

2.3. Построение графиков и диаграмм

Для построения графиков и диаграмм используется библиотека [`matplotlib.pyplot`](https://matplotlib.org/). Эта библиотека не является стандартной, в большинстве сред разработки (в том числе и IDLE(Python)) ее необходимо установить.

Перед первым применением библиотеки ее необходимо подключить. Чтобы упростить обращение к функциям библиотеки, можно присвоить ей имя, например, `plt`:

```
import matplotlib.pyplot as plt
```

С помощью этой библиотеки создается графическая область, на которой отображаются следующие элементы:

- линии;
- оси координат;
- заголовок;
- легенды линий;
- многое другое.

Графическая область представляет собой прямоугольную сетку, координаты узлов которой определяются либо автоматически, либо могут быть установлены пользователем. В графическую область можно вывести точку с координатами (**x**, **y**) (рисунок 5). Значение **x** увеличивается по горизонтали слева направо, значение **y** увеличивается по вертикали снизу вверх, как это принято в декартовой системе координат.

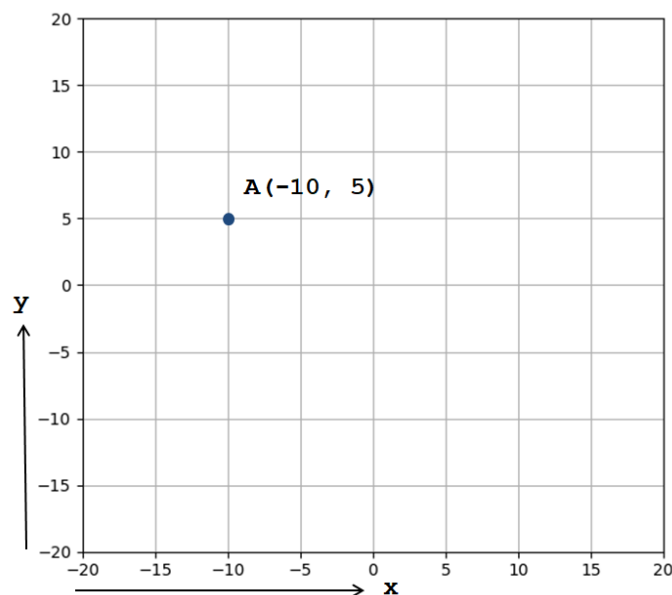


Рисунок 5. Область рисования

2.3.1. Построение линий

Для построения линий по точкам на плоскости используется метод `plot()`:

```
plot(x, y, параметры),
```

где **x** – список значений по оси X,

y – список значений по оси Y;

параметры – формат отображения линии на графике (ее характеристики).

Например, построим график по точкам, координаты которых представлены в виде двух списков (последний оператор нужен, чтобы отобразить линию в области построения):

```
import matplotlib.pyplot as plt
plt.plot([1, 5, -3, -0.5], [1, 25, 9, 0.25])
plt.show()
```

В результате будет построена ломаная линия (рисунок 6).

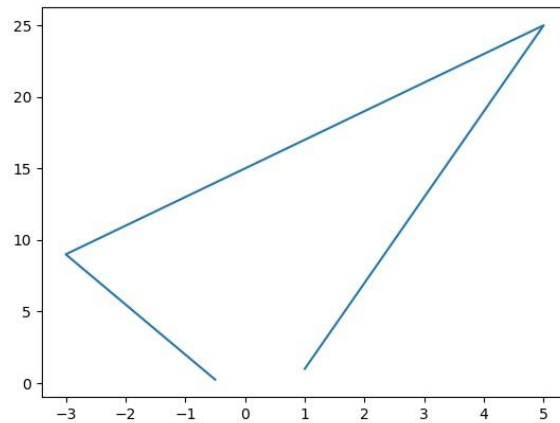


Рисунок 6. Линия, построенная по точкам

Библиотека [matplotlib.pyplot](https://matplotlib.org/) позволяет устанавливать следующие характеристики линии:

- маркер точек, по которым строится линия, его размер;
- стиль (сплошная, пунктирная, отсутствует и пр);
- цвет (красный, зеленый и пр);
- толщина (в пикселях).

Это можно сделать, либо указав соответствующие параметры в методе **plot()**, как показано в таблице 15, либо использовать метод **setp()**, который устанавливает характеристики заданной первым параметром линии.

Таблица 15 – Установка характеристик линии в **plot()**

Метод с параметрами	Описание
<code>plot(x, y)</code>	Создает линию, в которой используются параметры отображения, установленные по умолчанию
<code>plot(x, y, "bo")</code>	Создает точки на графике, которые обозначаются маркерами круглой формы синего цвета
<code>plot(y)</code>	Создает линию, в качестве значения x используется интервал от 0 до количества значений в списке y минус 1
<code>plot(y, "r+")</code>	Создает точки на графике, которые обозначаются маркером «+» красного цвета
<code>plot(x, y, "go--", linewidth=2, markersize=12)</code>	Создает пунктирную линию зеленого цвета, толщиной 2, в качестве маркеров используются круги радиусом 12 пикселей
<code>plot(x, y, color="green", marker="o", linestyle="dashed", linewidth=2, markersize=12)</code>	Другой способ описания предыдущей линии

Для того чтобы использовать метод **setp()** для установки характеристик линий, необходимо

- дать линии имя:
`line = plt.plot([1, 5, -3, -0.5], [1, 25, 9, 0.25])`

• задать необходимые характеристики для линии, которая указывается первым параметром метода **setp()**:

```
plt.setp(line, color="red", linewidth=2, marker="o")
```

- показать линии в области построения:

```
plt.show()
```

Результат построения показан на рисунке 7.

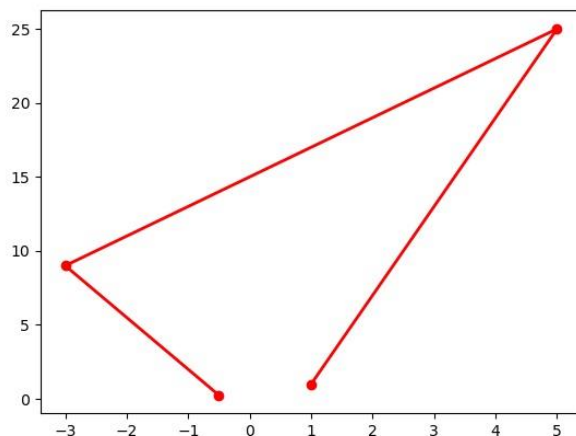


Рисунок 7. Красная линия с маркерами, построенная по точкам

2.3.2. Оси координат

Для построения и настройки координатных осей используется метод `gca()`. На графике могут быть отображены 4 оси, обращения к которым осуществляются следующим образом:

```
gca().spines["left"] - левая ось  
gca().spines["bottom"] - нижняя ось  
gca().spines["top"] - верхняя ось  
gca().spines["right"] - правая ось
```

Для каждой из них можно указать ее **расположение** с помощью функции:

```
set_position("значение")
```

где значение – положение оси на графике, при этом можно задать:

zero – в точке с координатой 0 по соответствующей оси;

center – по центру области вывода графиков.

Управление видимостью оси осуществляется с помощью метода:

```
set_visible(значение)
```

в котором значение может быть либо **True** (если ось нужно показать, установлено по умолчанию), либо **False** (если ось нужно «спрятать»).

Например, нарисуем нашу ломаную в традиционной декартовой системе координат.

Для этого уберем верхнюю и правую ось, а левую и нижнюю установим в позицию **zero**:

```
import matplotlib.pyplot as plt  
# формируем линию  
line = plt.plot([1, 5, -3, -0.5], [1, 25, 9, 0.25])  
  
# задаем формат ее вывода  
plt.setp(line, color="red", linewidth=2, marker="o")  
  
# устанавливаем две оси в положение zero  
plt.gca().spines["left"].set_position("zero")  
plt.gca().spines["bottom"].set_position("zero")  
  
# скрываем остальные две  
plt.gca().spines["top"].set_visible(False)  
plt.gca().spines["right"].set_visible(False)  
  
# отображаем область построения  
plt.show()
```

Результат выполнения программы показан на рисунке 8.

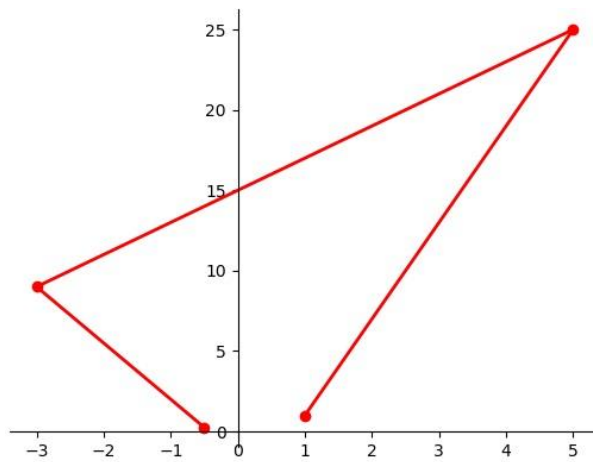


Рисунок 8. Линия в декартовой системе координат

2.3.3. Заголовок

В область построения можно вывести заголовок с помощью метода:

```
title("текст").
```

Если в нашу программу добавить оператор:

```
plt.title("Пример рисования линий")
```

то в области построения появится новый элемент – заголовок (рисунок 9)

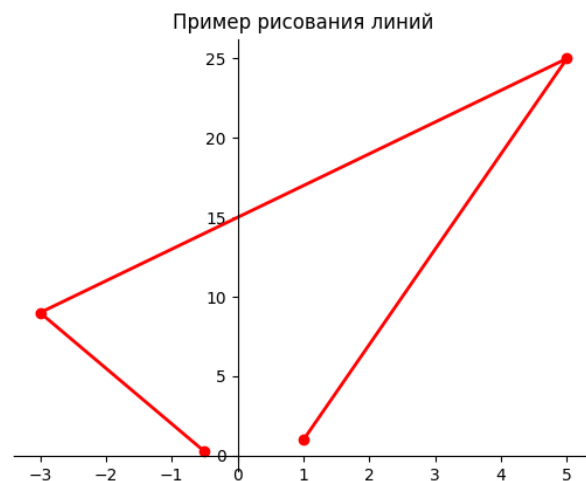


Рисунок 9. График с заголовком

2.3.4. Легенда

Легенда – это условное обозначение графиков. Легенда показывает названия и маркеры линий, используемых при их отображении в области построения.

Для каждой линии задается параметр – ее подпись, которая будет выводиться в качестве легенды. Это можно сделать двумя способами: либо задать параметр **label** при создании линии:

```
line_blue = plt.plot([1, 5, -3, -0.5], [1, 25, 9, 0.25], label='синяя линия')
```

либо задать этот параметр одновременно с установкой параметров стиля линии:

```
plt.setp(line_red, color="red", linewidth=2, label='красная линия')
```

Для вывода в область построения легенды используется метод **legend()**. В этом случае для расположения легенды выбирается «наилучшее» ее положение в области построения, то есть то, которое не перекрывает линии в области построения.

Если необходимо изменить положение вывода, используется метод:

```
legend(loc="значение")
```

Параметр значение определяет положение легенды на графике. Он может принимать следующие значения: `best`, `upper right`, `upper left`, `lower left`, `lower right`, `right`, `center left`, `center right`, `lower center`, `upper center`, `center`.

Например, построим две ломанные линии, зададим для них легенду.

```
import matplotlib.pyplot as plt

# создаем ломанные линии
line_blue = plt.plot([1, 5, -3, -0.5], [1, 25, 9, 0.25 ],
                     label='синяя линия')
line_red = plt.plot([-6, -5, 0, 8], [-5, -2, -3, 4 ])

# задаем стили для линий
plt.setp(line_blue, color="blue", linewidth=2, marker="v" )
plt.setp(line_red, color="red", linewidth=2, marker="o",
         label='красная линия')

# отображаем легенду
plt.legend()

# показываем область рисования
plt.show()
```

Результат выполнения программы показан на рисунке 10.

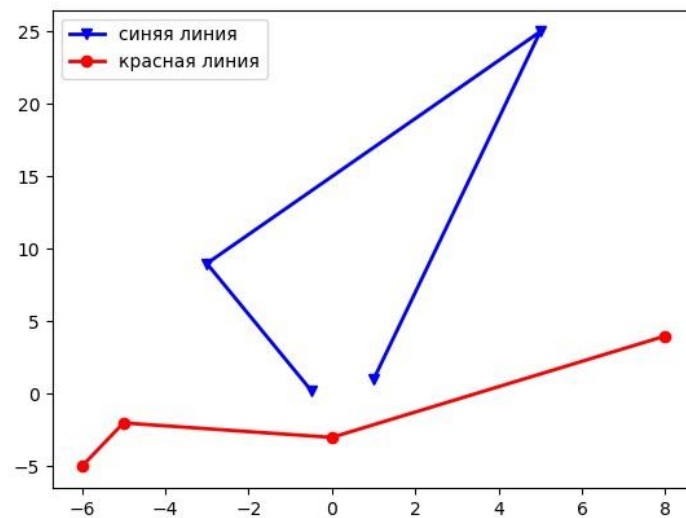


Рисунок 10. Вывод линий с легендой

Задания для самопроверки

1. Дан фрагмент программного кода:

```
import matplotlib.pyplot as plt
x_list = [1, 5, -3]
y_list = [-5, 6, 1]
#функция создания линии
plt.show()
```

Вставить оператор вместо комментария, чтобы была выведена пунктирная ломаная линия зеленого цвета, маркеры отображаются в виде знака плюс.

2. Дан фрагмент программного кода, в котором строится линия по координатам точек, занесенных в списки **x_list** и **y_list**.

```
import matplotlib.pyplot as plt
plt.plot(x_list, y_list, 'ro--', linewidth=4, markersize=10)
```

Выберите фрагменты кода, с помощью которых можно построить линию с теми же характеристиками:

- a. `plt.plot(x, y, color='red', marker='o',
 linestyle='dashed', linewidth=4, markersize=10)`
- b. `plt.plot(x, y, color='red', marker='o', linewidth=4, markersize=10)`
- c. `plt.plot(x, y, color='red', marker='o', linestyle='solid',
 linewidth=4, markersize=10)`
- d. `line = plt.plot(x, y)
 plt.setp(line, color='red', marker='o', linestyle='dashed',
 linewidth=4, markersize=10)`