# Лабораторный практикум 3. Векторная алгебра

## Линейные операции над векторами и их свойства

Загрузите необходимые библиотеки:

>> import numpy as np

>> import matplotlib.pyplot as plt

>> import math

>> from sympy import \*

Для построения векторов будем использовать функции ***arrow*** и ***quiver***(*см. help*(*plt.arrow*)и *help*(*plt.quiver*)). Чтобы построить вектор с помощью этих функций, необходимо задать начальную точку и координаты данного вектора.

### Пример 1. Даны три точки с координатами *A*(–2,0), *B*(1,2), *C*(1,–1). Найти координаты *AB*, *BC* и *AC* векторов и убедиться, что *АВ* + *ВС* = *AC*. Изобразить данные векторы.

OA = np.array([-2, 0])  
OB = np.array([1, 2])  
OC = np.array([1, -1])  
print('Векторы имеют координаты:')  
AB = OB - OA  
BC = OC - OB  
AC = OC - OA  
print('AB:', AB)  
print('BC:', BC)  
print('AC:', AC)

plt.grid()  
plt.ylim(-2, 3)  
plt.xlim(-3, 2)  
plt.arrow(-2, 0, AB[0], AB[1], width=0.07, length\_includes\_head=True, facecolor='Yellow')  
plt.arrow(1, 2, BC[0], BC[1], width=0.07, length\_includes\_head=True, facecolor='Red')  
plt.arrow(-2, 0, AC[0], AC[1], width=0.07, length\_includes\_head=True, facecolor='Blue')  
plt.show()

В самом простом случае ***arrow*(*x, y, dx, dy*)** стрелка задается всего четырьмя параметрами: ***x, y*** - координаты начала стрелки, ***dx, dy*** - длина стрелки по оси *x* и *y*. Параметр***width*** изменяет толщину стрелки (по умолчанию *width* = 0.001). Параметр ***length\_includes\_head*** установленный в ***True*** соответствует тому, что длина стрелки будет включать в себя длину острия. Параметры ***head\_width*** и ***head\_length*** позволяют изменять соответственно ширину и длину острия. Параметр ***facecolor*** устанавливает цвет стрелки.

### Упражнение 3.1. Дан параллелограмм *ABCD*, известны координаты трех его точек *A*(–2,0), *B*(1,2), *C*(1,–1). Найти координаты четвертой вершины *D* параллелограмма. Изобразить разными цветами векторы, используемые в решении задачи.

## Линейная зависимость векторов. Базис векторов

### Упражнение 3.2. Доказать, что векторы , и образуют базис. Изобразить эти векторы (см.пример 3).

### Упражнение 3.3. Проверить являются ли векторы линейно зависимыми и, если возможно, разложить вектор по этим векторам (при решении системы использовать формулы Крамера или обратную матрицу). Векторы задать в символьном виде.



## Скалярное произведение векторов

В Python скалярное произведение двух векторов ***a*** и ***b*** вычисляется с помощью функции ***dot*()**.

**Пример 2.** Найти угол между векторами (0,5,–1) и (–4,9,3).

a = np.array([0, 5, -1])

b = np.array([-4, 9, 3])

print ('Вектор a:', a)

print ('Вектор b:', b)

cos\_angle = np.dot(a, b) / np.linalg.norm(a) / np.linalg.norm(b)

print ('Косинус угла между a и b:', cos\_angle)

alfa= np.arccos(cos\_angle) # арккосинус угла

alfa=math.degrees(alfa)

print ('Угол в градусах:', alfa)

**Упражнение 3.4.** Даны три точки *A*(–2,0), *B*(3,4), *C*(4,–1). Найти величины углов треугольника *АВС*. Проверить свойство суммы углов треугольника. Сделать рисунок.

## Векторное произведение векторов

### В Python векторное произведение двух векторов *a* и *b* вычисляется с помощью функции *cross*().

### Упражнение 3.5. Найти векторное произведение векторов и .

a = np.array([1, 2, 0])  
b = np.array([2, 1, 0])  
print(np.cross(a, b))

### Пример 3. Найти векторное произведение векторов и . Сделать рисунок: первый вектор изобразить синим, второй зеленым, результат красным. Как связано определение векторного произведения с полученным изображением?

ax = plt.axes(projection='3d')  
  
z = np.array([0, 0, 0])  
a = np.array([1, 2, 0])  
b = np.array([2, 1, 0])  
c = np.cross(a, b)  
  
ax.set\_xlim([-2, 2])  
ax.set\_ylim([-2, 2])  
ax.set\_zlim([-2, 2])  
  
ax.quiver(\*z, \*a, color='Blue')  
ax.quiver(\*z, \*b, color='Green')  
ax.quiver(\*z, \*c, color='Red')  
ax.axis()  
plt.grid()  
plt.show()

Векторы ***a****,* ***b****,* ***c*** образуют правую тройку. Вектор ***c*** перпендикулярен плоскости векторов ***a*** и ***b***.

***axes*** — это та область, на которой чаще всего и отражаются графики, а также все вспомогательные атрибуты (линии сетки, метки, указатели и т.д.). Часто, установка этой области сопровождается вызовом *subplot*, который и помещает *axes* на регулярную сетку. Поэтому *axes* и *subplot* можно считать синонимами.

### Упражнение 3.6. Вычислить площадь треугольника *АВС* с вершинами *A*(1,3,–1), *B*(2,–1,4), *C*(5,0,3). Как соотносятся площадь треугольника и модуль векторного произведения?

## Дополнительное задание

Используя определение смешанного произведения, вычислить объем пирамиды *ABCD*, если . Проверить с помощью определителя.