Автор: Анна Смелова

```
In [1]: import numpy as np
```

Решение домашнего задания к уроку 3 “Матрицы и матричные операции. Часть 1”

1. Задача

Установить, какие произведения матриц AB и BA определены, и найти размерности полученных матриц:

а) A — матрица 4×2, B — матрица 4×2;

б) A — матрица 2×5, B — матрица 5×3;

в) A — матрица 8×3, B — матрица 3×8;

г) A — квадратная матрица 4×4, B — квадратная матрица 4×4.

Решение:
Произведением матрицы $A = \|a_{ij}\|$, имеющей порядки m и n, и матрицы $B = \|b_{ij}\|$, имеющей порядки n и k, называется матрица $C = \|c_{ij}\|$, имеющая порядки m и k
Матрицу A можно умножить не на всякую матрицу B: необходимо, чтобы число столбцов матрицы A было равно числу строк матрицы B.

а) A — матрица 4×2, B — матрица 4×2
$4 \neq 2 \Rightarrow$ произведение AB не определено, аналогично произведение BA не определено.

б) A — матрица 2×5, B — матрица 5×3
$2 \neq 3 \Rightarrow$ произведение AB не определено
$5 = 5 \Rightarrow$ произведение BA определено и имеет размерность 5×5

в) A — матрица 8×3, B — матрица 3×8
$8 = 8 \Rightarrow$ произведение AB определено и имеет размерность 8×8
$3 = 3 \Rightarrow$ произведение BA определено и имеет размерность 3×3

г) A — квадратная матрица 4×4, B — квадратная матрица 4×4
$4 = 4 \Rightarrow$ произведения AB и BA определены и имеют размерности 4×4 и 4×4.

2. Задача

Найти сумму и произведение матриц

$A = \begin{pmatrix} 1 & -2 \\ 3 & 0 \end{pmatrix}$ и $B = \begin{pmatrix} 4 & -1 \\ 0 & 5 \end{pmatrix}$.

Решение:

Сумма: $A + B = \begin{pmatrix} 1 & -2 \\ 3 & 0 \end{pmatrix} + \begin{pmatrix} 4 & -1 \\ 0 & 5 \end{pmatrix} = \begin{pmatrix} 5 & -3 \\ 3 & 5 \end{pmatrix}$

Произведение: $A \cdot B = \begin{pmatrix} 1 & -2 \\ 3 & 0 \end{pmatrix} \cdot \begin{pmatrix} 4 & -1 \\ 0 & 5 \end{pmatrix} = \begin{pmatrix} 4 - 0 & -1 - 10 \\ 12 + 0 & -3 + 0 \end{pmatrix} = \begin{pmatrix} 4 & -11 \\ 12 & -3 \end{pmatrix}$

Произведение: $B \cdot A = \begin{pmatrix} 4 & -1 \\ 0 & 5 \end{pmatrix} \cdot \begin{pmatrix} 1 & -2 \\ 3 & 0 \end{pmatrix} = \begin{pmatrix} 4 - 3 & -8 + 0 \\ 0 + 15 & 0 + 0 \end{pmatrix} = \begin{pmatrix} 1 & -8 \\ 15 & 0 \end{pmatrix}$

In [2]:
```python
A = np.array([[1, -2], [3, 0]])
B = np.array([[4, -1], [0, 5]])

print(f'Матрица A:\n {A}')
print(f'Матрица B:\n {B}')
print(f'Сумма A+B:\n {A+B}')
print(f'Матрица AB:\n {np.dot(A, B)}')
print(f'Матрица BA:\n {np.dot(B, A)}')
```

```
Матрица A:
 [[ 1 -2]
 [ 3  0]]
Матрица B:
 [[ 4 -1]
 [ 0  5]]
Сумма A+B:
 [[ 5 -3]
 [ 3  5]]
Матрица AB:
 [[  4 -11]
 [ 12  -3]]
Матрица BA:
 [[ 1 -8]
 [15  0]]
```

3. Задача

Из закономерностей сложения и умножения матриц на число можно сделать вывод, что матрицы одного размера образуют линейное пространство. Вычислить линейную комбинацию $3A - 2B + 4C$ для матриц

$A = \begin{pmatrix} 1 & 7 \\ 3 & -6 \end{pmatrix}$, $B = \begin{pmatrix} 0 & 5 \\ 2 & -1 \end{pmatrix}$, $C = \begin{pmatrix} 2 & -4 \\ 1 & 1 \end{pmatrix}$.

Решение:

$A = \begin{pmatrix} 1 & 7 \\ 3 & -6 \end{pmatrix}$, $B = \begin{pmatrix} 0 & 5 \\ 2 & -1 \end{pmatrix}$, $C = \begin{pmatrix} 2 & -4 \\ 1 & 1 \end{pmatrix}$.

1) $3A = \begin{pmatrix} 3 & 21 \\ 9 & -18 \end{pmatrix}$

2) $2B = \begin{pmatrix} 0 & 10 \\ 4 & -2 \end{pmatrix}$

3) $4C = \begin{pmatrix} 8 & -16 \\ 4 & 4 \end{pmatrix}$

4) $3A - 2B + 4C = \begin{pmatrix} 3 & 21 \\ 9 & -18 \end{pmatrix} - \begin{pmatrix} 0 & 10 \\ 4 & -2 \end{pmatrix} + \begin{pmatrix} 8 & -16 \\ 4 & 4 \end{pmatrix} = \begin{pmatrix} 3-0+8 & 21-10-16 \\ 9-4+4 & -18+2+4 \end{pmatrix} = \begin{pmatrix} 11 & -5 \\ 9 & -12 \end{pmatrix}$

In [3]:
```python
A = np.array([[1, 7], [3, -6]])
B = np.array([[0, 5], [2, -1]])
C = np.array([[2, -4], [1, 1]])
print(f'Матрица A:\n {A}')
print(f'Матрица B:\n {B}')
print(f'Матрица C:\n {C}')
print(f'3*A-2*B+4*C:\n {3*A-2*B+4*C}')
```

```
Матрица A:
 [[ 1  7]
 [ 3 -6]]
Матрица B:
 [[ 0  5]
 [ 2 -1]]
Матрица C:
 [[ 2 -4]
 [ 1  1]]
3*A-2*B+4*C:
 [[ 11  -5]
 [  9 -12]]
```

4. Задача

Дана матрица

$$A = \begin{pmatrix} 4 & 1 \\ 5 & -2 \\ 2 & 3 \end{pmatrix}.$$

Вычислить

Решение:

$$A = \begin{pmatrix} 4 & 1 \\ 5 & -2 \\ 2 & 3 \end{pmatrix}$$

$$A^T = \begin{pmatrix} 4 & 5 & 2 \\ 1 & -2 & 3 \end{pmatrix}$$

$$AA^T = \begin{pmatrix} 4 & 1 \\ 5 & -2 \\ 2 & 3 \end{pmatrix} \cdot \begin{pmatrix} 4 & 5 & 2 \\ 1 & -2 & 3 \end{pmatrix} = \begin{pmatrix} 16+1 & 20-2 & 8+3 \\ 20-2 & 25+4 & 10-6 \\ 8+3 & 10-6 & 4+9 \end{pmatrix} = \begin{pmatrix} 17 & 18 & 11 \\ 18 & 29 & 4 \\ 11 & 4 & 13 \end{pmatrix}$$

$$A^TA = \begin{pmatrix} 4 & 5 & 2 \\ 1 & -2 & 3 \end{pmatrix} \cdot \begin{pmatrix} 4 & 1 \\ 5 & -2 \\ 2 & 3 \end{pmatrix} = \begin{pmatrix} 16+25+4 & 4-10+6 \\ 4-10+6 & 1+4+9 \end{pmatrix} = \begin{pmatrix} 45 & 0 \\ 0 & 14 \end{pmatrix}$$

In [4]:
```
A = np.array([[4, 1], [5, -2], [2, 3]])
print(f'Матрица A:\n {A}')
A_T = A.T
print(f'Транспонированная матрица AT:\n {A_T}')
print(f'Матрица AA_T:\n {np.dot(A, A_T)}')
print(f'Матрица A_TA:\n {np.dot(A_T, A)}')
```

```
Матрица A:
 [[ 4  1]
 [ 5 -2]
 [ 2  3]]
Транспонированная матрица AT:
 [[ 4  5  2]
 [ 1 -2  3]]
Матрица AA_T:
 [[17 18 11]
 [18 29  4]
 [11  4 13]]
Матрица A_TA:
 [[45  0]
 [ 0 14]]
```

5*. Задача

Написать на Python функцию для перемножения двух произвольных матриц, не используя NumPy.

Решение:

In [5]:
```python
# Вспомогательная функция
# Возвращает транспонированную матрицу
def get_matrix_transpose(A):
    A_T = []
    n = len(A)
    m = len(A[0])
    for j in range(m):
        col = []
        for i in A:
            col.append(i[j])
        A_T.append(col)
    return A_T

# Функция для перемножения двух матриц
# Матрицы задаем обычным двумерным массивом
# Возвращает результат произведения
def get_matrix_multiply(A,B):
    if not A or not A[0] or not B or not B[0] or type(A[0])==int or type(B[0])==int:
        print(f'Укажите матрицы, используя двумерные массивы.')
        return 'error'
    n = len(A)  # количество строк A
    m = len(A[0])  # количество столбцов A
    p = len(B)  # количество строк B
    k = len(B[0])  # количество столбцов B
    if n != k:
        print(f'Произведение не определено, причина: размерность указанных матриц')
        return 'error'
    B_T = get_matrix_transpose(B)
    C = [[None]*n for _ in range(n)]  # результирующая матрица
    for i in range(n):  # зафиксировали строку A
        for j in range(k):  # зафиксировали столбец B(строку B_T)
            el = 0  # [i,j] элемент результирующей матрицы
            for ii in range(m):  # зафиксировали элемент строки A
                el += A[i][ii]*B_T[j][ii]
            C[i][j] = el

    print(f'Первая матрица:\n {A}')
    print(f'Вторая матрица:\n {B}')
    print(f'Произведение:\n {C}')
    return C
```

In [6]:
```python
# Test
A = [[1, -2], [3, 0]]
B = [[4, -1], [0, 5]]
get_matrix_multiply(A,B)
print(f'Проверка через numpy:\n {np.dot(np.array(A), np.array(B))}\n')
C = [[4, 1], [5, -2], [2, 3]]
C_T = [[4, 5, 2], [1, -2, 3]]
get_matrix_multiply(C,C_T)
print(f'Проверка через numpy:\n {np.dot(np.array(C), np.array(C_T))}\n')
```

```
Первая матрица:
 [[1, -2], [3, 0]]
Вторая матрица:
 [[4, -1], [0, 5]]
Произведение:
 [[4, -11], [12, -3]]
Проверка через numpy:
 [[  4 -11]
 [ 12  -3]]

Первая матрица:
 [[4, 1], [5, -2], [2, 3]]
Вторая матрица:
 [[4, 5, 2], [1, -2, 3]]
Произведение:
 [[17, 18, 11], [18, 29, 4], [11, 4, 13]]
Проверка через numpy:
 [[17 18 11]
 [18 29  4]
 [11  4 13]]
```

Решение домашнего задания к уроку 4 “Матрицы и матричные операции. Часть 2”

1. Задача

Вычислить определитель:

а)

$$\begin{vmatrix} sinx & -cosx \\ cosx & sinx \end{vmatrix};$$

б)

$$\begin{vmatrix} 4 & 2 & 3 \\ 0 & 5 & 1 \\ 0 & 0 & 9 \end{vmatrix};$$

в)

$$\begin{vmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{vmatrix}.$$

Решение:

а) $\begin{vmatrix} sinx & -cosx \\ cosx & sinx \end{vmatrix} = sin^2x + cos^2x = 1$

б) $\begin{vmatrix} 4 & 2 & 3 \\ 0 & 5 & 1 \\ 0 & 0 & 9 \end{vmatrix} = 4 \cdot 5 \cdot 9 + 2 \cdot 1 \cdot 0 + 3 \cdot 0 \cdot 0 - 3 \cdot 5 \cdot 0 - 2 \cdot 0 \cdot 9 - 4 \cdot 1 \cdot 0 = 180 + 0 + 0 - 0 - 0 - 0 = 180$

в) $\begin{vmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{vmatrix} = 1 \cdot 5 \cdot 9 + 3 \cdot 4 \cdot 8 + 2 \cdot 6 \cdot 7 - 3 \cdot 5 \cdot 7 - 2 \cdot 4 \cdot 9 - 1 \cdot 6 \cdot 8 = 45 + 96 + 84 - 105 - 72 - 48 = 0$

In [7]:
```
b = np.array([[4, 2, 3], [0, 5, 1], [0, 0, 9]])
print(f'Матрица б):\n {b}')
print(f'Определитель б):\n {np.linalg.det(b):.0f}')
c = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print(f'Матрица в):\n {c}')
print(f'Определитель в):\n {np.linalg.det(c):.0f}')
```

```
Матрица б):
 [[4 2 3]
 [0 5 1]
 [0 0 9]]
Определитель б):
 180
Матрица в):
 [[1 2 3]
 [4 5 6]
 [7 8 9]]
Определитель в):
 -0
```

2. Задача

Определитель матрицы A равен 4. Найти:

а) $det(A^2)$;

б) $det(A^T)$;

в) $det(2A)$.

Решение:

а) $det(AB) = detA \cdot detB \Rightarrow det(A^2) = det(AA) = detA \cdot detA = 4^2 = 16$

б) $detA^T = detA = 4$

в) $det(2A)$ зависит от размерности матрицы A. Если матрица A размером $n \times n$, то $det(2A) = 2^n \cdot det(A) = 2^n \cdot 4$

3. Задача

Доказать, что матрица

$$\begin{pmatrix} -2 & 7 & -3 \\ 4 & -14 & 6 \\ -3 & 7 & 13 \end{pmatrix}$$

вырожденная.

Решение:

$A = \begin{pmatrix} -2 & 7 & -3 \\ 4 & -14 & 6 \\ -3 & 7 & 13 \end{pmatrix}$

$det(A) = (-2) \cdot (-14) \cdot 13 + 7 \cdot 6 \cdot (-3) + (-3) \cdot 4 \cdot 7 - (-3) \cdot (-14) \cdot (-3) - 7 \cdot 4 \cdot 13 - (-2) \cdot 6 \cdot 7 = 364 - 126 - 84 + 126 - 364 + 84 = 0$

Матрица называется сингулярной, или вырожденной, если ее определитель равен нулю. чтд.

In [8]:

```
A = np.array([[-2, 7, -3], [4, -14, 6], [-3, 7, 13]])
print(f'Матрица A:\n {A}')
print(f'Определитель A:\n {np.linalg.det(A):.0f}')
```

```
Матрица A:
 [[ -2   7  -3]
 [  4 -14   6]
 [ -3   7  13]]
Определитель A:
 0
```

4. Задача

Найти ранг матрицы:

а) $\begin{pmatrix} 1 & 2 & 3 \\ 1 & 1 & 1 \\ 2 & 3 & 4 \end{pmatrix}$;

б) $\begin{pmatrix} 0 & 0 & 2 & 1 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 4 & 3 \\ 2 & 3 & 5 & 6 \end{pmatrix}$.

Решение:

а) $A = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 1 & 1 \\ 2 & 3 & 4 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 0 & 1 & 2 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 0 & 0 & 0 \end{pmatrix}$

$\Rightarrow$ ранг матрицы A равен 2.

б) $B = \begin{pmatrix} 0 & 0 & 2 & 1 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 4 & 3 \\ 2 & 3 & 5 & 6 \end{pmatrix} \rightarrow \begin{pmatrix} 2 & 3 & 5 & 6 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 \end{pmatrix} \rightarrow \begin{pmatrix} 2 & 3 & 5 & 6 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$

$\Rightarrow$ ранг матрицы B равен 3.

In [9]:

```
A = np.array([[1, 2, 3], [1, 1, 1], [2, 3, 4]])
r_A = np.linalg.matrix_rank(A)
print(f'Ранг матрицы A: {r_A}')
B = np.array([[0, 0, 2, 1], [0, 0, 2, 2], [0, 0, 4, 3], [2, 3, 5, 6]])
r_B = np.linalg.matrix_rank(B)
print(f'Ранг матрицы B: {r_B}')
```

```
Ранг матрицы A: 2
Ранг матрицы B: 3
```

5*. Задача

Написать на Python функцию для вычисления определителя матрицы методом разложения по первой строке.

Решение:

In [10]:
```
def get_det_by_first_row(A):
    print(f'Считаем для матрицы: {A}')
    if not A or not A[0] or type(A[0])==int:
        print(f'Укажите матрицу, используя двумерный массив.')
        return 'error'
    n = len(A)  # количество строк=количество столбцов
    if n == 1:
        print(f'det={n}\n')
        return n
    if n==2:
        print(f'det={A[0][0]*A[1][1] - A[0][1]*A[1][0]}\n')
        return A[0][0]*A[1][1] - A[0][1]*A[1][0]
    det = 0
    first_row = A[0]  # первая строка
    for i in range(n):
        B = [[None]*(n-1) for _ in range(n-1)]
        for p in range(n):
            for k in range(n):
                if p != 0:
                    if k<i:
                        B[p-1][k] = A[p][k]
                    elif k>i:
                        B[p-1][k-1] = A[p][k]
        print(f'Элемент разложения по первой строке: {first_row[i]*(-1)**(i)}*{B}\n')
        det += first_row[i]*(-1)**(i)*get_det_by_first_row(B)
    return det
```

In [11]:
```python
# Test
A = [[-1, 2, -3, 4], [5, -6, 7, 8], [9, 10, -11, 12], [13, -14, 15, 16]]
print(f'Определитель матрицы: {get_det_by_first_row(A)}')
print()
print(f'Проверка через numpy:\n {np.linalg.det(np.array(A)):.0f}')
```

```
Считаем для матрицы: [[-1, 2, -3, 4], [5, -6, 7, 8], [9, 10, -11, 12], [13, -14, 15, 16]]
Элемент разложения по первой строке: -1*[[-6, 7, 8], [10, -11, 12], [-14, 15, 16]]

Считаем для матрицы: [[-6, 7, 8], [10, -11, 12], [-14, 15, 16]]
Элемент разложения по первой строке: -6*[[-11, 12], [15, 16]]

Считаем для матрицы: [[-11, 12], [15, 16]]
det=-356

Элемент разложения по первой строке: -7*[[10, 12], [-14, 16]]

Считаем для матрицы: [[10, 12], [-14, 16]]
det=328

Элемент разложения по первой строке: 8*[[10, -11], [-14, 15]]

Считаем для матрицы: [[10, -11], [-14, 15]]
det=-4

Элемент разложения по первой строке: -2*[[5, 7, 8], [9, -11, 12], [13, 15, 16]]

Считаем для матрицы: [[5, 7, 8], [9, -11, 12], [13, 15, 16]]
Элемент разложения по первой строке: 5*[[-11, 12], [15, 16]]

Считаем для матрицы: [[-11, 12], [15, 16]]
det=-356

Элемент разложения по первой строке: -7*[[9, 12], [13, 16]]

Считаем для матрицы: [[9, 12], [13, 16]]
det=-12

Элемент разложения по первой строке: 8*[[9, -11], [13, 15]]

Считаем для матрицы: [[9, -11], [13, 15]]
det=278

Элемент разложения по первой строке: -3*[[5, -6, 8], [9, 10, 12], [13, -14, 16]]

Считаем для матрицы: [[5, -6, 8], [9, 10, 12], [13, -14, 16]]
Элемент разложения по первой строке: 5*[[10, 12], [-14, 16]]

Считаем для матрицы: [[10, 12], [-14, 16]]
det=328

Элемент разложения по первой строке: 6*[[9, 12], [13, 16]]

Считаем для матрицы: [[9, 12], [13, 16]]
det=-12

Элемент разложения по первой строке: 8*[[9, 10], [13, -14]]

Считаем для матрицы: [[9, 10], [13, -14]]
det=-256

Элемент разложения по первой строке: -4*[[5, -6, 7], [9, 10, -11], [13, -14, 15]]

Считаем для матрицы: [[5, -6, 7], [9, 10, -11], [13, -14, 15]]
Элемент разложения по первой строке: 5*[[10, -11], [-14, 15]]

Считаем для матрицы: [[10, -11], [-14, 15]]
det=-4

Элемент разложения по первой строке: 6*[[9, -11], [13, 15]]

Считаем для матрицы: [[9, -11], [13, 15]]
det=278

Элемент разложения по первой строке: 7*[[9, 10], [13, -14]]

Считаем для матрицы: [[9, 10], [13, -14]]
det=-256

Определитель матрицы: 1152

Проверка через numpy:
 1152
```

In [ ]: