ЛАБОРАТОРНАЯ РАБОТА 11

# ОБЪЕКТЫ «ТОЧКА НА ПЛОСКОСТИ» «ВЕКТОР НА ПЛОСКОСТИ»

Компьютерная математика
БГУ, ММФ, 1 курс, КМ
доц. Малевич А.Э.,
доц. Щеглова Н.Л.,
доц. Лаврова О.А.
ноябрь 2021

## Постановка задачи

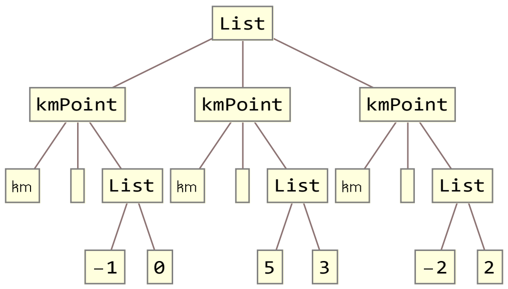
Решите предложенные задачи с использованием системы «Аналитическая геометрия на плоскости», используя методы объекта «точка на плоскости».

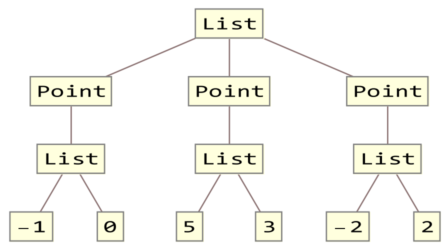
Спроектируйте и создайте класс «вектор на плоскости».

## Графический образ объекта «точка на плоскости»

***Задание 1.*** Создайте функцию-препроцессор **`kmPointShow[gp_List, opts___]`**, которая отображает в графической области объект «точка на плоскости» со всплывающей подсказкой, содержащей внутреннее представление точки. Используйте локальные правила преобразований,

Протестируйте функцию на примерах.




```
ReplaceAll[{P1, P2, P3},
 P_kmPoint :→ Point@P@"coord"]
```

## Задачи о точке в системе «Аналитическая геометрия на плоскости»

***Задание 2.1*** Координаты точки M в полярной системе координат равны $(7\pi/4, 1)$. Начала координат прямоугольной декартовой и полярной систем координат совпадают, а полярная ось сонаправлена с положительным направлением оси абсцисс.

a) Создайте точку M, используя конструктор объекта «точка на плоскости».

b) Получите декартовы координаты точки M.

c) Получите имя точки.

d) В графической области изобразите точку M зеленым цветом с использованием функции **`kmPointShow`** для отображения.

### *Задание 2.2*

a) Используя конструкторы объекта «точка на плоскости», создайте точки **`M`** и **`N`**, имеющие полярные координаты **`M`**($\pi/4$, 2) и **`N`**($3\pi/4$, 6).

b) Напишите функцию **`dist[P1_kmPoint,P2_kmPoint]`**, которая вычисляет расстояние между двумя точками **`P1`** и **`P2`**, заданными в системе «Аналитическая геометрия на плоскости».

c) В графической области отобразите точку **`M`** красным цветом, точку **`N`** – синим, отрезок, их соединяющий – серым; при этом точку, расположенную дальше от начала координат, отобразите большей по размеру, а также подпишите длину отрезка. Используйте функцию **`kmPointShow`** для отображения.

### *Задание 2.3*

a) Точка **`C`** лежит на отрезке **`AB`** и делит его в отношении $\lambda$. Создайте конструктор, который строит объект «точка на плоскости» для представления точки **`C`**, если концы отрезка **`A`** и **`B`** заданы как объекты «точка на плоскости».

b) Отобразите в графической области следующие объекты: отрезок – серым цветом, концы произвольного отрезка – красным; середину отрезка, а также точку, делящую отрезок в отношении 1/3 и точку, делящую отрезок в отношении 2/3 – синим цветом. Используйте функцию **`kmPointShow`** для отображения.

### *Задание 2.4*

a) Треугольник **`ABC`** задан координатами своих вершин. Создайте три объекта «точка на плоскости», каждый объект представляет одну вершину треугольника.

b) Создайте конструктор объекта «точка на плоскости» **`kmPoint[triangle:{A_,B_,C_},”bisAcrossBC”]`** для представления точки **`D`** пересечения биссектрисы внутреннего угла **`A`** со стороной **`BC`**.

Используйте свойство биссектрисы: точка **D** делит сторону **BC** в отношении $\lambda$ = (длина **AC**)/(длина **AB**).

При реализации конструктора используйте глобальные определения, введенные при выполнении заданий 2.2b и 2.3a.

c) В графической области отобразите произвольный треугольник. Отобразите точки пересечения биссектрис с противолежащими сторонами треугольника красным цветом. Отобразите биссектрисы зеленым цветом. Используйте функцию **kmPointShow** для отображения.

### *Задание 2.5*

a) Напишите пользовательскую функцию **polygonVertices**, которая возвращает список вершин правильного **n**-угольника, вписанного в окружность радиуса **r**, в виде геометрических объектов «точка на плоскости».

Используйте метод построения вершин полигона в полярных координатах. Расположите полюс в точке O, являющейся центром **n**-угольника, а одну из вершин **n**-угольника – на полярной оси при нулевом угле ее поворота.

b) В графической области отобразите произвольный **n**-угольник, его вершины, объекты «точка на плоскости», красным цветом, а также описанную вокруг **n**-угольника окружность.

Используйте функцию **kmPointShow** для отображения.

## Объект «вектор на плоскости»

***Задание 3.1*** Спроектируйте внутреннее представление объекта «вектор на плоскости».

### *Выполнение задания 3.1*

Поставим в соответствие математическому понятию «вектор» объект «вектор на плоскости» в среде *Mathematica*.

Предположим, на плоскости зафиксирована прямоугольная декартова система координат. Следуем соглашениям, прописанным ранее.

**Представлением объекта «вектор на плоскости»** в *Mathematica* будем называть выражение вида

**kmVector[km, "id", {x, y}]** (3.1)

где **x, y** – одноименные координаты вектора в прямоугольной декартовой системе координат.

Запишите в текстовую ячейку образец, описывающий множество выражений вида (3.1).

***Задание 3.2*** Постройте булеву функцию **NamedQ**, позволяющую определить, имеет ли представленный объект «вектор на плоскости» имя.

### *Выполнение задания 3.2*

Выполнить самостоятельно.

***Задание 3.3*** Спроектируйте функции-конструкторы объекта «вектор на плоскости».

### *Выполнение задания 3.3*

Проектируем конструкторы вектора таким образом, чтобы при расширении и/или уточнении способов задания вектора изменялось как можно меньшее их количество. Для этого выделим базовые конструкторы.

Такой подход позволит удобен тем, что если возникнет необходимость пере-/до- проектировать нашу систему, то достаточно переписать именно эти базовые конструкторы. Остальные же глобальные определения не потребуют переделки, они будут продолжать работать.

Выявим способы, которыми мы можем задавать вектор. Задаваемой информацией может быть

1) имя и декартовы координаты вектора;
2) декартовы координаты вектора;
3) имя вектора и две различные точки: начало и конец направленного отрезка, соответствующего вектору;
4) две различные точки: начало и конец направленного отрезка, соответствующего вектору;
5) имя, орт (единичный вектор, со направленный с задаваемым вектором), и длина вектора.
6) орт и длина вектора.

Введем обозначения, которые будем использовать в образцах: **coord** – координаты вектора, **id** – имя вектора, **P0** и **P1** – экземпляры объекта «точка на плоскости», указывающие соответственно начало и конец направленного отрезка, соответствующего вектору.

Тогда формальные, входные параметры конструкторов для случаев 1) – 4) можно записать в виде:

1) **kmVector[id_String, coord_List]**, где **id** – имя, **coord** – координаты вектора;

2) **kmVector[coord_List]**;

3) **kmVector[id_String,P0_kmPoint,P1_kmPoint]**;

4) **kmVector[P0_kmPoint,P1_kmPoint]**.

Выберем в качестве базового способ задания 1). Покажем далее, что каждый из конструкторов 2) – 4) можно свести к работе конструктора 1).

При способе задания 2), когда указываются только координаты вектора, мы вызовем базовый конструктор 1), указав в качестве имени вектора пустую строку

```
kmVector[coord_List] := kmVector["", coord]

kmVector[{2, -1}]

kmVector[, {2, -1}]
```

Далее построим определение 3), сведя его к базовому способу задания вектора. При этом в правой части правила для вызова конструктора 1) в качестве второго аргумента укажем выражение, которое вычислит координаты вектора, используя координаты начала и конца направленного отрезка, соответствующего вектору.

```
kmVector[id_String,P0_kmPoint,P1_kmPoint]:=
    kmVector[id, выражение для вычисления
                                    координат вектора]
```

Рассмотрим теперь случай, когда вектор задан способом 4).

В отличие от способа задания 3), здесь имя вектора не указывается. Тогда мы можем вычислить имя вектора, если именованы начало и конец направленного отрезка, соответствующего вектору. В противном случае зададим экземпляру объекта нарицательное имя «Вектор».

Таким образом, при способе задания 4) мы вызываем конструктор вида 3), который, в свою очередь, передаст вычисление базовому конструктору 1).

```
kmVector[P0_kmPoint,P1_kmPoint]:=
    kmVector[выражение для вычисления имени вектора,
                                   P0,P1]
```

Конструкторы 5) и 6) спроектируйте самостоятельно, по аналогии с проектированием конструкторов 2), 3) и 4).

***Задание 3.4*** Создайте функции-конструкторы объекта «вектор на плоскости» для случаев, когда вектор задается двумя различными точками: началом и концом направленного отрезка, соответствующего вектору, с указанием имени либо нет. Функции должны сводить вычисления к базовому конструктору 1).

### *Выполнение задания 3.4*

Рассмотрим случай 3), когда вектор задается именем и двумя различными точками: началом и концом направленного отрезка, соответствующего вектору.

Зададим точки, имеющие имена `A` и `B` соответственно, ассоциируем их с символами `P0` и `P1`

```
{P0, P1} = MapThread[kmPoint[#1, #2] &,
  {{"A", "B"}, {{3, 4}, {-2, 3}}}]
```

```
{A(3, 4), B(-2, 3)}
```

Чтобы использовать базовый конструктор 1), следует вычислить координаты вектора

```
P1@"coord" - P0@"coord"                    {-5, -1}
```

и указать выражение для вычисления координат вектора в качестве второго аргумента вызываемого конструктора 1)

```
kmVector[id_String, P0_kmPoint, P1_kmPoint] :=          (3.2)
 kmVector[id, P1@"coord" - P0@"coord"]
```

Таким образом, в случае задания вектора по имени и координатам начала и конца направленного отрезка, соответствующего вектору, мы готовим информацию для работы базового конструктора 1)

```
kmVector["CD", P0, P1]                    kmVector[CD, {-5, -1}]
```

Базовый конструктор, строящий представление вектора при его способе задания 1), мы напишем позже.

Следует отметить также принцип формирования имени вектора. В случае 3), когда мы задаем имя вектора непосредственно в аргументах конструктора, игнорируются имена точек, посредством которых этот вектор определяется.

Рассмотрим теперь случай 4), когда вектор задается двумя различными точками: началом и концом направленного отрезка, соответствующего вектору. Этот способ задания отличается от способа 3) тем, что имя вектора не указано явно.

Тогда формирование имени может взять на себя создаваемое глобальное определение, и как только имя сформировано, можно вызвать конструктор 3).

Для формирования имени примем следующее, общепринятое в математике, соглашение: если обе заданные точки имеют имя, то вектор именуем, подряд записывая имя точки-начала и имя точки-конца направленного отрезка.

```
StringJoin[P0@"id", P1@"id"]                         AB
```

Узнать, именованы ли обе точки, поможет функция **NamedQ**, которая должна быть определена и для объекта **kmPoint**.

```
And @@ NamedQ /@ {P0, P1}                        True
```

Тогда возможность 4) задания вектора определяется конструктором

```
kmVector[P0_kmPoint, P1_kmPoint] :=
 kmVector[
  If[And @@ NamedQ /@ {P0, P1},
   StringJoin[P0@"id", P1@"id"], ""], P0, P1]
```
(3.3)

При выполнении правила (3.3) процесс вычисления передается правилу (3.2), которое, в свою очередь, вызывает базовый конструктор 1)

```
kmVector[P0, P1]                    kmVector[AB, {-5, -1}]
```

***Задание 3.5*** Постройте конструктор объекта «вектор на плоскости» в случае, когда он именован либо нет и задается направлением (вектором-ортом) и длиной. Функция должны сводить вычисления к базовому конструктору.

### *Выполнение задания 3.5*

Выполнить самостоятельно, аналогично выполнению задания 3.4.

***Задание 3.6*** Создайте возможность задания объекта «вектор на плоскости» посредством указания имени вектора и его декартовых координат.

### *Выполнение задания 3.6*

Выполнить самостоятельно.

***Задание 3.7*** Напишите оператор, возвращающий имя объекта «вектор на плоскости», если оно существует, и нарицательное имя **"Вектор"** в противном случае.

### *Выполнение задания 3.7*

Выполнить самостоятельно.

***Задание 3.8*** Напишите оператор, возвращающий координаты заданного объекта «вектор на плоскости».

### *Выполнение задания 3.8*

Выполнить самостоятельно.

### *Задание 3.9*

Напишите оператор, возвращающий длину заданного объекта «вектор на плоскости».

### *Выполнение задания 3.9*

Построим сначала компьютерную модель для вычисления длины вектора по его заданным координатам.

Пусть вектор *V* имеет координаты *(Vx, Vy)*. В *Mathematica* координаты вектора представлены списком, содержащим два элемента `{Vx, Vy}`. Тогда операция извлечения корня квадратного из скалярного произведения списка самого на себя (**`Dot`**), представленная в виде

```
√{Vx, Vy}.{Vx, Vy}
```

$$\sqrt{Vx^2 + Vy^2}$$

может служить компьютерной моделью для вычисления длины вектора.

Завершите выполнение задания самостоятельно.

***Задание 3.10*** Напишите оператор, возвращающий орт заданного объекта «вектор на плоскости».

### *Выполнение задания 3.10*

*Ортом вектора* $V \neq 0$ называют вектор единичной длины, со направленный с вектором *V*.

Орт вектора *V* будем обозначать *Ve* и вычислять по формуле

$$Ve = \frac{V}{|V|}, \text{ где } |V| \text{ – длина вектора } V,\ V \neq 0.$$

Завершите выполнение задания самостоятельно.

## Литература


1. Голубева Л.Л., Малевич А.Э., Щеглова Н.Л. Компьютерная математика. Символьный пакет *Mathematica*. Лаб. практикум в 2 ч. Ч 1.- Минск: БГУ, 2012. – 235 с.