Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

«Основы работы с пакетом matplotlib»

ОТЧЕТ
по лабораторной работе №4
дисциплины
«Технологии распознавания образов»

Выполнил:
Сотников Андрей Александрович
2 курс, группа ПИЖ-б-о-21-1,
011.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

_____
(подпись)

Проверил:

_____
(подпись)

Отчет защищен с оценкой_____Дата защиты_____

Ставрополь, 2022 г.

Проработка примеров из лабораторной работы:

```
In [1]: import matplotlib.pyplot as plt
        %matplotlib inline
        plt.plot([1, 2, 3, 4, 5], [1, 2, 3, 4, 5])
```

```
Out[1]: [<matplotlib.lines.Line2D at 0x8a4f252760>]
```

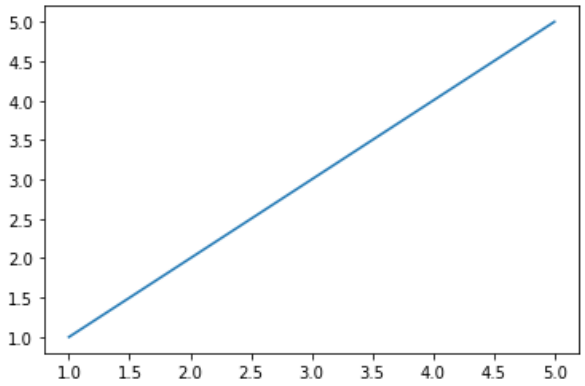


Рисунок 1 – Проработка примеров

Построение графика

```
In [3]: import numpy as np

        # Независимая (x) и зависимая (y) переменные
        x = np.linspace(0, 10, 50)
        y = x

        # Построение графика
        plt.title("Линейная зависимость y = x") # заголовок

        plt.xlabel("x") # ось абсцисс

        plt.ylabel("y") # ось ординат

        plt.grid() # включение отображение сетки

        plt.plot(x, y) # построение графика
```

```
Out[3]: [<matplotlib.lines.Line2D at 0x8a4f375760>]
```

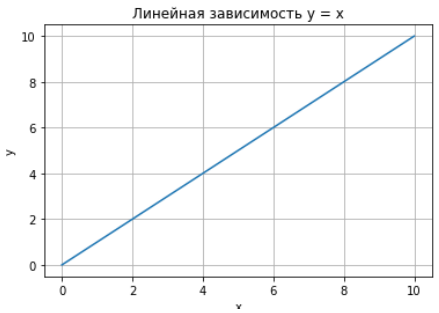


Рисунок 2 – Проработка примеров

In [5]:
```
import numpy as np

# Независимая (x) и зависимая (y) переменные
x = np.linspace(0, 10, 50)
y = x

# Построение графика
plt.title("Линейная зависимость y = x с пунктироной линией") # заголовок

plt.xlabel("x") # ось абсцисс

plt.ylabel("y") # ось ординат

plt.grid() # включение отображение сетки

plt.plot(x, y, 'r--') # построение графика
```

Out[5]:
```
[<matplotlib.lines.Line2D at 0x8a4bd38580>]
```

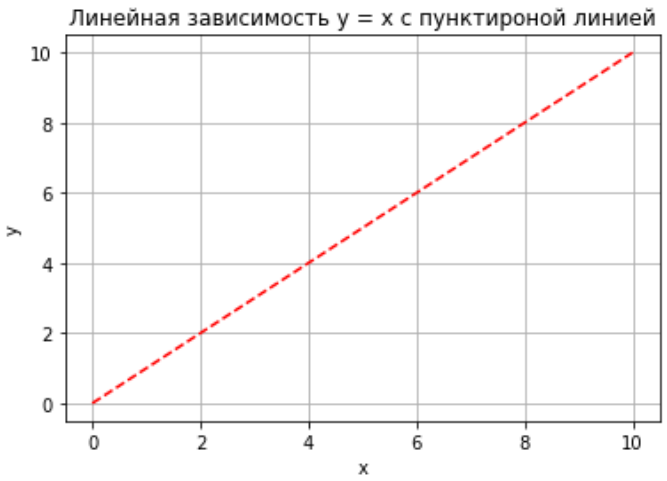


Рисунок 3 – Проработка примеров

Несколько графиков в одном окне (так же была устранена ошибка в коде из примера)

In [8]:
```
#линейная зависимость
x = np.linspace(0, 10, 50)
y1 = x

# Квадратичная зависимость
y2 = [i**2 for i in x]

# Построение графика
plt.title("Зависимости: y1 = x, y2 = x^2") # заголовок

plt.xlabel("x") # ось абсцисс

plt.ylabel("y1, y2") # ось ординат

plt.grid() # включение отображение сетки

plt.plot(x, y1, x, y2) # построение графика
```

Out[8]:
```
[<matplotlib.lines.Line2D at 0x8a4bdedee0>,
 <matplotlib.lines.Line2D at 0x8a4bdedfd0>]
```

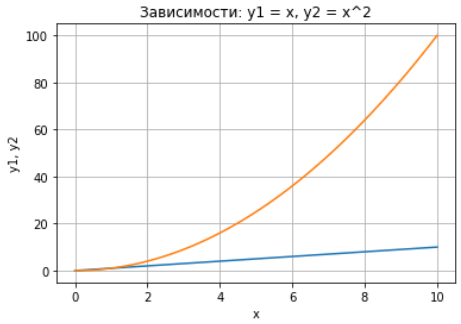


Рисунок 4 – Проработка примеров

Несколько раздельных полей с графиками

In [9]:
```
# Линейная зависимость
x = np.linspace(0, 10, 50)
y1 = x

# Квадратичная зависимость
y2 = [i**2 for i in x]

# Построение графиков
plt.figure(figsize=(9, 9))
plt.subplot(2, 1, 1)

plt.plot(x, y1) # построение графика

plt.title("Зависимости: y1 = x, y2 = x^2") # заголовок

plt.ylabel("y1", fontsize=14) # ось ординат

plt.grid(True) # включение отображение сетки
plt.subplot(2, 1, 2)

plt.plot(x, y2) # построение графика

plt.xlabel("x", fontsize=14) # ось абсцисс

plt.ylabel("y2", fontsize=14) # ось ординат
plt.grid(True)
```

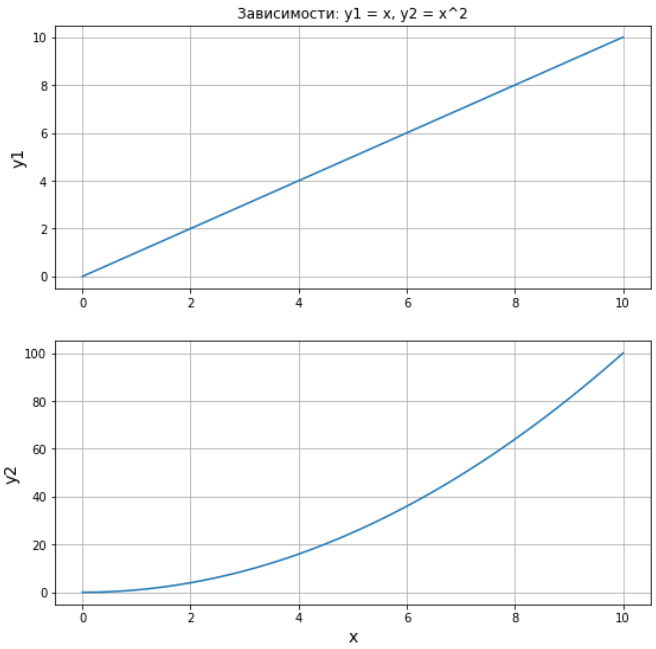


Рисунок 5 – Проработка примеров

Построение диаграммы для категориальных данных ¶

In [10]:
```
fruits = ["apple", "peach", "orange", "bannana", "melon"]
counts = [34, 25, 43, 31, 17]

plt.bar(fruits, counts)
plt.title("Fruits!")
plt.xlabel("Fruit")
plt.ylabel("Count")
```

Out[10]:
```
Text(0, 0.5, 'Count')
```

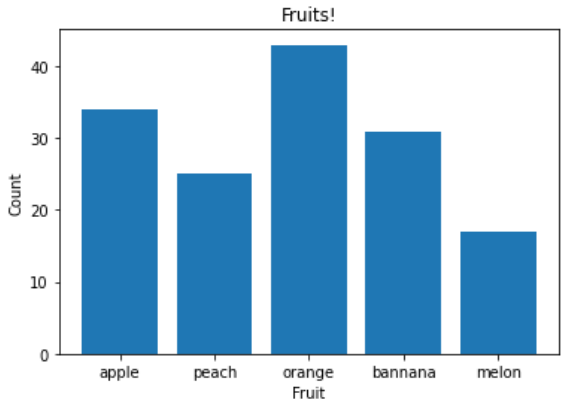


Рисунок 6 – Проработка примеров

Пример фигуры из лабораторной работы

In [12]:
```
import matplotlib.pyplot as plt
from matplotlib.ticker import (MultipleLocator, FormatStrFormatter,
AutoMinorLocator)
import numpy as np

x = np.linspace(0, 10, 10)
y1 = 4*x
y2 = [i**2 for i in x]

fig, ax = plt.subplots(figsize=(8, 6))

ax.set_title("Графики зависимостей: y1=4*x, y2=x^2", fontsize=16)

ax.set_xlabel("x", fontsize=14)

ax.set_ylabel("y1, y2", fontsize=14)

ax.grid(which="major", linewidth=1.2)

ax.grid(which="minor", linestyle="--", color="gray", linewidth=0.5)

ax.scatter(x, y1, c="red", label="y1 = 4*x")

ax.plot(x, y2, label="y2 = x^2")

ax.legend()

ax.xaxis.set_minor_locator(AutoMinorLocator())

ax.yaxis.set_minor_locator(AutoMinorLocator())

ax.tick_params(which='major', length=10, width=2)

ax.tick_params(which='minor', length=5, width=1)

plt.show()
```

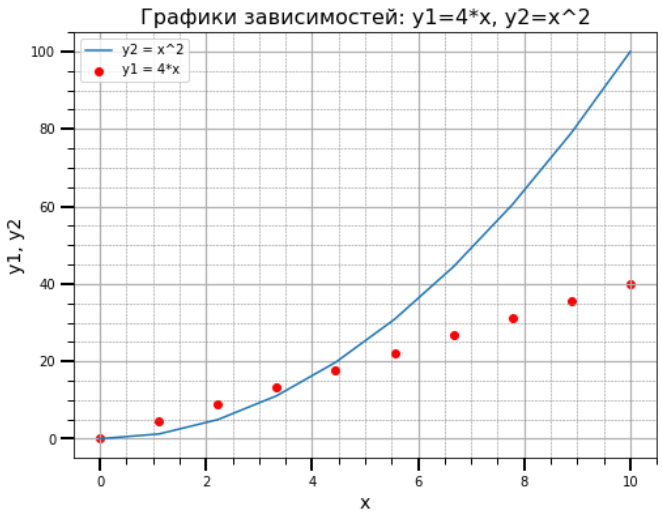


Рисунок 7 – Проработка примеров

Работа с инструментом pyplot

Построение графиков

In [13]:
```
import matplotlib.pyplot as plt
%matplotlib inline

plt.plot()
```

Out[13]: []

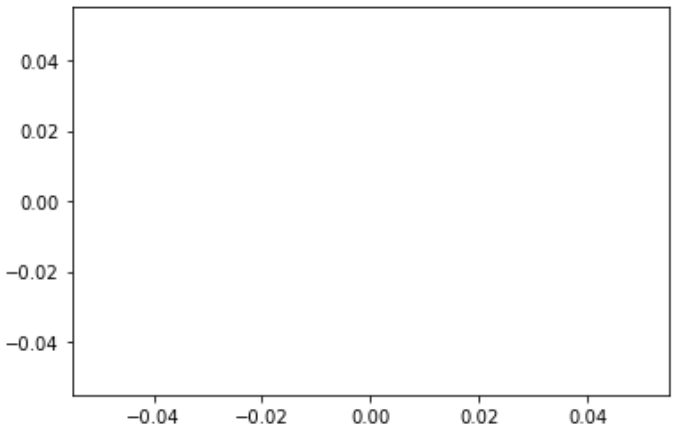
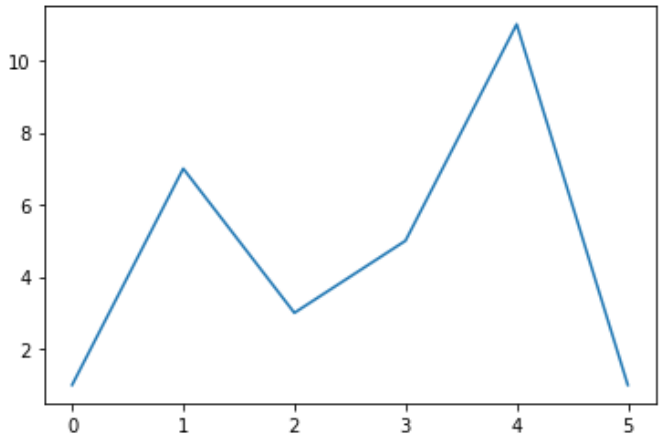


Рисунок 8 – Проработка примеров

```
In [14]: plt.plot([1, 7, 3, 5, 11, 1])
Out[14]: [<matplotlib.lines.Line2D at 0x8a4f4840a0>]
```




```
In [15]: plt.plot([1, 5, 10, 15, 20], [1, 7, 3, 5, 11])
Out[15]: [<matplotlib.lines.Line2D at 0x8a4f588c40>]
```

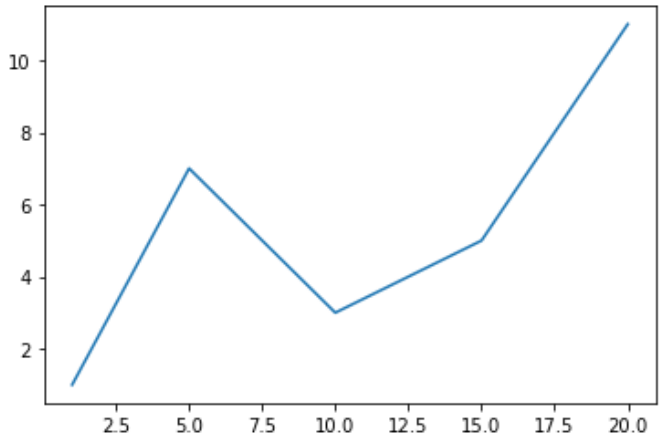


Рисунок 9 – Проработка примеров

Текстовые надписи на графике

```
In [16]: plt.xlabel('Day', fontsize=15, color='blue')
```

```
Out[16]: Text(0.5, 0, 'Day')
```

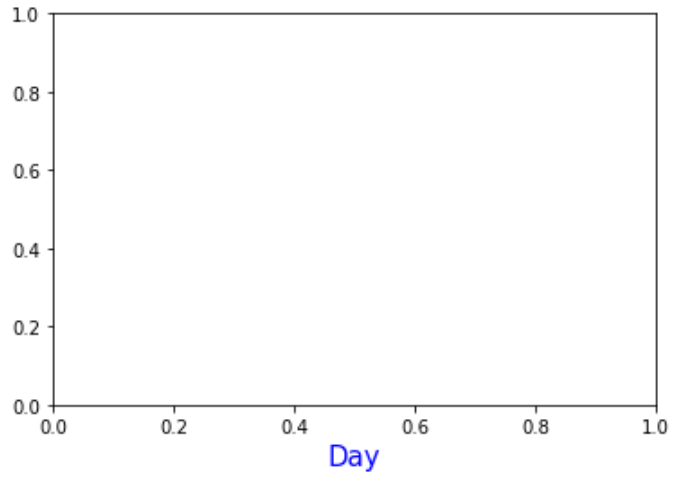
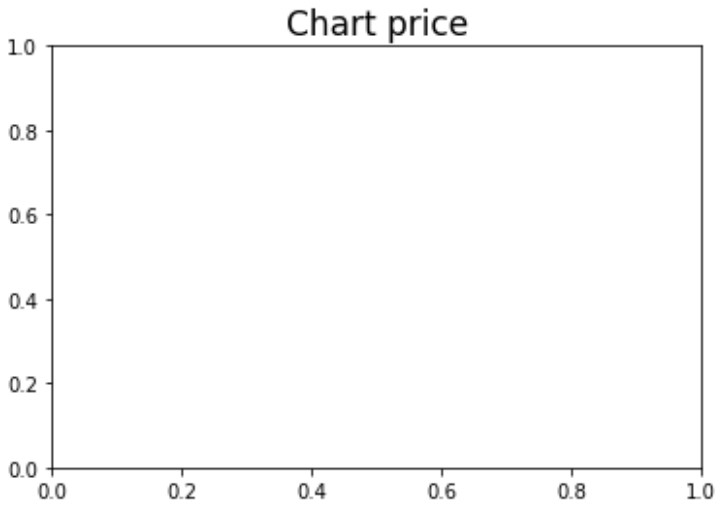


Рисунок 10 – Проработка примеров

```
In [17]: plt.title('Chart price', fontsize=17)
Out[17]: Text(0.5, 1.0, 'Chart price')
```




```
In [18]: plt.text(1, 1, 'type: Steel')
Out[18]: Text(1, 1, 'type: Steel')
```

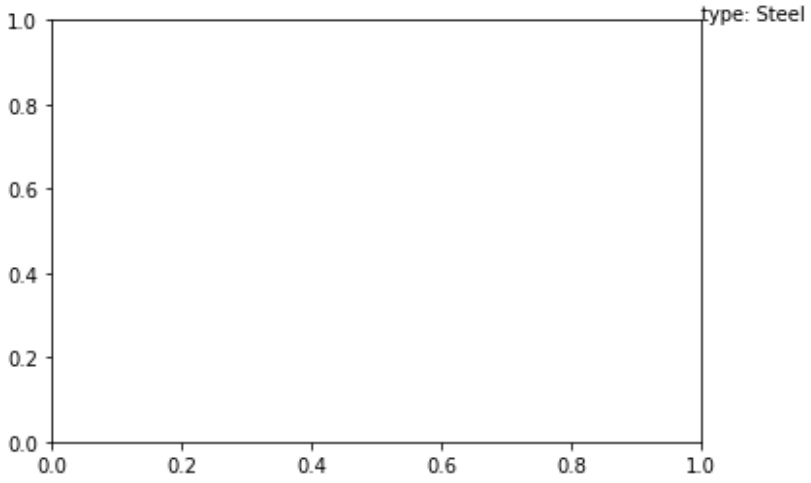


Рисунок 11 – Проработка примеров

```
In [21]: x = [1, 5, 10, 15, 20]
         y = [1, 7, 3, 5, 11]

         plt.plot(x, y, label='steel price')
         plt.title('Chart price', fontsize=15)
         plt.xlabel('Day', fontsize=12, color='blue')
         plt.ylabel('Price', fontsize=12, color='blue')

         plt.legend()
         plt.grid(True)

         plt.text(15, 4, 'grow up!')

Out[21]: Text(15, 4, 'grow up!')
```

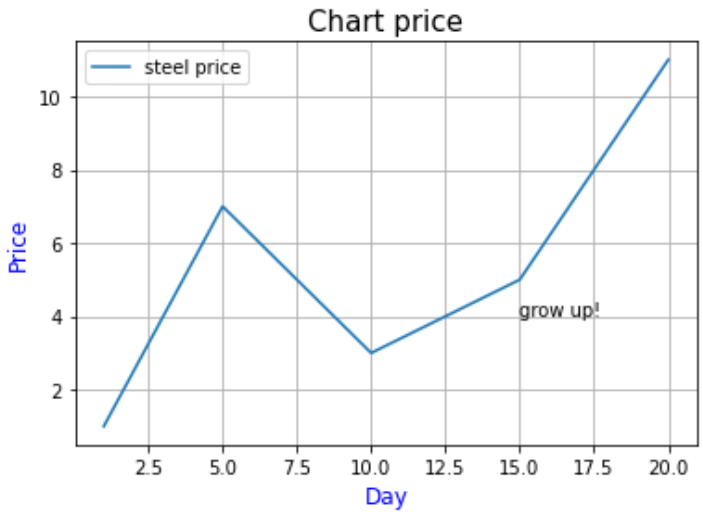


Рисунок 12 – Проработка примеров

Работа с линейным графиком

```
In [22]: plt.plot(x, y, color='red')

Out[22]: [<matplotlib.lines.Line2D at 0x8a4f5a82e0>]
```

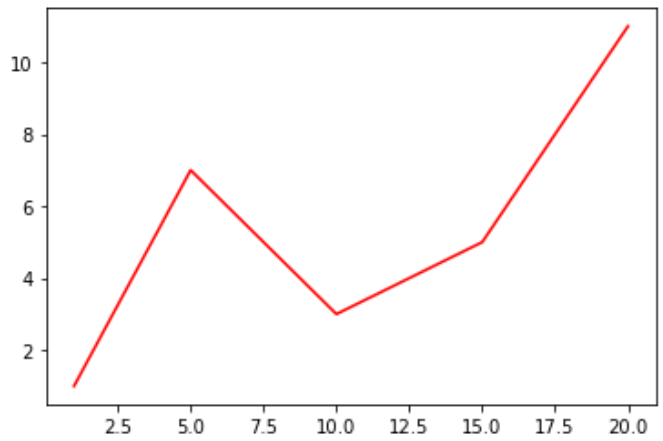


Рисунок 13 – Проработка примеров

Стиль линии графиков

```
In [25]: x = [1, 5, 10, 15, 20]
         y = [1, 7, 3, 5, 11]

         plt.plot(x, y, '--')
```

```
Out[25]: [<matplotlib.lines.Line2D at 0x8a507f8610>]
```

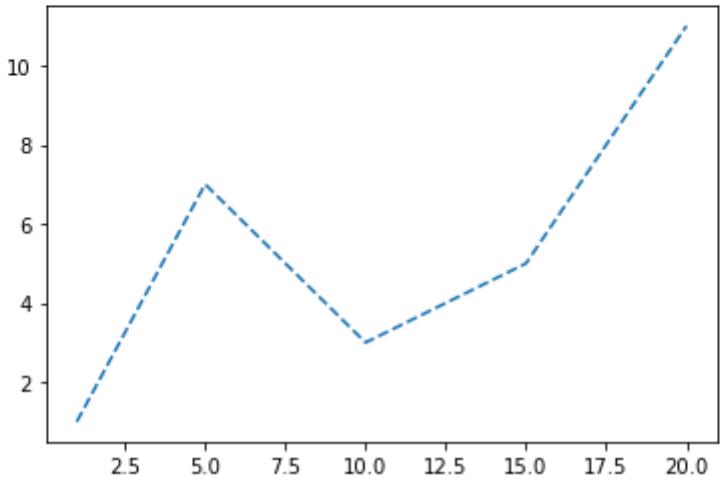


Рисунок 14 – Проработка примеров

```
In [26]: x = [1, 5, 10, 15, 20]
         y1 = [1, 7, 3, 5, 11]
         y2 = [i*1.2 + 1 for i in y1]
         y3 = [i*1.2 + 1 for i in y2]
         y4 = [i*1.2 + 1 for i in y3]

         plt.plot(x, y1, '-', x, y2, '--', x, y3, '-.', x, y4, ':')
```

```
Out[26]: [<matplotlib.lines.Line2D at 0x8a4f3fa190>,
          <matplotlib.lines.Line2D at 0x8a4f3fa8e0>,
          <matplotlib.lines.Line2D at 0x8a4f3fa100>,
          <matplotlib.lines.Line2D at 0x8a4f3fabe0>]
```

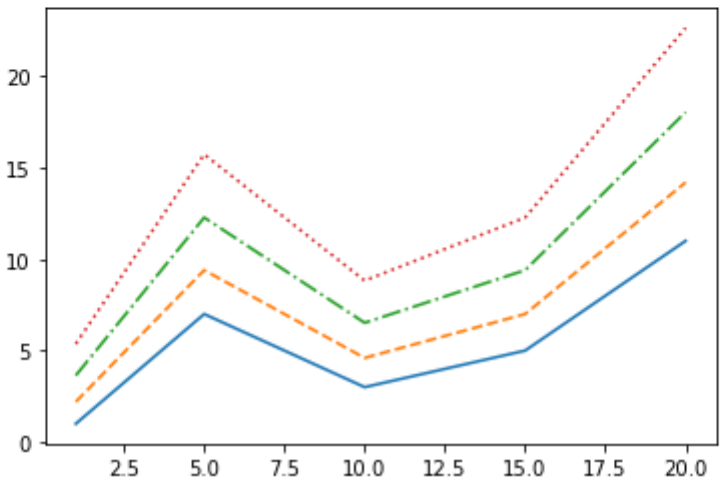
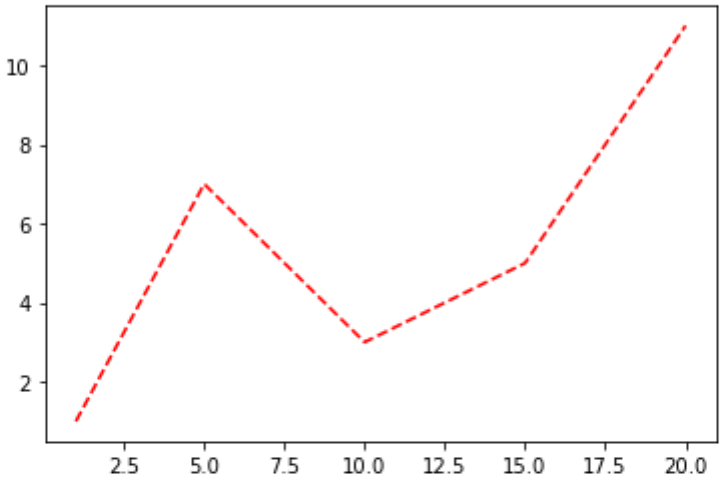


Рисунок 15 – Проработка примеров

Цвет линии

In [27]:
```
x = [1, 5, 10, 15, 20]
y = [1, 7, 3, 5, 11]

plt.plot(x, y, '--r')
```

Out[27]:
```
[<matplotlib.lines.Line2D at 0x8a4f468940>]
```



In [28]:
```
x = [1, 5, 10, 15, 20]
y = [1, 7, 3, 5, 11]

plt.plot(x, y, '--b')
```

Out[28]:
```
[<matplotlib.lines.Line2D at 0x8a5080e6a0>]
```

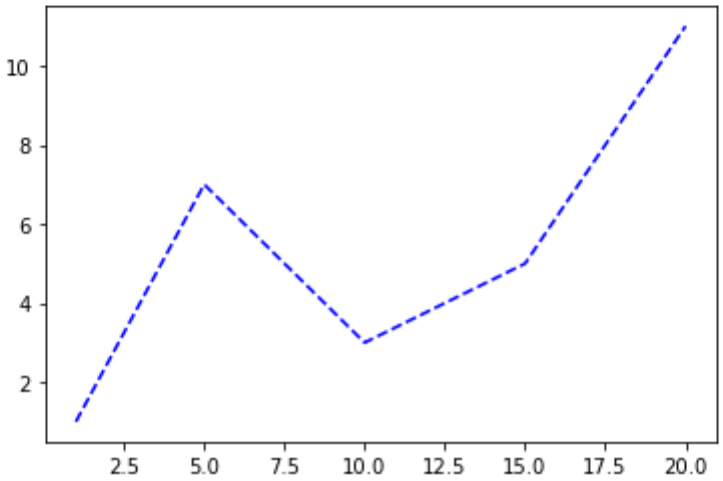
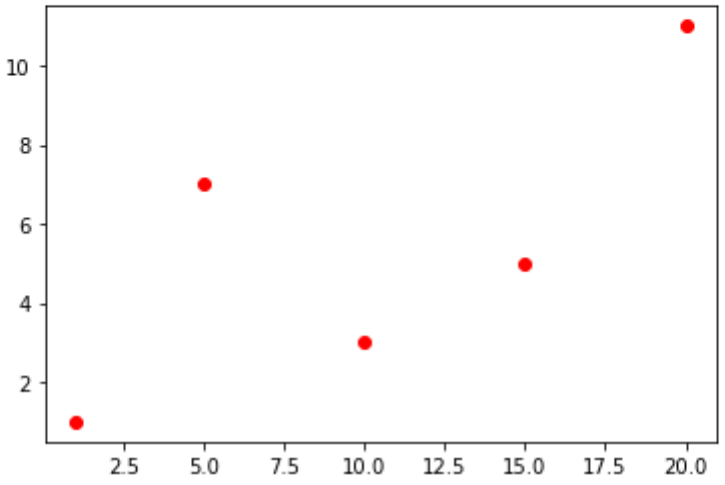


Рисунок 16 – Проработка примеров

Тип графика

```
In [29]: plt.plot(x, y, 'ro')
Out[29]: [<matplotlib.lines.Line2D at 0x8a5096be20>]
```




```
In [30]: plt.plot(x, y, 'bx')
Out[30]: [<matplotlib.lines.Line2D at 0x8a509c49a0>]
```

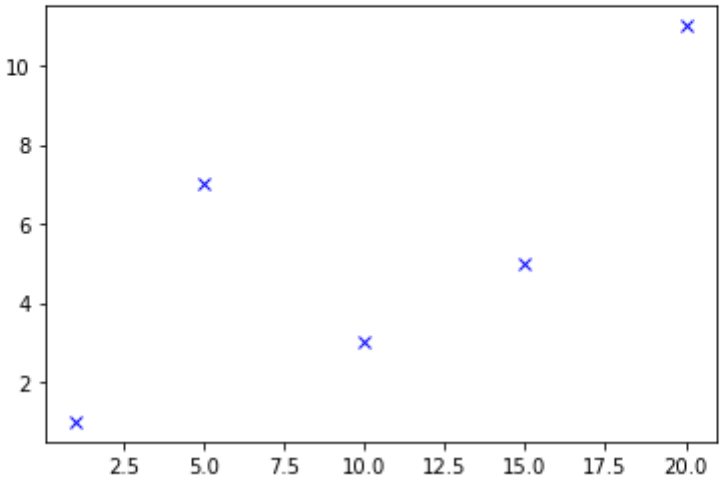


Рисунок 17 – Проработка примеров

Размещение графиков на разных полях

In [31]:
```
# Исходный набор данных
x = [1, 5, 10, 15, 20]
y1 = [1, 7, 3, 5, 11]
y2 = [i*1.2 + 1 for i in y1]
y3 = [i*1.2 + 1 for i in y2]
y4 = [i*1.2 + 1 for i in y3]

# Настройка размеров подложки
plt.figure(figsize=(12, 7))

# Вывод графиков
plt.subplot(2, 2, 1)
plt.plot(x, y1, '-')

plt.subplot(2, 2, 2)
plt.plot(x, y2, '--')

plt.subplot(2, 2, 3)
plt.plot(x, y3, '-.')

plt.subplot(2, 2, 4)
plt.plot(x, y4, ':')
```

Out[31]: [<matplotlib.lines.Line2D at 0x8a50abb5b0>]

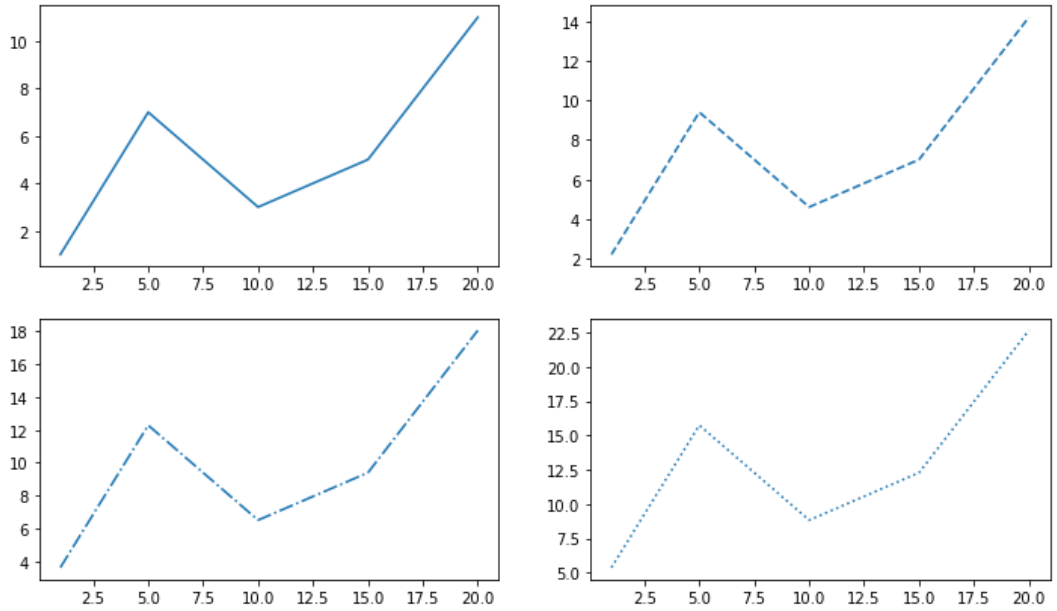


Рисунок 18 – Проработка примеров

Работа с функцией subplots()

In [32]:
```
fig, axs = plt.subplots(2, 2, figsize=(12, 7))

axs[0, 0].plot(x, y1, '-')
axs[0, 1].plot(x, y2, '--')
axs[1, 0].plot(x, y3, '-.')
axs[1, 1].plot(x, y4, ':')
```

Out[32]: [<matplotlib.lines.Line2D at 0x8a50bd0400>]

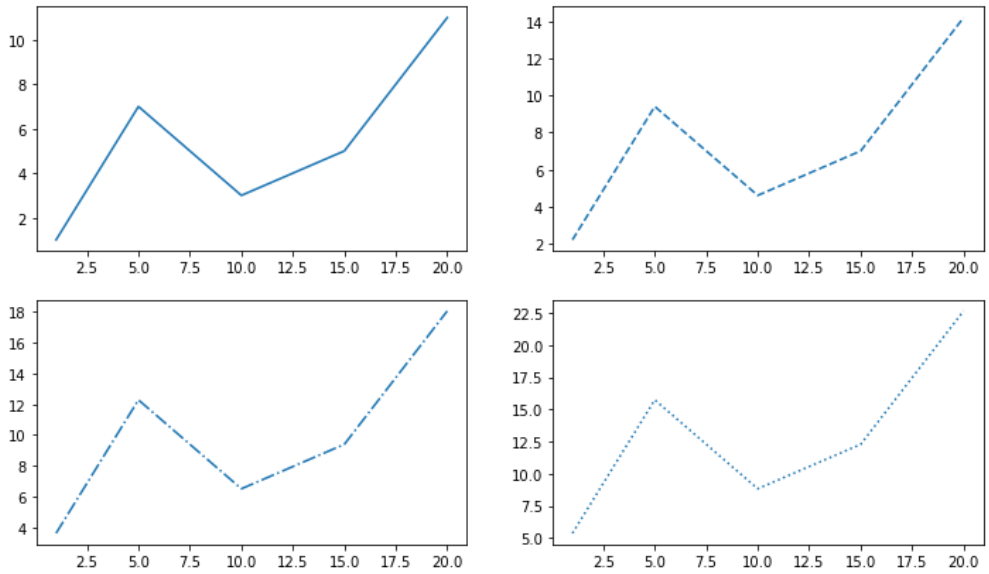


Рисунок 19 – Проработка примеров

Контрольные вопросы

1. Как осуществляется установка пакета matplotlib?

Существует два основных варианта установки этой библиотеки: в первом случае вы устанавливаете пакет Anaconda, в состав которого входит большое количество различных инструментов для работы в области машинного обучения и анализа данных (и не только); во втором – установить Matplotlib самостоятельно, используя менеджер пакетов.

Второй вариант – это воспользоваться менеджером pip и установить Matplotlib самостоятельно, для этого введите в командной строке вашей операционной системы следующие команды:

```
$ python -m pip install -U pip
$ python -m pip install -U matplotlib
```

2. Какая "магическая" команда должна присутствовать в ноутбуках Jupyter для корректного отображения графиков matplotlib?

```
import matplotlib.pyplot as plt
%matplotlib inline
```

3. Как отобразить график с помощью функции plot ?

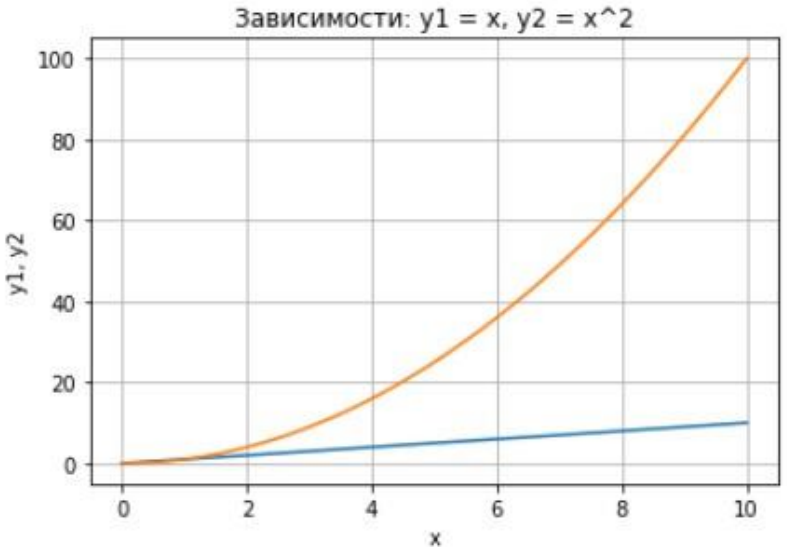
```
import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4, 5], [1, 2, 3, 4, 5])
plt.show()
```

4. Как отобразить несколько графиков на одном поле?

Построим несколько графиков на одном поле, для этого добавим квадратичную зависимость:

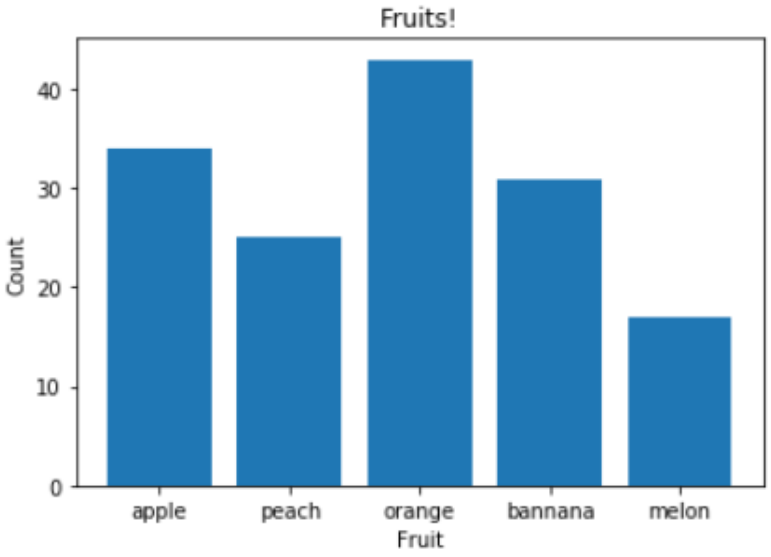
```
# Линейная зависимость
x = np.linspace(0, 10, 50)
y1 = x

# Квадратичная зависимость
y2 = [i**2 for i **in** x]
```

```
# Построение графика
plt.title("Зависимости: y1 = x, y2 = x^2") # заголовок
plt.xlabel("x")          # ось абсцисс
plt.ylabel("y1, y2")     # ось ординат
plt.grid()               # включение отображение сетки

plt.plot(x, y1, x, y2)  # построение графика
```

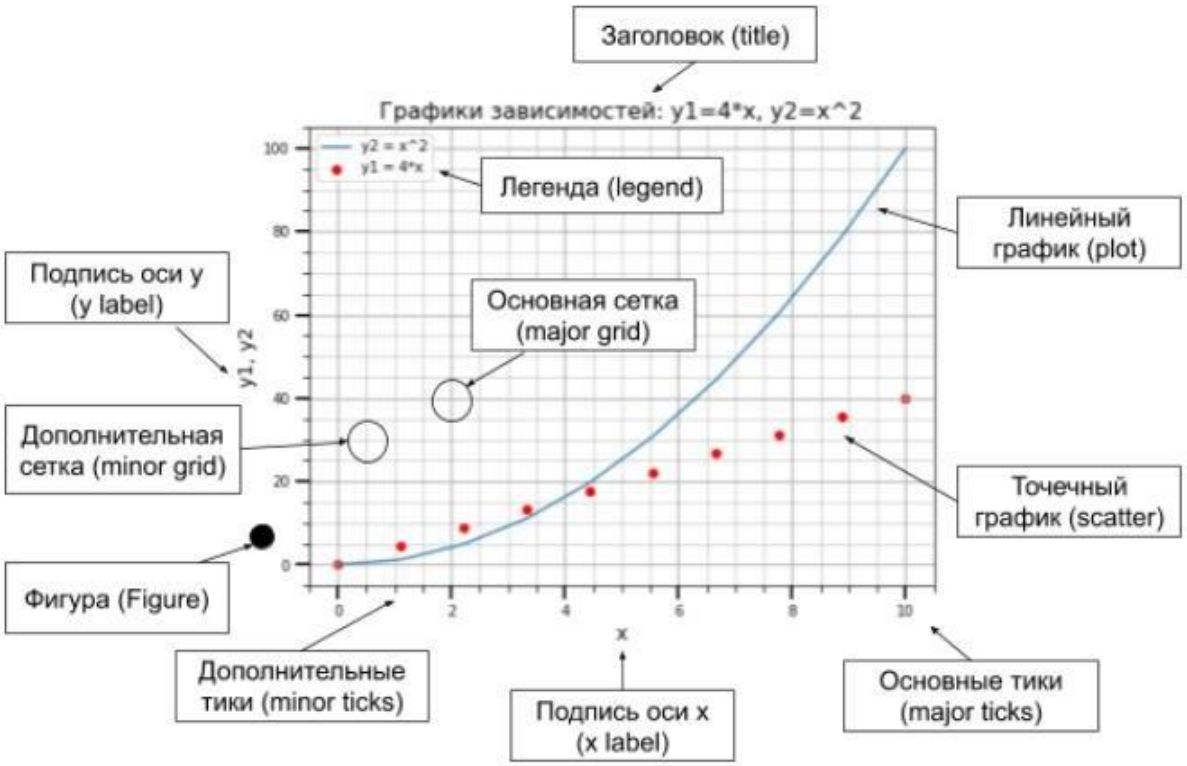


5. Какой метод Вам известен для построения диаграмм категориальных данных?

```
fruits = ["apple", "peach", "orange", "bannana", "melon"]
counts = [34, 25, 43, 31, 17]

plt.bar(fruits, counts)
plt.title("Fruits!")
plt.xlabel("Fruit")
plt.ylabel("Count")
```



6. Какие основные элементы графика Вам известны?



7. Как осуществляется управление текстовыми надписями на графике?

Наименование осей

Для задания подписи оси *x* используется функция xlabel(), оси *y* – ylabel(). Разберемся с аргументами данных функций.

Функции xlabel()/ylabel() принимают в качестве аргументов параметры конструктора класса matplotlib.text.Text. Пример использования:

```
plt.xlabel('Day', fontsize=15, color='blue')
```

Заголовок графика

Для задания заголовка графика используется функция *title()*:

```
plt.title('Chart price', fontsize=17)
```

Для функции *title()* также доступны параметры конструктора класса *matplotlib.text.Text*, часть из них представлена в описании аргументов функций xlabel() / ylabel().

Текстовое примечание

За размещение текста на поле графика отвечает функция *text()*, которой вначале передаются координаты позиции надписи, после этого – текст самой надписи.

```
plt.text(1, 1, 'type: Steel')
```

8. Как осуществляется управление легендой графика?

Легенда

Легенда будет размещена на графике, если вызвать функцию *legend()*, в рамках данного урока мы не будем рассматривать аргументы этой функции.

Разместим на уже знакомом нам графике необходимый набор подписей.

```
x = [1, 5, 10, 15, 20]
y = [1, 7, 3, 5, 11]

plt.plot(x, y, label='steel price')
plt.title('Chart price', fontsize=15)
plt.xlabel('Day', fontsize=12, color='blue')
plt.ylabel('Price', fontsize=12, color='blue')

plt.legend()
plt.grid(True)

plt.text(15, 4, 'grow up!')
```

9. Как задать цвет и стиль линий графика?

Стиль линии графика задается через параметр *linestyle*, который может принимать значения из приведенной ниже таблицы.

Значение параметра	**Описание**
'-' или *'solid'*	Непрерывная линия
'–' или *'dashed'*	Штриховая линия
'-.' или *'dashdot'*	Штрихпунктирная линия
':' или *'dotted'*	Пунктирная линия
'None' или ' ' или ''	Не отображать линию

Стиль линии можно передать сразу после указания списков с координатами без указания, что это параметр *linewidth*.

```
x = [1, 5, 10, 15, 20]
y = [1, 7, 3, 5, 11]

plt.plot(x, y, '--')
```

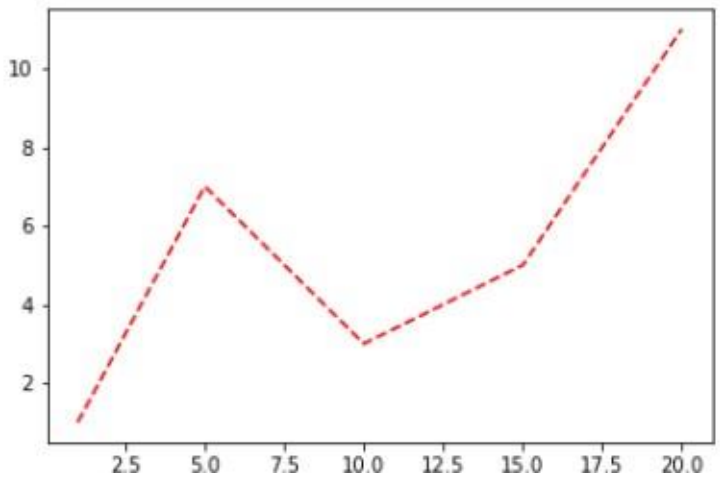
Задание цвета линии графика производится через параметр *color* (или *c*, если использовать сокращенный вариант). Значение может быть представлено в одном из следующих форматов:

- *RGB* или *RGBA* кортеж значений с плавающей точкой в диапазоне [0, 1] (пример: (0.1, 0.2, 0.3)
- *RGB* или *RGBA* значение в *hex* формате (пример: '#0a0a0a')
- строковое представление числа с плавающей точкой в диапазоне [0, 1] (определяет цвет в шкале серого) (пример: '0.7')
- символ из набора {*'b', 'g', 'r', 'c', 'm', 'y', 'k', 'w'*}
- имя цвета из палитры *X11/CSS4*
- цвет из палитры xkcd(https://xkcd.com/color/rgb/), должен начинаться с префикса *'xkcd:'*
- цвет из набора *Tableau Color* (палитра *T10*), должен начинаться с префикса *'tab:'*

Если цвет задается с помощью символа из набора {*'b', 'g', 'r', 'c', 'm', 'y', 'k', 'w'*}, то он может быть совмещен со стилем линии в рамках параметра *fmt* функции *plot()*.

Например штриховая красная линия будет задаваться так: '–r', а штрих пунктирная зеленая так '-.g'

```
x = [1, 5, 10, 15, 20]
y = [1, 7, 3, 5, 11]

plt.plot(x, y, '--r')
```



10. Как выполнить размещение графика в разных полях?

Размещение графиков на разных полях

Существуют три основных подхода к размещению нескольких графиков на разных полях:

- использование функции *subplot()* для указания места размещения поля с графиком;
- использование функции *subplots()* для предварительного задания сетки, в которую будут укладываться поля;
- использование *GridSpec*, для более гибкого задания геометрии размещения полей с графиками в сетке.

В этом уроке будут рассмотрены первые два подхода.

Работа с функцией subplot()

Самый простой способ представить графики в отдельных полях – это использовать функцию *supplot()* для задания их мест размещения. До этого момента мы не работали с Фигурой (*Figure*) напрямую, значения ее параметров, задаваемые по умолчанию, нас устраивали. Для решения текущей задачи придется один из параметров – размер подложки, задать вручную. За это отвечает аргумент *figsize* функции *figure()*, которому присваивается кортеж из двух *float* элементов, определяющих высоту и ширину подложки.

После задания размера, указывается местоположение, куда будет установлено поле с графиком с помощью функции subplot(). Чаще всего используют следующие варианты вызова subplot:

---

subplot(nrows, ncols, index)

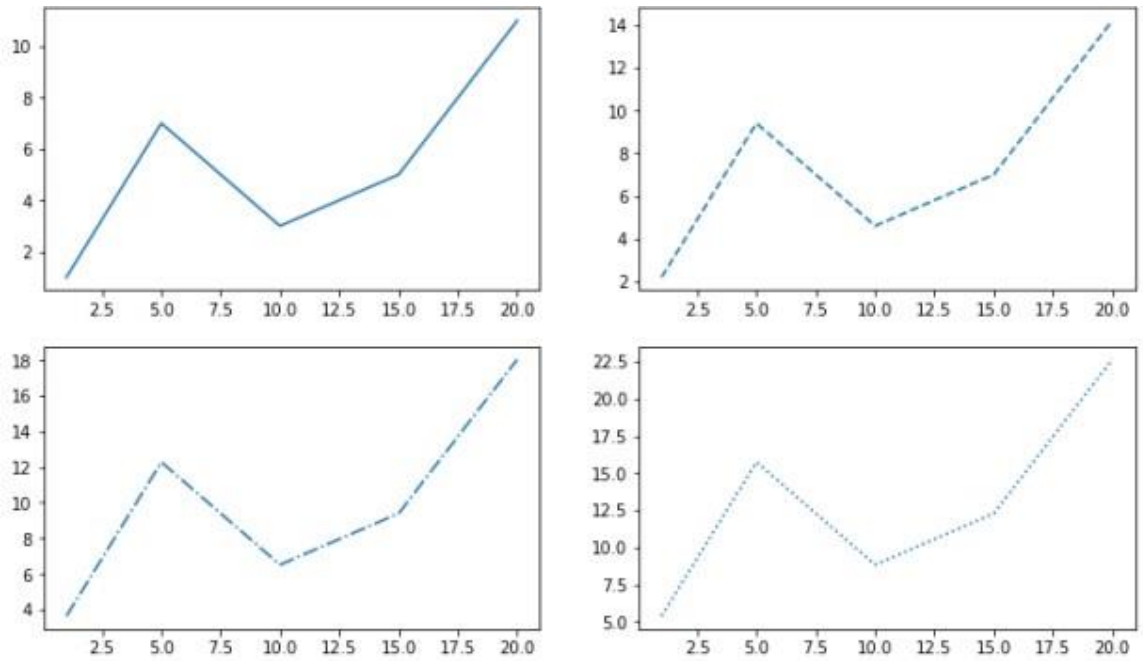
- nrows: int
  - Количество строк.
- ncols: int
  - Количество столбцов.
- index: int
  - Местоположение элемента.

subplot(pos)

- pos:int
  - Позиция, в виде трехзначного числа, содержащего информацию о количестве строк, столбцов и индексе, например 212, означает подготовить разметку с двумя строками и одним столбцов, элемент вывести в первую позицию второй строки. Этот вариант можно использовать, если количество строк и столбцов сетки не более 10, в ином случае лучше обратиться к первому варианту.

Рассмотрим на примере работу с данными функциями:

```
# Исходный набор данных
x = [1, 5, 10, 15, 20]
y1 = [1, 7, 3, 5, 11]
y2 = [i*1.2 + 1 for i in y1]
y3 = [i*1.2 + 1 for i in y2]
y4 = [i*1.2 + 1 for i in y3]

# Настройка размеров подложки
plt.figure(figsize=(12, 7))

# Вывод графиков
plt.subplot(2, 2, 1)
plt.plot(x, y1, '-')

plt.subplot(2, 2, 2)
plt.plot(x, y2, '--')

plt.subplot(2, 2, 3)
plt.plot(x, y3, '-.')

plt.subplot(2, 2, 4)
plt.plot(x, y4, ':')
```



Второй вариант использования *subplot()*:

```
# Вывод графиков
plt.subplot(221)
plt.plot(x, y1, '-')

plt.subplot(222)
plt.plot(x, y2, '--')

plt.subplot(223)
plt.plot(x, y3, '-.')

plt.subplot(224)
plt.plot(x, y4, ':')
```

Работа с функцией subplots()