

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»

Выполнила:

Кувшин Ирина Анатольевна
2 курс, группа ПИЖ-б-о-21-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

(подпись)

Проверил:

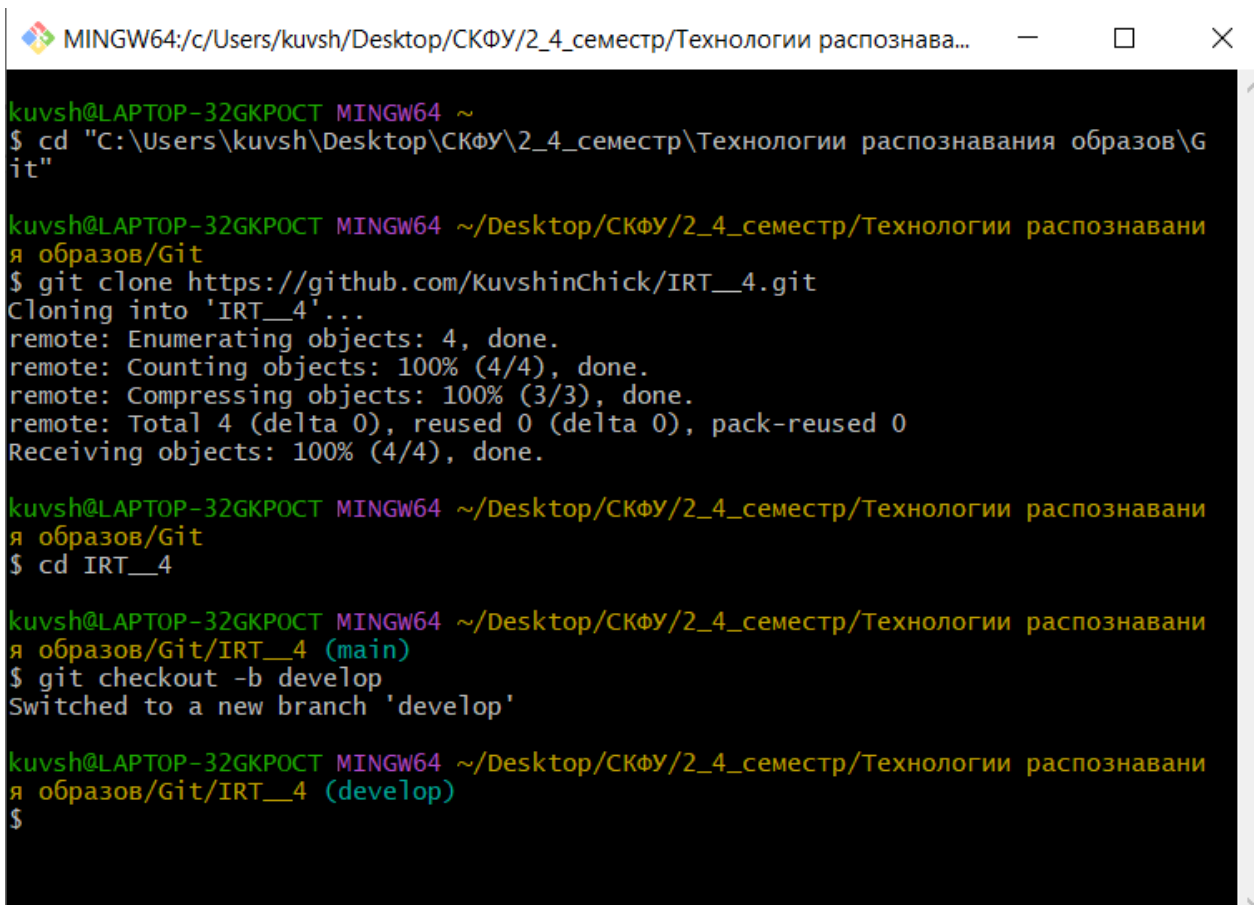
(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Цель работы: исследовать базовые возможности библиотеки matplotlib языка программирования Python.

1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и выбранный Вами язык программирования (выбор языка программирования будет доступен после установки флажка Add .gitignore).
3. Выполните клонирование созданного репозитория на рабочий компьютер.
4. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.



```
MINGW64:/c:/Users/kuvsh/Desktop/СКФУ/2_4_семестр/Технологии распознава...
kuvsh@LAPTOP-32GKPOCT MINGW64 ~
$ cd "C:\Users\kuvsh\Desktop\СКФУ\2_4_семестр\Технологии распознавания образов\Git"

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распознавания образов/Git
$ git clone https://github.com/KuvshinChick/IRT__4.git
Cloning into 'IRT__4'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.

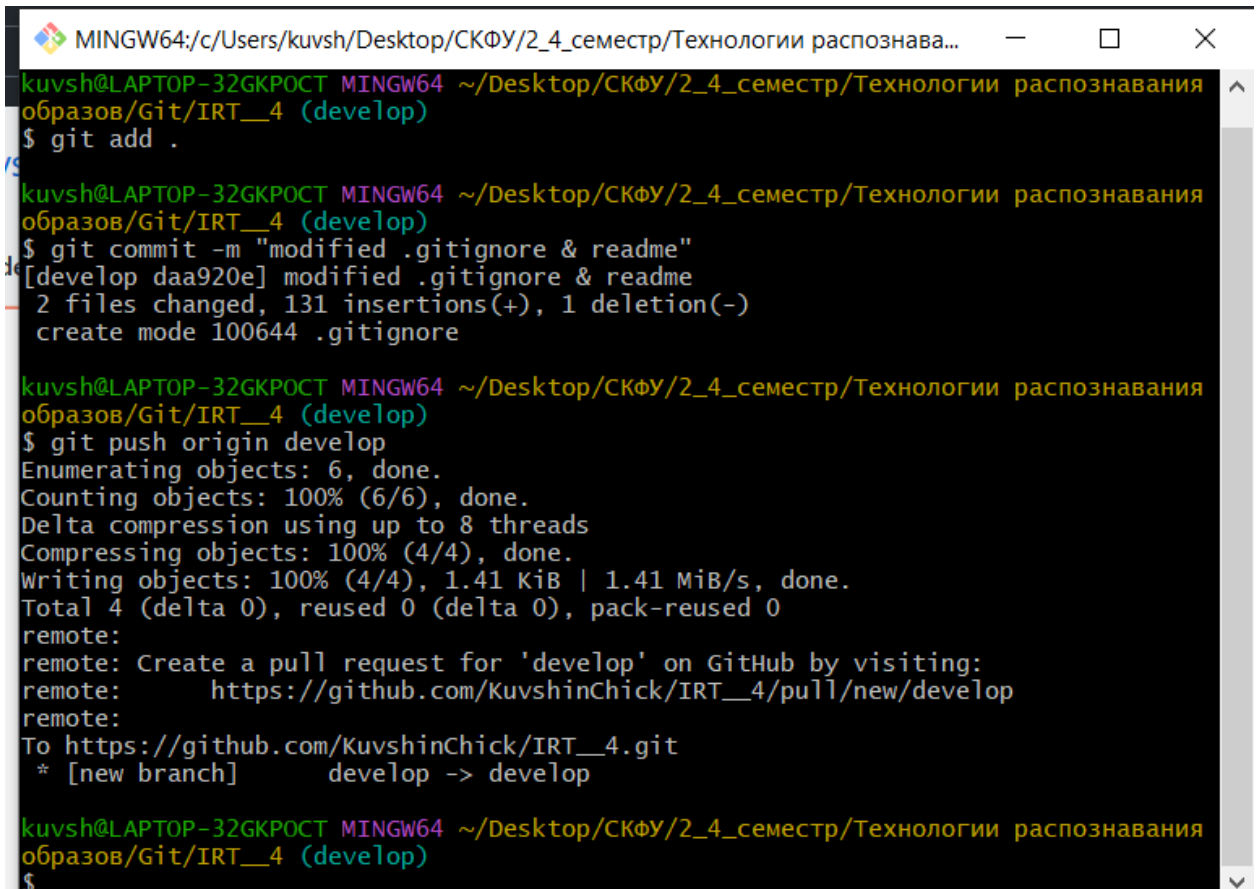
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распознавания образов/Git/IRT__4
$ cd IRT__4

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распознавания образов/Git/IRT__4 (main)
$ git checkout -b develop
Switched to a new branch 'develop'

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распознавания образов/Git/IRT__4 (develop)
$
```

Рисунок 4.1 – Клонирование репозитория и создание ветки develop

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



```
MINGW64:/c/Users/kuvsh/Desktop/СКФУ/2_4_семестр/Технологии распознава...
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распознавания
образов/Git/IRT__4 (develop)
$ git add .

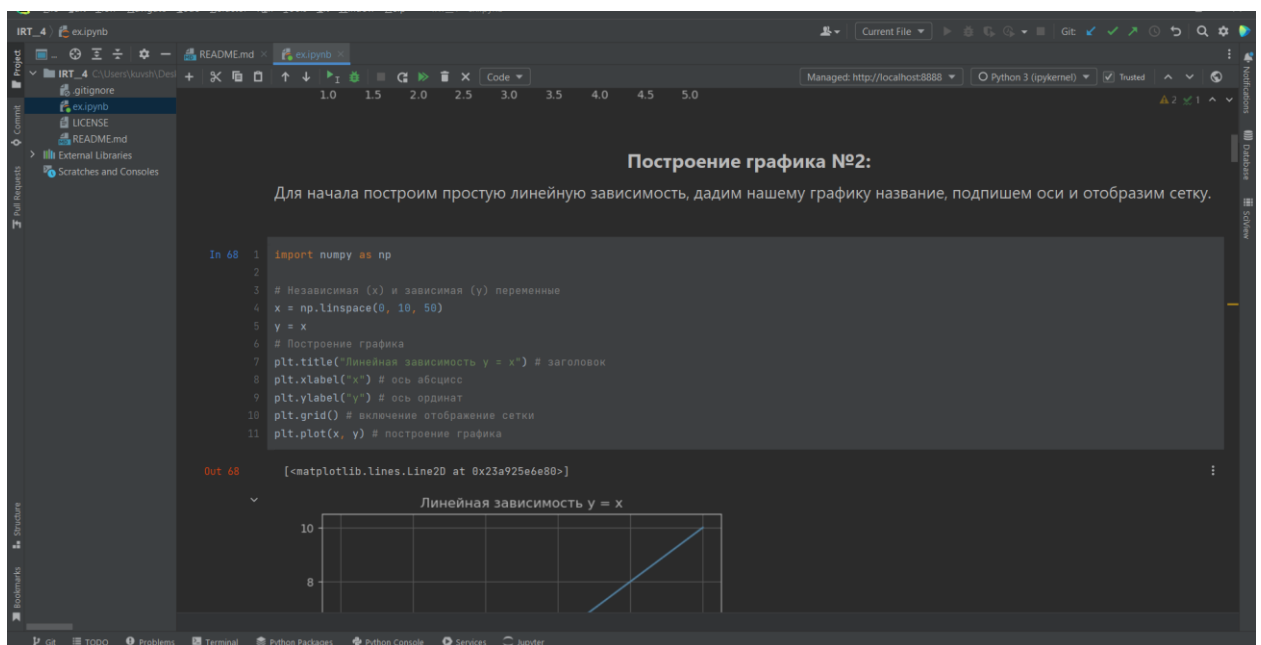
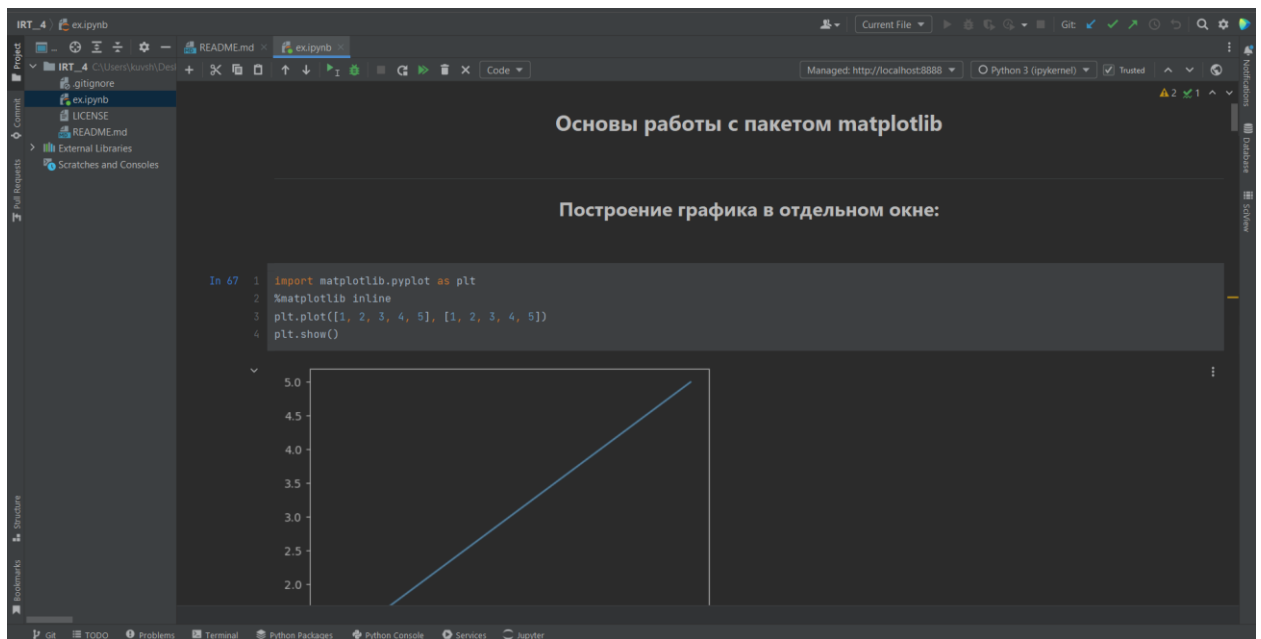
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распознавания
образов/Git/IRT__4 (develop)
$ git commit -m "modified .gitignore & readme"
[develop daa920e] modified .gitignore & readme
2 files changed, 131 insertions(+), 1 deletion(-)
create mode 100644 .gitignore

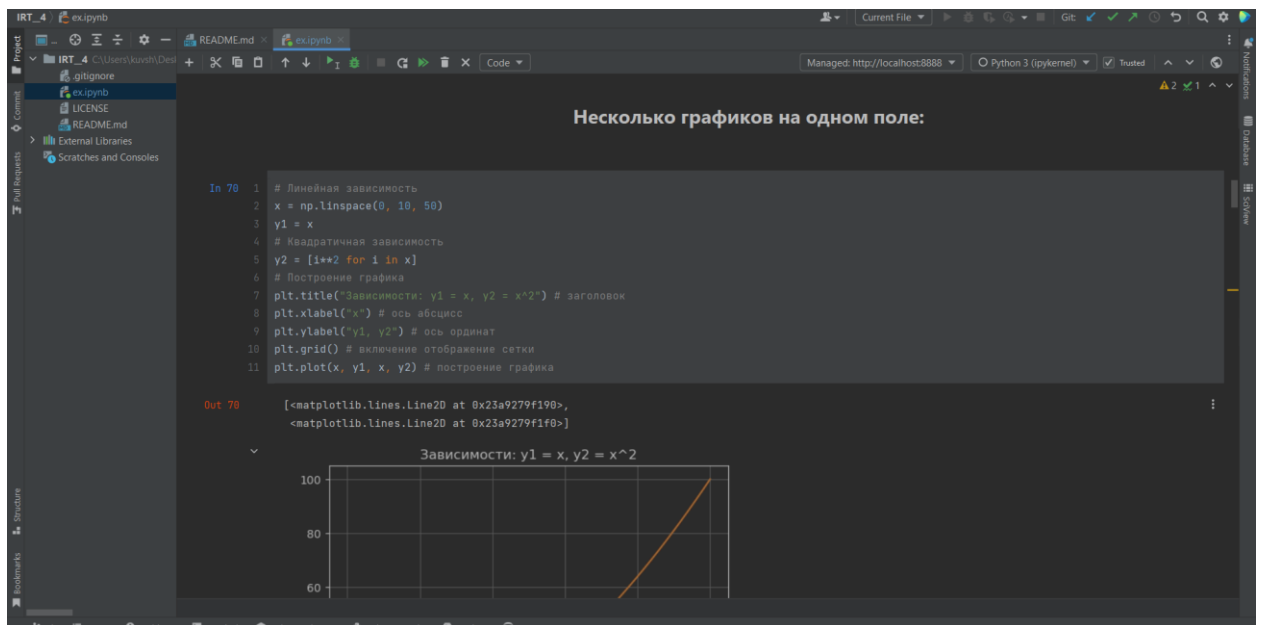
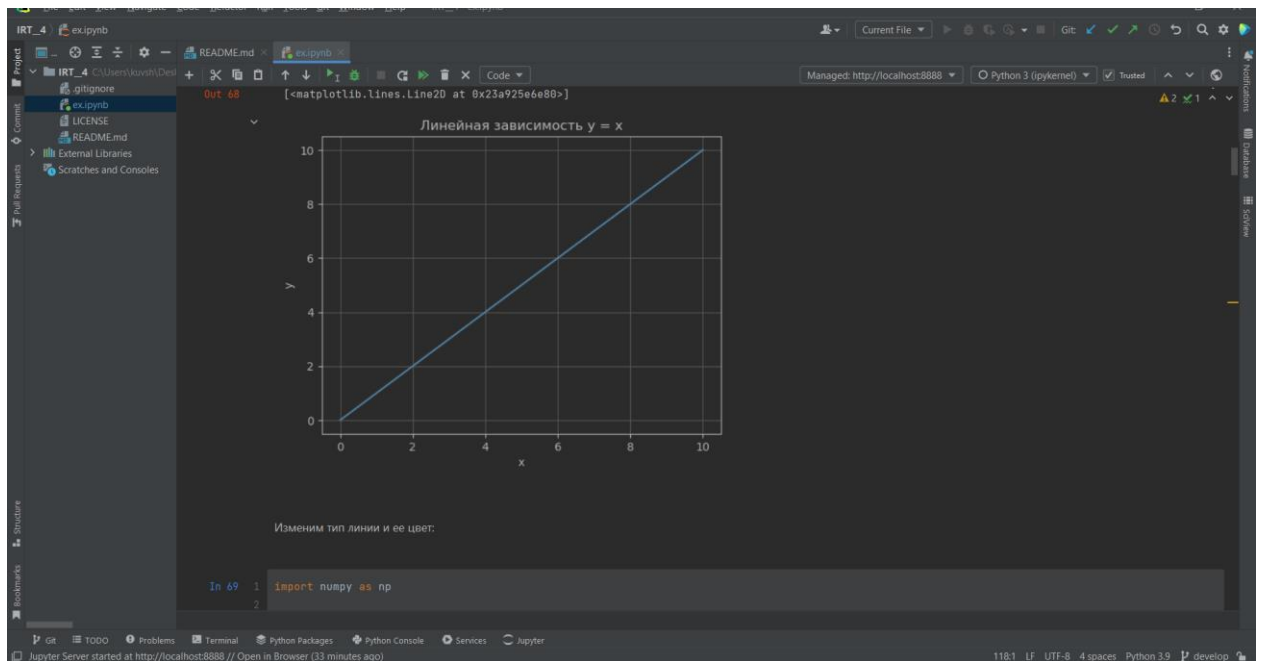
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распознавания
образов/Git/IRT__4 (develop)
$ git push origin develop
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 1.41 KiB | 1.41 MiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:      https://github.com/KuvshinChick/IRT__4/pull/new/develop
remote:
To https://github.com/KuvshinChick/IRT__4.git
 * [new branch]      develop -> develop

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распознавания
образов/Git/IRT__4 (develop)
$
```

Рисунок 4.2 – Обновление .gitignore и readme

6. Проработать примеры лабораторной работы.





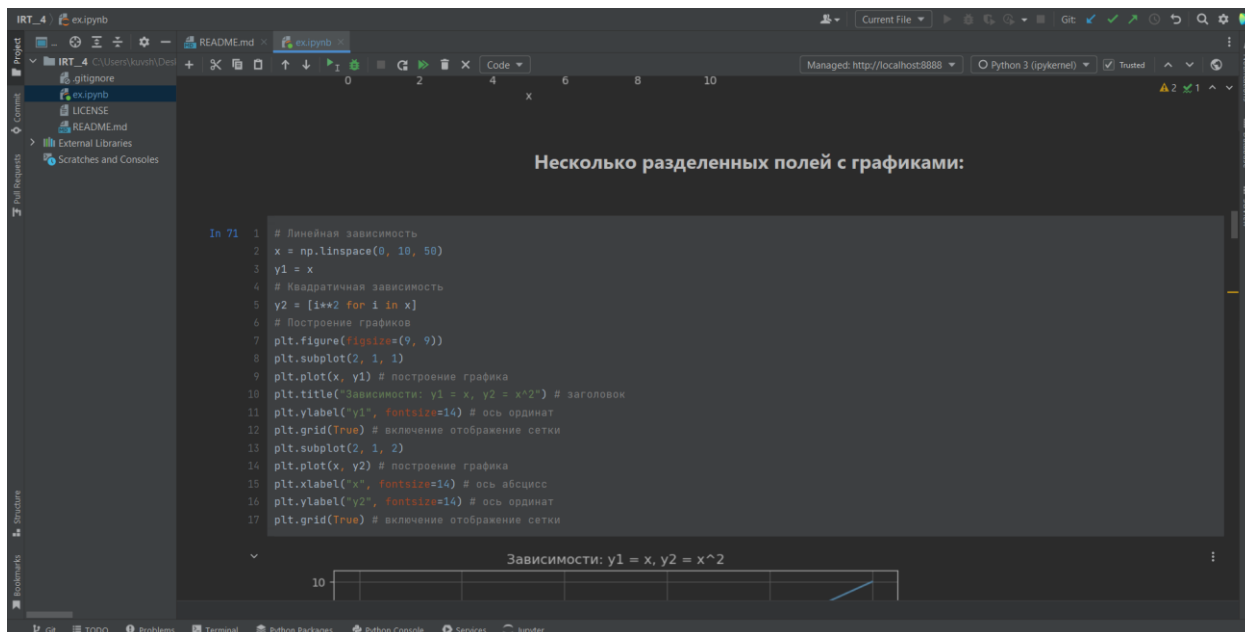


Рисунок 4.3 – Результат проработки примеров

7. Зафиксируйте сделанные изменения в репозитории.
8. Выполните слияние ветки для разработки с веткой main (master).
9. Отправьте сделанные изменения на сервер GitHub.

Рисунок 4.3 – Результат проработки примеров

9. Зафиксируйте сделанные изменения в репозитории.
10. Выполните слияние ветки для разработки с веткой main (master).
11. Отправьте сделанные изменения на сервер GitHub.

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?

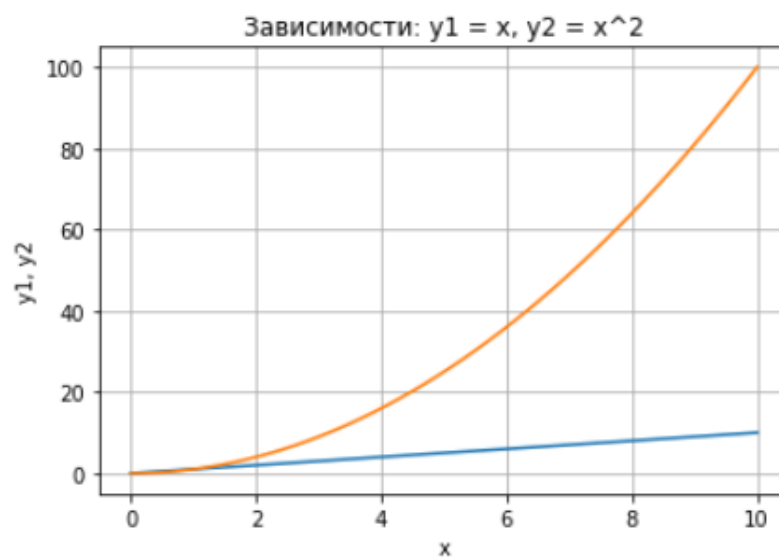
Построим несколько графиков на одном поле, для этого добавим квадратичную зависимость:

```
# 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

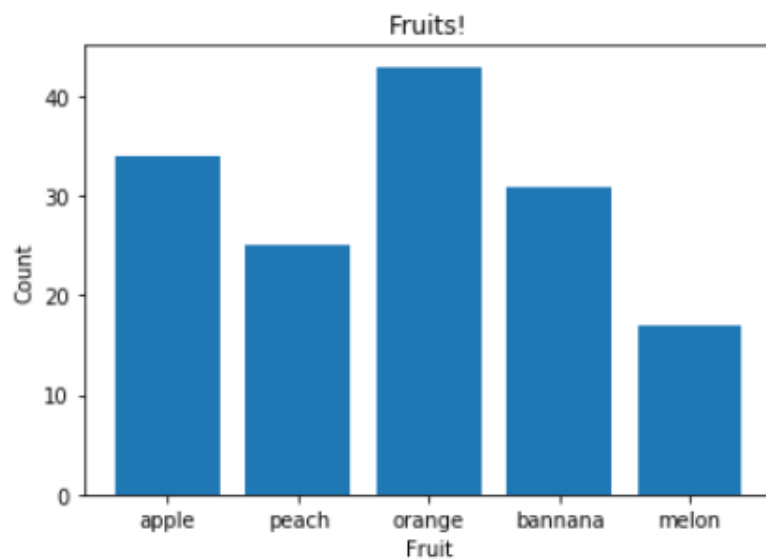
plt.plot(x, y1, x, y2) # построение графика
```



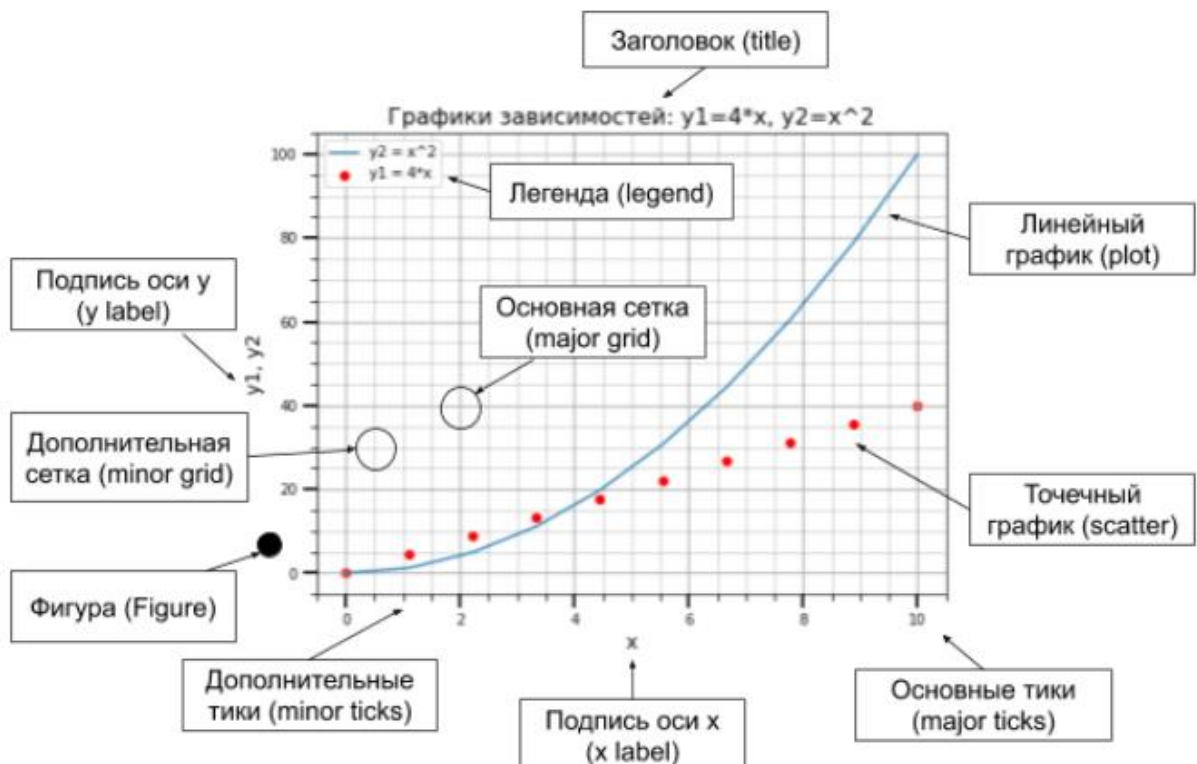
5. Какой метод Вам известен для построения диаграмм категориальных данных?


```
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:

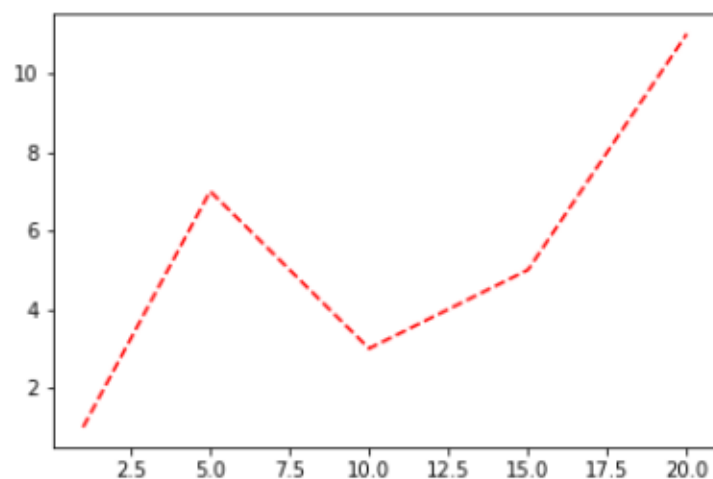
- *RGB* или *RGBA* кортеж значений с плавающей точкой в диапазоне [0, 1] (пример: (0.1, 0.2, 0.3))
- *RGB* или *RGBA* значение в *hex* формате (пример: '#0a0a0a')
- строковое представление числа с плавающей точкой в диапазоне [0, 1] (определяет цвет в шкале серого) (пример: '0.7')
- символ из набора {'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 *subplot()* 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]

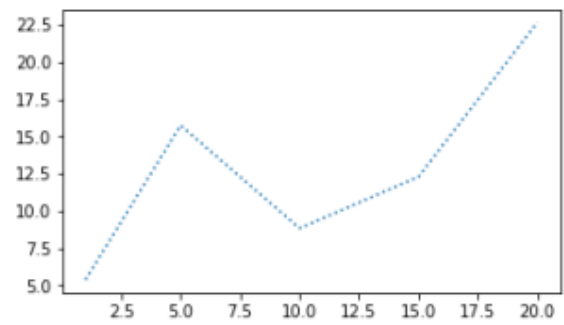
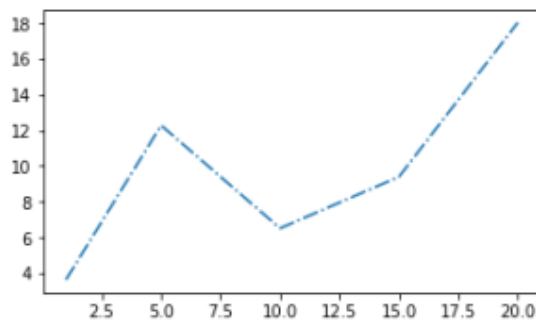
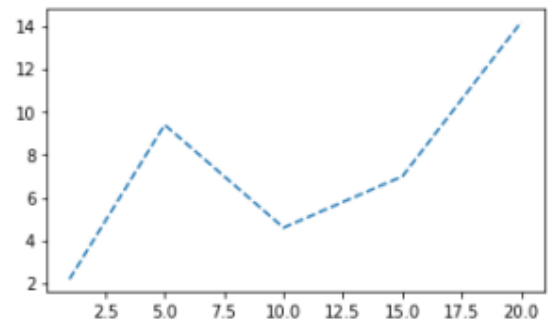
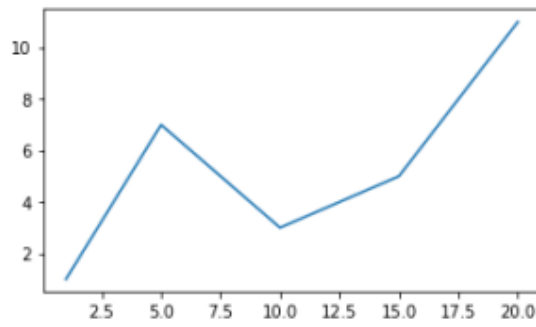
# Настройка размеров подложки
plt.figure(figsize=(12, 7))

# 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, ':')
```

Результат функции `subplot()`