In [86]:
```
#Biblioteka pandas:
import pandas as pd
#***linijka dodana ponieważ funkcja replace() w przyszłych wersjach będzie 
#a więc aby uniknąć niepotrzebnych warningów ją wkleiłem
pd.set_option('future.no_silent_downcasting', True)
def newExeInfo(number,label=""):
  print("--------"+str(number)+"-----------\n",label)
file ="https://marcingabryel.pl/ai/iris.csv"
dataframe = pd.read_csv(file)
newExeInfo(1,"10 pierwszych wierszy:")
print(dataframe.head(10))
```

```
--------1-----------
 10 pierwszych wierszy:
   sepal.length  sepal.width  petal.length  petal.width variety
0           5.1          3.5           1.4          0.2  Setosa
1           4.9          3.0           1.4          0.2  Setosa
2           4.7          3.2           1.3          0.2  Setosa
3           4.6          3.1           1.5          0.2  Setosa
4           5.0          3.6           1.4          0.2  Setosa
5           5.4          3.9           1.7          0.4  Setosa
6           4.6          3.4           1.4          0.3  Setosa
7           5.0          3.4           1.5          0.2  Setosa
8           4.4          2.9           1.4          0.2  Setosa
9           4.9          3.1           1.5          0.1  Setosa
```

In [87]:
```
#Biblioteka pandas:
import pandas as pd
def newExeInfo(number,label=""):
  print("--------"+str(number)+"-----------\n",label)
file ="https://marcingabryel.pl/ai/iris.csv"
dataframe = pd.read_csv(file)
#2
newExeInfo(2,"Unikalnie wartości kolumny variety:")
print(dataframe['variety'].unique())
```

```
--------2-----------
 Unikalnie wartości kolumny variety:
['Setosa' 'Versicolor' 'Virginica']
```

In [88]:
```
#Biblioteka pandas:
import pandas as pd
def newExeInfo(number,label=""):
  print("--------"+str(number)+"-----------\n",label)
file ="https://marcingabryel.pl/ai/iris.csv"
dataframe = pd.read_csv(file)

newExeInfo(3,"variety = Versicolor")
print(dataframe[dataframe['variety'] == "Versicolor"])
```

```
---------3-----------
 variety = Versicolor
    sepal.length  sepal.width  petal.length  petal.width     variety
50           7.0          3.2           4.7          1.4  Versicolor
51           6.4          3.2           4.5          1.5  Versicolor
52           6.9          3.1           4.9          1.5  Versicolor
53           5.5          2.3           4.0          1.3  Versicolor
54           6.5          2.8           4.6          1.5  Versicolor
55           5.7          2.8           4.5          1.3  Versicolor
56           6.3          3.3           4.7          1.6  Versicolor
57           4.9          2.4           3.3          1.0  Versicolor
58           6.6          2.9           4.6          1.3  Versicolor
59           5.2          2.7           3.9          1.4  Versicolor
60           5.0          2.0           3.5          1.0  Versicolor
61           5.9          3.0           4.2          1.5  Versicolor
62           6.0          2.2           4.0          1.0  Versicolor
63           6.1          2.9           4.7          1.4  Versicolor
64           5.6          2.9           3.6          1.3  Versicolor
65           6.7          3.1           4.4          1.4  Versicolor
66           5.6          3.0           4.5          1.5  Versicolor
67           5.8          2.7           4.1          1.0  Versicolor
68           6.2          2.2           4.5          1.5  Versicolor
69           5.6          2.5           3.9          1.1  Versicolor
70           5.9          3.2           4.8          1.8  Versicolor
71           6.1          2.8           4.0          1.3  Versicolor
72           6.3          2.5           4.9          1.5  Versicolor
73           6.1          2.8           4.7          1.2  Versicolor
74           6.4          2.9           4.3          1.3  Versicolor
75           6.6          3.0           4.4          1.4  Versicolor
76           6.8          2.8           4.8          1.4  Versicolor
77           6.7          3.0           5.0          1.7  Versicolor
78           6.0          2.9           4.5          1.5  Versicolor
79           5.7          2.6           3.5          1.0  Versicolor
80           5.5          2.4           3.8          1.1  Versicolor
81           5.5          2.4           3.7          1.0  Versicolor
82           5.8          2.7           3.9          1.2  Versicolor
83           6.0          2.7           5.1          1.6  Versicolor
84           5.4          3.0           4.5          1.5  Versicolor
85           6.0          3.4           4.5          1.6  Versicolor
86           6.7          3.1           4.7          1.5  Versicolor
87           6.3          2.3           4.4          1.3  Versicolor
88           5.6          3.0           4.1          1.3  Versicolor
89           5.5          2.5           4.0          1.3  Versicolor
90           5.5          2.6           4.4          1.2  Versicolor
91           6.1          3.0           4.6          1.4  Versicolor
92           5.8          2.6           4.0          1.2  Versicolor
93           5.0          2.3           3.3          1.0  Versicolor
94           5.6          2.7           4.2          1.3  Versicolor
95           5.7          3.0           4.2          1.2  Versicolor
96           5.7          2.9           4.2          1.3  Versicolor
97           6.2          2.9           4.3          1.3  Versicolor
98           5.1          2.5           3.0          1.1  Versicolor
99           5.7          2.8           4.1          1.3  Versicolor
```

In [89]:
```python
#Biblioteka pandas:
import pandas as pd
def newExeInfo(number,label=""):
  print("---------"+str(number)+"-----------\n",label)
file ="https://marcingabryel.pl/ai/iris.csv"
dataframe = pd.read_csv(file)

newExeInfo(4,"Po 5 wierszy dla każdej unikalnej wartości variety")
for value in dataframe["variety"].unique():
  print(dataframe[dataframe["variety"] == str(value)].head(5))
```

```
---------4-----------
 Po 5 wierszy dla każdej unikalnej wartości variety
   sepal.length  sepal.width  petal.length  petal.width variety
0           5.1          3.5           1.4          0.2  Setosa
1           4.9          3.0           1.4          0.2  Setosa
2           4.7          3.2           1.3          0.2  Setosa
3           4.6          3.1           1.5          0.2  Setosa
4           5.0          3.6           1.4          0.2  Setosa
    sepal.length  sepal.width  petal.length  petal.width     variety
50           7.0          3.2           4.7          1.4  Versicolor
51           6.4          3.2           4.5          1.5  Versicolor
52           6.9          3.1           4.9          1.5  Versicolor
53           5.5          2.3           4.0          1.3  Versicolor
54           6.5          2.8           4.6          1.5  Versicolor
     sepal.length  sepal.width  petal.length  petal.width    variety
100           6.3          3.3           6.0          2.5  Virginica
101           5.8          2.7           5.1          1.9  Virginica
102           7.1          3.0           5.9          2.1  Virginica
103           6.3          2.9           5.6          1.8  Virginica
104           6.5          3.0           5.8          2.2  Virginica
```

In [90]:
```
#Biblioteka pandas:
import pandas as pd
def newExeInfo(number,label=""):
  print("---------"+str(number)+"-----------\n",label)
file ="https://marcingabryel.pl/ai/iris.csv"
dataframe = pd.read_csv(file)
newExeInfo(5,"dane statystyczne:")
for value in dataframe:
  if value == "variety": break;
  print("min of",value +":", dataframe[value].min())
  print("max of",value +":", dataframe[value].max())
  print("mean of",value +":", dataframe[value].mean(),"\n")
```

```
---------5-----------
 dane statystyczne:
min of sepal.length: 4.3
max of sepal.length: 7.9
mean of sepal.length: 5.843333333333334

min of sepal.width: 2.0
max of sepal.width: 4.4
mean of sepal.width: 3.0573333333333337

min of petal.length: 1.0
max of petal.length: 6.9
mean of petal.length: 3.7580000000000005

min of petal.width: 0.1
max of petal.width: 2.5
mean of petal.width: 1.1993333333333336
```

In [91]:
```
#Biblioteka pandas:
import pandas as pd
#***linijka dodana ponieważ funkcja replace() w przyszłych wersjach będzie u
#a więc aby uniknąć niepotrzebnych warningów ją wkleiłem
pd.set_option('future.no_silent_downcasting', True)
def newExeInfo(number,label=""):
  print("---------"+str(number)+"-----------\n",label)
file ="https://marcingabryel.pl/ai/iris.csv"
dataframe = pd.read_csv(file)

newExeInfo(6,"nowa kolumna y:")
```

```
#dodanie nowej kolumny 'y' z wartościami z kolumny "variety"
dataframe['y'] = dataframe["variety"]
#iteracja przez wszystkie unikalne wartości z kolumny 'y'
for value in dataframe['y'].unique():
  print(value)
  #zastąpienie wartości na 1.0 i 0.0
  if value == "Setosa":
    dataframe['y'] = dataframe['y'].replace(value, 1.0).infer_objects(copy=I
    continue
  dataframe['y'] = dataframe['y'].replace(value,0.0).infer_objects(copy=Fals
for value in dataframe["variety"].unique():
  print(dataframe[dataframe["variety"] == str(value)].head(5))
```

```
--------6-----------
 nowa kolumna y:
Setosa
Versicolor
Virginica
   sepal.length  sepal.width  petal.length  petal.width variety    y
0           5.1          3.5           1.4          0.2  Setosa  1.0
1           4.9          3.0           1.4          0.2  Setosa  1.0
2           4.7          3.2           1.3          0.2  Setosa  1.0
3           4.6          3.1           1.5          0.2  Setosa  1.0
4           5.0          3.6           1.4          0.2  Setosa  1.0
    sepal.length  sepal.width  petal.length  petal.width     variety    y
50           7.0          3.2           4.7          1.4  Versicolor  0.0
51           6.4          3.2           4.5          1.5  Versicolor  0.0
52           6.9          3.1           4.9          1.5  Versicolor  0.0
53           5.5          2.3           4.0          1.3  Versicolor  0.0
54           6.5          2.8           4.6          1.5  Versicolor  0.0
     sepal.length  sepal.width  petal.length  petal.width    variety    y
100           6.3          3.3           6.0          2.5  Virginica  0.0
101           5.8          2.7           5.1          1.9  Virginica  0.0
102           7.1          3.0           5.9          2.1  Virginica  0.0
103           6.3          2.9           5.6          1.8  Virginica  0.0
104           6.5          3.0           5.8          2.2  Virginica  0.0
```

In [92]:
```
#Biblioteka pandas:
import pandas as pd
def newExeInfo(number,label=""):
  print("--------"+str(number)+"-----------\n",label)
file ="https://marcingabryel.pl/ai/iris.csv"
dataframe = pd.read_csv(file)
newExeInfo(7,"nowy obiekt DataFrame X")
#utworzenie pustego obiektu X typu DataFrame
X = pd.DataFrame()
#domyślam się, że poprostu napisanie tego miało od ręki a nie w pętli miało
#złożoność O(1) a wykorzystanie pętli daje nam złożoność O(n)
#ale for study reasons chciałem takiego podejścia zastosować z myślą o poter
#w przyszłości pracy na większych zbiorach gdzie katorgą byłoby pisać
#100 linijek oddzielnie
for value in dataframe:
  if(value != "sepal.length" and value != "sepal.width" and value != "petal.
  X[value] = dataframe[value]
print(X.head(5))
```

```
--------7-----------
 nowy obiekt DataFrame X
   sepal.length  sepal.width  petal.length  petal.width
0           5.1          3.5           1.4          0.2
1           4.9          3.0           1.4          0.2
2           4.7          3.2           1.3          0.2
3           4.6          3.1           1.5          0.2
4           5.0          3.6           1.4          0.2
```

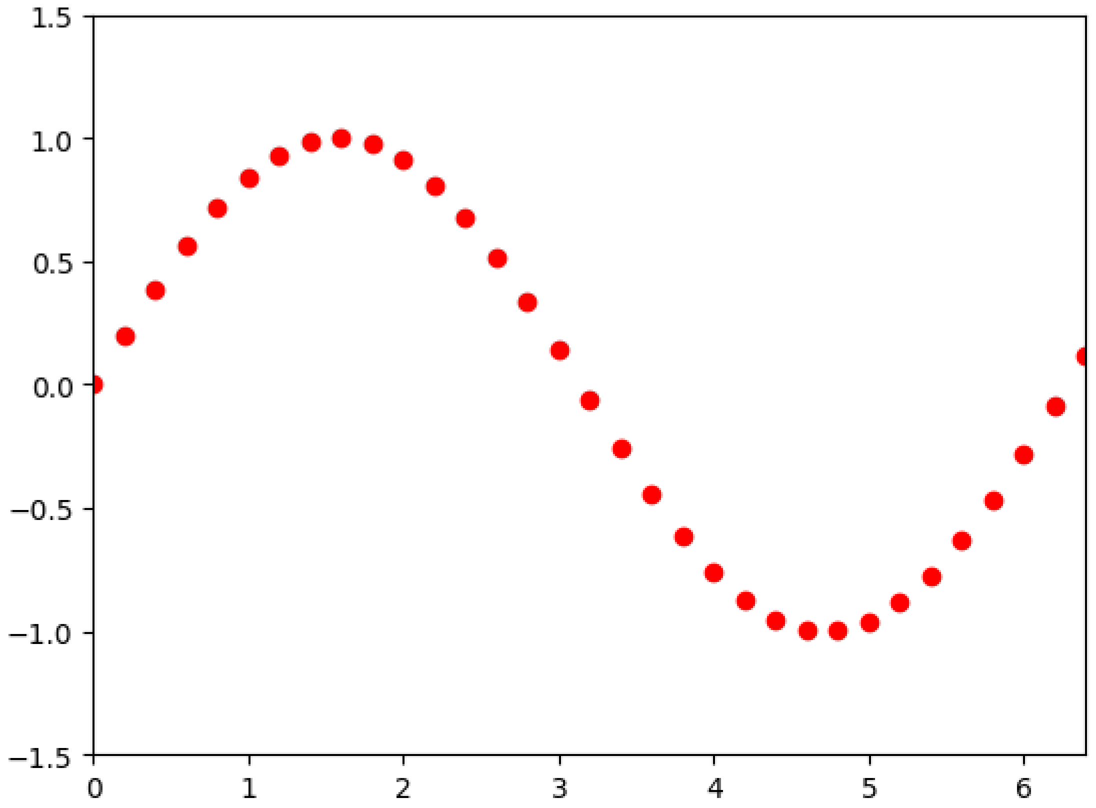
In [94]:
```
#Biblioteka pandas:
import pandas as pd
#***linijka dodana ponieważ funkcja replace() w przyszłych wersjach będzie u
#a więc aby uniknąć niepotrzebnych warningów ją wkleiłem
pd.set_option('future.no_silent_downcasting', True)
def newExeInfo(number,label=""):
  print("--------"+str(number)+"-----------\n",label)
file ="https://marcingabryel.pl/ai/iris.csv"
dataframe = pd.read_csv(file)
#--- z zadania 6 ---------
#dodanie nowej kolumny 'y' z wartościami z kolumny "variety"
dataframe['y'] = dataframe["variety"]
#iteracja przez wszystkie unikalne wartości z kolumny 'y'
for value in dataframe['y'].unique():
  print(value)
  #zastąpienie wartości na 1.0 i 0.0
  if value == "Setosa":
    dataframe['y'] = dataframe['y'].replace(value, 1.0).infer_objects(copy=F
    continue
  dataframe['y'] = dataframe['y'].replace(value,0.0).infer_objects(copy=Fals
#-------------------------
newExeInfo(8,"nowy obiekt DataFrame Y")
Y = pd.DataFrame()
Y['y'] = dataframe['y']
print(Y.head(5))
```

```
Setosa
Versicolor
Virginica
--------8-----------
 nowy obiekt DataFrame Y
     y
0  1.0
1  1.0
2  1.0
3  1.0
4  1.0
```

Muszę przynać, że pozytywnie zaskoczyła mnie intuicyjność biblioteki pandas.

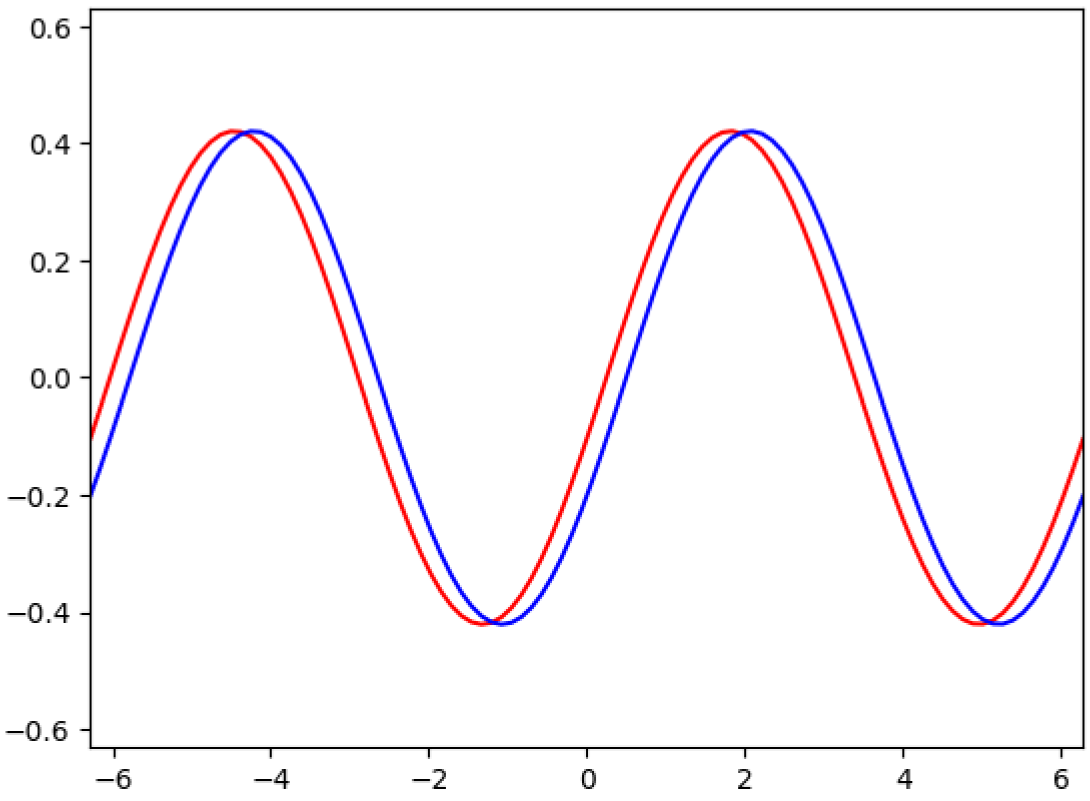
In [83]:
```
#Biblioteka matplotlib
import matplotlib.pyplot as plt
import numpy as np
def newExeInfo(number,label=""):
  print("--------"+str(number)+"-----------\n",label)

newExeInfo(1,"wykres sin(x) z punktów")
x1 = np.linspace(0,6.40,33)
y1 = np.sin(x1)
plt.plot(x1, y1, "ro")
plt.axis([0,6.4,-1.5,1.5])
plt.show()
```

```
--------1-----------
 wykres sin(x) z punktów
```



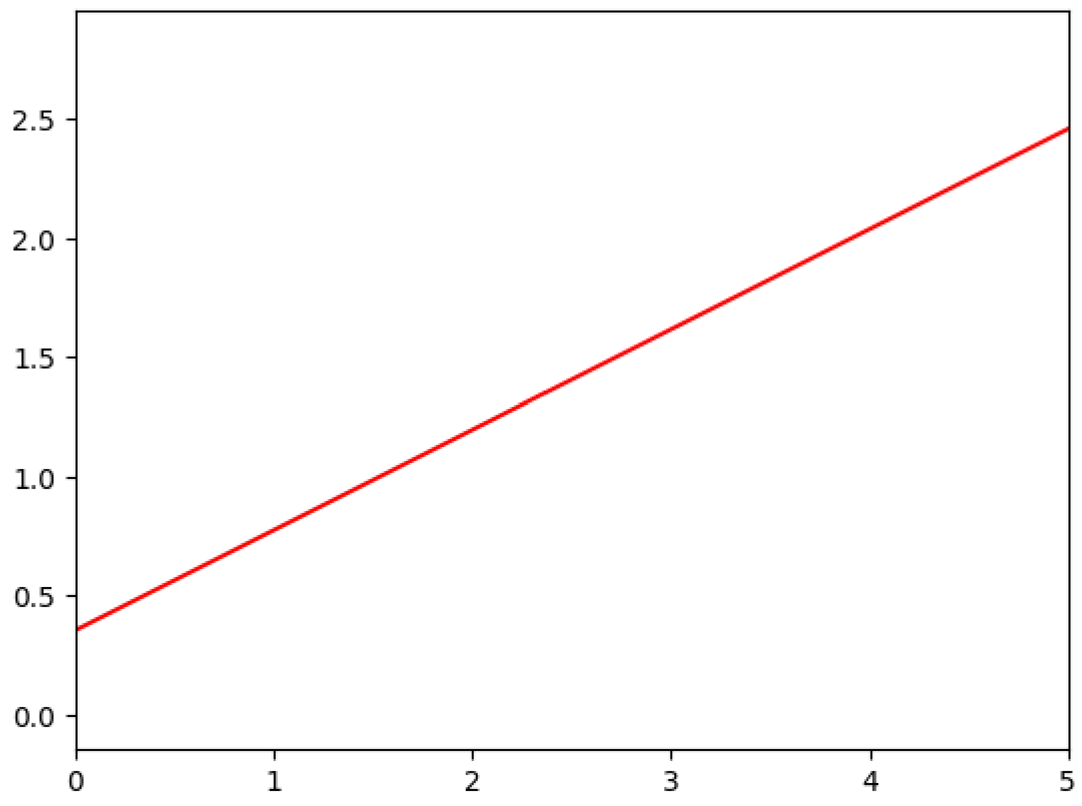
In [84]:
```
#Biblioteka matplotlib
import matplotlib.pyplot as plt
import numpy as np
def newExeInfo(number,label=""):
  print("---------"+str(number)+"-----------\n",label)

newExeInfo(2,"Dwa wykresy")
t = np.linspace(-2*np.pi,2*np.pi,100)
a = np.random.random()
print("a =",a)
plt.plot(t, a*np.sin(t-0.25), "r-", t, a*np.sin(t-0.5), "b-")
# automatyczne skalowanie osi y w celu uniknięcia widoku prostej lini
# przy bardzo małej wartości a
plt.axis([-2*np.pi, 2*np.pi, -1.5*a,1.5*a])
plt.show()
```

```
--------2-----------
 Dwa wykresy
a = 0.42059322223401774
```



```
In [85]: #Biblioteka matplotlib
         import matplotlib.pyplot as plt
         import numpy as np
         def newExeInfo(number,label=""):
           print("--------"+str(number)+"-----------\n",label)


         newExeInfo(3, "losowe wartosci a i b:")
         x3 = np.linspace(0,5,70)
         a3 = np.random.random()
         b3 = np.random.random()
         y3 = a*x3+b3
         plt.plot(x3, y3,"r-")
         plt.axis([0,5,min(y3)-0.5, max(y3)+0.5])
         plt.show()
```

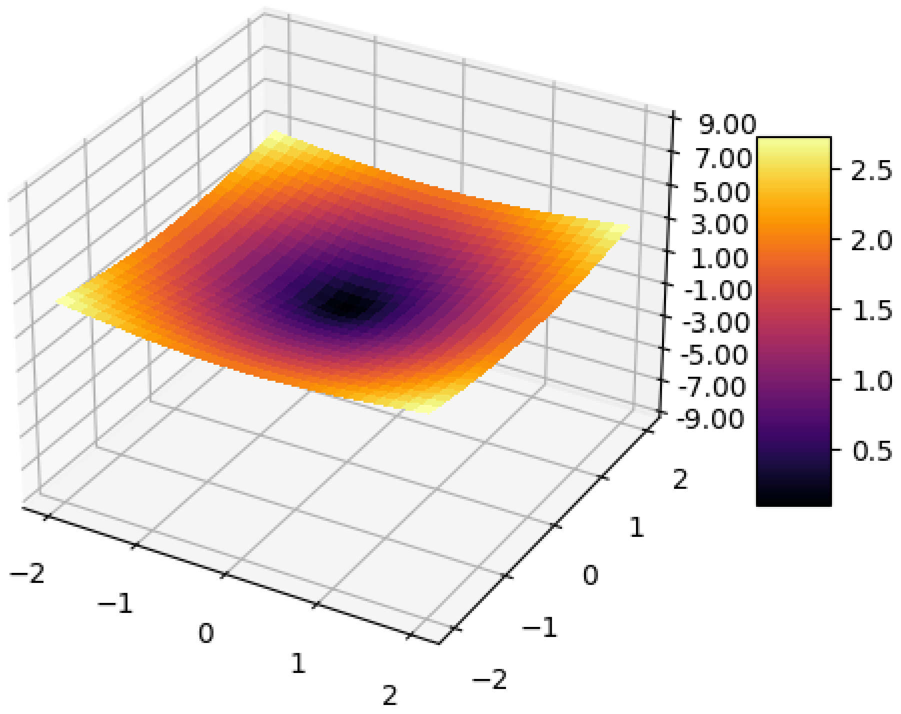
```
--------3-----------
 losowe wartosci a i b:
```



In [104…

```
#Biblioteka matplotlib
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
import numpy as np
import math

print("---------"+str(4)+"-----------\n","Powierzchnia z = sqrt(x^2 + y^2)")

fig = plt.figure()
#ax = fig.gca(projection='3d)
ax = fig.add_subplot(projection='3d')

X = np.linspace(-2,2,30)
Y = np.linspace(-2,2,30)
X, Y = np.meshgrid(X,Y)
Z = np.sqrt(X ** 2 + Y ** 2)

surf = ax.plot_surface(X,Y,Z, cmap=cm.inferno, linewidth=0, antialiased=Fals
ax.set_zlim(-9,9)
ax.zaxis.set_major_locator(LinearLocator(10))
ax.zaxis.set_major_formatter(FormatStrFormatter('%.02f'))
fig.colorbar(surf,shrink=0.5,aspect=5)
plt.show()
```

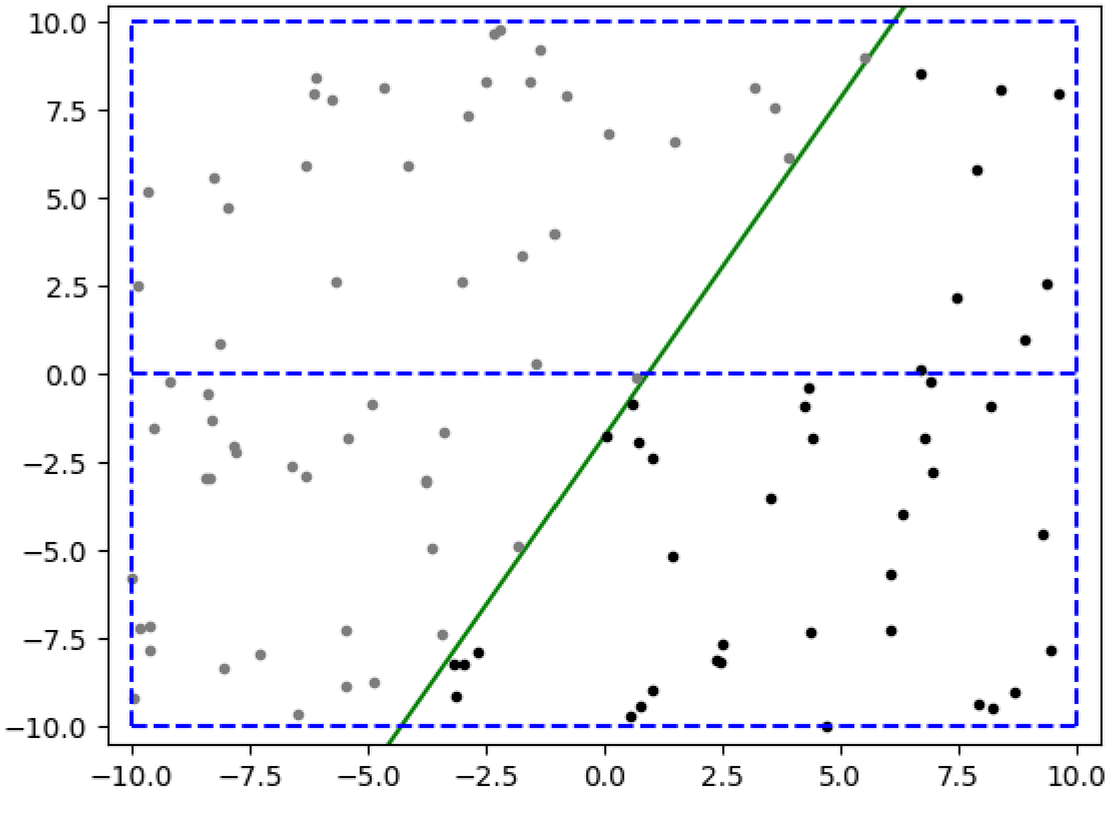
```
---------4-----------
 Powierzchnia z = sqrt(x^2 + y^2)
```



In [153…

```
import matplotlib.pyplot as plt
import numpy as np

#tablica p 100 punktów x,y z zakresu <-10,10>
N = 100
p = np.random.random((N,2)) * 20 -10
print("p:\n",p)

#prosta:
x = np.linspace(-10.5,10.5,2)
n = 10
a = np.random.random() * 4 - 2
b = a * (np.random.random()*(2*n)-n) *(-1)
y = a * x + b
plt.plot(x,y,"g-")
#punkty:
for punkt in p:
  if punkt[1] > a*punkt[0] +b:
    plt.plot(punkt[0],punkt[1], '.', color="grey")
  else:
    plt.plot(punkt[0],punkt[1], '.', color="black")
#linia sprawdzająca czy prosta przecina oś x w zadanym przedziale:
plt.plot(np.linspace(-10,10,2),np.full(2,0), "b--")
#ramka:
plt.plot(np.full(100, -10), np.linspace(-10,10,100), "b--")
plt.plot(np.full(100,  10), np.linspace(-10,10,100), "b--")
plt.plot(np.linspace(-10,10,100), np.full(100, -10), "b--")
plt.plot(np.linspace(-10,10,100), np.full(100,  10), "b--")

plt.axis([-10.5,10.5,-10.5,10.5])
plt.show()
```

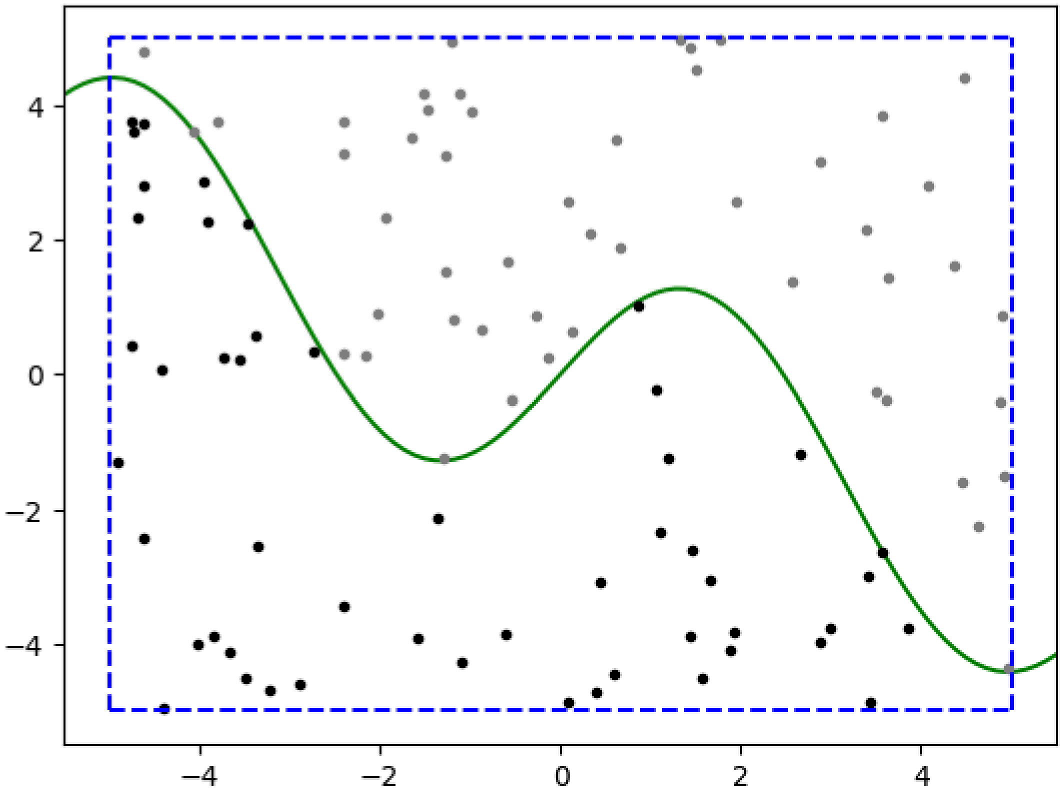
```
p:
 [[ 9.44615636 -7.84026271]
 [-6.09197214  8.45423569]
 [-2.32782913  9.68887261]
 [ 0.77645445 -9.38802814]
 [-5.66411075  2.62136914]
 [ 4.39606517 -1.81482161]
 [ 8.8981875   1.01345427]
 [-3.39908355 -1.59410122]
 [-3.17766355 -8.19405741]
 [-9.65060941  5.18873738]
 [-9.20631949 -0.17325372]
 [-8.28027853  5.5740056 ]
 [-6.15712182  7.96802928]
 [ 1.41891915 -5.14966017]
 [ 6.29334118 -3.93409562]
 [ 4.21345023 -0.89885352]
 [ 3.50472618 -3.48447481]
 [-9.87862886  2.50796439]
 [ 6.87822582 -0.17556826]
 [-7.29119121 -7.90416071]
 [-1.08086753  3.98795476]
 [ 3.15321327  8.13890488]
 [-1.7301783   3.35546932]
 [ 3.88225401  6.1676085 ]
 [-8.30019602 -1.2799683 ]
 [-5.76722495  7.81628004]
 [-2.69018915 -7.85074998]
 [-4.86555572 -8.71223205]
 [-7.83237826 -2.01423915]
 [ 6.05013308 -5.64387969]
 [ 8.37642431  8.0975998 ]
 [-2.97868042 -8.1976629 ]
 [-3.12278387 -9.11502424]
 [ 8.15126322 -0.90294798]
 [-1.83913524 -4.85331166]
 [-2.89684681  7.36372156]
 [-8.41842887 -2.91267753]
 [ 3.60686432  7.5857126 ]
 [ 8.20358578 -9.48735132]
 [ 4.30498916 -0.39373461]
 [ 1.01791228 -8.96077551]
 [ 9.35148398  2.5614664 ]
 [-8.14964642  0.89512669]
 [-3.75547973 -3.0349556 ]
 [ 0.08838888  6.82504832]
 [-9.59856126 -7.11638926]
 [ 0.05262883 -1.71799367]
 [ 0.58113415 -0.84262057]
 [-3.75390099 -2.9953112 ]
 [-8.35135756 -2.95374143]
 [-6.31567798  5.92927873]
 [ 4.37187819 -7.29031025]
 [ 7.9095681  -9.33128324]
 [-4.66930646  8.12551839]
 [-1.57457771  8.34665866]
 [ 2.45802667 -8.17775114]
 [-0.80962534  7.9377644 ]
 [ 5.49752305  9.00423478]
 [-6.46369884 -9.6450462 ]
 [-9.61996047 -7.82806269]
 [-5.41669233 -1.81456449]
 [ 0.7227319  -1.92553724]
 [-9.9592307  -9.15743195]
 [ 7.89146517  5.82792759]
 [ 2.51391691 -7.65485473]
 [ 6.93967903 -2.75609858]
 [-3.01008436  2.64772522]
 [ 8.6857026  -9.02510892]
 [ 6.04067491 -7.2415781 ]
 [ 2.3587664  -8.09261888]
 [ 1.47495439  6.60537122]
 [-3.44181537 -7.35498898]
 [ 9.2582342  -4.51489404]
 [-9.51232871 -1.494271  ]
 [-2.22328627  9.79588253]
 [-1.35352256  9.23337544]
 [ 4.70967695 -9.96258596]
 [-9.81643846 -7.20413396]
 [ 6.69970893  8.53853187]
 [ 6.75043201 -1.7777035 ]
 [-4.91640078 -0.82002485]
 [-4.13858694  5.96442908]
 [ 7.44544041  2.18905255]
 [-9.98532874 -5.76609087]
 [-5.48526348 -8.8557252 ]
 [-3.65292608 -4.93277644]
 [-8.03253222 -8.31651264]
 [ 0.53286761 -9.7100283 ]
 [-6.31058651 -2.85369228]
 [-7.95608899  4.73468907]
 [-7.80386729 -2.16836353]
 [-2.48452985  8.30069277]
 [-6.62125312 -2.57497318]
 [-1.45860235  0.28586941]
 [ 1.00234544 -2.33669392]
 [ 9.62296223  7.99042293]
 [-8.38453204 -0.55330024]
 [ 0.67863342 -0.10450073]
 [-5.48375298 -7.26692797]
 [ 6.6876503   0.16772385]]
```



In [186…

```
import numpy as np
import matplotlib.pyplot as plt
n = 5
p = np.random.random((100,2)) * (2*n) -n
print("p:\n",p)

#prosta:

x = np.linspace(-n-0.5,n+0.5,100)
y = 2* np.sin(x) - x/2
plt.plot(x,y,"g-")
#d dla punktów:
d = np.zeros(100)
for i in range(len(d)):
  if p[i,1] > 2* np.sin(p[i,0]) - p[i,0]/2:
    d[i] = 1
  else:
    d[i] = 0
print("d:\n",d)
#punkty:
print("len:",len(p))
for i in range(len(p)):
  if d[i] == 1:
    plt.plot(p[i,0],p[i,1],'.',color='grey')
  else:
    plt.plot(p[i,0],p[i,1],'.',color='black')
#ramka:
plt.plot(np.full(2, -n), np.linspace(-n,n,2), "b--")
plt.plot(np.full(2,  n), np.linspace(-n,n,2), "b--")
plt.plot(np.linspace(-n,n,2), np.full(2, -n), "b--")
plt.plot(np.linspace(-n,n,2), np.full(2,  n), "b--")
plt.axis([-n-0.5,n+0.5,-n-0.5,n+0.5])
plt.show()
```

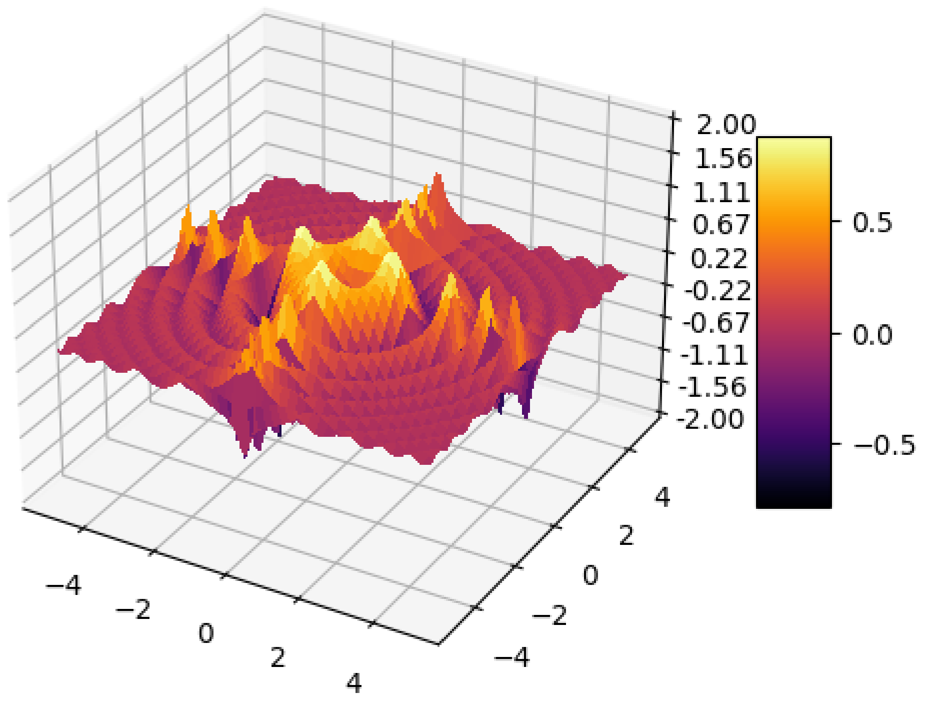
```
p:
 [[-2.39314609  3.76300228]
 [ 3.63986039  1.42910824]
 [-3.89918379  2.28721829]
 [-4.42590388  0.08492872]
 [-3.38266867  0.57561026]
 [-4.74183529  3.77452054]
 [-4.39800122 -4.95751959]
 [-1.64637502  3.51662972]
 [ 4.88690162 -0.38895314]
 [ 2.87141849 -3.98810743]
 [ 1.95710125  2.58381806]
 [-4.05680131  3.61817442]
 [ 0.12294075  0.6470032 ]
 [ 3.57656341  3.84713134]
 [-2.40628121  3.27465472]
 [-3.8032703   3.76535384]
 [ 0.33294403  2.10945122]
 [ 3.3905917   2.16493796]
 [-1.0860873  -4.26934098]
 [ 1.1119568  -2.33801079]
 [ 0.08265692 -4.86318855]
 [ 3.55837545 -2.64844446]
 [-2.15073294  0.29253259]
 [-1.47515403  3.95295832]
 [ 4.97066934 -4.36808557]
 [ 0.45124076 -3.07421963]
 [ 1.76380517  4.99819214]
 [-1.36144234 -2.11822746]
 [ 0.59308334 -4.45535517]
 [ 1.06755692 -0.23448511]
 [ 4.48488436  4.42429092]
 [ 4.92285818 -1.49563448]
 [-1.26420198  3.25938407]
 [-3.45648059  2.25420396]
 [-3.5501548   0.22678214]
 [ 4.08915666  2.82661313]
 [-1.5687644  -3.92886622]
 [-0.60922455 -3.85444786]
 [-1.16813749  0.83156508]
 [ 2.65558291 -1.1813138 ]
 [-3.48313761 -4.52163559]
 [ 0.09003481  2.58554357]
 [ 3.60815743 -0.36248194]
 [-1.10435453  4.17687361]
 [-0.52710926 -0.36527281]
 [-0.25861262  0.88411696]
 [ 2.98316335 -3.75082308]
 [-4.61767056  3.73270325]
 [-4.62345202  4.80285131]
 [-3.95890728  2.85806358]
 [-0.13682509  0.25677332]
 [-0.87658318  0.67956869]
 [ 4.89293102  0.86678883]
 [-0.98620768  3.90041821]
 [ 1.19898887 -1.23130911]
 [ 4.36519531  1.62504584]
 [ 1.51240089  4.53737694]
 [-4.01032783 -3.99493967]
 [-2.72130698  0.33250619]
 [-1.26607339  1.52520983]
 [-1.93811663  2.32220917]
 [-4.68715745  2.33747208]
 [ 1.33950243  4.98759591]
 [ 1.56988234 -4.51191492]
 [-3.357341   -2.55973164]
 [-0.58526071  1.6882714 ]
 [ 4.44984544 -1.58830582]
 [-4.75502259  0.44272564]
 [-3.7387314   0.2604926 ]
 [-2.404443    0.31751455]
 [-3.84346661 -3.88351515]
 [ 2.56489852  1.37083617]
 [-4.91181167 -1.29597389]
 [ 0.86118347  1.0257672 ]
 [ 0.62765803  3.49928323]
 [-3.6648509  -4.11401717]
 [ 1.43433506  4.86695644]
 [-1.28440687 -1.22437546]
 [-1.20681644  4.9580646 ]
 [-1.51990345  4.17848101]
 [ 4.63941587 -2.23733531]
 [ 1.87547643 -4.09474609]
 [ 2.89034848  3.16215599]
 [ 1.431714   -3.88984808]
 [ 0.40589963 -4.72320859]
 [-4.62181568 -2.42107241]
 [-2.39416259 -3.42714102]
 [ 3.43845586 -4.8560532 ]
 [-2.02295845  0.89365982]
 [ 0.67238952  1.88263638]
 [ 1.93818547 -3.83354242]
 [ 3.50434491 -0.25785511]
 [ 3.8585924  -3.76987192]
 [-3.21913842 -4.69860305]
 [-4.73630936  3.61723678]
 [ 3.42272757 -2.99950429]
 [ 1.46228798 -2.60844168]
 [-4.6171088   2.80450466]
 [ 1.66103433 -3.04107566]
 [-2.89037818 -4.59909691]]
d:
 [1. 1. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 0. 0. 1. 1.
 1. 0. 1. 0. 0. 0. 1. 1. 1. 0. 0. 1. 0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 0. 0.
 1. 0. 1. 1. 1. 1. 0. 1. 1. 0. 0. 1. 1. 0. 1. 0. 0. 1. 1. 0. 0. 1. 0. 1.
 0. 0. 1. 0. 1. 1. 1. 1. 1. 0. 1. 0. 0. 0. 0. 0. 1. 1. 0. 1. 0. 0. 0. 0.
 0. 0. 0. 0.]
len: 100
```



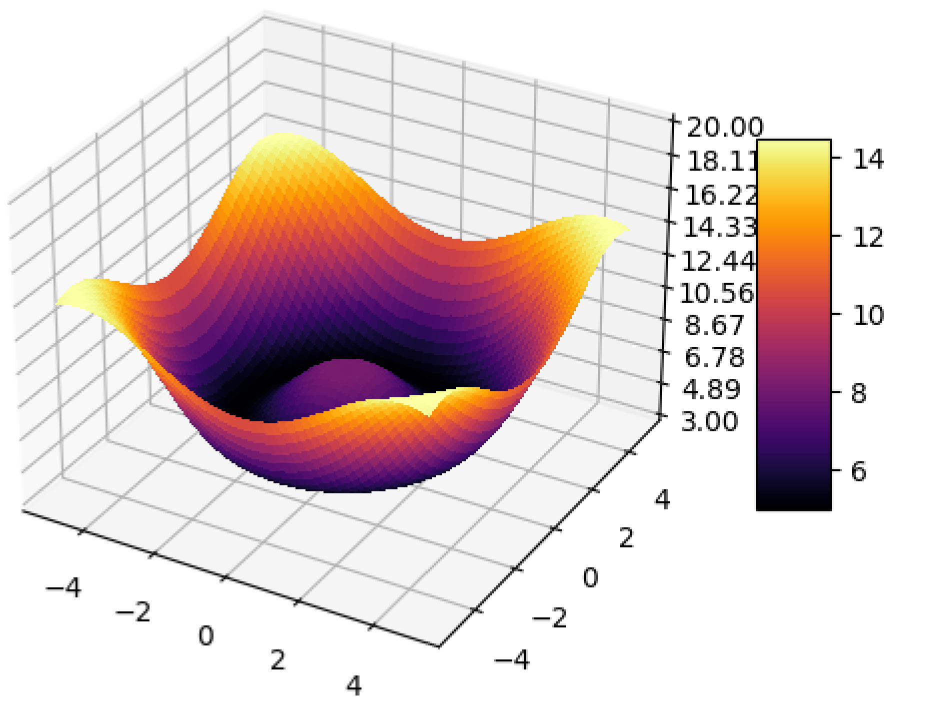
In [192…

```
#Wykresy 3D
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
import numpy as np
import math



#dane:
X = np.linspace(-5,5,100)
Y = np.linspace(-5,5,100)
X, Y = np.meshgrid(X,Y)
Z1 = (np.sin(X ** 2 + Y ** 2))/(abs(X * Y) + 1)
Z2 = np.sqrt(X ** 2 + Y ** 2 ) + 3 * np.cos(np.sqrt(X ** 2 + Y ** 2)) + 5
#plot Z1:
print("Wykres 3D nr 1:")
fig = plt.figure()
ax = fig.add_subplot(projection='3d')
surf = ax.plot_surface(X,Y,Z1, cmap=cm.inferno, linewidth=0, antialiased=Fal
ax.set_zlim(-2,2)
ax.zaxis.set_major_locator(LinearLocator(10))
ax.zaxis.set_major_formatter(FormatStrFormatter('%.02f'))
fig.colorbar(surf,shrink=0.5,aspect=5)
plt.show()
#plot Z2:
print("Wykres 3D nr 2:")
fig1 = plt.figure()
ax1 = fig1.add_subplot(projection="3d")
surf1 = ax1.plot_surface(X,Y,Z2, cmap=cm.inferno, linewidth=0, antialiased=
ax1.set_zlim(3,20)
ax1.zaxis.set_major_locator(LinearLocator(10))
ax1.zaxis.set_major_formatter(FormatStrFormatter('%.02f'))
fig1.colorbar(surf1,shrink=0.5,aspect=5)
plt.show()
```

```
Wykres 3D nr 1:
```



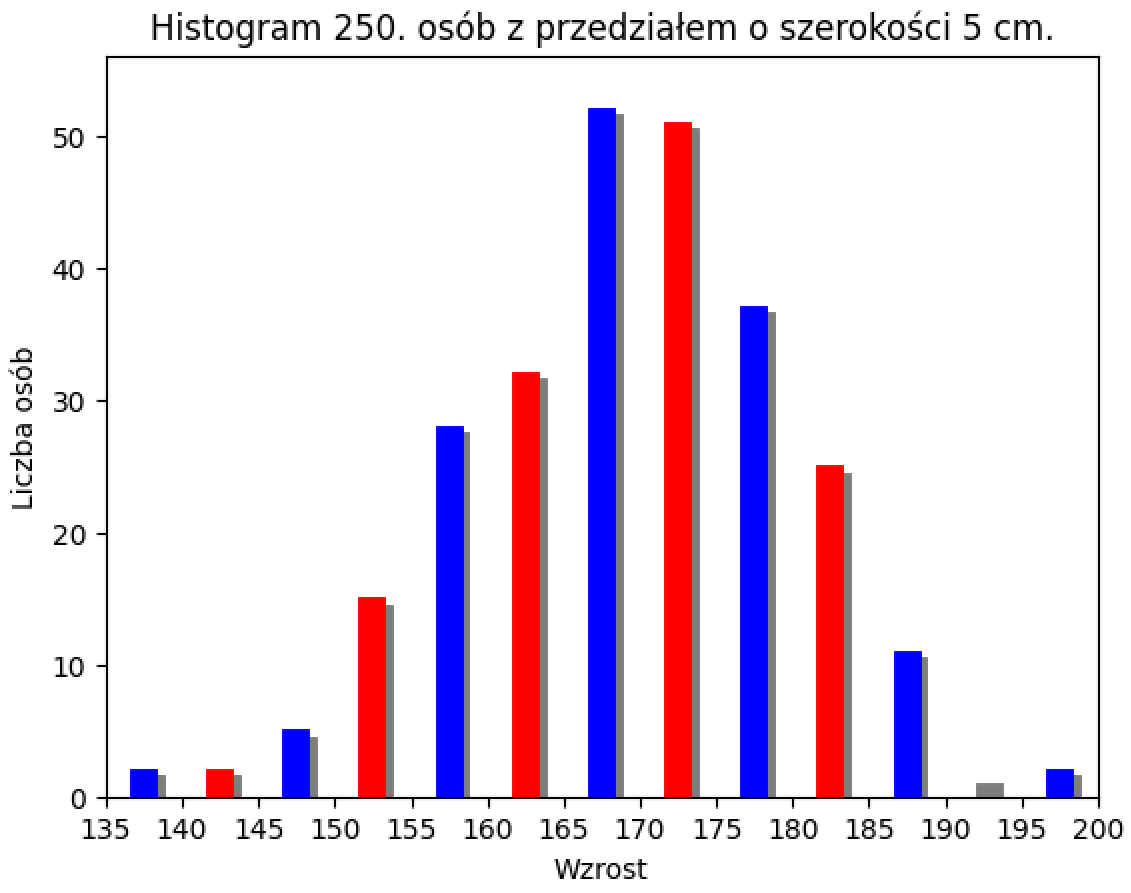
```
Wykres 3D nr 2:
```



In [263…

```python
#Histogram
import numpy as np
import matplotlib.pyplot as plt


#x - tablica wartości
#n - szerokość przedziału np. dla przedziału 120 - 125 szerokość wynosi 5
def HistogramWzrostu(x,n):
  shadowOffset= n/10
  startingValueOfFirstInterval = (min(x)//n)*n
  endingValueOfLastInterval = (max(x)//n +1)*n
  countOfIntervals = (endingValueOfLastInterval - startingValueOfFirstInter
  y = np.zeros(int(countOfIntervals))
  for val in x:
    ind = int((val-startingValueOfFirstInterval)//n)
    y[ind] += 1
  for i in range(len(y)):
    xPosition = startingValueOfFirstInterval + i*n
    xPosition += n/2
    plt.plot(np.full(2,xPosition+shadowOffset), [0,y[i]-shadowOffset], '-',
    if i%2 == 0:
      plt.plot(np.full(2,xPosition), [0,y[i]], 'b-',linewidth=10 )
    else:
      plt.plot(np.full(2,xPosition), [0,y[i]], 'r-',linewidth=10)
  plt.title("Histogram "+str(int(len(x)))+". osób z przedziałem o szerokośc
  plt.ylabel("Liczba osób")
  plt.xlabel("Wzrost")
  plt.axis([startingValueOfFirstInterval,endingValueOfLastInterval,0,max(y)
  plt.xticks(np.arange(startingValueOfFirstInterval,endingValueOfLastInterv
  plt.show()
X = np.random.normal(170,10,250)
N = 5
print("X:\n",X)
print("N:",N)
HistogramWzrostu(X,N)
```

```
X:
 [167.27291474 172.52448145 169.45704318 171.38759134 174.78252166
 162.92915338 167.80611086 171.63371185 184.90444141 159.4828468
 167.87069334 158.968717   173.09153777 169.65437073 176.73642121
 170.15894679 175.1311203  154.95788218 182.09643169 174.14713044
 160.35363159 159.64128878 168.76608634 157.35029635 179.01797081
 165.34844092 172.79409224 164.1116145  161.93681097 179.03810039
 167.32833287 170.22359285 174.93797054 153.76081872 169.26824277
 174.68709254 165.15633089 160.96904331 164.32576089 162.69802712
 168.34917955 178.79532215 172.97423845 158.81157119 175.26503575
 163.4784374  187.63627633 157.02485951 167.1589057  165.21637022
 163.01615051 162.09300992 159.79267529 164.93205453 168.65110001
 170.32185198 158.42446673 180.64405831 169.89885544 176.05535878
 180.63886726 164.24711448 187.17099435 179.8251459  164.78753169
 178.71766634 177.50551399 169.99337201 169.21596606 171.98310842
 168.90293191 173.65094914 168.58844097 179.36668371 146.85857253
 171.24433373 178.66138534 158.08203204 162.95694987 164.11092562
 158.63822441 172.90099522 172.66478005 178.38495675 152.89761251
 169.40100144 159.09277301 168.85741016 178.8607692  164.34622014
 178.66156089 180.58146938 175.85804808 170.06338656 154.3299521
 173.36696542 165.00575092 176.10285888 173.43653001 188.52554679
 187.81242826 184.58124343 168.72674283 166.2055516  142.28585562
 162.377948   172.17004803 154.04878617 168.82691453 154.29936107
 185.85500379 171.54058306 174.30723844 181.3007093  171.38110087
 157.82538804 168.62977063 181.96874019 164.97329206 183.03428358
 162.84243302 174.92593761 159.35491126 173.83998112 154.83742239
 183.73122915 167.25463856 169.29737851 162.99095511 151.94012191
 168.04982724 176.56074676 171.93927278 163.44957851 164.34402745
 171.05810351 170.24273413 171.37016237 183.09037476 153.8409821
 174.75290535 175.11140493 189.78005244 162.16862299 184.88340934
 176.98545391 169.02772545 157.87913709 163.30356163 177.15205136
 167.09725294 139.08794883 179.17926937 149.16838714 169.29022754
 181.95464793 174.50590674 169.95003922 165.25046495 165.0604974
 157.3452624  168.20427814 171.59921103 181.09936758 157.81514902
 164.74128805 183.31300243 184.82455167 171.73183319 172.12397541
 175.06070597 166.59996233 158.02510383 161.14044407 184.29317678
 147.92378694 179.44849272 150.22416017 172.29021233 168.12977527
 173.54814977 164.59686026 157.46715776 175.09294103 180.14681647
 164.61467727 161.16287246 167.50621815 157.5149914  171.26280987
 175.5473382  187.92593771 149.26590447 180.27920799 180.72912515
 158.03581437 172.45283871 155.6950641  171.82175624 179.70349568
 169.78389131 164.96898792 168.8077138  188.88164306 175.02798768
 180.07257205 179.42336919 153.05002433 155.22060802 172.20730631
 155.36897489 165.18135935 181.62272616 156.26489255 181.25904448
 169.34630852 163.97703665 166.43060275 154.28584893 195.91842332
 177.32190253 177.14455865 153.43285632 182.76679019 167.81262984
 167.27218026 185.73257554 156.33039153 176.73979235 165.83125195
 175.39803549 159.40907253 159.01253552 172.27705251 153.58159352
 175.93386417 166.83161286 171.43557554 173.65085971 173.37203215
 169.37855368 176.34253289 189.08474829 174.57316928 167.4459677
 173.81971379 171.27413212 169.59218436 171.95011623 177.71626141]
N: 5
```

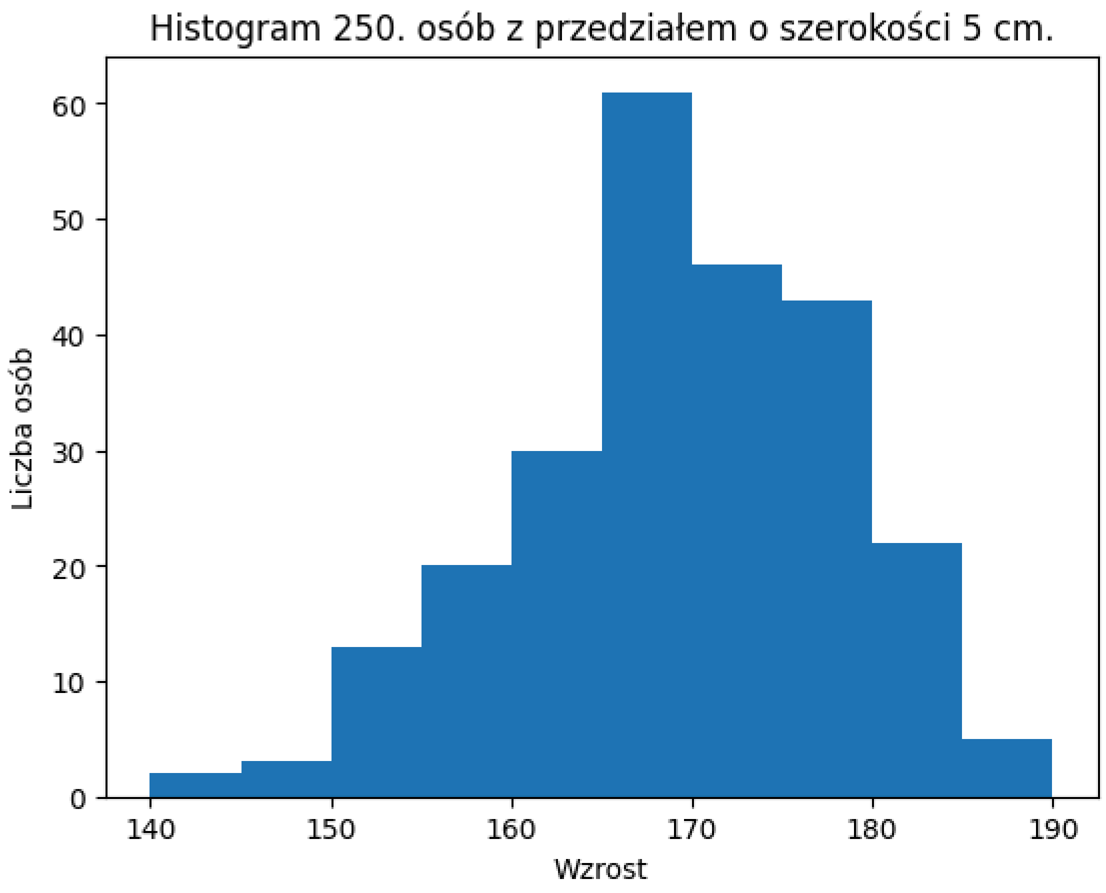


Z histogramu możemy odczytać jak rozkłada się w tym przypadku wzrost w badanej grupie osób. Pozwala szybko zauważyć, jakich miar jest statystycznie najwięcej a jakich najmniej. Dowiadujemy się dzięki temu w szybki sposób jak wygląda w całości badana przez nas pula danych.

In [264…

```
#Histogram
import numpy as np
import matplotlib.pyplot as plt

def HistogramWzrostu2(x,n):
  shadowOffset= n/10
  startingValueOfFirstInterval = (min(x)//n)*n
  endingValueOfLastInterval = (max(x)//n +1)*n
  countOfIntervals = (endingValueOfLastInterval - startingValueOfFirstInterv
  y = np.zeros(int(countOfIntervals))
  for val in x:
    ind = int((val-startingValueOfFirstInterval)//n)
    y[ind] += 1
  b = np.arange(startingValueOfFirstInterval,endingValueOfLastInterval,n)
  plt.hist(x,b)
  plt.title("Histogram "+str(int(len(x)))+". osób z przedziałem o szerokośc
  plt.ylabel("Liczba osób")
  plt.xlabel("Wzrost")
  plt.show()
X = np.random.normal(170,10,250)
N = 5
print("X:\n",X)
print("N:",N)
HistogramWzrostu2(X,N)
```

```
X:
 [163.68207175 171.00969758 162.08036955 167.70652041 168.89479965
 184.59655097 150.55013376 168.35596022 180.92527628 172.02617878
 148.7088017  164.33403808 171.43755037 176.6207665  172.28892243
 156.05569723 151.61579576 171.82300284 182.38923601 176.93970874
 170.6428587  167.71060518 188.2429814  171.15191643 170.39379148
 155.21547867 172.85636725 151.28528513 168.66693526 178.91057773
 166.91859581 177.03463498 179.29770234 162.35857675 175.13281749
 157.9751396  169.18500351 159.66875958 161.88114355 177.2709342
 169.1958712  168.67957818 178.40643092 164.39659295 166.30925027
 179.37170212 159.47767685 164.68810846 155.32945757 153.80890146
 175.79470939 155.78499579 167.01897276 181.98157804 183.58567356
 183.35648417 165.22566218 181.69876363 184.56032618 160.93164299
 154.58794457 172.49715788 168.43784308 156.72291563 165.08207783
 179.16325437 159.0942764  178.98799035 174.59648746 181.35981288
 170.36587299 153.17014896 171.59260532 172.5585797  173.58829475
 150.71755502 172.24018411 166.6266529  161.38829209 157.8701069
 167.31077101 173.02328909 167.48586526 176.1910132  176.13573859
 165.89618025 179.51692752 172.68576374 166.39713145 173.88548975
 167.22506085 169.94150057 168.21591341 179.8684649  167.9168171
 155.47240884 166.07067944 178.77801696 177.26672234 174.87811101
 165.70659278 175.11800063 173.50625989 174.59064356 169.07857563
 186.78105101 173.45998128 160.63015478 190.39202014 171.84817385
 177.71638764 157.31064096 182.8446549  152.0927691  173.5901856
 190.16800151 188.8079111  180.18503072 169.51592175 169.2914178
 179.8468836  159.40158468 183.57822503 182.41611168 172.24337786
 156.32714097 143.65832752 176.24837151 170.30482182 176.33880351
 192.27205508 173.93416892 167.46507224 167.34835666 170.69522057
 143.80474227 156.78415103 179.03898127 154.73450753 165.67288456
 152.73888658 172.04536927 145.69170154 172.12512468 177.6133545
 179.99032381 190.21790088 165.31847638 167.93825659 166.69389242
 150.3942923  161.33975759 172.11281142 146.93014013 176.26144932
 188.03326378 162.98120172 179.70676692 163.37178636 166.66932973
 167.39562367 174.15802515 170.2870634  164.25396479 167.2314305
 174.30117629 173.79663972 172.25068138 151.4167436  167.27857176
 168.61875134 167.88411635 170.50107309 176.87977779 156.75794328
 161.6645837  169.58803588 165.27924026 156.98471108 182.73558955
 169.74221876 167.5181745  181.05310766 172.02494361 160.34906314
 169.72486319 169.95771529 170.82440934 164.5949531  171.70220747
 182.84293636 179.09879228 168.53930766 164.84304151 170.86349408
 177.40147172 180.64994817 179.48307221 163.31779356 181.70727243
 167.60991975 169.3694978  176.92767524 166.46994704 166.52368922
 163.05229697 177.37579682 173.57234476 178.27074189 172.99542665
 165.81572394 176.90555455 167.31883174 166.49836787 156.86819098
 177.42591796 193.37122035 176.10319626 156.10831737 164.2677644
 160.63094447 175.52835158 183.56964946 176.31416746 168.78423947
 164.1709983  161.11569973 163.9522116  176.58511909 168.11335083
 153.83527909 169.17976409 174.05633912 185.81909008 176.76392354
 163.07205215 164.9558502  161.90415547 166.99078305 160.92965547
 180.48114386 157.71807505 180.99328051 168.26714106 169.71915368
 175.88690148 184.1259975  160.85808562 172.54957642 169.61032313]
N: 5
```



Wcześniejszy wykres histogramu napisałem sam korzystając zwyczajnie z funkcji plot. Później uświadomiłem sobie, że pewnie jest do tego funkcja, więc też ją wykonałem i również tutaj zamieściłem.