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

Институт цифрового развития
Кафедра инфокоммуникаций

«Визуализация данных с помощью matplotlib»

ОТЧЕТ
по лабораторной работе №5
дисциплины
«Технологии распознавания образов»

Выполнил:
Мизин Глеб Егорович
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, 2023 г.

Проработка примеров:

Линейный график

Построение графика

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

In [2]:
```
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)
```

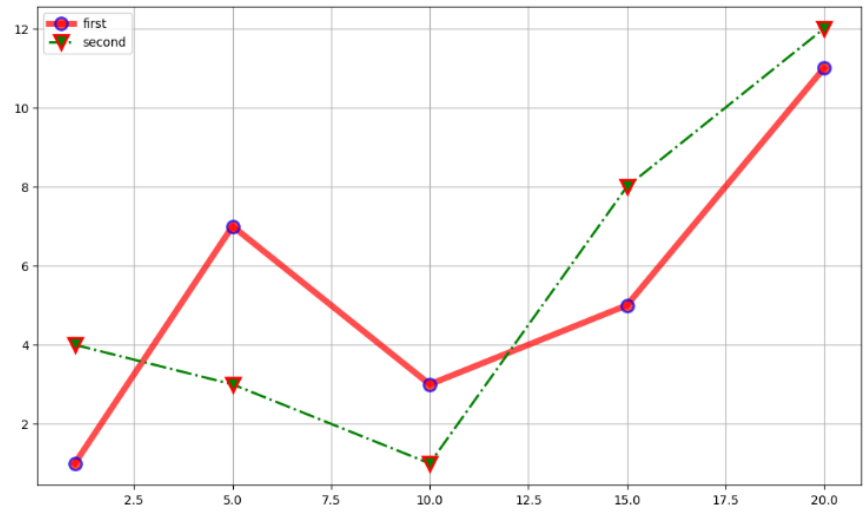


Рисунок 1 – Проработка примеров

Заливка области между графиком и осью

In [3]:
```
import numpy as np
x = np.arange(0.0, 5, 0.01)
y = np.cos(x*np.pi)
```

In [4]:
```
plt.plot(x, y, c = "r")
plt.fill_between(x, y)
```

Out[4]:
```
<matplotlib.collections.PolyCollection at 0x2109cd356d0>
```

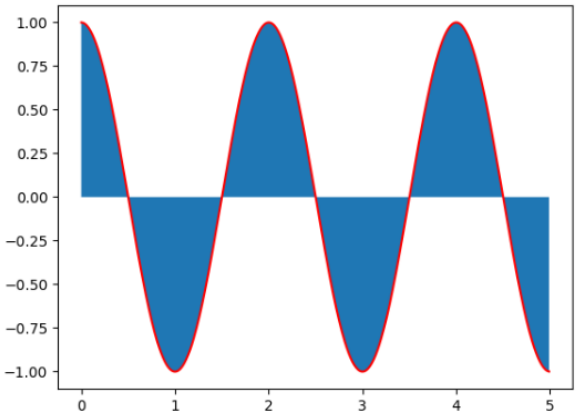


Рисунок 2 – Проработка примеров

```
In [5]: plt.plot(x, y, c="r")
        plt.fill_between(x, y, where=(y > 0.75) | (y < -0.75))
```

```
Out[5]: <matplotlib.collections.PolyCollection at 0x2109cdb4190>
```

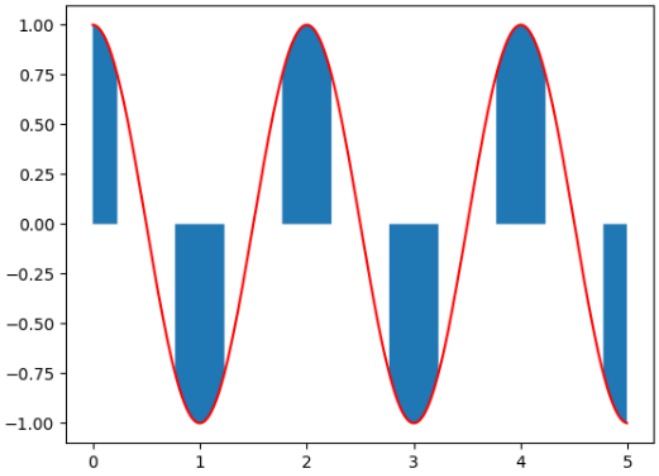


Рисунок 3 – Проработка примеров

```
In [6]: plt.plot(x, y, c="r")
        plt.fill_between(x, y, where=(y > 0))
```

```
Out[6]: <matplotlib.collections.PolyCollection at 0x2109cd67e90>
```

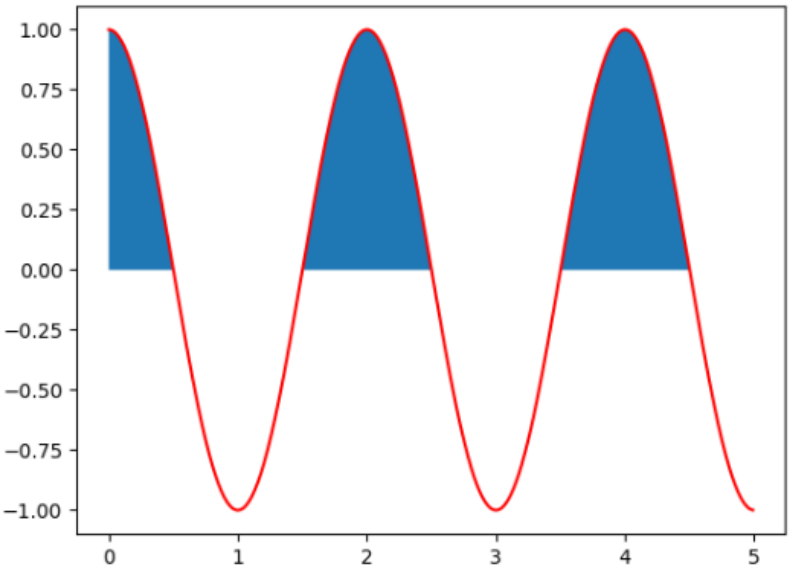


Рисунок 4 – Проработка примеров

```
In [7]: plt.plot(x, y, c="r")
        plt.grid()
        plt.fill_between(x, 0.5, y, where=(y>=0.5))
Out[7]: <matplotlib.collections.PolyCollection at 0x2109ceb0e10>
```

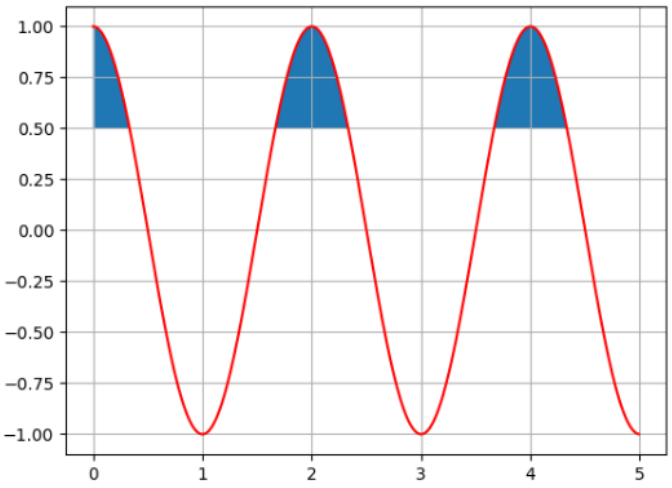
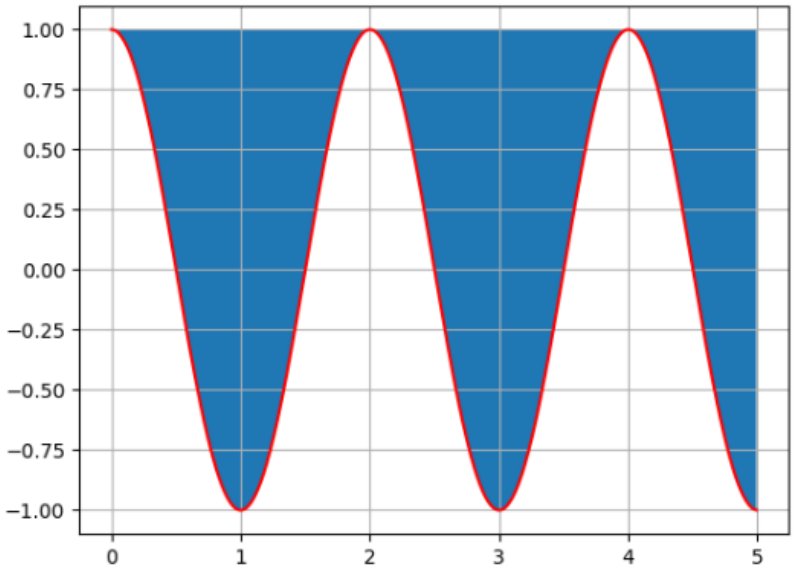


Рисунок 5 – Проработка примеров

```
In [8]: plt.plot(x, y, c="r")
        plt.grid()
        plt.fill_between(x, y, 1)
Out[8]: <matplotlib.collections.PolyCollection at 0x2109cf32050>
```



Так же библиотека matplotlib предоставляет возможности заливки графиков несколькими цветами

Рисунок 6 – Проработка примеров

```
In [9]: 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[9]: <matplotlib.collections.PolyCollection at 0x2109cf4db90>
```

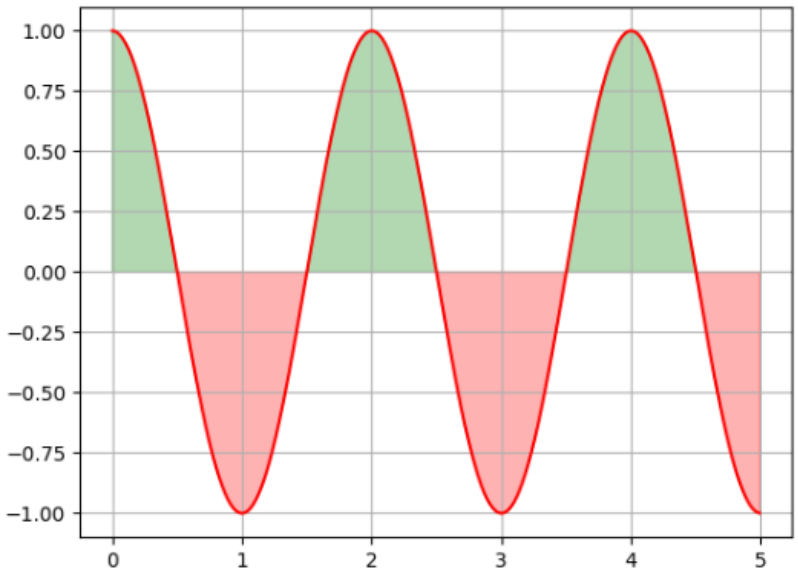


Рисунок 7 – Проработка примеров

Настройка маркировки графиков

```
In [10]: 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[10]: [<matplotlib.lines.Line2D at 0x2109cfd1990>]
```

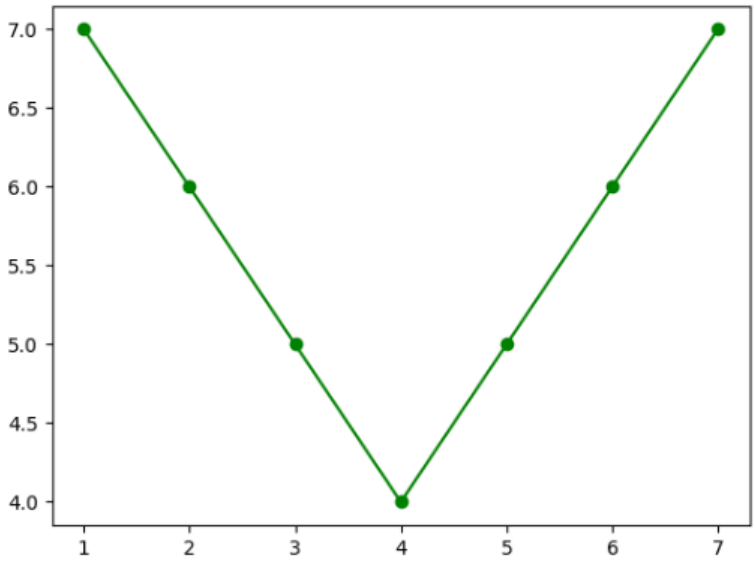
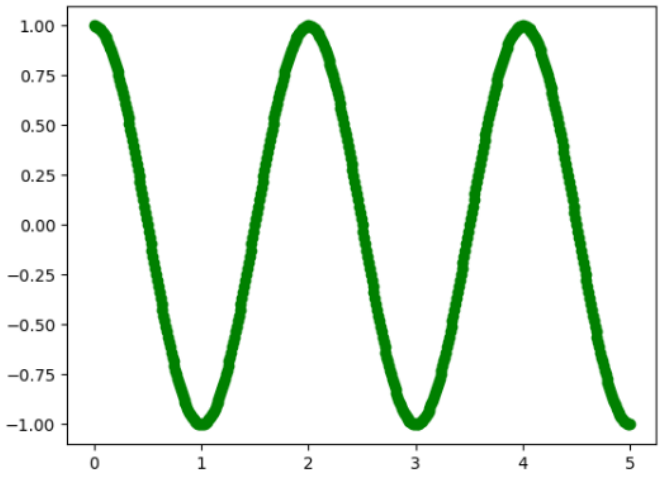


Рисунок 8 – Проработка примеров

In [11]:
```
import numpy as np
x = np.arange(0.0, 5, 0.01)
y = np.cos(x*np.pi)

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

Out[11]: [<matplotlib.lines.Line2D at 0x210a02876d0>]



В примере выше на графике создаётся точка для каждого высчитанного значения, из-за этого кажется что линия графика просто стала шире За интервал отображения маркеров отвечает параметр markevery который имеет стандартное значение None, что в свою очередь задаёт отображение каждой точки.

Рисунок 9 – Проработка примеров

In [33]:
```
x = np.arange(0.0, 5, 0.01)
y = np.cos(x * np.pi)
m_ev_case = [None, 10, (100, 30), slice(100,400,15), [0, 100, 200, 300], [10,
50, 100]]
fig, ax = plt.subplots(2, 3, figsize=(10, 7))
axs = [ax[i, j] for i in range(2) for j in range(3)]
for i, case in enumerate(m_ev_case):
    axs[i].set_title(str(case))
    axs[i].plot(x, y, "o", ls='-', ms=7, markevery=case)
```

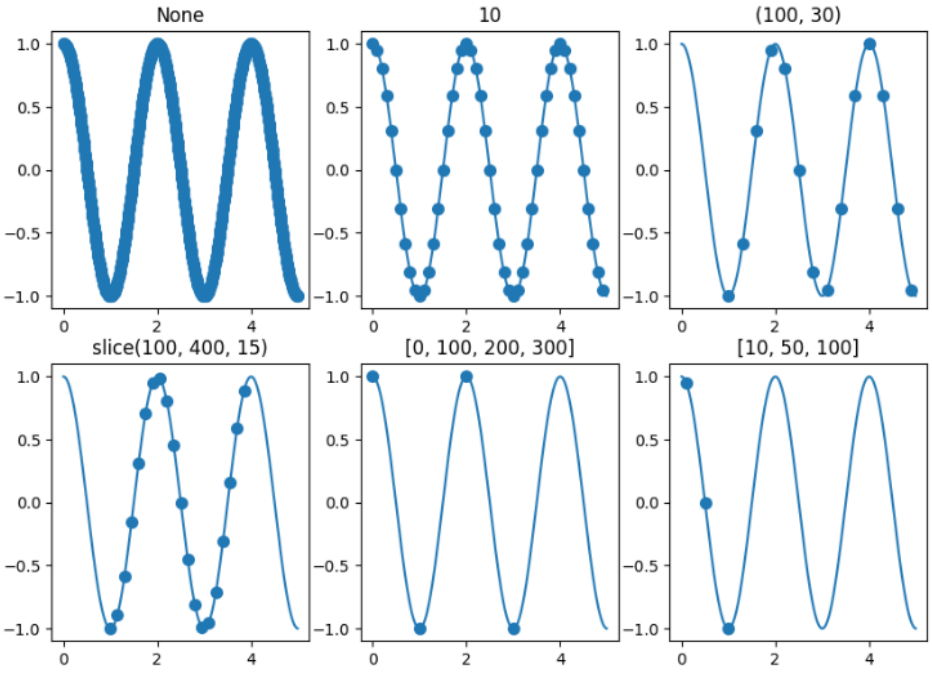


Рисунок 10 – Проработка примеров

Обрезка графика

Для обрезки графика используется функция masked_where

In [34]:
```
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)
```

Out[34]:
```
[<matplotlib.lines.Line2D at 0x210a3958950>]
```

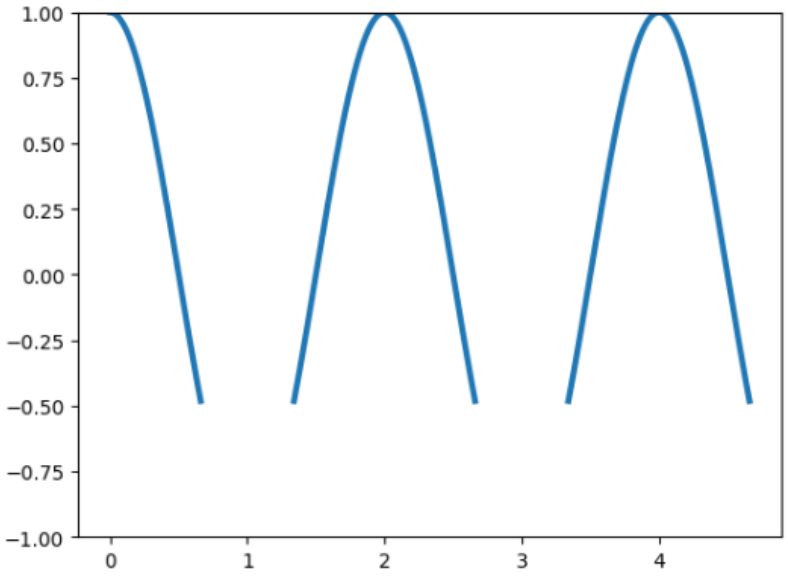


Рисунок 11 – Проработка примеров

Ступенчатый график

In [35]:
```
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()
```

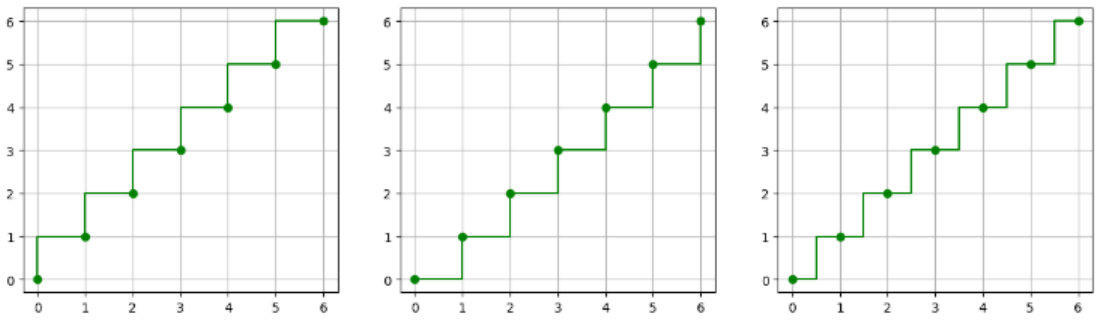


Рисунок 12 – Проработка примеров

Стековый график

In [36]:
```
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[36]:
```
<matplotlib.legend.Legend at 0x210a0af2950>
```

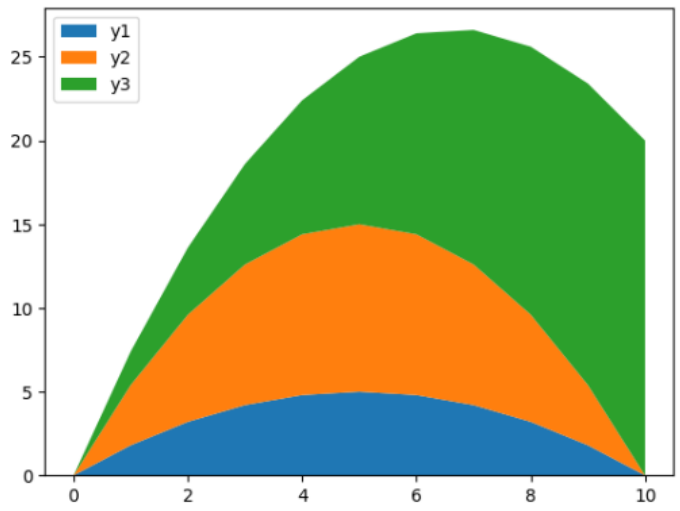


Рисунок 13 – Проработка примеров

Stem-график

In [37]:
```
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[37]:
```
<StemContainer object of 3 artists>
```

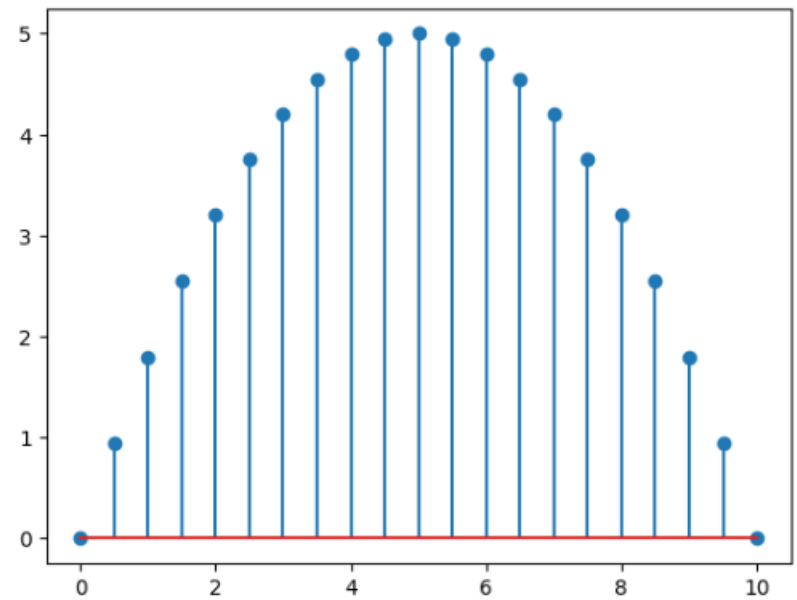


Рисунок 14 – Проработка примеров

```
In [38]: plt.stem(x, y, linefmt="r--", markerfmt="^", bottom=1)
Out[38]: <StemContainer object of 3 artists>
```

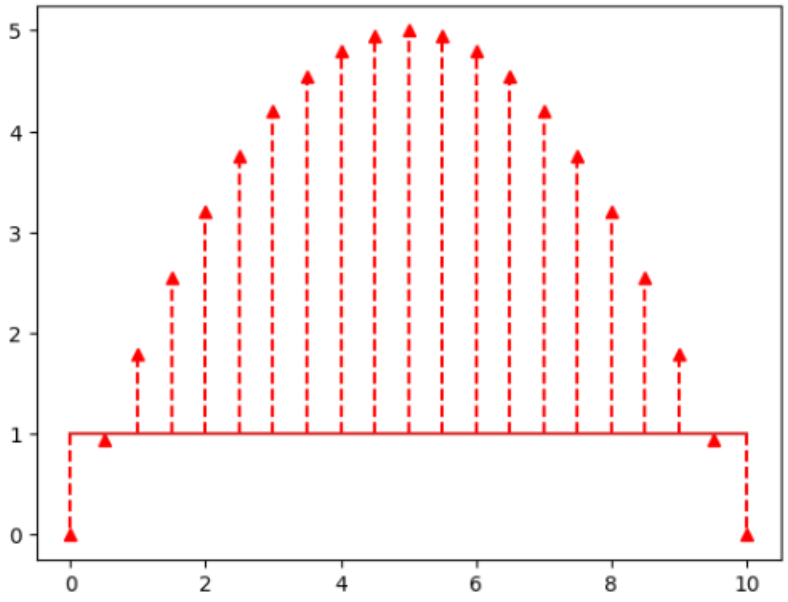


Рисунок 15 – Проработка примеров

Точечный график

```
In [39]: x = np.arange(0, 10.5, 0.5)
         y = np.cos(x)
         plt.scatter(x, y)
Out[39]: <matplotlib.collections.PathCollection at 0x210a3dc65d0>
```

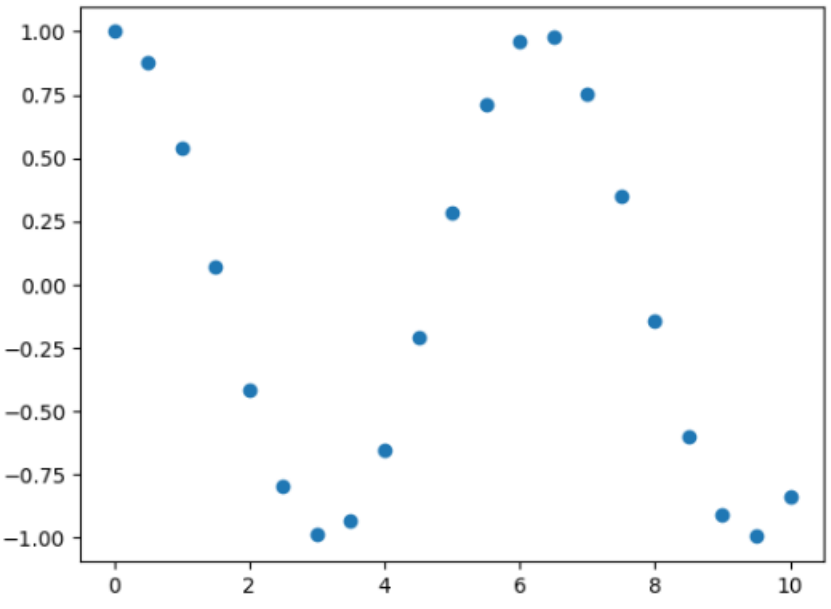


Рисунок 16 – Проработка примеров

```
In [40]: 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[40]: <matplotlib.collections.PathCollection at 0x210a3e063d0>
```

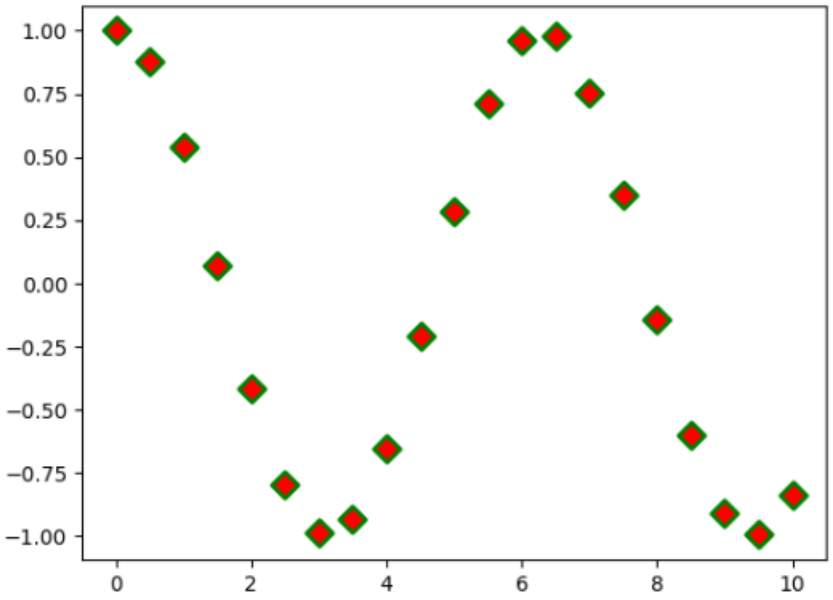


Рисунок 17 – Проработка примеров

```
In [41]: import matplotlib.colors as mcolors
         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[41]: [<matplotlib.lines.Line2D at 0x210a403c190>]
```

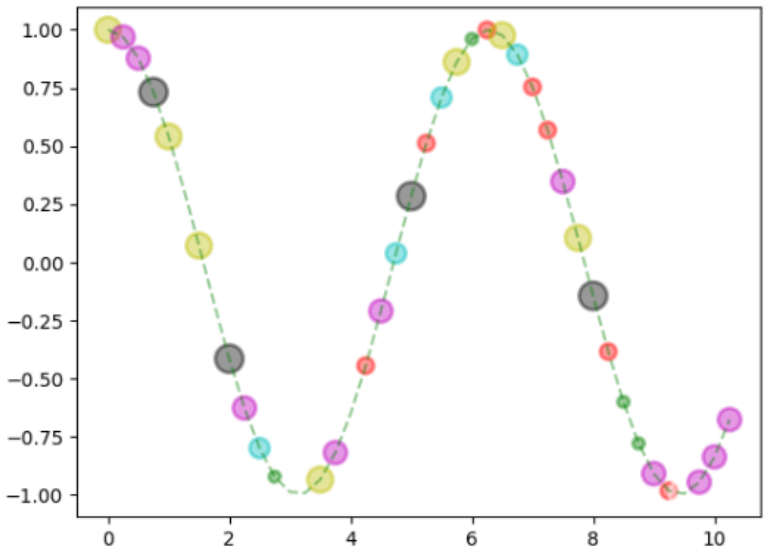


Рисунок 18 – Проработка примеров

Столбчатые диаграммы

In [42]:
```
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[42]: <BarContainer object of 7 artists>

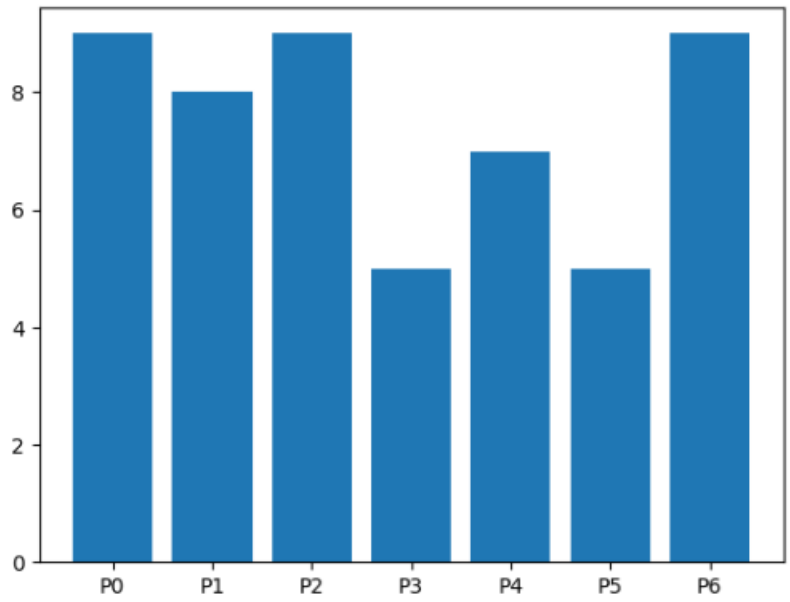


Рисунок 19 – Проработка примеров

In [43]:
```
plt.barh(groups, counts)
```

Out[43]: <BarContainer object of 7 artists>

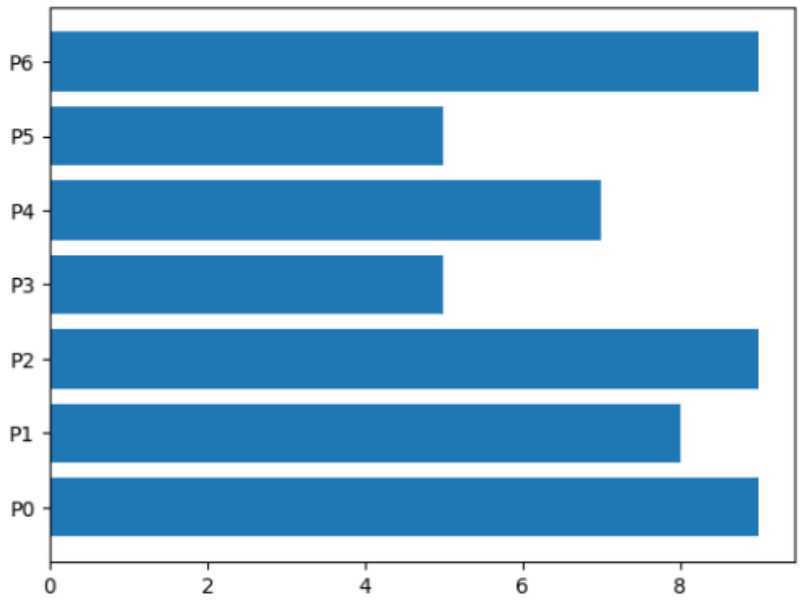


Рисунок 20 – Проработка примеров

```
In [44]: import matplotlib.colors as mcolors
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[44]: <BarContainer object of 7 artists>
```

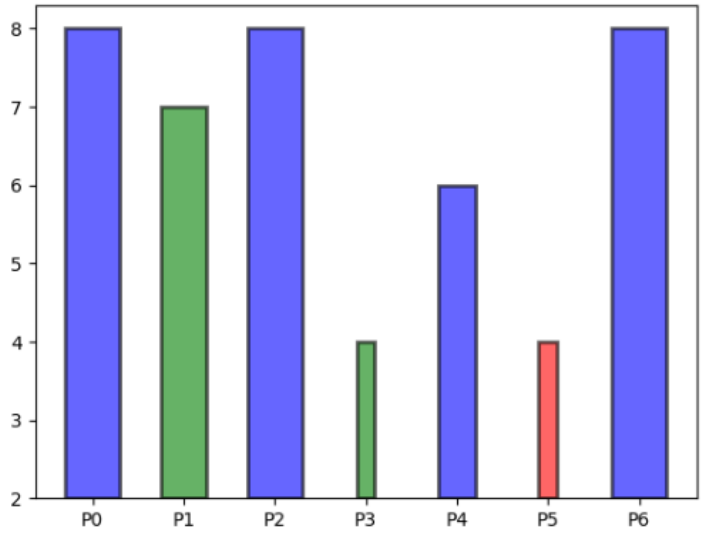


Рисунок 21 – Проработка примеров

Групповые столбчатые диаграммы

```
In [45]: 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[45]: <matplotlib.legend.Legend at 0x210a075add0>
```

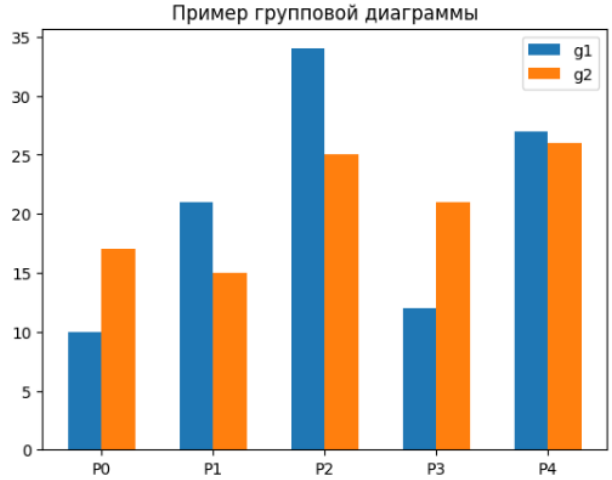


Рисунок 22 – Проработка примеров

Круговые диаграммы

Классические круговые диаграммы

In [46]:
```
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[46]:
```
(-1.0999964134742897,
 1.0999998292130615,
 -1.0999996332200421,
 1.0999997447175798)
```

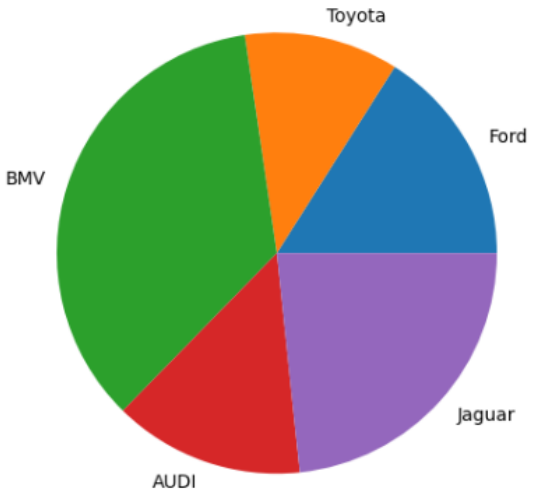


Рисунок 23 – Проработка примеров

In [47]:
```
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[47]:
```
(-1.2541693468510096,
 1.1991449549931066,
 -1.1017808888721372,
 1.1374061134115763)
```

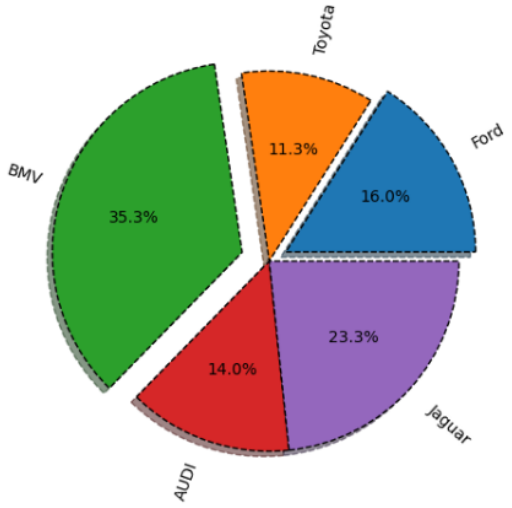


Рисунок 24 – Проработка примеров

Вложенные круговые диаграммы

In [48]:
```
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[48]:
```
([<matplotlib.patches.Wedge at 0x210a38fe750>,
  <matplotlib.patches.Wedge at 0x210a22fec50>,
  <matplotlib.patches.Wedge at 0x210a22fd550>,
  <matplotlib.patches.Wedge at 0x210a22fc490>,
  <matplotlib.patches.Wedge at 0x210a22902d0>,
  <matplotlib.patches.Wedge at 0x210a2293090>,
  <matplotlib.patches.Wedge at 0x210a2292110>,
  <matplotlib.patches.Wedge at 0x210a06873d0>,
  <matplotlib.patches.Wedge at 0x210a2293310>],
 [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, '')])
```

Рисунок 25 – Проработка примеров

Круговая диаграмма в виде бублика

In [49]:
```
vals = [24, 17, 53, 21, 35]
labels = ["Ford", "Toyota", "BMV", "AUDI", "Jaguar"]
fig, ax = plt.subplots()
ax.pie(vals, labels=labels, wedgeprops=dict(width=0.5))
```

Out[49]:
```
([<matplotlib.patches.Wedge at 0x210a2268990>,
  <matplotlib.patches.Wedge at 0x210a0531950>,
  <matplotlib.patches.Wedge at 0x210a0584710>,
  <matplotlib.patches.Wedge at 0x210a0586310>,
  <matplotlib.patches.Wedge at 0x210a0585f90>],
 [Text(0.9639373540021144, 0.5299290306818474, 'Ford'),
  Text(0.22870287165240302, 1.075962358309037, 'Toyota'),
  Text(-1.046162158377023, 0.3399187231970734, 'BMV'),
  Text(-0.3617533684721028, -1.0388139873909512, 'AUDI'),
  Text(0.8174592712713289, -0.7360437078139777, 'Jaguar')])
```

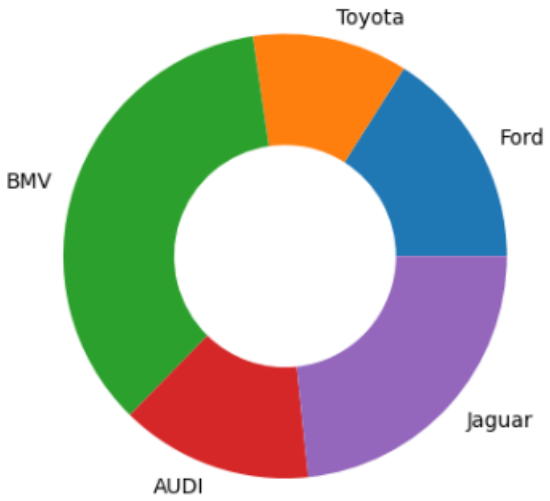


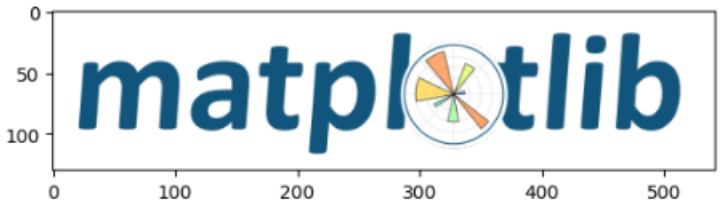
Рисунок 26 – Проработка примеров

Цветовые карты (colormaps)

Построение цветовой сетки и отображение изображений

In [50]:
```
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[50]: <matplotlib.image.AxesImage at 0x210a22a8790>



In [51]:
```
np.random.seed(196808121)
data = np.random.randn(25, 25)
plt.imshow(data)
```

Out[51]: <matplotlib.image.AxesImage at 0x2109cc476d0>

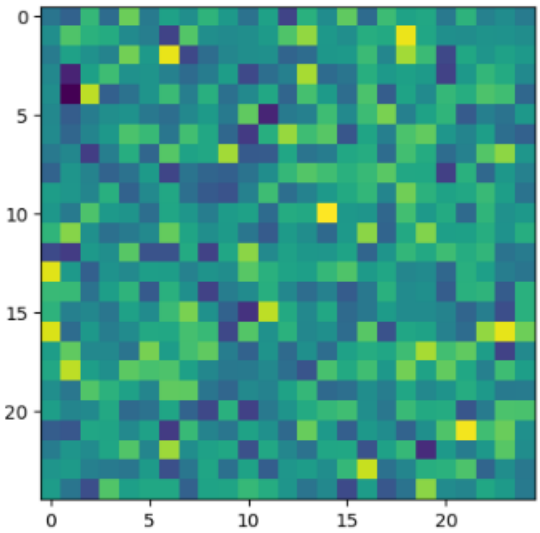
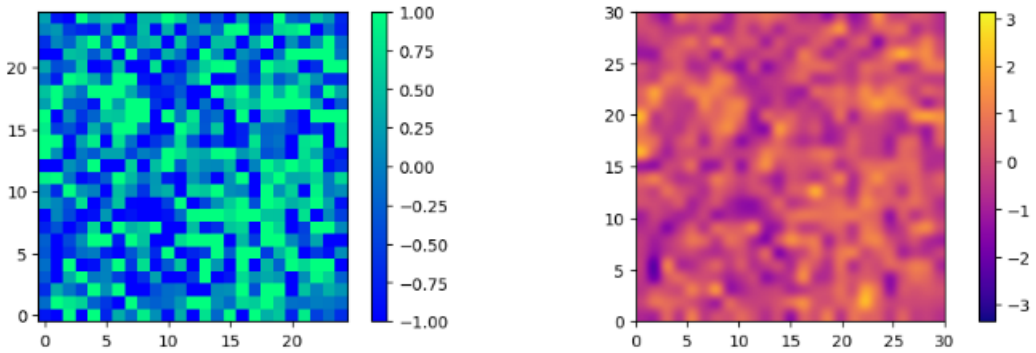


Рисунок 27 – Проработка примеров

```
In [52]: 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[52]: <matplotlib.colorbar.Colorbar at 0x2109f153b90>
```



Отображение тепловой карты

```
In [53]: np.random.seed(123)
         data = np.random.rand(5, 7)
         plt.pcolormesh(data, cmap='plasma', edgecolors="k", shading='flat')
```

```
Out[53]: <matplotlib.collections.QuadMesh at 0x210a39e32d0>
```

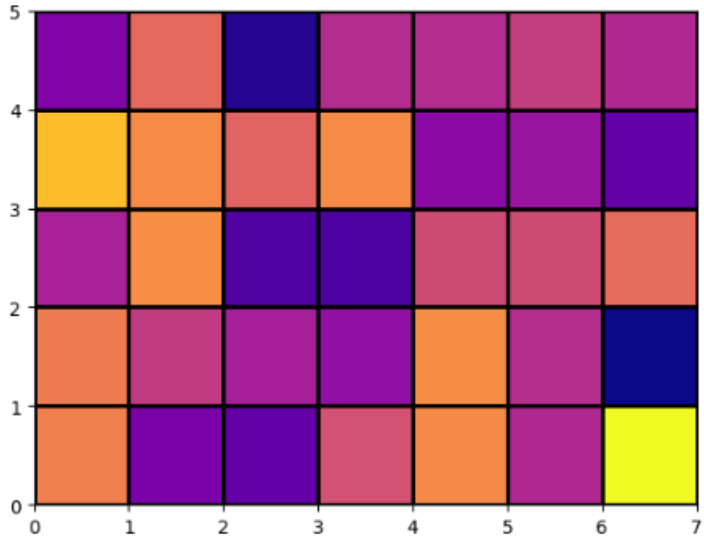


Рисунок 28 – Проработка примеров

Задание №1: Демонстрация работы с линейным графиком

Задание №1

Решение физической задачи с использованием линейного графика

Условие: Автомобильная камера накачана до давления $p_1 = 220\text{кПА}$ при температуре $T_1 = 290\text{K}$. Во время движения она нагрелась до температуры $T_2 = 330\text{K}$ и с шумом лопнула. Считая процесс, происходящий после повреждения камеры, адиабатным, определить изменение температуры вышедшего из неё воздуха. Внешнее давление $p_0 = 100\text{кПА}$

Аналитическое решение: Выделим два термодинамических процесса задачи. Первый - изохорическое нагревание воздуха в камере, а второй адиабатическое расширение воздуха:

1. $p_1/T_1 = p_2/T_2$, $p_2 = p_1 T_2/T_1$
2. $T_2 p_2^{(1-\gamma)/\gamma} = T_0 p_0^{(1-\gamma)/\gamma}$, где T_0 - температура воздуха в конце адиабатического расширения, $T_0 = T_2 p_2^{(1-\gamma)/\gamma} / p_0^{(1-\gamma)/\gamma}$.
3. Теперь найдём разность температур $\triangle T = T_0 - T_2$, $\triangle T = T_2 (p_2/p_0)^{(1-\gamma)/\gamma} - T_2$ и $\triangle T = T_2 \times ((p_2/p_0)^{(1-\gamma)/\gamma} - 1)$

Рисунок 29 – Задача №1

Составим программу для графического решения данной задачи:

In [7]:
```
import matplotlib.pyplot as plt
import math as mt
R=8.31
p0=1.0e5
p1=2.2e5
T1=290
T2=330
g=1.4
b=(1-g)/g
c=1/b
p2=p1*T2/T1
T0=T2*mt.pow(p2/p0,b)
DT=T2-T0
print ("p2=%8.3e"%p2,"T0=%5.1f"%T0,"DT=%5.1f"%DT)
N1=100; dT1=(T2-T1)/N1; dT2=(T0-T2)/N1;
t=[]
DN=[]
t.append(T1)
DN.append(p1)
for i in range(1,N1):
 t1=T1+i*dT1; t.append(t1); DN.append(p1*t1/T1)
for i in range(1,N1):
 t1=T2+i*dT2
 t.append(t1);DN.append(p2*mt.pow(T2/t1,c))
plt.plot(t,DN,'k-')
plt.xlabel('$T$',fontsize=14)
plt.ylabel('$p$',fontsize=14)
plt.show()
```

```
p2=2.503e+05 T0=253.9 DT= 76.1
```

Рисунок 30 – Задача №1

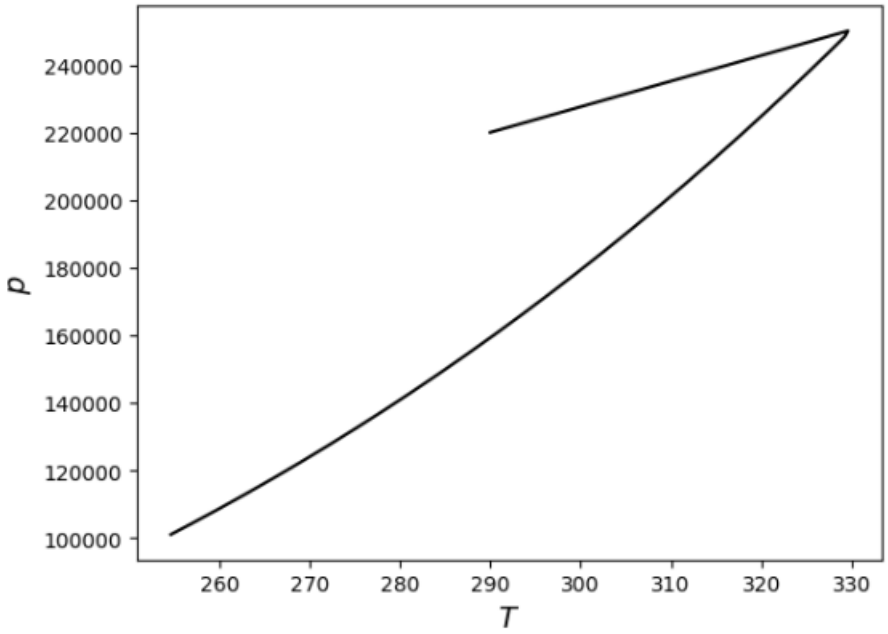


Рисунок 31 – Задача №1

Задание №2: Демонстрация работы с гистограммой

> Условие задачи: Найти давление воздуха в откачиваемом сосуде как функцию времени откачки t. Объем сосуда V = 100 л. Процесс считать изотермическим и скорость откачки независимой от давления и равной C = 0.01 л/с. Скоростью откачки называют объем газа, откачиваемый за единицу времени, причем этот объем измеряется при давлении газа в данный момент времени.

Аналитическое решение поставленной задачи: Рассмотрим изотермический процесс при изменении объема на dV и давления на dp , тогда $pV = (p + dp)(V + dV)$, или $pV = pV + pdV + Vdp + dpdV$; пренебрегая последним слагаемым, мы получаем дифференциальное уравнение первого порядка с разделяющимися переменными $dp/p = -dV/V$. Учитывая, что изменение объема $dV = Cdt$, уравнение можно проинтегрировать:
$ln(p) - ln(p_0) = -\frac{C}{V}t$ или $\frac{p}{p_(0)} = exp(-\frac{C}{V}t)$ Полученная формула является решением задачи

Программа для решения поставленной задачи:

In [6]:
```
import matplotlib.pyplot as plt
import math
N=20; C=0.01; tmax=50.0; dt=tmax/N ; V=0.1;
dV=0.0005;qV=V/(V+dV); b1=C/V;dt1=V*(1.0/qV-1.0)/C
p0=1.0
print (" q =",qV," b1=",b1," \Delta t=",dt1)
t=[]; p1=[]; p2=[]
t.append(0); p1.append(p0); p2.append(p0)
for i in range(1,N):
    t1=i*dt; t.append(t1)
    p1.append(math.exp(-b1*t1))
    p2.append(qV**(t1/dt1))
plt.bar(t,p1)
plt.xlabel('$t$',fontsize=14)
plt.ylabel('$\eta_1, \eta_2$',fontsize=14)
plt.show()
```

```
 q = 0.9950248756218906  b1= 0.09999999999999999  \Delta t= 0.049999999999998934
```

Рисунок 32 – Задача №2

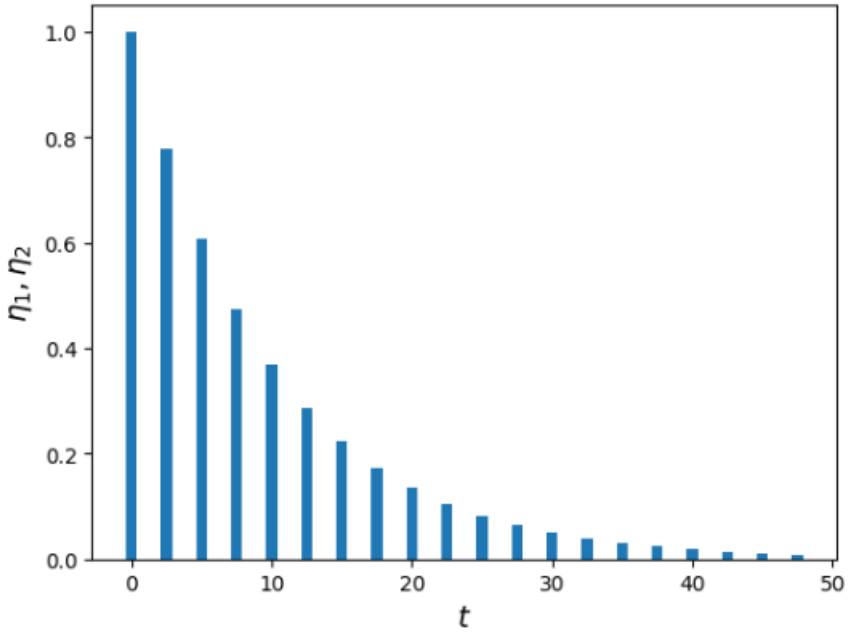


Рисунок 33 – Задача №2

Задание №3: Демонстрация работы с круговой диаграммой

Соотношение людей женского и мужского пола которые встречались с булингом.

In [2]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

In [3]:
```
import csv


with open('bullying.csv', 'r', newline='') as csvfile:
    data = csv.reader(csvfile, delimiter=';')
    values=[]
    male = 0
    female = 0
    for rowdata in data:
        if rowdata[5] == "Female" and (rowdata[1] or rowdata[2] or rowdata[3]) == "Yes":
            female +=1
        elif rowdata[5] == "Male" and (rowdata[1] or rowdata[2] or rowdata[3]) == "Yes":
            male +=1
    values.append(male)
    values.append(female)
    plt.figure(figsize=(30, 30))
    explode = (0.1, 0.15)
    labels = ["Женщины","Мужчины"]
    fig, ax = plt.subplots()
    ax.pie(values, labels=labels, shadow=True, explode=explode)
    plt.show()
```

```
<Figure size 3000x3000 with 0 Axes>
```

Рисунок 34 – Задача №3

Задание №4: Работа с изображениями

```
In [1]: import matplotlib.pyplot as plt
        %matplotlib inline
        from PIL import Image
        import requests
        from io import BytesIO
        plt.figure(figsize=(10, 10))
        response_cat1 = requests.get('https://cdn141.picsart.com/365953143048211.png')
        img_cat1 = Image.open(BytesIO(response_cat1.content))
        plt.imshow(img_cat1)

Out[1]: <matplotlib.figure.AxesImage at 0x26506ef3410>
```

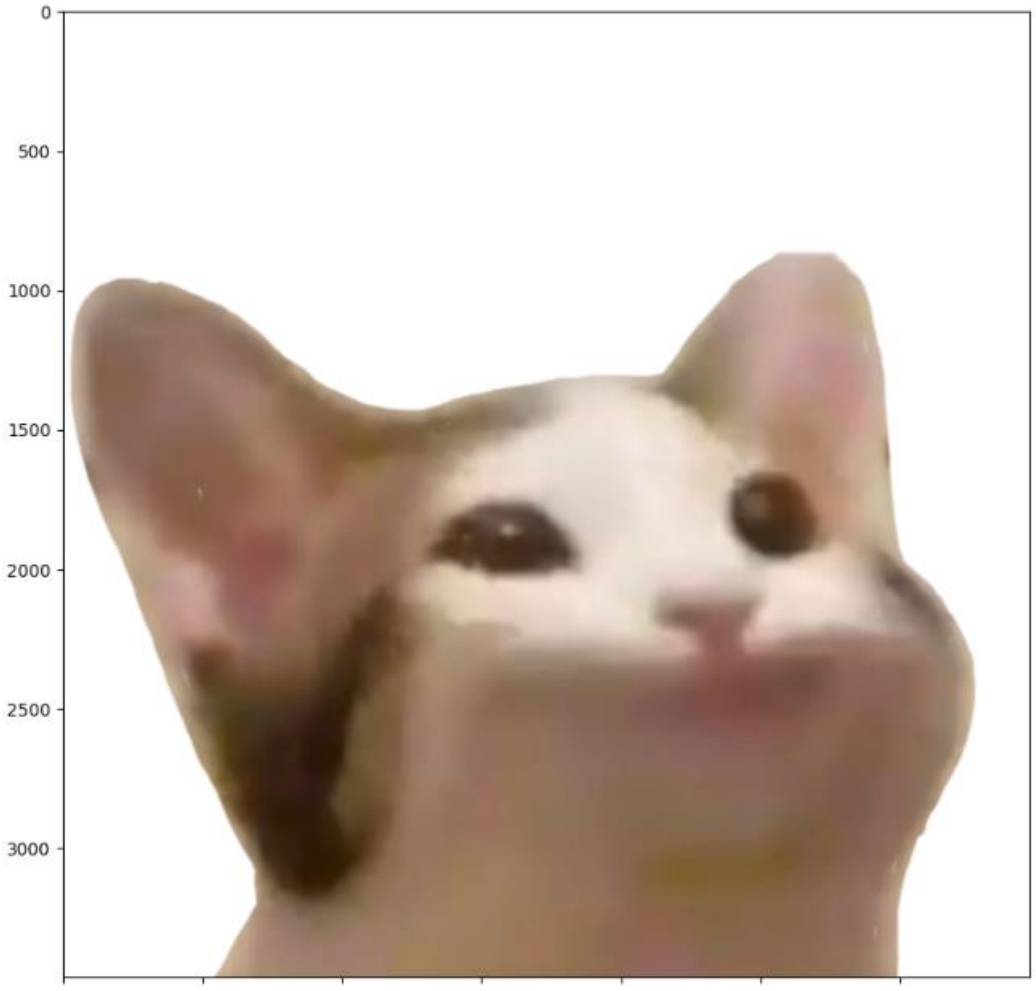


Рисунок 35 – Задача №4

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)
```

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?

```
plt.plot(x, y, c="r")
plt.fill_between(x, y, where=(y > 0))
```

4. Как выполнить двухцветную заливку?

Вариант двухцветной заливки:

```
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?

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

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*.

```
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?

```
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?

```
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?

```
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?

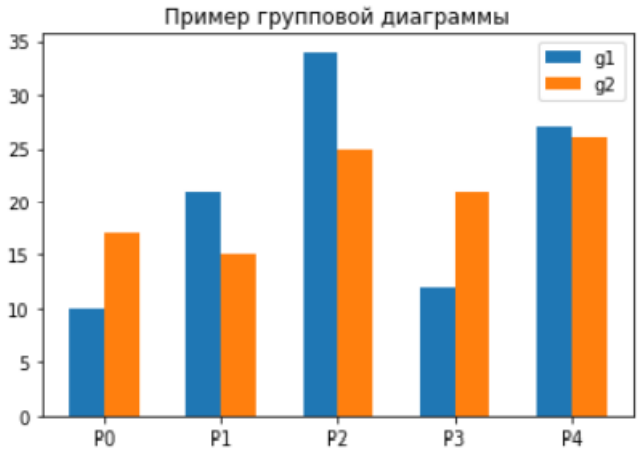
```
x = np.arange(0, 10.5, 0.5)
y = np.cos(x)

plt.scatter(x, y)
```

11. Как осуществляется построение столбчатых диаграмм с помощью matplotlib?

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. Подробное руководство по цветовым картам вы можете найти на официальном сайте *Matplotlib* (https://matplotlib.org/tutorials/colors/colormaps.html#sphx-glr-tutorials-colors-colormaps-py). Также отметим, что такие карты можно создавать самостоятельно, если среди существующих нет подходящего решения. Ниже представлены примеры некоторых цветовых схем, из библиотеки *Matplotlib*.

15. Как отобразить изображение средствами matplotlib?

```
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)
```

16. Как отобразить тепловую карту средствами matplotlib?

```
np.random.seed(123)

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