МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования «Северо-Кавказский федеральный университет»

Кафедра инфокоммуникаций

Отчѐт по практическому занятию №3.5

«Визуализация данных с помощью matplotlib»

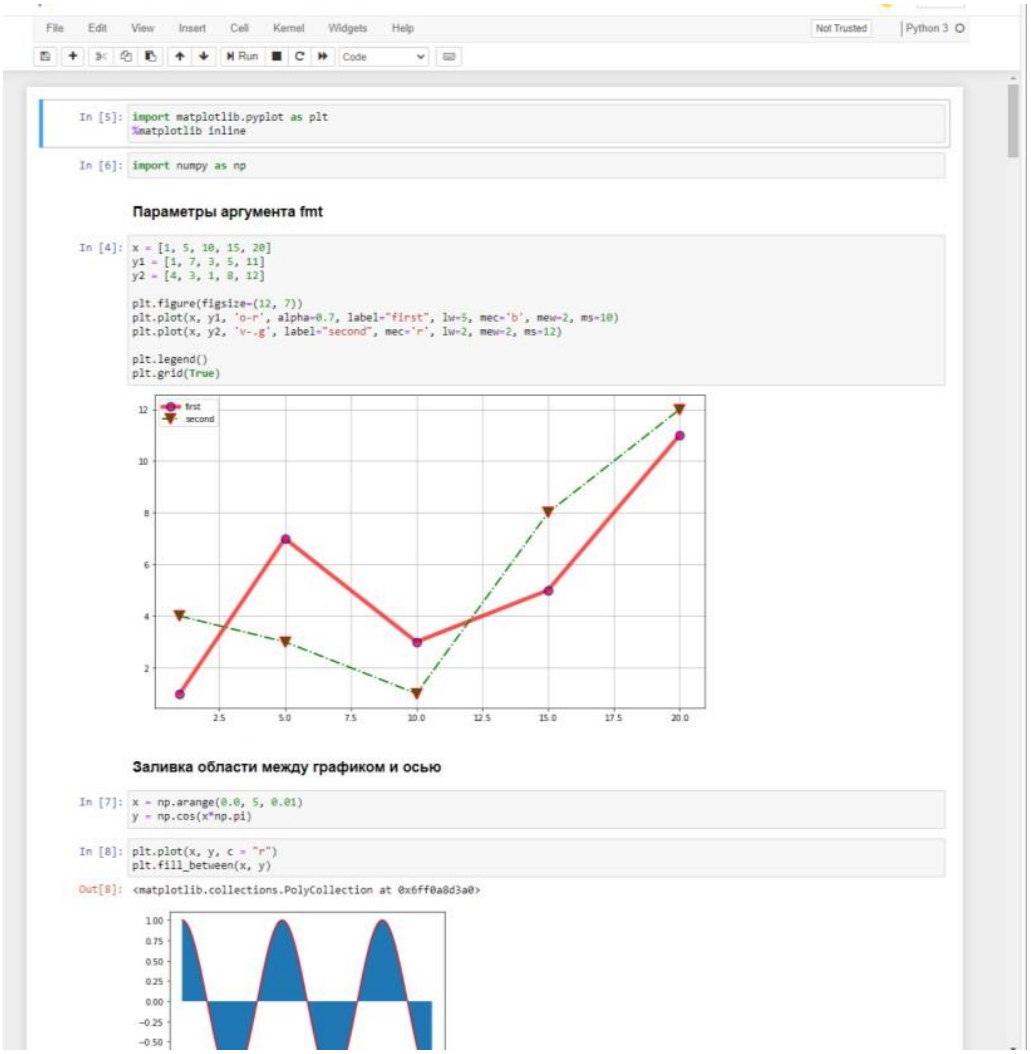
по дисциплине «Теории распознавания образов»

Выполнил студент группы ПИЖ-б-о-21-1
Коновалова В.Н. « »_____20__г.
Подпись студента _____
Работа защищена « »_____20__г.
Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2023

1. 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).

2. Проработать примеры лабораторной работ

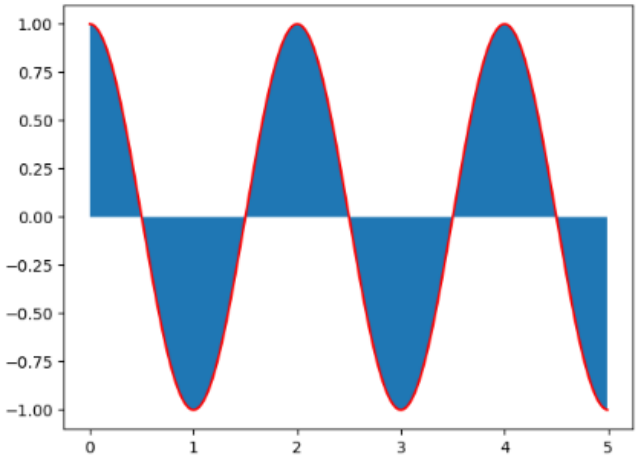


Заливка области между графиком и осью

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

In [6]:
```
plt.plot(x, y, c = "r")
plt.fill_between(x, y)
```

Out[6]: <matplotlib.collections.PolyCollection at 0x2bc19267a00>



Заливка области между графиком и осью

Создадим набор данных для эксперимента:

In [7]:
```
x = np.arange(0.0, 5, 0.01)
y = np.cos(x*np.pi)
```

Отобразим график с заливкой:

In [8]:
```
plt.plot(x, y, c = "r")
plt.fill_between(x, y)
```

Out[8]: <matplotlib.collections.PolyCollection at 0x2bc192a6d00>

Создать ноутбук, в котором выполнить решение трех вычислительных задач (например, задачи из области физики, экономики, математики, статистики и т. д.) требующих построения графика (линейного, кругового, столбчатой), условия которых предварительно необходимо согласовать с преподавателем.

Создать ноутбук, в котором выполнить решение вычислительной задачи из области экономики требующей построения:

1)линейного графика; 2)столбчатой диаграммы; 3)круговой диаграммы;

Условие:

Себестоимость 1 тонны картофеля, выращенной на первом участке составляет 5000 руб., на втором участке – 8 тыс. руб., на третьем – 12 тыс. руб. Оптовая цена 1 тонны картофеля составляет 12 тыс. руб.

Чему равна дифференциальная рента, получаемая на первом и втором участке при производстве от 1 до 10 тонн картофеля?

Чему равна дифференциальная рента на каждом участке при производстве 10 тонн картофеля?

In [1]:
```
import matplotlib.pyplot as plt
```

In [2]:
```
# Создаем список себестоимостей на каждом участке
costs = [5000, 8000, 12000]

#Оптовая цена 1 тонны картофеля
wholesale_price = 12000

# Создаем список обьемов производства картофеля
productions = range(1, 11)
```

1)Линейный график

Чтобы решить задачу, нужно сначала вычислить себестоимость производства от 1 до 10 тонн картофеля на первом участке и оптовую выручку от их продажи, а затем вычислить разницу между ними - это и будет дифференциальной рентой.

In [3]:
```
# Создаем список стоимостей производства 1 тонны картофеля для каждого обьема производства
cost_1 = [costs[0]/ p for p in productions]

# Создаем список дифференциальной ренты для каждого обьема производства
rent_1 = [wholesale_price - c for c in cost_1]
```

In [4]:
```
# Строим график Рикардо
plt.plot(productions, cost_1, label='Стоимость производства')
plt.axhline(y=wholesale_price, color='r', linestyle='-', label='Оптовая цена')
plt.plot(productions, rent_1, label='Дифференциальная рента')
plt.xlabel('Обьем производства, тонн')
plt.ylabel('Стоимость, руб.')
plt.legend()
plt.show()
```

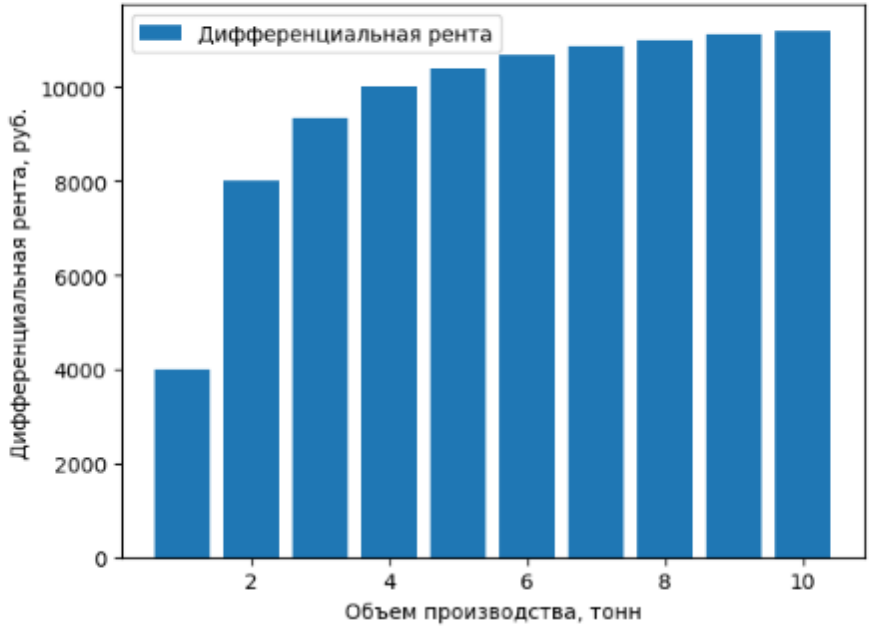
Объем производства, тонн

2)Столбчатая диаграмма

Получим дифференциальнаю ренту анналогично вычислениям в первом пункте

In [5]:
```
# Создаем список стоимостей производства 1 тонны картофеля для каждого объема производства
cost_2  = [costs[1] / p for p in productions]

# Создаем список дифференциальной ренты для каждого обьема производства
rent_2 = [wholesale_price - c for c in cost_2 ]
```

In [6]:
```
# Строим столбчатый график Рикардо для второго участка
plt.bar(productions, rent_2, label='Дифференциальная рента')
plt.xlabel('Обьем производства, тонн')
plt.ylabel('Дифференциальная рента, руб.')
plt.legend()
plt.show()
```



3)Круговая диаграмма

Высчитываем дифференциальную ренту на каждом участке при производстве 10 тонн картофеля

In [7]:
```
# Вычисляем обьем дифференциальной ренты на каждом участке при производстве 10 тонн картофеля
rent1 = wholesale_price - costs[0]/10
rent2 = wholesale_price - costs[1]/10
rent3 = wholesale_price - costs[2]/10

# Создаем список дифференциальной ренты на каждом участке
rents = [rent1, rent2, rent3]

# Задаем названия секторов
labels = ['Участок 1', 'Участок 2', 'Участок 3']
```

In [8]:
```
# Строим круговую диаграмму
plt.pie(rents, labels=labels, autopct='%1.1f%%')
plt.title('Дифференциальная рента на каждом участке')
plt.show()
```

Найти какое-либо изображение в сети Интернет. Создать ноутбук, в котором будет отображено выбранное изображение средствами библиотеки matplotlib по URL из сети Интернет

Найти какое-либо изображение в сети Интернет. Создать ноутбук, в котором будет отображено выбранное изображение средствами библиотеки matplotlib по URL из сети Интернет.

In [1]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
from PIL import Image
import requests
```

In [3]:
```
from io import BytesIO
```

In [4]:
```
plt.figure(figsize=(20, 20))

url = requests.get('https://chudo-prirody.com/uploads/posts/2021-08/thumbs/1628693056_39-p-novogodnii-kotik-foto-50.jpg?size=807
img = Image.open(BytesIO(url.content))

plt.imshow(img)
```

Out[4]: `<matplotlib.figure.AxesImage at 0x24122024400>`

Вопросы для защиты работы

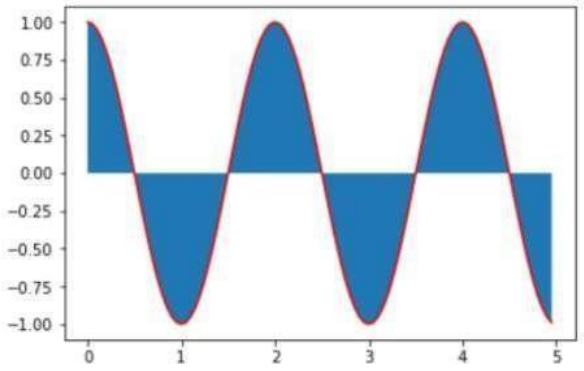
1. Как выполнить построение линейного графика с помощью matplotlib?

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)
```

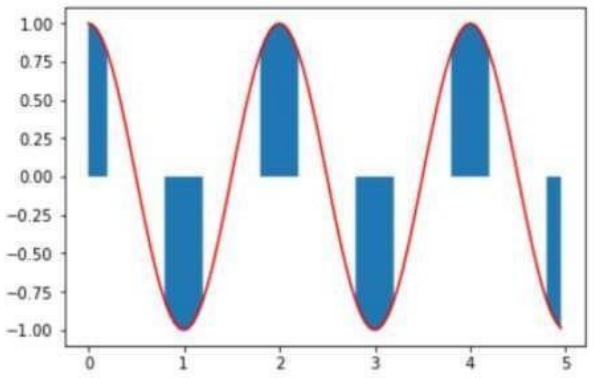
2. Как выполнить заливку области между графиком и осью? Между двумя графиками?

```
plt.plot(x, y, c = "r")
plt.fill_between(x, y)
```



3. Как выполнить выборочную заливку, которая удовлетворяет некоторому условию?

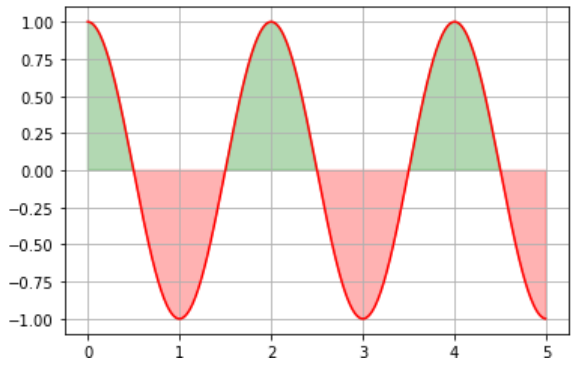
```
plt.plot(x, y, c="r")
plt.fill_between(x, y, where=(y > 0.75) | (y < -0.75))
```



4. Как выполнить двухцветную заливку?

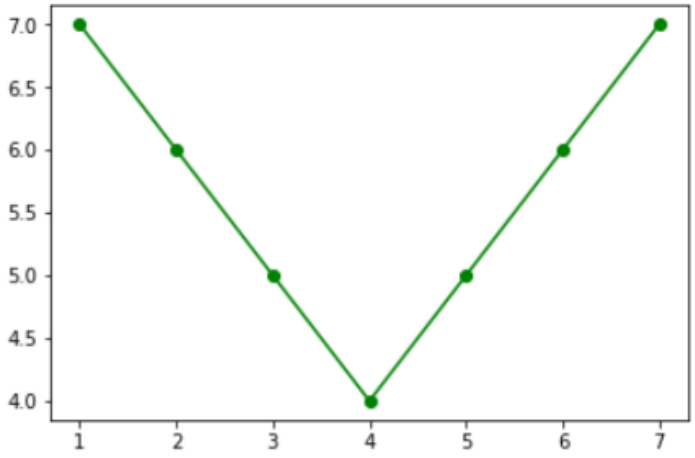
In [14]:
```
plt.plot(x, y, c="r")
plt.grid()

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)
```

Out[14]: <matplotlib.collections.PolyCollection at 0x1b7583e04f0>



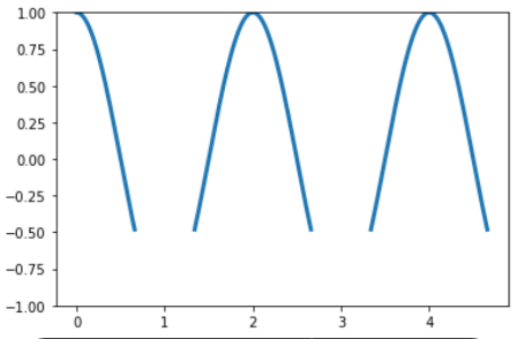
5. Как выполнить маркировку графиков?

```
x = [1, 2, 3, 4, 5, 6, 7]
y = [7, 6, 5, 4, 5, 6, 7]

plt.plot(x, y, marker="o", c="g")
```



6. Как выполнить обрезку графиков?

```
x = np.arange(0.0, 5, 0.01)
y = np.cos(x * np.pi)

y_masked = np.ma.masked_where(y < -0.5, y)
plt.ylim(-1, 1)

plt.plot(x, y_masked, linewidth=3)
```

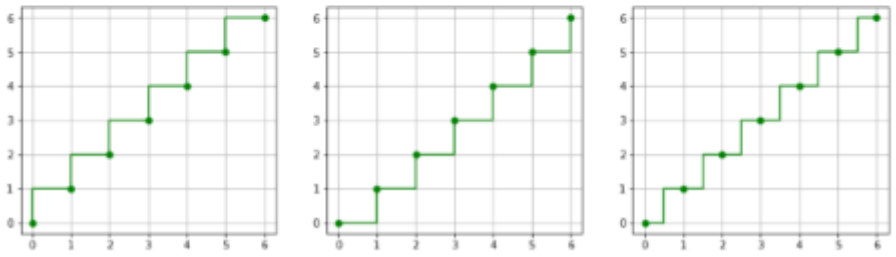


7. Как построить ступенчатый график? В чем особенность ступенчатого графика?

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.

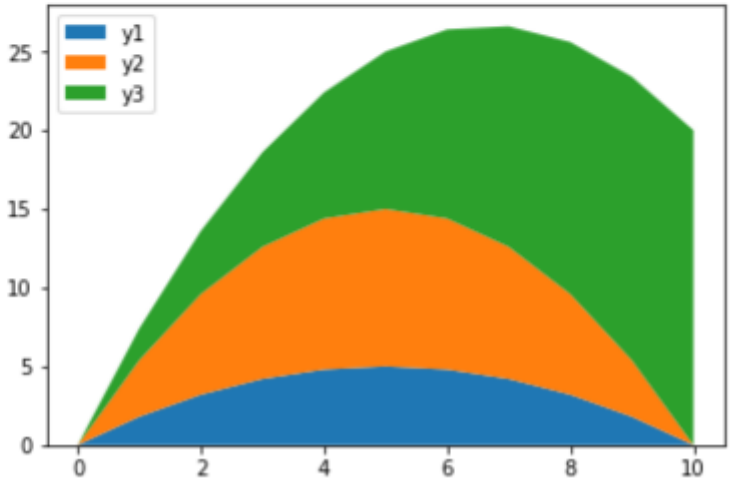
```
x = np.arange(0, 7)
y = x

where_set = ['pre', 'post', 'mid']
fig, axs = plt.subplots(1, 3, figsize=(15, 4))

for i, ax in enumerate(axs):
  ax.step(x, y, "g-o", where=where_set[i])
  ax.grid()
```



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.

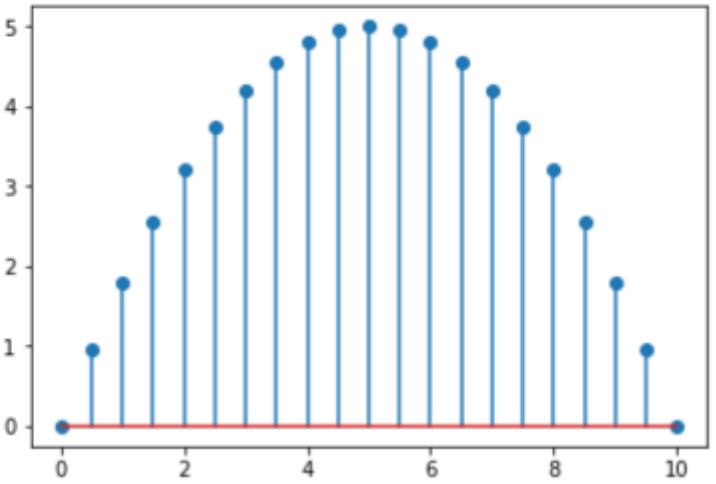
```
x = np.arange(0, 11, 1)

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')
```



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.

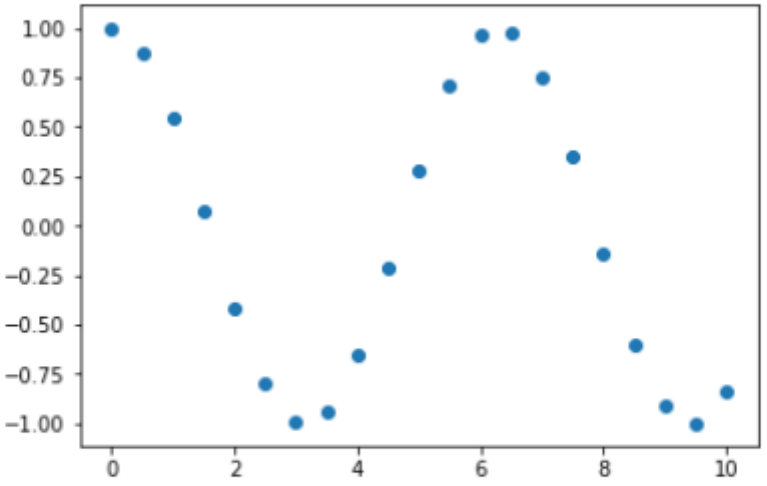
```
x = np.arange(0, 10.5, 0.5)
y = np.array([(-0.2)*i**2+2*i for i in x])

plt.stem(x, y)
```



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.

```
x = np.arange(0, 10.5, 0.5)
y = np.cos(x)

plt.scatter(x, y)
```

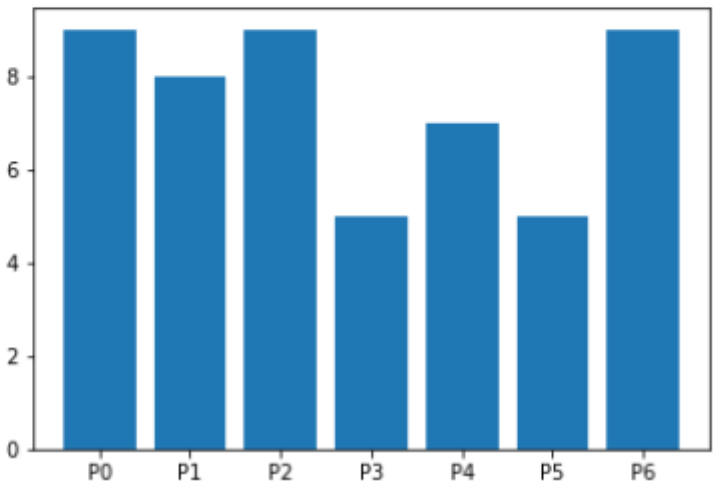


Для более детальной настройки отображения необходимо воспользоваться дополнительными параметрами функции *scatter()*, сигнатура ее вызова имеет следующий вид:

```
scatter(x, y, s=None, c=None, marker=None, cmap=None, norm=None, vmin=None,
vmax=None, alpha=None, linewidths=None, verts=None, edgecolors=None, *,
plotnonfinite=False, data=None, **kwargs)
```

11. Как осуществляется построение столбчатых диаграмм с помощью matplotlib?

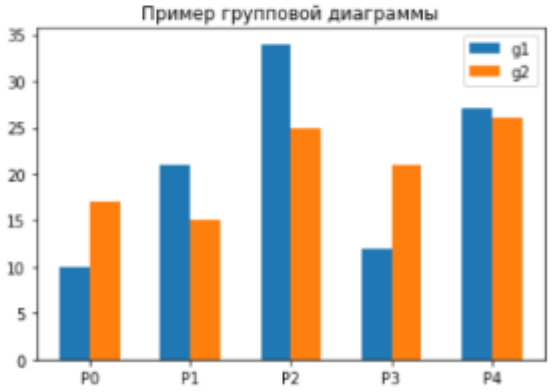
```
np.random.seed(123)

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

plt.bar(groups, counts)
```



12. Что такое групповая столбчатая диаграмма? Что такое столбчатая диаграмма с errorbar элементом?

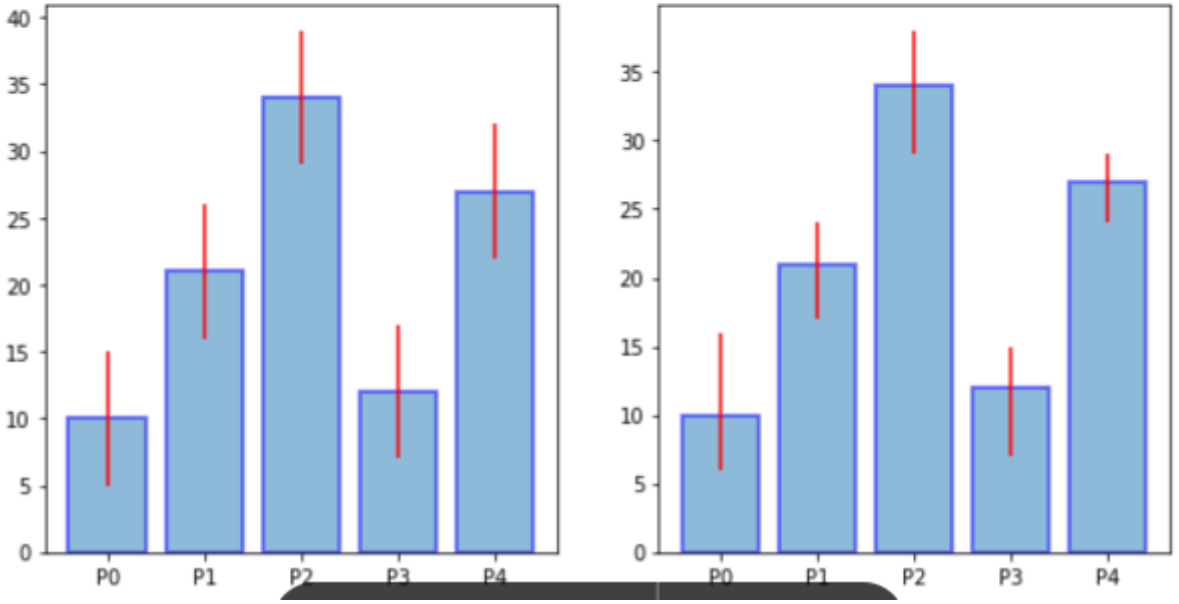
```
cat_par = [f"P{i}" for i in range(5)]

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()
```

В



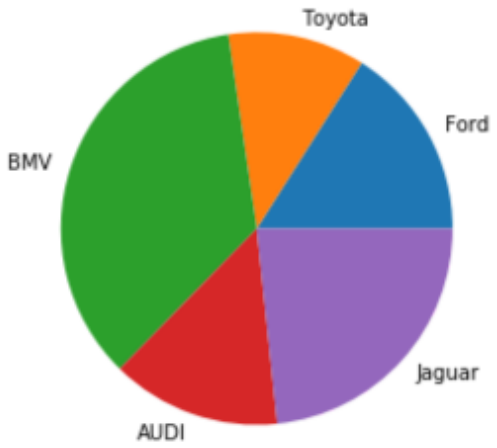
Errorbar элемент позволяет задать величину ошибки для каждого элемента графика. Для этого используются параметры xerr, yerr и ecolor (для задания цвета).

```
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)
```



13. Как выполнить построение круговой диаграммы средствами matplotlib?

```
vals = [24, 17, 53, 21, 35]

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

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



Рассмотрим параметры функции *pie()*:

- `x`: массив - массив с размерами долей.
- `explode`: массив, optional, значение по умолчанию: None - если параметр не равен *None*, то часть долей, который перечислены в передаваемом значении будут вынесены из диаграммы на заданное расстояние, пример диаграммы:

14. Что такое цветовая карта? Как осуществляется работа с цветовыми картами в matplotlib?

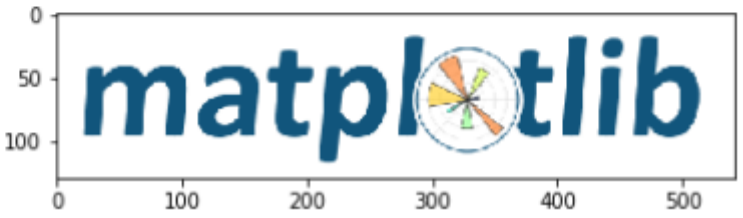
Цветовая карта представляет собой подготовленный набор цветов, который хорошо подходит для визуализации того или иного набора данных.

15. Как отобразить изображение средствами matplotlib?

Рассмотрим две функции для построения цветовой сетки: imshow() и pcolormesh().

```
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)
```

В результате получим изображение логотипа *Matplotlib*.



Рассмотрим параметры функции *pcolormesh()*:

- `C` : массив - *2D* массив скалярных значений
- `cmap` : str или *Colormap*, optional - см. *cmap* в *imshow()*
- `norm` : *Normalize*, optional - см. *norm* в *imshow()*

- `vmin`, `vmax` : scalar, optional, значение по умолчанию: None - см. *vmin, vmax* в *imshow()*
- `edgecolors` : {'none', None, 'face', color, color sequence}, optional - цвет границы, по умолчанию: 'none', возможны следующие варианты:
  - *'none'* or '': без отображения границы.
  - *None*: черный цвет.
  - *'face'*: используется цвет ячейки.
  - Можно выбрать цвет из доступных наборов.
- `alpha` : scalar, optional, значение по умолчанию: None - см. *alpha* в *imshow()*.
- `shading` : {'flat', 'gouraud'}, optional - стиль заливки, доступные значения:
  - *'flat'*: сплошной цвет заливки для каждого квадрата.
  - *'gouraud'*: для каждого квадрата будет использован метод затенения *Gouraud*.
- `snap` : bool, optional, значение по умолчанию: False - привязка сетки к границам пикселей.

16. Как отобразить тепловую карту средствами matplotlib?

Пример использования функции *pcolormesh()*:

```
np.random.seed(123)

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