In [8]:
```
import numpy as np
np_height = np.array([1.73, 1.08, 1.89, 1.71])

np_weight = np.array([65.4, 63.0, 98.4, 68.7])
bmi = np_weight / np_height**2

print(np_height +  np_weight)
```

```
[ 67.13  64.08 100.29  70.41]
```

In [14]:
```
height = [1.73, 1.08, 1.89, 1.71]
np_height = np.array([height])

weight = [65.4, 63.0, 98.4, 68.7]
np_weight = np.array([65.4, 63.0, 98.4, 68.7])

bmi = np_weight / np_height**2
print(bmi)
print(weight[3])
print(type(np_height))   #nd - n-dimensional array
```

```
[[21.85171573 54.01234568 27.5468212  23.49440854]]
68.7
<class 'numpy.ndarray'>
```

In [12]:
```
bmi
bmi = np.array([65.4, 63.0, 98.4, 68.7])

print(bmi[0])
```

```
65.4
```

In [34]:
```
np_2d =  np.array([[1.75, 1.08, 1.89, 1.71], [65.4, 63.0, 98.4, 68.7]])
print(np_2d[:, 0:1])
print(np_2d)
print(np_2d.shape)
print(np_2d[0][2]) #subsetting
print(np_2d[1, 3])
print("\n")
print(np_2d[:, 0])
print("\n")
print(np_2d[1:, 3 ])
print(np.mean(np_2d[:, 1]))
```

```
[[ 1.75]
 [65.4 ]]
[[ 1.75  1.08  1.89  1.71]
 [65.4  63.   98.4  68.7 ]]
(2, 4)
1.89
68.7


[ 1.75 65.4 ]


[68.7]
32.04
```

In [8]:
```

x = [["a", "b"], ["c", "d"]]

np_x = np.array([x])
print(np_x[:, 1])
```

```
[['c' 'd']]
```

In [19]:
```
np_city = np.array([[1.1, 2.2], [2.1, 3.2], [1.1, 2.2], [1.1, 4.3], [1.1, 2.2], [
print(np.mean(np_city))
print(np.mean(np_city[:, 1]))
print(np.std(np_city[:, 0]))
print(np.corrcoef(np_city[:, 0], np_city[:, 1]))
```

```
3.239444444444444
3.0994444444444444
3.2507853277190697
[[1.         0.52413228]
 [0.52413228 1.        ]]
```

In [33]:
```
height = np.round(np.random.normal(1.73, 0.20, 5000), 2)
weight = np.round(np.random.normal(65.4, 15, 5000), 2)
np_city = np.column_stack((height, weight))
print(np_city)
print(np.mean(np_city))
```

```
[[ 1.6  59.45]
 [ 1.54 60.56]
 [ 1.8  91.9 ]
 ...
 [ 2.11 55.85]
 [ 1.85 66.45]
 [ 1.87 80.94]]
33.456975
```

In [29]:
```
a = 10
b = 2
c = np.array([1, 2, 3, 5])
print(a == b)
print(a != b)
print(a < b)
print(a > b)
print(b < a)
print(b > a)
print(c < 3)
print(c[c < 3]) # that met the conditions
```

```
False
True
False
True
True
False
[ True  True False False]
[1 2]
```

In [30]:
```

np_height = np.array([1.73, 1.08, 1.89, 1.71])
np_weight = np.array([65.4, 63.0, 98.4, 68.7])
np_bmi = np.column_stack((np_height, np_weight))
print(np_bmi)
```

```
[[ 1.73 65.4 ]
 [ 1.08 63.  ]
 [ 1.89 98.4 ]
 [ 1.71 68.7 ]]
```

In [35]:
```
print(np.random.normal(4, 0.2, 1000))
```

```
 3.8504111  4.09218304 4.0337253  4.20830315 3.90032719 4.18783403
 4.26781087 4.05216611 3.7158903  3.80396535 4.07883917 3.80986929
 3.75288833 3.51064444 4.02456916 4.20963599 3.69386923 4.15133437
 4.31881538 3.93200182 4.5286735  3.99826416 4.41909768 3.77387029
 3.73278893 3.85837888 4.33988336 4.01016696 3.7748329  3.6655455
 4.05823594 3.87807228 3.94489947 4.05130928 4.1067146  4.14239661
 3.95142981 3.94757049 4.00503873 3.9392076  4.52713905 3.85296368
 4.15387737 4.02057362 3.75614724 4.03249103 3.84807883 3.91806646
 3.81868922 4.05771175 3.89755817 3.97005586 3.78526696 4.16143041
 4.00691684 4.04594555 3.74129974 4.28742988 3.96802923 4.1531488
 3.90148391 3.65183955 4.22063073 3.72539875 4.2524976  4.27063797
 4.30273304 4.23960412 4.20426576 3.8108066  3.82812559 4.39809654
 3.72213897 3.9877428  4.19300343 4.33655094 3.89989356 4.01170649
 3.8488945  4.22328531 4.12109165 3.69780787 3.96367618 3.83221735
 4.23964834 3.84579688 4.00609447 3.92899762 4.23288688 4.1061279
 4.27249306 3.77321951 3.92656904 4.24165397 4.26201329 3.97989551
 4.17146465 3.63928791 4.10552392 4.23368528 4.41623699 4.09689546
 4.23124912 3.85044104 4.04881322 4.27160955 4.02172341 4.25459066
 4.09725066 3.79435761 3.82051584 3.79189604 3.76287748 4.08028484
 4.16356536 4.06372953 4.21246755 4.21362951 4.12227239 3.89058863
```

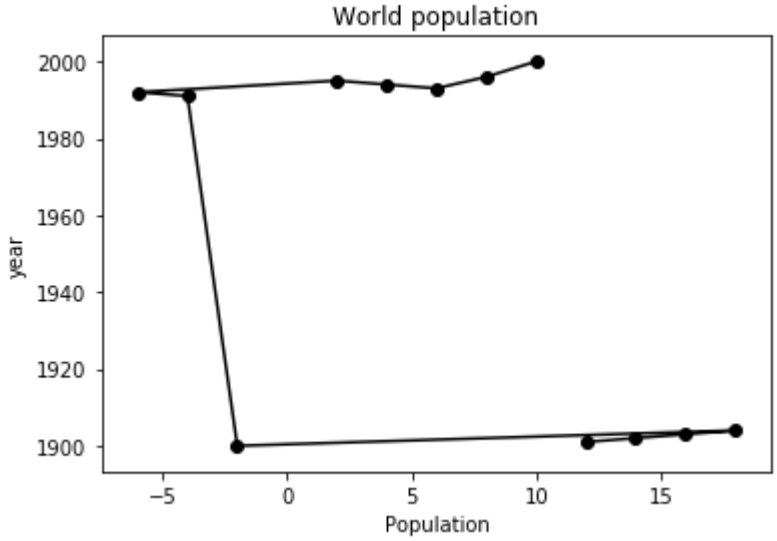
In [37]:
```
fam = [1.73, 1.08, 1.89, 1.71]
tallest = max(fam)
print(tallest)
print(max(fam))
```

```
1.89
1.89
```

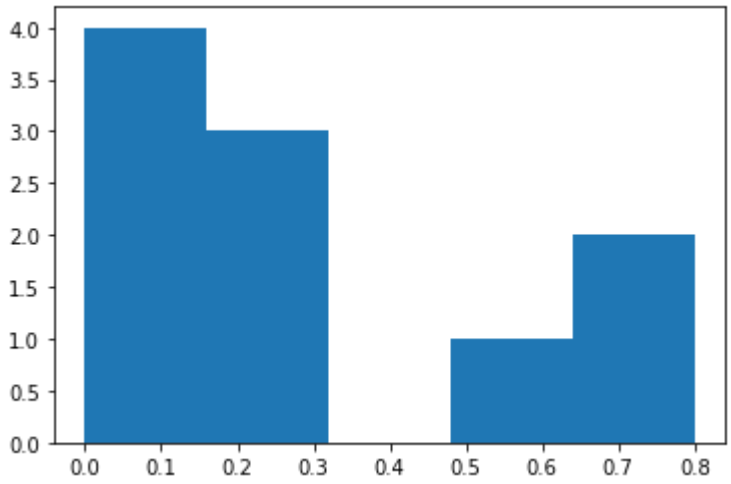
In [43]:
```
fam = [1.73, 1.08, 1.89, 1.71]  #round()
print(round(1.73, 1))
print(round(1.08))
print(round(1.08, 1))
print(round(1.89))
```

```
1.7
1
1.1
2
```

In [14]:
```
import matplotlib.pyplot as plt
year = [1900, 1991, 1992, 1995, 1994, 1993, 1996, 2000]
Pop = [-2, -4, -6, 2, 4, 6, 8, 10]
year = [1901, 1902, 1903, 1904] + year
Pop = [12, 14, 16, 18] + Pop
plt.plot(Pop , year, color = '0')
plt.xlabel('Population')
plt.ylabel('year')
plt.title('World population')
plt.scatter(Pop, year, color = '0')
#plt.yticks([1999, 2000, 2001, 2002, 2001, 2003, 2008, 2010]) # change y-axis
#plt.grid()
plt.show()
```

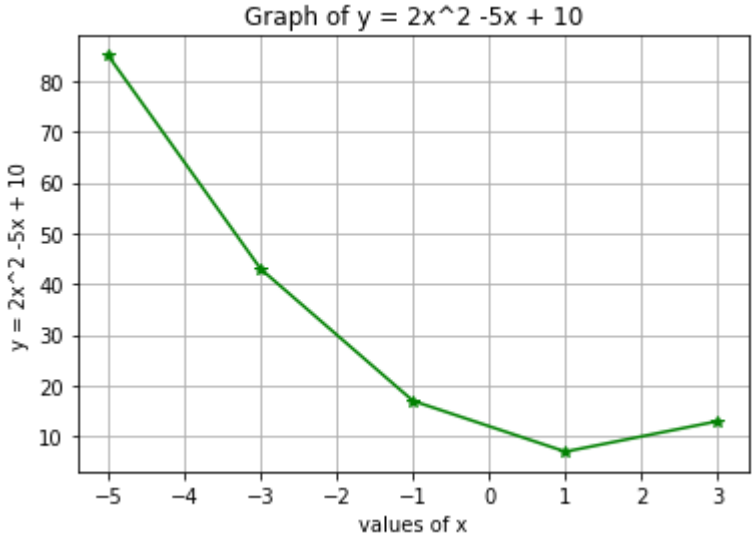


In [10]:
```
# histogram
#help(plt.hist)
import matplotlib.pyplot as plt
values = [0,0.6,0.8,0.8,0.10,0.12,0.14,0.16,0.18,0.20]
plt.hist(values, bins = 5)

plt.show()
plt.clf()
```

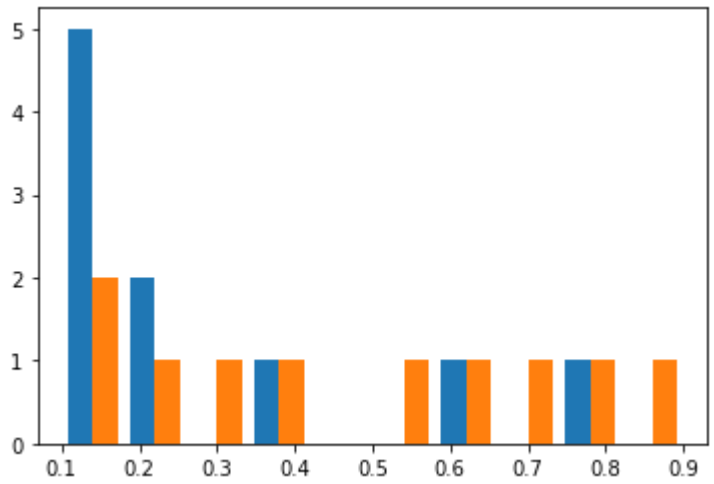



```
<Figure size 432x288 with 0 Axes>
```

In [63]:
```
# plotting of graphs
# plot a graph of y = 2x + 5 for -5<=x<=5
from numpy import *
from matplotlib.pyplot import * # graph module
# some text = import matplotlib.pyplot
x = arange(-5, 5, 2)
y = 2*x**2 - 5* x + 10

plot(x,y,'-*', color = 'g')
title("Graph of y = 2x^2 -5x + 10")
xlabel("values of x")
ylabel("y = 2x^2 -5x + 10")
grid()
show()
```



In [20]:
```
import matplotlib.pyplot as plt
life_span = [[0.2,0.4,0.6,0.8,0.10,0.12,0.14,0.14,0.16,0.18], [0.1,0.2,0.3,0.4,0.
plt.hist(life_span)
plt.show()
plt.clf()
```




```
<Figure size 432x288 with 0 Axes>
```

In [3]:
```
pop = [40.9, 35.9, 36.8]
countries = ["afghanistan", "albania", "algeria"]
ind_alb = countries.index("albania")
ind_alb

pop[ind_alb]
```

Out[3]:
```
35.9
```

In [7]:
```
#dictionaries
world = {"afghanistan":40.9, "albania":35.9, "algeria":36.8}
print(world["albania"])
```

```
35.9
```

In [6]:
```
world = {"afghanistan":40.9, "albania":35.9, "algeria":36.8}
print(world.values())
print(world.keys())
```

```
dict_values([40.9, 35.9, 36.8])
dict_keys(['afghanistan', 'albania', 'algeria'])
```

In [9]:
```
x = 6
if x % 2 == 0:
    print("x is even")
```

```
x is even
```

In [15]:
```
import numpy as np
numpy1 = np.array([17.2, 20.0, 8.25, 9.50])
numpy2 = np.array([13.0, 24.0, 8.25, 9.0])
np.logical_not(numpy1 > 10, numpy2 < 20)
#np.logical_and(numpy1 > 10, numpy2 < 20)
```

Out[15]:
```
array([False, False,  True,  True])
```

In [3]:
```
world = {"afghanistan":40.9, "albania":35.9, "algeria":36.8, "albania":36.0}
world["sealand"] = 45.90

del(world["sealand"])
print(world)
```

```
{'afghanistan': 40.9, 'albania': 36.0, 'algeria': 36.8}
```

In [29]:
```
import pandas as pd
world = {"country":["afghanistan","albania","algeria","sealand","africa"],
        "capital":["russia","iraq","india","china","pretoria"],
        "area":[8.56, 3.67, 4.98, 2.123, 4.90],
        "population":[200.4, 143.6, 125.2, 134.5, 52.6] }
brics = pd.DataFrame(world)
print(brics)
```

```
       country   capital   area  population
0  afghanistan    russia  8.560       200.4
1      albania      iraq  3.670       143.6
2      algeria     india  4.980       125.2
3      sealand     china  2.123       134.5
4       africa  pretoria  4.900        52.6
```

In [24]:
```
import pandas as pd
dict = {"Country":["Brazil", "Russia", "China", "South Africa", "India"],
        "Capital":["Brazilia", "Moscow", "New Delhi", "Beijing", "Pretoria"],
        "area":[8.516, 17.10, 3.286, 1.221, 12.22],
        "population":[200.4, 143.5, 1252, 1357, 52.98] }
brics = pd.DataFrame(dict)
brics.index = ["BR", "RU", "CH", "SA", "IN"]
print(brics)
```

```
         Country    Capital    area  population
BR        Brazil   Brazilia   8.516      200.40
RU        Russia     Moscow  17.100      143.50
CH         China  New Delhi   3.286     1252.00
SA  South Africa    Beijing   1.221     1357.00
IN         India   Pretoria  12.220       52.98
```

In [31]:
```
names = ["Brazil", "Russia", "China", "South Africa", "India", "Moscow", "Africa"
dr = [True, False, True, False, False, False, False]
cpc = [809, 731, 588, 18, 200, 70, 45]

import pandas as pd
my_dict = {"Country": names, "Drives_right": dr, "Cars_per_cap": cpc}
cars = pd.DataFrame(my_dict)
cars.index = ["BR", "RU", "CH", "SA", "IN", "MO", "AF"]
print(cars)
```

```
         Country  Drives_right  Cars_per_cap
BR        Brazil          True           809
RU        Russia         False           731
CH         China          True           588
SA  South Africa         False            18
IN         India         False           200
MO        Moscow         False            70
AF        Africa         False            45
```

In [2]:
```
import numpy as np
print(np.random.normal(3, 0.1,1000))
```

```
[2.95894703 2.97849399 2.89191453 3.03242511 3.144757   3.06633276
 2.94263928 2.92528279 2.98188513 3.01876461 3.12640149 2.97335539
 2.96289746 3.09255819 2.84522918 2.7880327  2.85677057 3.12993613
 2.8562205  2.95520579 2.97882788 3.1190239  3.00524618 2.93743766
 2.84484797 3.02885828 3.00404371 2.97790365 3.19425172 3.08825807
 3.11637278 3.00261712 2.9667356  3.01944391 3.13255576 2.94648294
 3.00144346 3.03113789 2.92693429 3.0321206  3.13464289 2.93065489
 3.12947577 2.90907356 2.95237535 3.02503665 3.06808951 2.85309153
 3.04720087 3.03463129 3.05774218 2.9177099  3.06011215 2.86638613
 3.02134975 3.14559077 2.86979982 2.97077305 3.01508605 2.95141928
 2.89142502 3.07016738 2.89001158 2.90533642 3.08919482 3.04613176
 2.97135175 2.96858899 3.09203518 2.97877721 2.85106297 3.11072098
 2.92223872 2.89753209 2.99474256 3.16174626 2.91248955 3.05826836
 2.94836202 2.86241445 2.88470649 3.07575912 2.87278504 3.00691493
 2.84717072 2.98141108 3.05952518 3.15286326 2.94083424 2.95788542
 3.08199343 3.00919136 3.00884628 2.81342305 2.97633357 2.99592875
 3.1290946  2.97295313 2.99207941 2.95394206 3.06985072 2.98257919
 2.98665947 3.06381707 3.01683905 3.04942162 3.14234643 3.00220844
 3.13377396 3.0559595  3.01743105 2.83004126 3.06494083 3.0110901
```

In [24]:
```
import pandas as pd
dict = {"Country":["Brazil", "Russia", "China", "South Africa", "India"],
        "Capital":["Brazilia", "Moscow", "New Delhi", "Beijing", "Pretoria"],
        "area":[8.516, 17.10, 3.286, 1.221, 12.22],
        "population":[200.4, 143.5, 1252, 1357, 52.98] }
brics = pd.DataFrame(dict)
brics.index = ["BR", "RU", "CH", "SA", "IN"]
print(brics["Country"])   #sinle [] gives a pandas series
```

```
BR          Brazil
RU          Russia
CH           China
SA    South Africa
IN           India
Name: Country, dtype: object
```

In [7]:
```
import pandas as pd
dict = {"Country":["Brazil", "Russia", "China", "South Africa", "India"],
        "Capital":["Brazilia", "Moscow", "New Delhi", "Beijing", "Pretoria"],
        "area":[8.516, 17.10, 3.286, 1.221, 12.22],
        "population":[200.4, 143.5, 1252, 1357, 52.98] }
brics = pd.DataFrame(dict)
brics.index = ["BR", "RU", "CH", "SA", "IN"]
print(type(brics["Capital"]))
```

```
<class 'pandas.core.series.Series'>
```

In [8]:
```
import pandas as pd
dict = {"Country":["Brazil", "Russia", "China", "South Africa", "India"],
        "Capital":["Brazilia", "Moscow", "New Delhi", "Beijing", "Pretoria"],
        "area":[8.516, 17.10, 3.286, 1.221, 12.22],
        "population":[200.4, 143.5, 1252, 1357, 52.98] }
brics = pd.DataFrame(dict)
brics.index = ["BR", "RU", "CH", "SA", "IN"]
print(brics[["Country", "area"]])
```

```
         Country    area
BR        Brazil   8.516
RU        Russia  17.100
CH         China   3.286
SA  South Africa   1.221
IN         India  12.220
```

In [10]:
```
import pandas as pd
dict = {"Country":["Brazil", "Russia", "China", "South Africa", "India"],
        "Capital":["Brazilia", "Moscow", "New Delhi", "Beijing", "Pretoria"],
        "area":[8.516, 17.10, 3.286, 1.221, 12.22],
        "population":[200.4, 143.5, 1252, 1357, 52.98] }
brics = pd.DataFrame(dict)
brics.index = ["BR", "RU", "CH", "SA", "IN"]
print(brics[1:4])  #rows use slice
```

```
         Country    Capital    area  population
RU        Russia     Moscow  17.100       143.5
CH         China  New Delhi   3.286      1252.0
SA  South Africa    Beijing   1.221      1357.0
```

In [16]:
```
import pandas as pd
dict = {"Country":["Brazil", "Russia", "China", "South Africa", "India"],
        "Capital":["Brazilia", "Moscow", "New Delhi", "Beijing", "Pretoria"],
        "area":[8.516, 17.10, 3.286, 1.221, 12.22],
        "population":[200.4, 143.5, 1252, 1357, 52.98] }
brics = pd.DataFrame(dict)
brics.index = ["BR", "RU", "CH", "SA", "IN"]
print(brics.loc[["RU"]])  #row loc label-based

print(brics.loc["RU"])
```

```
   Country Capital  area  population
RU  Russia  Moscow  17.1       143.5
Country       Russia
Capital       Moscow
area            17.1
population     143.5
Name: RU, dtype: object
```

In [17]:
```
import pandas as pd
dict = {"Country":["Brazil", "Russia", "China", "South Africa", "India"],
        "Capital":["Brazilia", "Moscow", "New Delhi", "Beijing", "Pretoria"],
        "area":[8.516, 17.10, 3.286, 1.221, 12.22],
        "population":[200.4, 143.5, 1252, 1357, 52.98] }
brics = pd.DataFrame(dict)
brics.index = ["BR", "RU", "CH", "SA", "IN"]
print(brics.loc[["RU", "BR", "CH"]]) #double [[]] gives a panda
```

```
   Country    Capital    area  population
RU  Russia     Moscow  17.100       143.5
BR  Brazil   Brazilia   8.516       200.4
CH   China  New Delhi   3.286      1252.0
```

In [20]:
```
import pandas as pd
dict = {"Country":["Brazil", "Russia", "China", "South Africa", "India"],
        "Capital":["Brazilia", "Moscow", "New Delhi", "Beijing", "Pretoria"],
        "area":[8.516, 17.10, 3.286, 1.221, 12.22],
        "population":[200.4, 143.5, 1252, 1357, 52.98] }
brics = pd.DataFrame(dict)
brics.index = ["BR", "RU", "CH", "SA", "IN"]
print(brics.loc[["BR", "RU", "CH"], ["Capital", "Country"]])
```

```
      Capital Country
BR   Brazilia  Brazil
RU     Moscow  Russia
CH  New Delhi   China
```

In [21]:
```
import pandas as pd
dict = {"Country":["Brazil", "Russia", "China", "South Africa", "India"],
        "Capital":["Brazilia", "Moscow", "New Delhi", "Beijing", "Pretoria"],
        "area":[8.516, 17.10, 3.286, 1.221, 12.22],
        "population":[200.4, 143.5, 1252, 1357, 52.98] }
brics = pd.DataFrame(dict)
brics.index = ["BR", "RU", "CH", "SA", "IN"]
print(brics.iloc[1])
```

```
Country       Russia
Capital       Moscow
area            17.1
population     143.5
Name: RU, dtype: object
```

In [22]:
```
import pandas as pd
dict = {"Country":["Brazil", "Russia", "China", "South Africa", "India"],
        "Capital":["Brazilia", "Moscow", "New Delhi", "Beijing", "Pretoria"],
        "area":[8.516, 17.10, 3.286, 1.221, 12.22],
        "population":[200.4, 143.5, 1252, 1357, 52.98] }
brics = pd.DataFrame(dict)
brics.index = ["BR", "RU", "CH", "SA", "IN"]
print(brics.iloc[[1,2,3], [0,1]])
```

```
         Country    Capital
RU        Russia     Moscow
CH         China  New Delhi
SA  South Africa    Beijing
```

In [23]:
```
import pandas as pd
dict = {"Country":["Brazil", "Russia", "China", "South Africa", "India"],
        "Capital":["Brazilia", "Moscow", "New Delhi", "Beijing", "Pretoria"],
        "area":[8.516, 17.10, 3.286, 1.221, 12.22],
        "population":[200.4, 143.5, 1252, 1357, 52.98] }
brics = pd.DataFrame(dict)
brics.index = ["BR", "RU", "CH", "SA", "IN"]
print(brics.iloc[:, [0,1]])
```

```
         Country    Capital
BR        Brazil   Brazilia
RU        Russia     Moscow
CH         China  New Delhi
SA  South Africa    Beijing
IN         India   Pretoria
```

In [26]:
```
import pandas as pd
dict = {"Country":["Brazil", "Russia", "China", "South Africa", "India"],
        "Capital":["Brazilia", "Moscow", "New Delhi", "Beijing", "Pretoria"],
        "area":[8.516, 17.10, 3.286, 1.221, 12.22],
        "population":[200.4, 143.5, 1252, 1357, 52.98] }
brics = pd.DataFrame(dict)
brics.index = ["BR", "RU", "CH", "SA", "IN"]
print(brics.iloc[[4, 2]])
```

```
   Country    Capital    area  population
IN   India   Pretoria  12.220       52.98
CH   China  New Delhi   3.286     1252.00
```

In [ ]: