def input_matrix(name):
    """Ввод матрицы с клавиатуры"""
    while True:
        try:
            rows = int(input(f"Введите количество строк матрицы {name}: "))
            cols = int(input(f"Введите количество столбцов матрицы {name}: "))
            if rows <= 0 or cols <= 0:
                print("Размеры должны быть положительными числами!")
                continue
            break
        except ValueError:
            print("Ошибка: введите целое число!")

    print(f"Введите элементы матрицы {name} построчно (через пробел):")
    matrix = []
    for i in range(rows):
        while True:
            row = input(f"Строка {i + 1}: ").strip().split()
```

```
            if len(row) != cols:
                print(f"Ошибка: ожидается {cols} элементов, получено {len(row)}")
                continue

            try:
                matrix.append([float(x) for x in row])
                break
            except ValueError:
                print("Ошибка: все элементы должны быть числами!")

    return matrix


def print_matrix(matrix, name, width=10):
    """Вывод матрицы на экран с форматированием"""
    if not matrix:
        print(f"\nМатрица {name}: [пустая]")
        return

    print(f"\nМатрица {name} ({len(matrix)}×{len(matrix[0])}):")
    for row in matrix:
        formatted_row = [f"{x:{width}.4f}" if isinstance(x, (int, float)) else
f"{x:^{width}}" for x in row]
        print(" ".join(formatted_row))


def main():
    print("Программа умножения матриц A(m×n) и B(n×l)")

    try:
        # Ввод матриц
        A = input_matrix("A")
        B = input_matrix("B")

        # Умножение матриц
        C = matrix_multiply(A, B)

        # Вывод результата
        if C is not None:
            print_matrix(A, "A")
            print_matrix(B, "B")
            print_matrix(C, "C = A×B")

    except KeyboardInterrupt:
        print("\nПрограмма прервана пользователем")
    except Exception as e:
        print(f"Произошла непредвиденная ошибка: {str(e)}")


if __name__ == "__main__":
    main()
```

## Результаты работы программы

```
Программа умножения матриц A(m×n) и B(n×l)
Введите количество строк матрицы A: 3
Введите количество столбцов матрицы A: 3
Введите элементы матрицы A построчно (через пробел):
Строка 1: 1 1 1
Строка 2: 1 1 1
Строка 3: 1 1 1
Введите количество строк матрицы B: 3
Введите количество столбцов матрицы B: 3
Введите элементы матрицы B построчно (через пробел):
Строка 1: 1 2 3
Строка 2: 4 5 6
Строка 3: 7 8 9

Матрица A (3×3):
    1.0000     1.0000     1.0000
    1.0000     1.0000     1.0000
    1.0000     1.0000     1.0000

Матрица B (3×3):
    1.0000     2.0000     3.0000
    4.0000     5.0000     6.0000
    7.0000     8.0000     9.0000

Матрица C = A×B (3×3):
   12.0000    15.0000    18.0000
   12.0000    15.0000    18.0000
   12.0000    15.0000    18.0000

Process finished with exit code 0
```

***(Рис. 1)***

```
Программа умножения матриц A(m×n) и B(n×l)
Введите количество строк матрицы A: 4
Введите количество столбцов матрицы A: 5
Введите элементы матрицы A построчно (через пробел):
Строка 1: 5
Ошибка: ожидается 5 элементов, получено 1
Строка 1: 1 2 3 4
Ошибка: ожидается 5 элементов, получено 4
Строка 1: 1 2 3 4 5
Строка 2: 2 3 4 5 6
Строка 3: 3 4 5 6 7
Строка 4: 4 5 6 7 8
Введите количество строк матрицы B: 5
Введите количество столбцов матрицы B: 4
Введите элементы матрицы B построчно (через пробел):
Строка 1: 1 2 3 4
Строка 2: 2 3 4 5
Строка 3: 3 4 5 6
Строка 4: 4 5 6 7
Строка 5: 5 6 7 8

Матрица A (4×5):
    1.0000     2.0000     3.0000     4.0000     5.0000
    2.0000     3.0000     4.0000     5.0000     6.0000
    3.0000     4.0000     5.0000     6.0000     7.0000
    4.0000     5.0000     6.0000     7.0000     8.0000

Матрица B (5×4):
    1.0000     2.0000     3.0000     4.0000
    2.0000     3.0000     4.0000     5.0000
    3.0000     4.0000     5.0000     6.0000
    4.0000     5.0000     6.0000     7.0000
    5.0000     6.0000     7.0000     8.0000
```

```
Матрица C = A×B (4×4):
   55.0000    70.0000    85.0000   100.0000
   70.0000    90.0000   110.0000   130.0000
   85.0000   110.0000   135.0000   160.0000
  100.0000   130.0000   160.0000   190.0000

Process finished with exit code 0
```

***(Рис. 2)***

## Тестовые кейсы

$$\mathbf{C} = \mathbf{A} \cdot \mathbf{B} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} = \begin{pmatrix} 12 & 15 & 18 \\ 12 & 15 & 18 \\ 12 & 15 & 18 \end{pmatrix}$$

***(Рис. 1)***

$$\mathbf{C} = \mathbf{A} \cdot \mathbf{B} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 5 & 6 & 7 \\ 4 & 5 & 6 & 7 & 8 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 4 & 5 & 6 & 7 \\ 5 & 6 & 7 & 8 \end{pmatrix} = \begin{pmatrix} 55 & 70 & 85 & 100 \\ 70 & 90 & 110 & 130 \\ 85 & 110 & 135 & 160 \\ 100 & 130 & 160 & 190 \end{pmatrix}$$

***(Рис. 2)***

## Список используемой литературы

1. Н.А. Прохоренок, В.А. Дронов, Python 3 и PyQt 5. Разработка приложений: СПб.: БХВ- Петербург, 2017

2. В.П. Рядченко, Методическое пособие по выполнению лабораторных работ.