

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ  
ФЕДЕРАЦИИ**

**Федеральное государственное автономное  
образовательное учреждение высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций  
«Основы работы с пакетом matplotlib»**

**Отчет по лабораторной работе № 3.4  
по дисциплине «Технологии распознавания образов»**

Выполнил студент группы ПИЖ-б-о-21-1

Халимендик Я. Д. « » 2023г.

Подпись студента \_\_\_\_\_

Работа защищена « » \_\_\_\_\_ 20\_\_ г.

Проверил Воронкин Р.А. \_\_\_\_\_  
(подпись)

Ставрополь 2023

Цель работы: исследовать базовые возможности библиотеки matplotlib языка программирования Python

Ход работы:

1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.

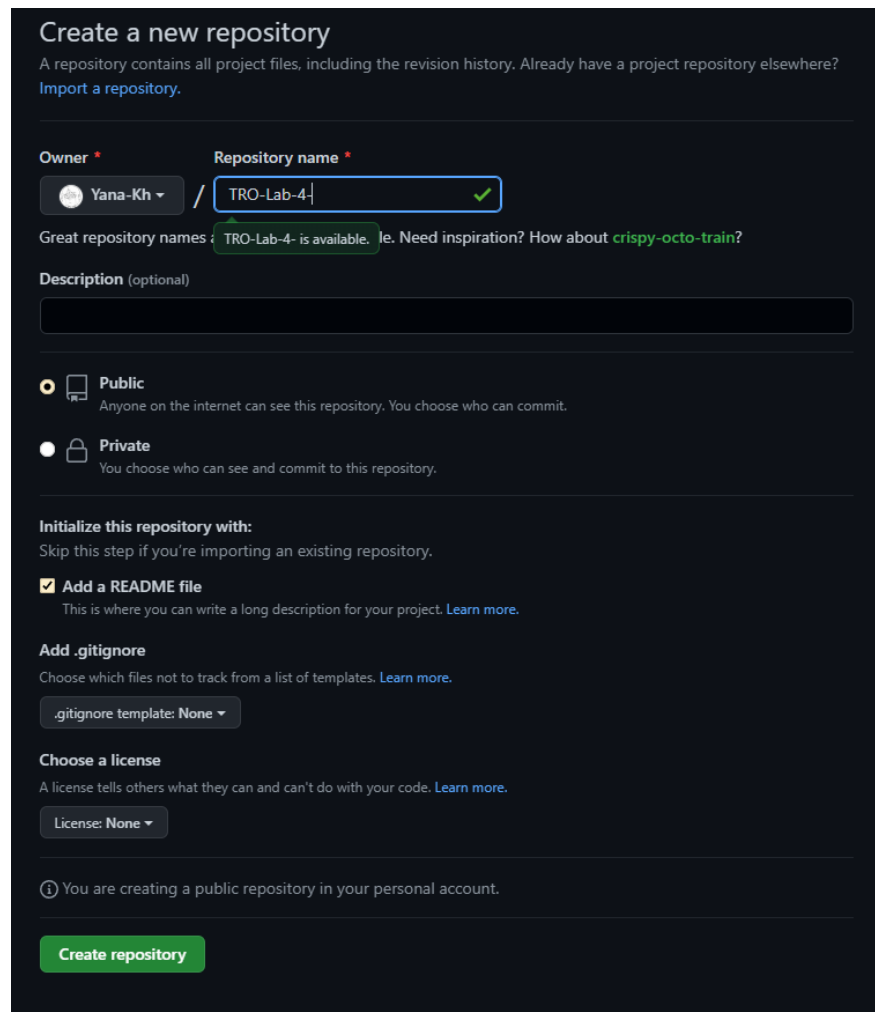


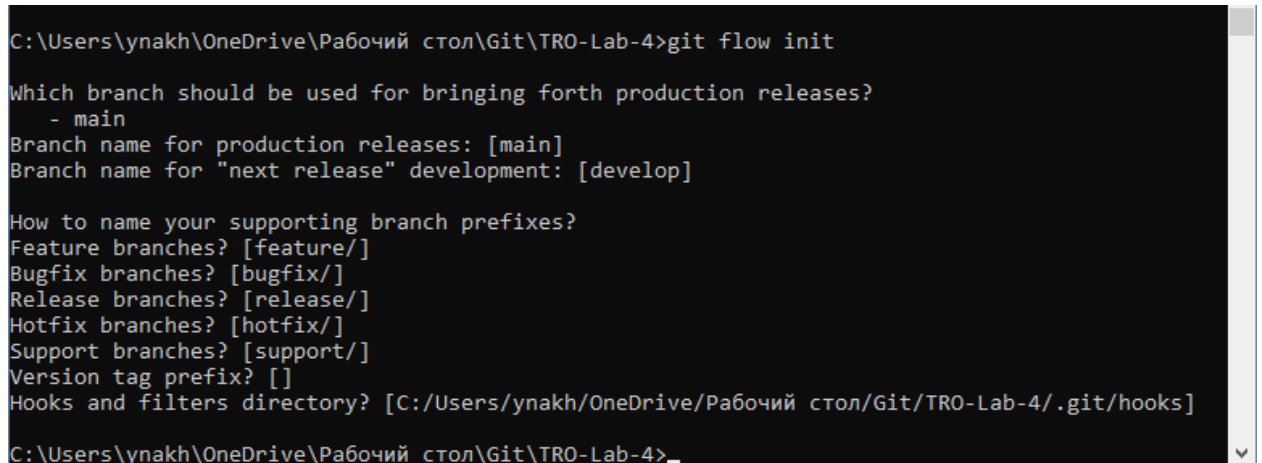
Рисунок 1 – Создание репозитория

3. Выполните клонирование созданного репозитория.

```
C:\Users\ynakh\OneDrive\Рабочий стол\Git>git clone https://github.com/Yana-Kh/TRO-Lab-4.git
Cloning into 'TRO-Lab-4'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

Рисунок 2 – Клонирование репозитория

4. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

A terminal window with a black background and white text. The prompt is 'C:\Users\ynakh\OneDrive\Рабочий стол\Git\TRO-Lab-4>'. The command 'git flow init' has been executed. The output shows a series of prompts and default values for setting up a git-flow repository. The prompts include: 'Which branch should be used for bringing forth production releases?' (default: main), 'Branch name for "next release" development:' (default: develop), 'How to name your supporting branch prefixes?' (with defaults for feature, bugfix, release, hotfix, and support branches), 'Version tag prefix?' (default: []), and 'Hooks and filters directory?' (default: C:/Users/ynakh/OneDrive/Рабочий стол/Git/TRO-Lab-4/.git/hooks). The terminal ends with a prompt 'C:\Users\ynakh\OneDrive\Рабочий стол\Git\TRO-Lab-4>\_'.

```
C:\Users\ynakh\OneDrive\Рабочий стол\Git\TRO-Lab-4>git flow init
Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/ynakh/OneDrive/Рабочий стол/Git/TRO-Lab-4/.git/hooks]

C:\Users\ynakh\OneDrive\Рабочий стол\Git\TRO-Lab-4>_
```

Рисунок 3 – Организация репозитория в соответствии с моделью git-flow

5. Дополните файл .gitignore необходимыми правилами для выбранного языка программирования, интерактивной оболочки Jupyter notebook и интегрированной среды разработки.

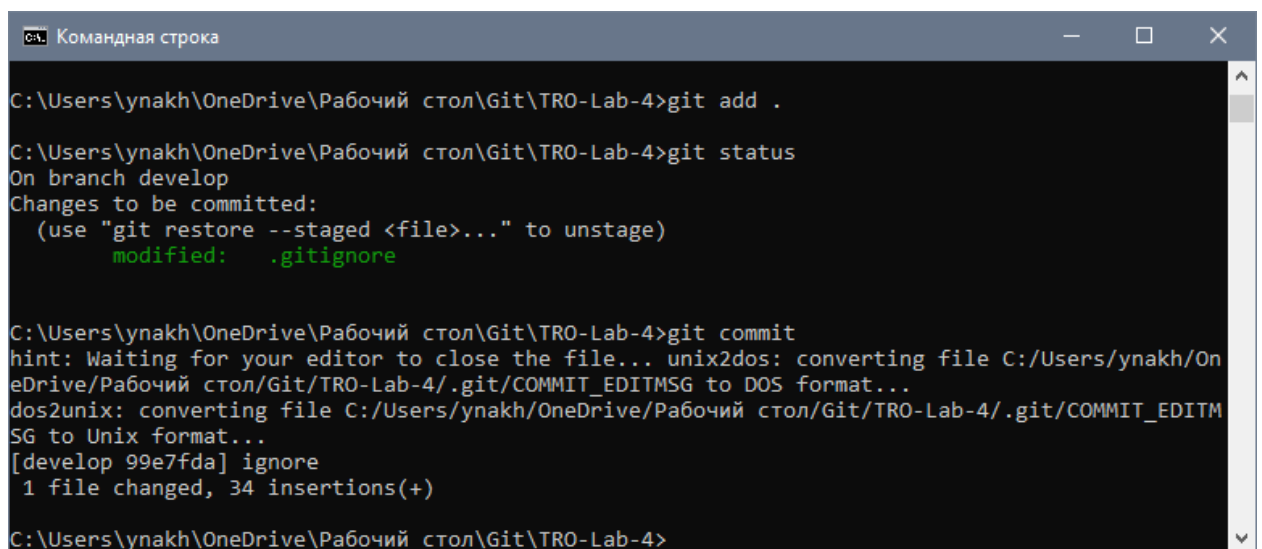
A terminal window titled 'Командная строка' (Command Prompt) with a grey title bar. The prompt is 'C:\Users\ynakh\OneDrive\Рабочий стол\Git\TRO-Lab-4>'. The commands 'git add .' and 'git status' have been executed. The output of 'git status' shows 'On branch develop' and 'Changes to be committed: (use "git restore --staged <file>..." to unstage) modified: .gitignore'. The command 'git commit' has been executed, and the output shows a hint about converting file formats, followed by '[develop 99e7fda] ignore' and '1 file changed, 34 insertions(+)'.

Рисунок 4 – Дополнение файла .gitignore

## 6. Проработать примеры лабораторной работы.

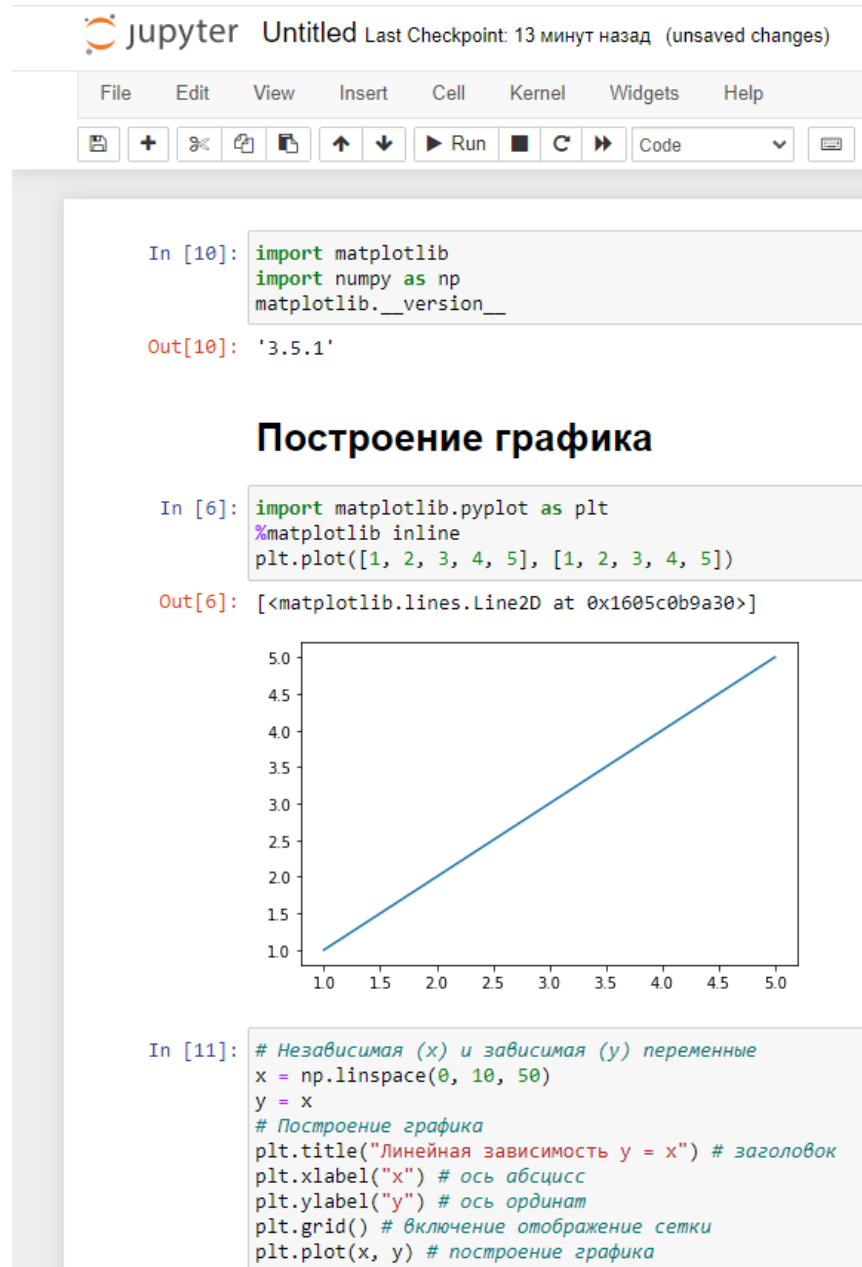
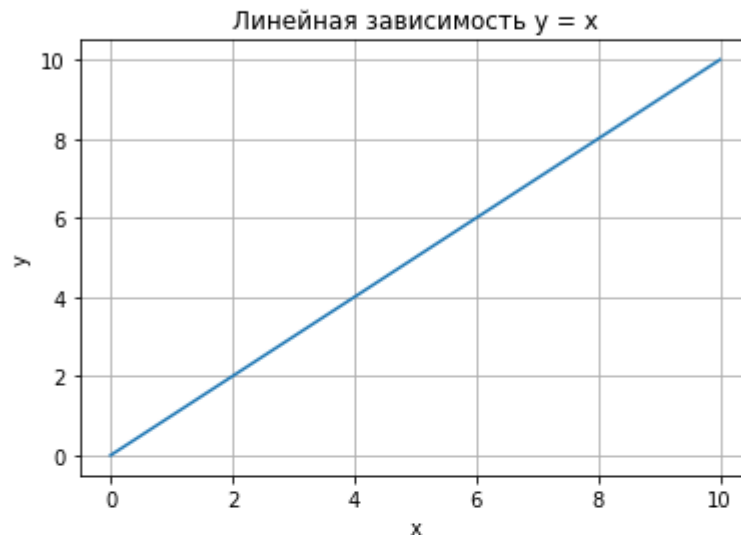


Рисунок 5 – Проработка примеров лабораторной работы

```
In [11]: # 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[11]: [matplotlib.lines.Line2D at 0x1605c215d00>]



```
In [17]: #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, "r--") # построение графика
```

Out[17]: [matplotlib.lines.Line2D at 0x1605cd7f880>]

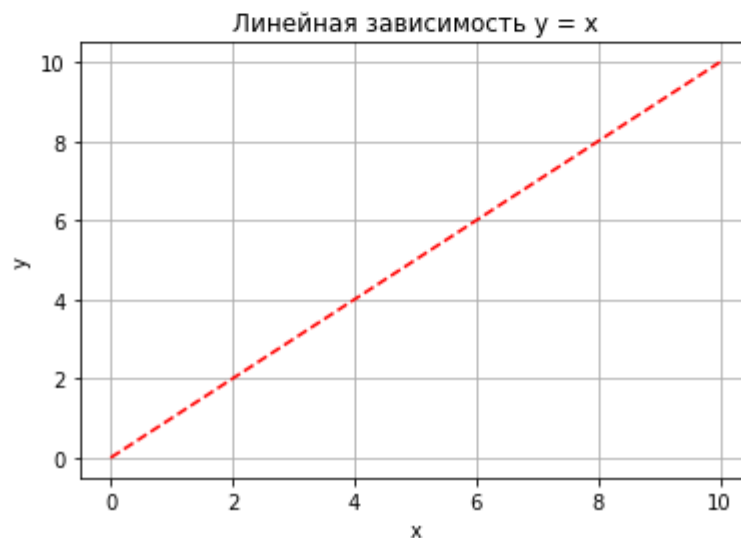


Рисунок 6 – Проработка примеров лабораторной работы

## Несколько графиков на одном поле

```
In [16]: # Линейная зависимость
x = np.linspace(0, 10, 50)
y1 = x
# Квадратичная зависимость
y2 = [i**2 for i in x]
# Построение графика
plt.title("Зависимости:  $y_1 = x$ ,  $y_2 = x^2$ ") # 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[16]: [<matplotlib.lines.Line2D at 0x1605cd12820>,
<matplotlib.lines.Line2D at 0x1605cd12880>]
```

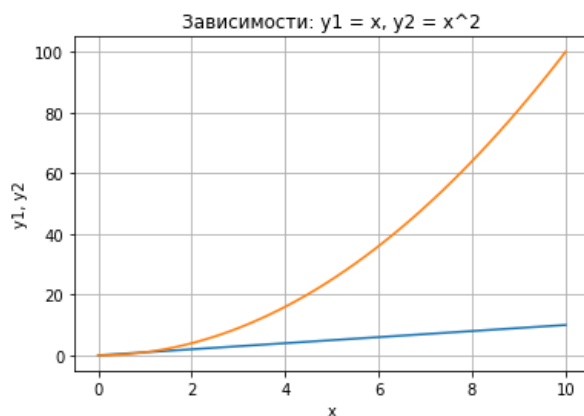


Рисунок 7 – Проработка примеров лабораторной работы

## Несколько разделенных полей с графиками

```
In [18]: # 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:  $y_1 = x$ ,  $y_2 = x^2$ ") # 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) # включение отображение сетки
```

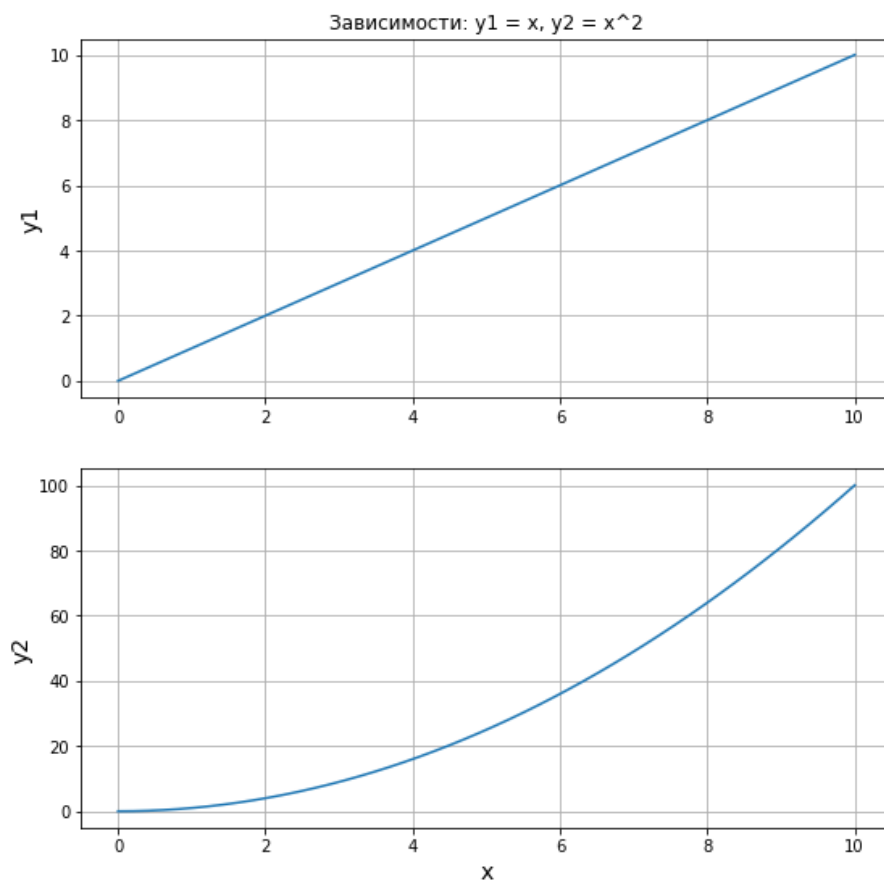


Рисунок 8 – Проработка примеров лабораторной работы

## Построение диаграммы для категориальных данных

```
In [19]: 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[19]: Text(0, 0.5, 'Count')

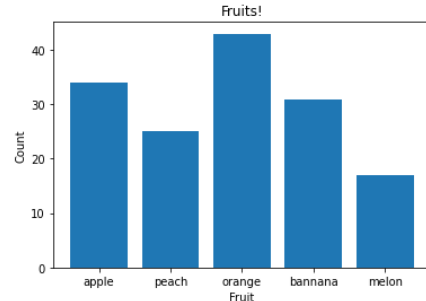


Рисунок 9 – Проработка примеров лабораторной работы

## Основные элементы графика

```
In [20]: 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()
```

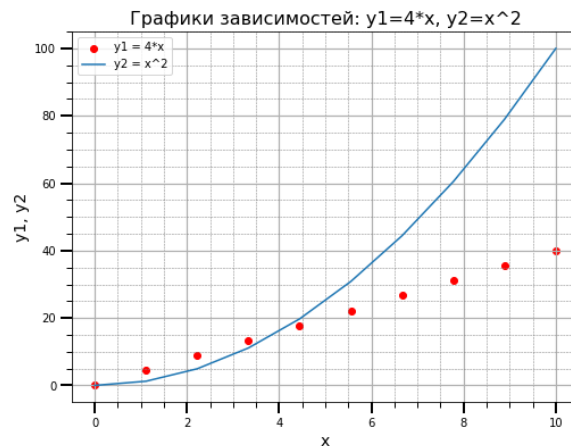


Рисунок 10 – Проработка примеров лабораторной работы

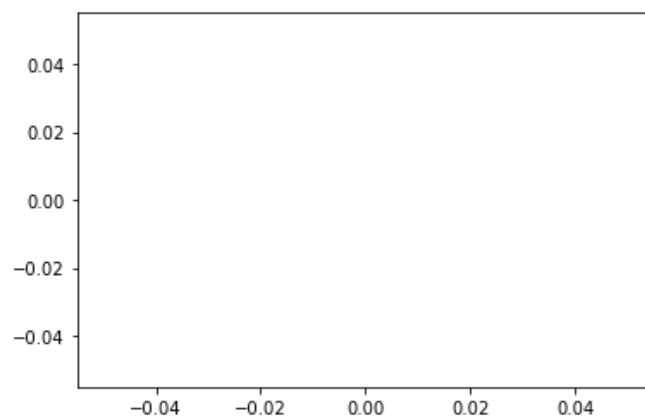


# Работа с инструментом pyplot

Построение графиков

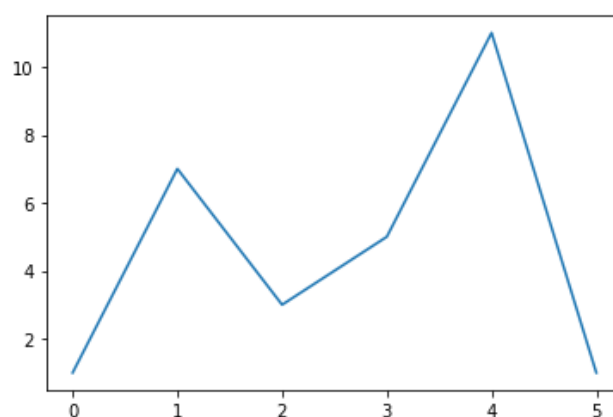
```
In [21]: plt.plot()
```

```
Out[21]: []
```



```
In [22]: plt.plot([1, 7, 3, 5, 11, 1])
```

```
Out[22]: [matplotlib.lines.Line2D at 0x1605e18bc70]
```



```
In [23]: plt.plot([1, 5, 10, 15, 20], [1, 7, 3, 5, 11])
```

```
Out[23]: [matplotlib.lines.Line2D at 0x1605e1ee940]
```

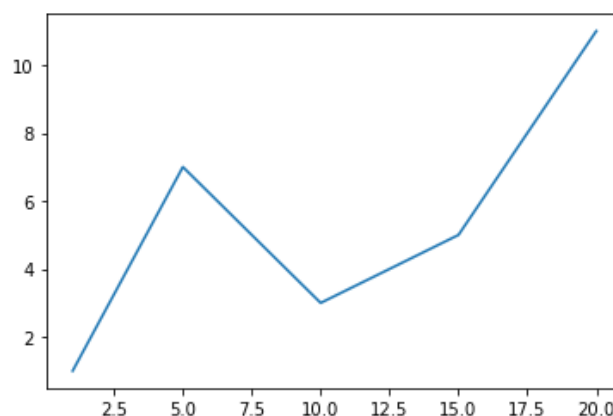


Рисунок 11 – Проработка примеров лабораторной работы

## Текстовые надписи на графике

```
In [24]: plt.xlabel('Day', fontsize=15, color='blue')
plt.title('Chart price', fontsize=17) #Заголовок графика
plt.text(1, 1, 'type: Steel') #Текстовое примечание

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() #легенда
plt.grid(True)

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

Out[24]: Text(15, 4, 'grow up!')

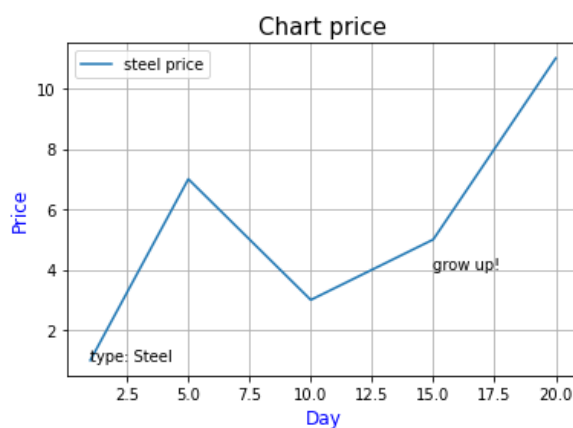


Рисунок 12 – Проработка примеров лабораторной работы

## Работа с линейным графиком

```
In [26]: plt.plot(x, y, color='red')

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

line = plt.plot(x, y)
plt.setp(line, linestyle='--')
```

Out[26]: [None]

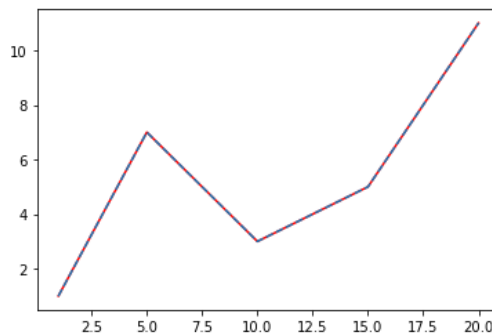
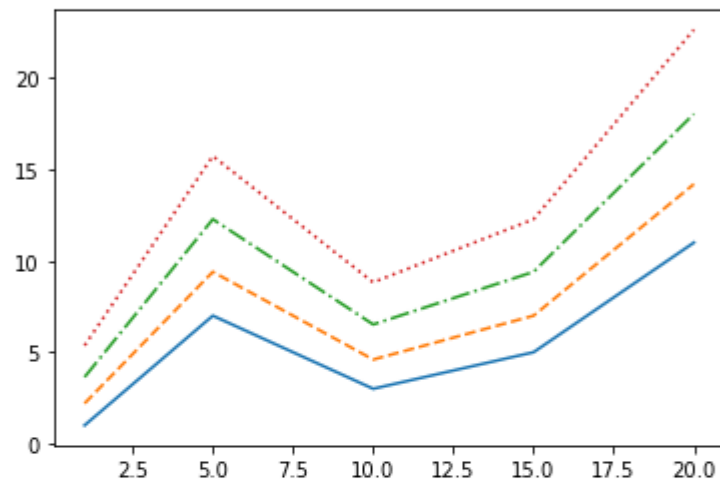


Рисунок 13 – Проработка примеров лабораторной работы

```
In [27]: 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[27]: [<matplotlib.lines.Line2D at 0x1605e2491c0>,
<matplotlib.lines.Line2D at 0x1605e249340>,
<matplotlib.lines.Line2D at 0x1605e249310>,
<matplotlib.lines.Line2D at 0x1605e249550>]
```



```
In [28]: plt.plot(x, y1, '-')
plt.plot(x, y2, '--')
plt.plot(x, y3, '-.')
plt.plot(x, y4, ':')
```

```
Out[28]: [<matplotlib.lines.Line2D at 0x1605d0a8400>]
```

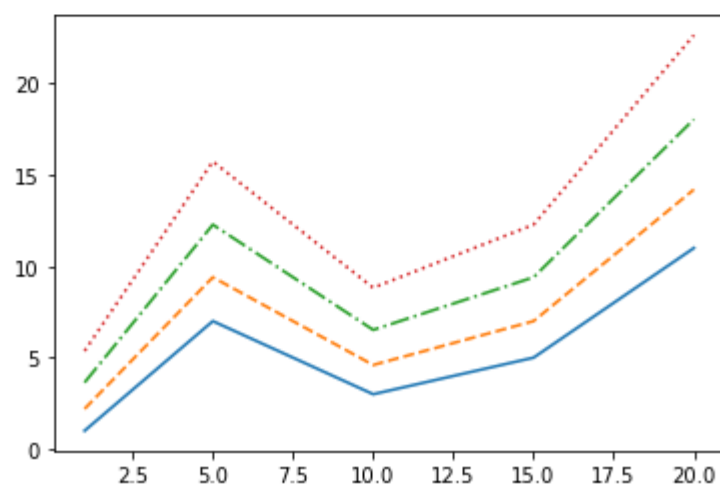


Рисунок 14 – Проработка примеров лабораторной работы

Цвет линии

```
In [29]: x = [1, 5, 10, 15, 20]  
y = [1, 7, 3, 5, 11]  
plt.plot(x, y, '--r')
```

```
Out[29]: [<matplotlib.lines.Line2D at 0x1605d109790>]
```

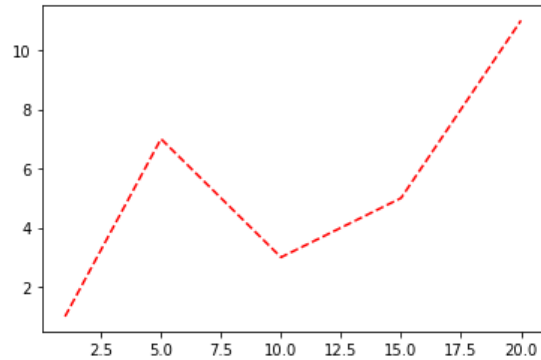
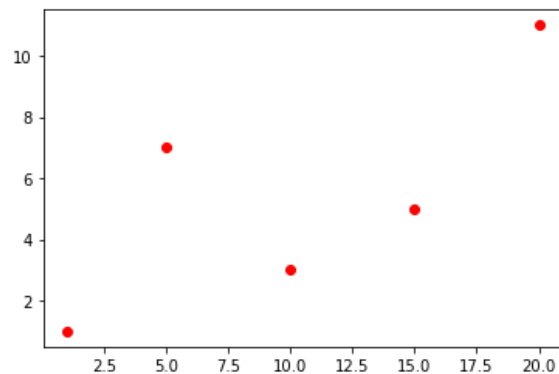


Рисунок 15 – Проработка примеров лабораторной работы

Тип графика

```
In [30]: plt.plot(x, y, 'ro')
```

```
Out[30]: [<matplotlib.lines.Line2D at 0x1605d16bca0>]
```



```
In [31]: plt.plot(x, y, 'bx')
```

```
Out[31]: [<matplotlib.lines.Line2D at 0x1605e29fe20>]
```

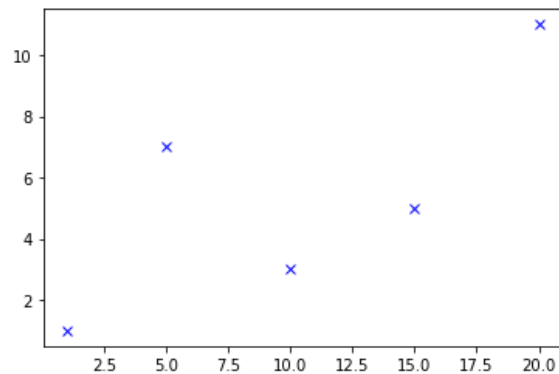


Рисунок 16 – Проработка примеров лабораторной работы

## Размещение графиков на разных полях

Работа с функцией subplot()

```
In [32]: # 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[32]: [

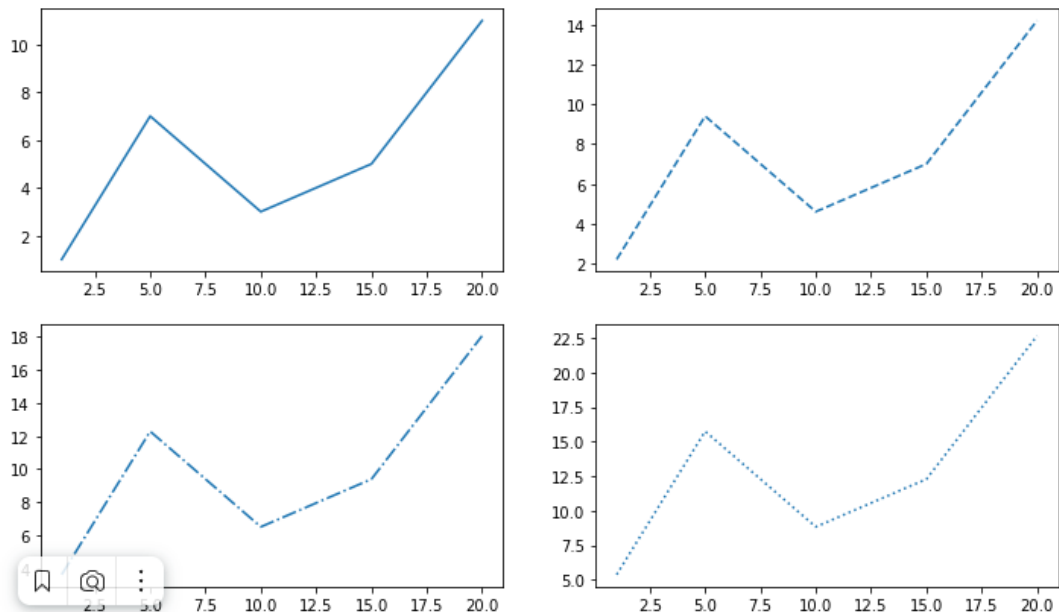


Рисунок 17 – Проработка примеров лабораторной работы

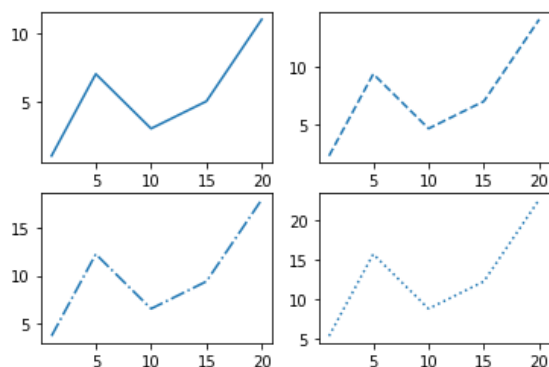
```
In [33]: # Вывод графиков
plt.subplot(221)
plt.plot(x, y1, '-')

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

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

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

Out[33]: [matplotlib.lines.Line2D at 0x1605e524460]



```
In [34]: fig, axes = plt.subplots(2, 2, figsize=(12, 7))
axes[0, 0].plot(x, y1, '-')
axes[0, 1].plot(x, y2, '--')
axes[1, 0].plot(x, y3, '-.')
axes[1, 1].plot(x, y4, ':')
```

Out[34]: [matplotlib.lines.Line2D at 0x1605e7c33d0]

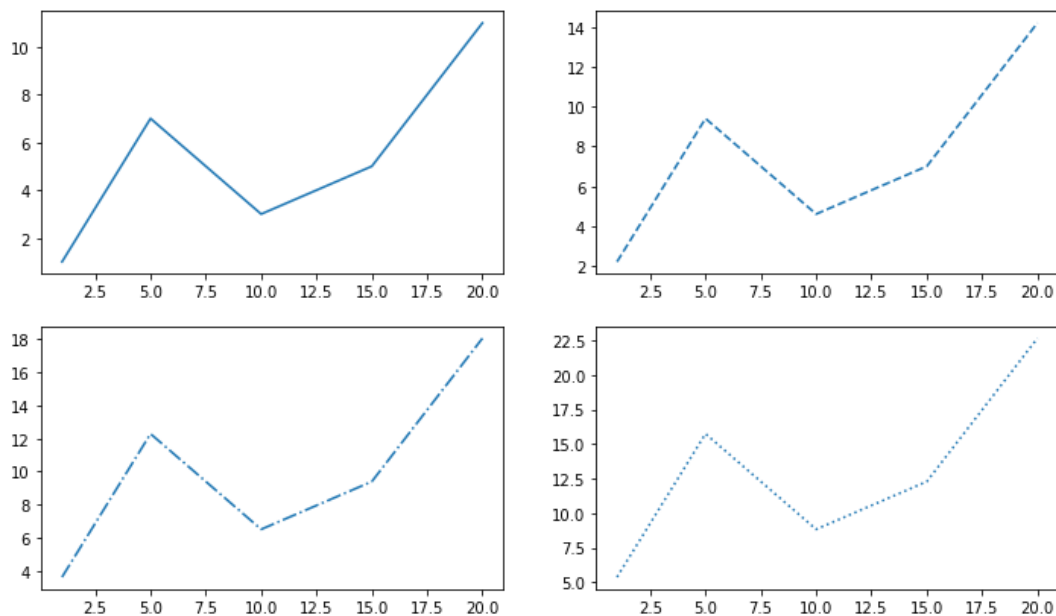


Рисунок 18 – Проработка примеров лабораторной работы

7. Зафиксируйте сделанные изменения в репозитории.



 .gitignore	ignore
 LICENSE	Initial commit
 README.md	Initial commit
 ex.ipynb	ex

Рисунок 19 – Фиксирование изменений в репозитории

## Вопросы для защиты работы

### 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.

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

```
%matplotlib inline
```

### 3. Как отобразить график с помощью функции plot?

Для построения графика используется команда plot(). Если в качестве параметра функции plot() передать список, то значения из этого списка будут отложены по оси ординат (ось y), а по оси абсцисс (ось x) будут отложены индексы элементов массива.

Для того, чтобы задать значения по осям x и y необходимо в plot() передать два списка.

```
plt.plot([1, 2, 3, 4, 5], [1, 2, 3, 4, 5])
```

### 4. Как отобразить несколько графиков на одном поле?

Для того, чтобы вывести несколько графиков на одном поле необходимо передать соответствующие наборы значений в функцию plot().

```
plt.plot(x, y1, x, y2)
```

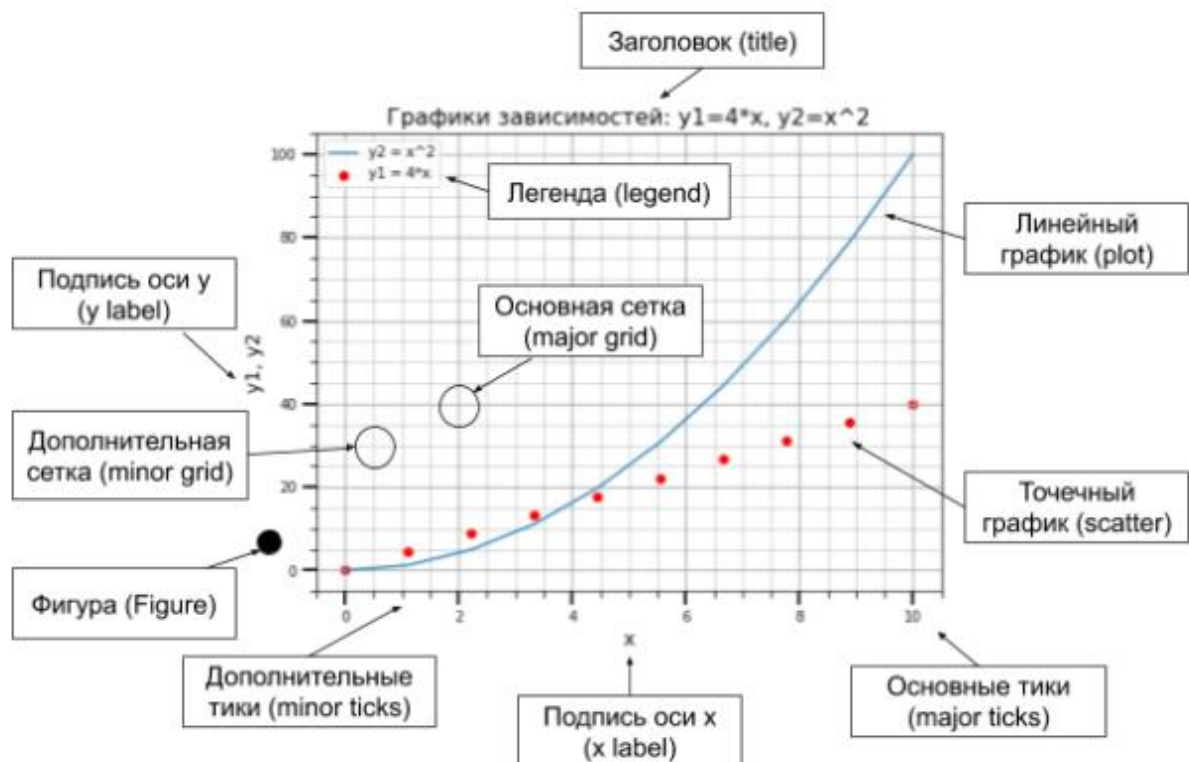
### 5. Какой метод Вам известен для построения диаграмм категориальных данных?

Метод bar()



6. Какие основные элементы графика Вам известны?

- Заголовок (title)
- Легенда (legend)
- Основная сетка (major grid)
- Линейный график (plot)
- Точечный график (scatter)
- Дополнительные тики (minor ticks) Фигура (figure)
- Дополнительная сетка (minor grid) Подпись оси y (y label)
- Основные тики (major ticks) Подпись оси x (x label)



7. Как осуществляется управление текстовыми надписями на графике?

Наименование осей: `plt.xlabel()`, `plt.ylabel()`

Заголовок графика: `plt.title()`

Текстовое примечание: `plt.text()`

Легенда: `plt.legend()`

## 8. Как осуществляется управление легендой графика?

Легенда будет размещена на графике, если вызвать функцию `legend()`

## 9. Как задать цвет и стиль линий графика?

Задание цвета: `plt.plot(x, y, color='red')`, `plt.setp( color='red', linewidth=1)`

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'`

Задание стиля линии: `plt.plot(x, y, '--')`

## 10. Как выполнить размещение графика в 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.

Самый простой способ представить графики в отдельных полях – это использовать функцию `subplot()` для задания их мест размещения.

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.