5МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе № 3.5
Визуализация данных с помощью matplotlib

Выполнил студент группы ИВТ-б-о-21-1

Пентухов С. А. « »_____20__г.

Подпись студента_____

Работа защищена « »_____20__г.

Проверил Воронкин Р.А. _____

(подпись)

Ставрополь, 2023

Цель работы: Исследовать базовые возможности визуализации данных на плоскости средствами библиотеки matplotlib языка программирования Python

Ссылка https://github.com/Pentuhov/Lab3.5

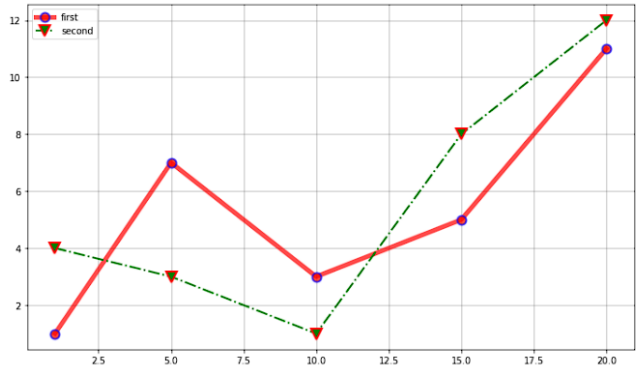
Ход работы

1. Проработать примеры из лабораторной

Реализуем возможности plot() на примере:

In [2]:
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
```

In [3]:
```
x = [1, 5, 10, 15, 20]
y1 = [1, 7, 3, 5, 11]
y2 = [4, 3, 1, 8, 12]

plt.figure(figsize=(12, 7))
plt.plot(x, y1, 'o-r', alpha=0.7, label="first", lw=5, mec='b', mew=2, ms=10)
plt.plot(x, y2, 'v-.g', label="second", mec='r', lw=2, mew=2, ms=12)

plt.legend()
plt.grid(True)
```

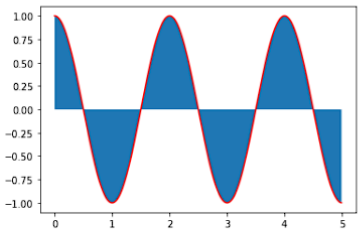


In [4]:
```
x = np.arange(0.0, 5, 0.01)
y = np.cos(x*np.pi)
```

Рис. 1 Пример линейных графиков

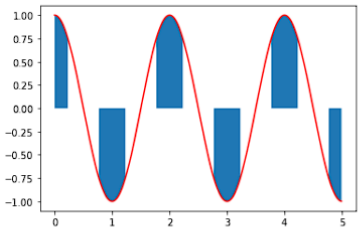
```
plt.fill_between(x, y)
```

```
Out[5]: <matplotlib.collections.PolyCollection at 0x25675f4c5e0>
```




```
In [6]: plt.plot(x, y, c="r")
        plt.fill_between(x, y, where=(y > 0.75) | (y < -0.75))
```

```
Out[6]: <matplotlib.collections.PolyCollection at 0x25675fa8100>
```




```
In [7]: plt.plot(x, y, c="r")
        plt.fill_between(x, y, where=(y > 0))
```

```
Out[7]: <matplotlib.collections.PolyCollection at 0x25675fdb610>
```

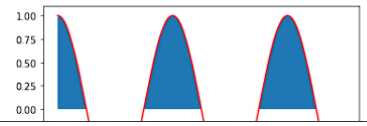
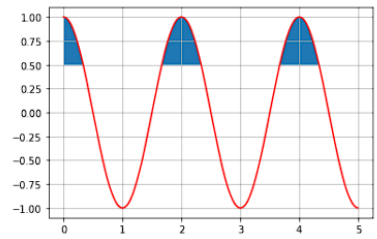


Рис. 2 Пример линейных графиков

```
In [22]: plt.plot(x, y, c="r")
         plt.grid()

         plt.fill_between(x, 0.5, y, where=(y>=0.5))
```

```
Out[22]: <matplotlib.collections.PolyCollection at 0x1b5dd217e20>
```




```
In [23]: plt.plot(x, y, c="r")
         plt.grid()

         plt.fill_between(x, y, 1)
```

```
Out[23]: <matplotlib.collections.PolyCollection at 0x1b5dd09aca0>
```

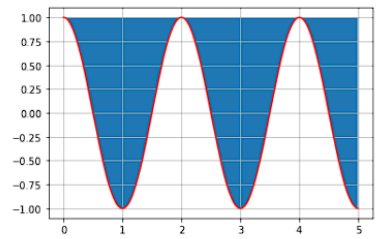
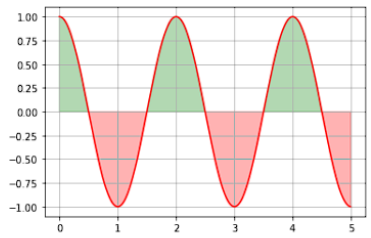


Рис. 3 Пример линейных графиков

```
plt.grid()

plt.fill_between(x, y, where=y>=0, color="g", alpha=0.3)
plt.fill_between(x, y, where=y<=0, color="r", alpha=0.3)
```

Out[24]: <matplotlib.collections.PolyCollection at 0x1b5dd15f670>



Настройка маркировки графиков

In [25]:
```
x = [1, 2, 3, 4, 5, 6, 7]
y = [7, 6, 5, 4, 5, 6, 7]
plt.plot(x, y, marker="o", c="g")
```

Out[25]: [<matplotlib.lines.Line2D at 0x1b5dd2814f0>]

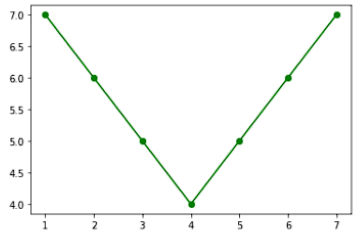
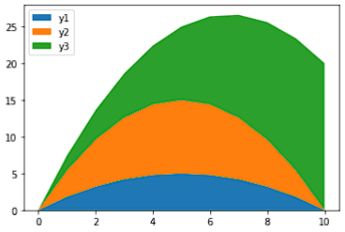


Рис. 4 Пример линейных графиков

Стековый график

In [17]:
```
x = np.arange(0, 11, 1)

y1 = np.array([(-0.2)*i**2+2*i for i in x])
y2 = np.array([(-0.4)*i**2+4*i for i in x])
y3 = np.array([2*i for i in x])

labels = ["y1", "y2", "y3"]

fig, ax = plt.subplots()

ax.stackplot(x, y1, y2, y3, labels=labels)
ax.legend(loc='upper left')
```

Out[17]: <matplotlib.legend.Legend at 0x170399d68e0>



In [18]:
```
x = np.arange(0, 10.5, 0.5)
y = np.array([(-0.2)*i**2+2*i for i in x])

plt.stem(x, y)
```

Out[18]: <StemContainer object of 3 artists>

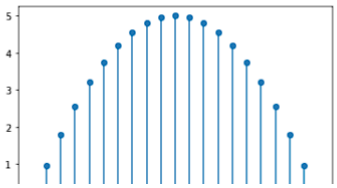
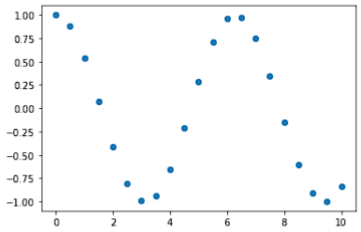


Рис. 5 Пример стековых графиков

Точечный график

In [19]:
```
x = np.arange(0, 10.5, 0.5)
y = np.cos(x)
plt.scatter(x, y)
```

Out[19]: <matplotlib.collections.PathCollection at 0x1703ab74df0>



In [34]:
```
x = np.arange(0, 10.5, 0.5)
y = np.cos(x)
plt.scatter(x, y, s=80, c="r", marker="D", linewidths=2, edgecolors="g")
```

Out[34]: <matplotlib.collections.PathCollection at 0x1b5de4e6bb0>

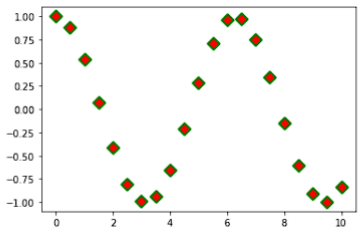
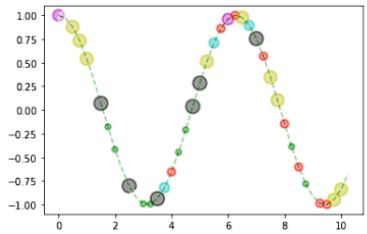


Рис. 6 Пример точечных графиков

In [36]:
```
bc = mcolors.BASE_COLORS

x = np.arange(0, 10.5, 0.25)
y = np.cos(x)

num_set = np.random.randint(1, len(mcolors.BASE_COLORS), len(x))
sizes = num_set * 35
colors = [list(bc.keys())[i] for i in num_set]

plt.scatter(x, y, s=sizes, alpha=0.4, c=colors, linewidths=2, edgecolors="face")

plt.plot(x, y, "g--", alpha=0.4)
```

Out[36]: [<matplotlib.lines.Line2D at 0x1b5de55f4c0>]



In [37]:
```
np.random.seed(123)

groups = [f"P{i}" for i in range(7)]
counts = np.random.randint(3, 10, len(groups))

plt.bar(groups, counts)
```

Out[37]: <BarContainer object of 7 artists>

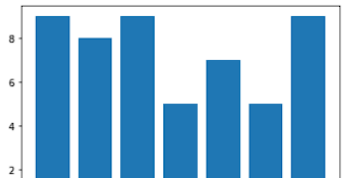
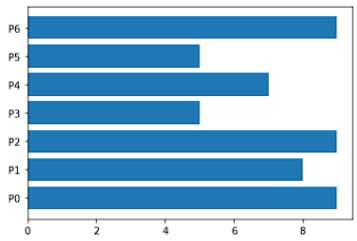


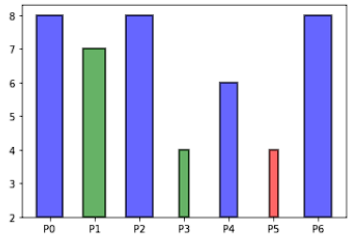
Рис. 7 Пример точечных графиков

Out[38]: <BarContainer object of 7 artists>



In [39]:
```
bc = mcolors.BASE_COLORS

np.random.seed(123)

groups = [f"P{i}" for i in range(7)]
counts = np.random.randint(0, len(bc), len(groups))

width = counts*0.1

colors = [["r", "b", "g"][int(np.random.randint(0, 3, 1))] for _ in counts]
plt.bar(groups, counts, width=width, alpha=0.6, bottom=2, color=colors,
edgecolor="k", linewidth=2)
```

Out[39]: <BarContainer object of 7 artists>



In [40]:
```
cat_par = [f"P{i}" for i in range(5)]

g1 = [10, 21, 34, 12, 27]
```

Рис. 8 Пример диаграмм

In [40]:
```
cat_par = [f"P{i}" for i in range(5)]

g1 = [10, 21, 34, 12, 27]
g2 = [17, 15, 25, 21, 26]

width = 0.3

x = np.arange(len(cat_par))

fig, ax = plt.subplots()
rects1 = ax.bar(x - width/2, g1, width, label='g1')
rects2 = ax.bar(x + width/2, g2, width, label='g2')

ax.set_title('Пример групповой диаграммы')
ax.set_xticks(x)
ax.set_xticklabels(cat_par)

ax.legend()
```

Out[40]: <matplotlib.legend.Legend at 0x1b5de736130>

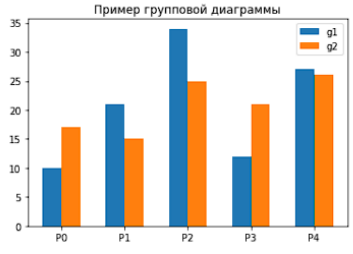


Диаграмма с errorbar элементом

Рис. 9 Пример диаграмм

Диаграмма с errorbar элементом

In [41]:
```
np.random.seed(123)

rnd = np.random.randint

cat_par = [f"P{i}" for i in range(5)]
g1 = [10, 21, 34, 12, 27]

error = np.array([[rnd(2,7),rnd(2,7)] for _ in range(len(cat_par))]).T
fig, axs = plt.subplots(1, 2, figsize=(10, 5))

axs[0].bar(cat_par, g1, yerr=5, ecolor="r", alpha=0.5, edgecolor="b",
linewidth=2)
axs[1].bar(cat_par, g1, yerr=error, ecolor="r", alpha=0.5, edgecolor="b",
linewidth=2)
```

Out[41]: <BarContainer object of 5 artists>

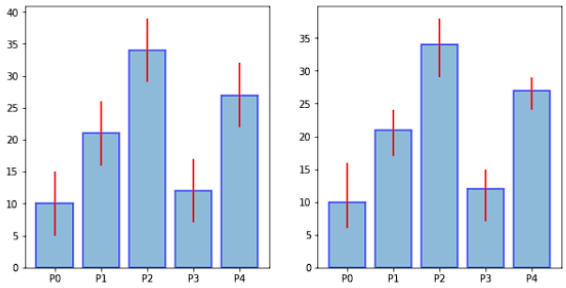


Рис. 10 Пример диаграмм

Круговые диаграммы

In [42]:
```
vals = [24, 17, 53, 21, 35]

labels = ["Ford", "Toyota", "BMV", "AUDI", "Jaguar"]

fig, ax = plt.subplots()
ax.pie(vals, labels=labels)
ax.axis("equal")
```

Out[42]:
```
(-1.1163226287452406,
 1.1007772680354877,
 -1.1107362350259515,
 1.1074836529113834)
```

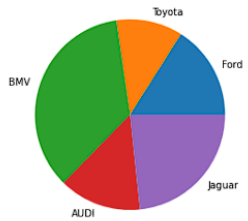


Рис. 11 Пример круговых диаграмм

In [43]:
```
vals = [24, 17, 53, 21, 35]

labels = ["Ford", "Toyota", "BMV", "AUDI", "Jaguar"]
explode = (0.1, 0, 0.15, 0, 0)

fig, ax = plt.subplots()

ax.pie(vals, labels=labels, autopct='%1.1f%%', shadow=True, explode=explode,
wedgeprops={'lw':1, 'ls':'--','edgecolor':"k"}, rotatelabels=True)

ax.axis("equal")
```

Out[43]:
```
(-1.2704955621219602,
 1.1999223938155328,
 -1.1121847055183558,
 1.1379015332518725)
```

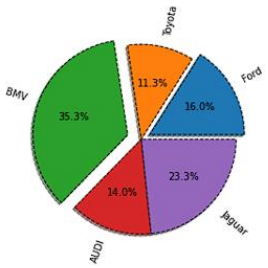


Рис. 12 Пример круговых диаграмм

In [44]:
```
fig, ax = plt.subplots()

offset=0.4

data = np.array([[5, 10, 7], [8, 15, 5], [11, 9, 7]])
cmap = plt.get_cmap("tab20b")

b_colors = cmap(np.array([0, 8, 12]))
sm_colors = cmap(np.array([1, 2, 3, 9, 10, 11, 13, 14, 15]))

ax.pie(data.sum(axis=1), radius=1, colors=b_colors,
wedgeprops=dict(width=offset, edgecolor='w'))

ax.pie(data.flatten(), radius=1-offset, colors=sm_colors,
wedgeprops=dict(width=offset, edgecolor='w'))
```

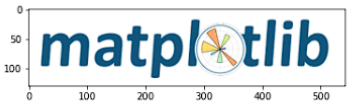
Out[44]:
```
([<matplotlib.patches.Wedge at 0x1b5de9421c0>,
  <matplotlib.patches.Wedge at 0x1b5de942730>,
  <matplotlib.patches.Wedge at 0x1b5de942c10>,
  <matplotlib.patches.Wedge at 0x1b5de94f160>,
  <matplotlib.patches.Wedge at 0x1b5de94f640>,
  <matplotlib.patches.Wedge at 0x1b5de94fb20>,
  <matplotlib.patches.Wedge at 0x1b5de94ffd0>,
  <matplotlib.patches.Wedge at 0x1b5de95b520>,
  <matplotlib.patches.Wedge at 0x1b5de95ba00>],
 [Text(0.646314344414094, 0.13370777166859046, ''),
  Text(0.4521935266177387, 0.48075047008298655, ''),
  Text(0.040366679721656945, 0.6587643973138266, ''),
  Text(-0.34542288787409087, 0.5623904591409097, ''),
  Text(-0.6578039053946477, 0.05379611554331286, ''),
  Text(-0.48987451889717687, -0.44229283934431896, ''),
  Text(-0.12049606360635531, -0.6489073112975174, ''),
  Text(0.39011356818311405, -0.532363976917521, ''),
  Text(0.6332653697075483, -0.1859434632601054, '')])
```

Рис. 13 Пример круговых диаграмм

Цветовые карты, отображение изображений

In [46]:
```
from PIL import Image
import requests

from io import BytesIO

response = requests.get('https://matplotlib.org/_static/logo2.png')
img = Image.open(BytesIO(response.content))

plt.imshow(img)
```

Out[46]:
```
<matplotlib.image.AxesImage at 0x1b5debc0ee0>
```



In [47]:
```
np.random.seed(19680801)

data = np.random.randn(25, 25)
plt.imshow(data)
```

Out[47]:
```
<matplotlib.image.AxesImage at 0x1b5dec2bf40>
```

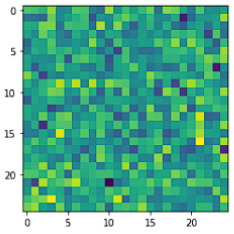
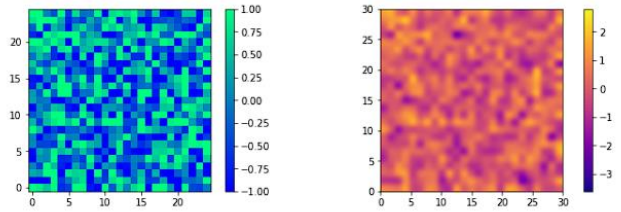


Рис. 14 Пример цветовых карт

```
In [48]: fig, axs = plt.subplots(1, 2, figsize=(10,3), constrained_layout=True)

         p1 = axs[0].imshow(data, cmap='winter', aspect='equal', vmin=-1, vmax=1,
         origin="lower")
         fig.colorbar(p1, ax=axs[0])

         p2 = axs[1].imshow(data, cmap='plasma', aspect='equal',
         interpolation='gaussian', origin="lower", extent=(0, 30, 0, 30))
         fig.colorbar(p2, ax=axs[1])

Out[48]: <matplotlib.colorbar.Colorbar at 0x1b5dec5fe80>
```



```
In [49]: np.random.seed(123)

         data = np.random.rand(5, 7)
         plt.pcolormesh(data, cmap='plasma', edgecolors="k", shading='flat')

Out[49]: <matplotlib.collections.QuadMesh at 0x1b5dfde0bb0>
```

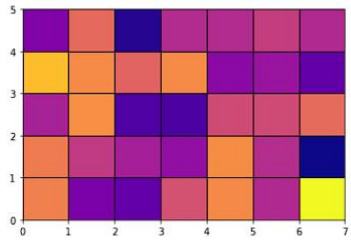


Рис. 15 Пример цветовых карт

2. Создать ноутбук, в котором выполнить решение вычислительной задачи (например, задачи из области физики, экономики, математики, статистики и т. д.) требующей построения линейного графика

ЗАДАЧА: С противоположных концов города длиной 360 км едут навстречу друг другу грузовик и автомобиль.Автомобиль начинает движение на 2 часа позже грузовика. Через сколько часов они встретятся, если скорость грузовика – 48 км/час, а автомобиля – 72 км/час?

РЕШЕНИЕ: Пусть x ч.– время движения второго мальчика, тогда время первого – (x+2) ч. Составим уравнение, учитывая, что сумма расстояний равна 240 км:72(x + 2) + 48x = 240 решив которое, получим, что x= 2.9ч. Тогда время, через которое они встретятся – 4.9 ч.

```
[14]: import matplotlib.pyplot as plt
      plt.plot([240, 200, 160, 120, 100, 80, 60, 40, 20, 10])
      plt.plot([0, 5, 10, 15, 18, 20, 25])
      plt.ylabel('x, м', fontsize=12, color='blue')
      plt.xlabel('t, c', fontsize=12, color='blue')
      plt.text(4.8, 270, '')
      plt.text(4.8, 35, '')
      plt.grid()
      plt.show()
```

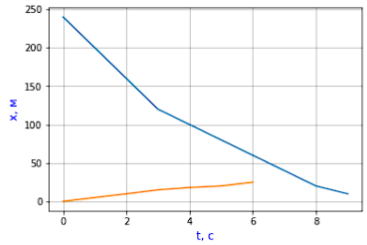


Рис. 16 Решение заданий 1

3. Создать ноутбук, в котором выполнить решение вычислительной задачи (например, задачи из области физики, экономики,

математики, статистики и т. д.) требующей построения столбчатой диаграммы

Список высочайших гор

Высота гор: Мауна-Кеа – 10203 м.

Джомолунгма – 8868 м.

Эльбрус – 5642 м.

Базардюзю - 4466 м

```
In [10]: import matplotlib.pyplot as plt
         %matplotlib inline
         fruits = ["Мауна-Кеа","Джомолунгма", "Эльбрус", "Базардюзю"]
         counts = [10203,8849,5642, 4466]
         plt.bar(fruits, counts)
         plt.title("Высочайшие горы")
         plt.xlabel("Название гор")
         plt.ylabel("Высота
Out[10]: Text(0, 0.5, 'Длина')
```

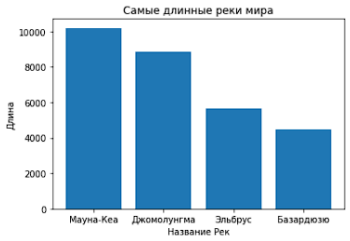


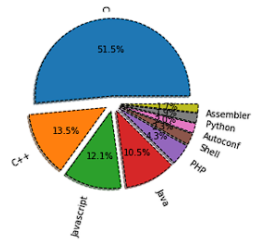
Рис. 17 Решение заданий 2

4. Создать ноутбук, в котором выполнить решение вычислительной задачи (например, задачи из области физики, экономики, математики, статистики и т. д.) требующей построения круговой диаграммы

```
In [8]: import numpy as np
        import matplotlib
        import matplotlib.pyplot as plt
        import matplotlib.colors as mcolors
```

Рейтинг языков программирования на WIIIDev

```
In [6]: vals = [46, 12.1, 10.78, 9.39, 3.88, 2.08, 1.82, 1.74, 1.56]

        labels = ["C", "C++", "Javascript", "Java", "PHP", "Shell", "Autoconf", "Python", "Assembler"]
        explode = (0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1)

        fig, ax = plt.subplots()

        ax.pie(vals, labels=labels, autopct='%1.1f%%', shadow=True, explode=explode,
        wedgeprops={'lw':1, 'ls':'--','edgecolor':"k"}, rotatelabels=True)
        ax.axis("equal")
Out[6]: (-1.1916308825110455,
         1.2089677480655996,
         -1.2154325588218327,
         1.2156064978201757)
```




```
In [ ]:
```

Рис. 18 Решение заданий 3

5. Найти какое-либо изображение в сети Интернет. Создать

ноутбук, в котором будет отображено выбранное изображение средствами библиотеки matplotlib по URL из сети Интернет.

Рис. 19 Решение домашнего задания 4

Вывод: В результате выполнения работы были исследованы возможности визуализации данных на плоскости средствами библиотеки matplotlib языка программирования Python

.

Контрольные вопросы:

1. Как выполнить построение линейного графика с помощью matplotlib? - Для построения линейного графика используется функция plot(), со следующей сигнатурой:

plot([x], y, [fmt], *, data=None, **kwargs)

plot([x], y, [fmt], [x2], y2, [fmt2], …, **kwargs)

Если вызвать функцию plot() с одним аргументом – вот так: plot(y), то мы получим график, у которого по оси ординат (ось y) будут отложены значения из переданного списка, по по оси абсцисс (ось x) – индексы элементов массива

2. Как выполнить заливку области между графиком и осью? Между двумя графиками? - Для заливки областей используется функция

fill_between(). Сигнатура функции:

fill_between(x, y1, y2=0, where=None, interpolate=False, step=None, *, data=None, **kwargs)

3. Как выполнить выборочную заливку, которая удовлетворяет некоторому условию? - Используя параметры y1 и y2 можно формировать более сложные решения.

4. Как выполнить двухцветную заливку? –

```
plt.plot(x, y, c="r")
plt.grid()
plt.fill_between(x, y, where=y>=0, color="g", alpha=0.3)
plt.fill_between(x, y, where=y<=0, color="r", alpha=0.3)
```

5. Как выполнить маркировку графиков? - В этой случае нужно задать интервал отображения маркеров, для этого используется параметр markevery, который может принимать одно из следующих значений: None – отображаться будет каждая точка; N – отображаться будет каждая N-я точка; (start, N) – отображается каждая N-я точка начиная с точки start; slice(start, end, N) – отображается каждая N-я точка в интервале от start до end; [i, j, m, n] – будут отображены только точки i, j, m, n

6. Как выполнить обрезку графиков? - Для того, чтобы отобразить только часть графика, которая отвечает определенному условию используйте предварительное маскирование данных с помощью функции masked_where из пакета numpy.

7. Как построить ступенчатый график? В чем особенность ступенчатого графика? - Рассмотрим еще одни график – ступенчатый. Такой график строится с помощью функции step(), которая принимает следующий набор параметров:

x: array_like - набор данных для оси абсцисс

y: array_like - набор данных для оси ординат

fmt: str, optional - задает отображение линии (см. функцию plot()). data: indexable object, optional - метки.

where : {‘pre’, ‘post’, ‘mid’}, optional , по умолчанию ‘pre’ - определяет место, где будет установлен шаг.

‘pre’: значение y ставится слева от значения x, т.е. значение y[i] определяется для интервала (x[i-1]; x[i]).

‘post’: значение y ставится справа от значения x, т.е. значение y[i] определяется для интервала (x[i]; x[i+1]).

‘mid’: значение y ставится в середине интервала.

8. Как построить стековый график? В чем особенность стекового графика? - Для построения стекового графика используется функция stackplot(). Суть его в том, что графики отображаются друг над другом, и каждый следующий является суммой предыдущего и заданного набора данных

9. Как построить stem-график? В чем особенность stem-графика? - Визуально этот график выглядит как набор линий от точки с координатами (x, y) до базовой линии, в верхней точке ставится маркер

10. Как построить точечный график? В чем особенность точечного графика? - Для отображения точечного графика предназначена функция scatter(). В простейшем виде точечный график можно получить передав функции scatter() наборы точек для x, y координат

11. Как осуществляется построение столбчатых диаграмм с помощью matplotlib? - Для визуализации категориальных данных хорошо подходят столбчатые диаграммы. Для их построения используются функции: bar() – для построения вертикальной диаграммы

barh() – для построения горизонтальной диаграммы.

12. Что такое групповая столбчатая диаграмма? Что такое столбчатая диаграмма с errorbar элементом? - Errorbar элемент позволяет задать величину ошибки для каждого элемента графика. Для этого используются параметры xerr, yerr и ecolor (для задания цвета)

13. Как выполнить построение круговой диаграммы средствами matplotlib? - Круговые диаграммы – это наглядный способ показать доли компонент в наборе. Они идеально подходят для отчетов, презентаций и т.п. Для построения круговых диаграмм в Matplotlib используется функция pie().

14. Что такое цветовая карта? Как осуществляется работа с цветовыми картами в matplotlib? - Цветовая карта представляет собой подготовленный набор цветов, который хорошо подходит для визуализации того или иного набора данных. Также отметим, что такие карты можно создавать самостоятельно, если среди существующих нет подходящего решения

15. Как отобразить изображение средствами matplotlib? - Основное назначение функции imshow() состоит в представлении 2d растров. Это могут быть картинки, двумерные массивы данных, матрицы и т.п. Напишем простую программу, которая загружает картинку из интернета по заданному URL и отображает ее с использованием библиотеки Matplotlib

16. Как отобразить тепловую карту средствами matplotlibбораторной работе. - В библиотеке Matplotlib есть ещё одна функция с аналогичным функционалом – pcolor(), в отличии от нее рассматриваемая нами pcolormesh() более быстрая и является лучшим вариантом в большинстве случаев. Функция pcolormesh() похожа по своим возможностям на imshow(), но есть и отличия.