```
import pandas as pd
import numpy as np
data = pd.read_csv('PersonsData.csv')
print(data)
```

```
   Wzrost (cm)  Waga (kg)  Staz (lata)  Zarobki (tys.)  Ocena (pkt.)  Pietro  \
A          190         88            3             3.5             7       6
B          172         70           12             4.3             5       1

   Dzieci  Odleglosc (km)  Ubezp.
A       1              25       1
B       4              12       0
```

```
np.sqrt(np.sum((data.iloc[0,:]-data.iloc[1,:])**2))
```

```
30.620907889871585
```

```
data_copy=data.copy()
```

```
#data_copy.iloc[:,3]=data_copy.iloc[:,3]/1000
```

```
data_copy.iloc[:,3]=data_copy.iloc[:,3]*1000
```

```
print(data_copy)
```

```
   Wzrost (cm)  Waga (kg)  Staz (lata)  Zarobki (tys.)  Ocena (pkt.)  Pietro  \
A          190         88            3          3500.0             7       6
B          172         70           12          4300.0             5       1

   Dzieci  Odleglosc (km)  Ubezp.
A       1              25       1
B       4              12       0
```

```
np.sqrt(np.sum((data_copy.iloc[0,:]-data_copy.iloc[1,:])**2))
```

```
800.5854108088655
```

```
data_copy2=data_copy.copy()
```

```
data_copy.iloc[:,:]
```

	Wzrost (cm)	Waga (kg)	Staz (lata)	Zarobki (tys.)	Ocena (pkt.)	Pietro	Dzieci	Odleglosc (km)	Ubezp.
A	190	88	3	3500.0	7	6	1	25	1
B	172	70	12	4300.0	5	1	4	12	0

```
data_copy.iloc[:,:].std()
```

```
Wzrost (cm)        12.727922
Waga (kg)          12.727922
Staz (lata)         6.363961
Zarobki (tys.)    565.685425
Ocena (pkt.)        1.414214
Pietro              3.535534
Dzieci              2.121320
Odleglosc (km)      9.192388
Ubezp.              0.707107
dtype: float64
```

```
data_copy3=data.copy()
```

```
data_copy3.iloc[:,:]=data.iloc[:,:]/data.iloc[:,:].std()
```

```
data_copy2.iloc[:,:]=data_copy2.iloc[:,:]/data_copy2.iloc[:,:].std()
```

```
print(data_copy2)
```

```
   Wzrost (cm)  Waga (kg)  Staz (lata)  Zarobki (tys.)  Ocena (pkt.)  \
A    14.927810   6.913933     0.471405        6.187184      4.949747
B    13.513596   5.499719     1.885618        7.601398      3.535534

     Pietro    Dzieci  Odleglosc (km)    Ubezp.
```

```
A  1.697056  0.471405        2.719641  1.414214
B  0.282843  1.885618        1.305428  0.000000
```

```
np.sqrt(np.sum((data_copy2.iloc[0,:]-data_copy2.iloc[1,:])**2))
```

```
4.242640687119286
```

```
np.sqrt(np.sum((data_copy3.iloc[0,:]-data_copy3.iloc[1,:])**2))
```

```
4.242640687119285
```

```
data.iloc[:,3].std()
```

```
0.5656854249492379
```

minskowskiego

```
np.sqrt(np.sum((data.iloc[0,:]-data.iloc[1,:])**2))
```

```
30.620907889871585
```

miejska

```
np.sum(np.abs(data.iloc[0,:]-data.iloc[1,:]))
```

```
69.8
```

Zad 2

```
data2 = pd.read_csv('BinaryData.csv')
print(data2)
```

```
   A1  A2  A3  A4  A5  A6  A7  A8  A9  A10
```

```
A   0   0   1   1   0   1   0   1   0   1
B   0   0   0   1   1   1   0   0   0   1
C   1   0   1   1   0   1   1   1   0   0
D   1   1   1   1   0   0   0   0   0   0
E   1   0   1   0   1   0   0   0   0   0
F   0   0   0   1   0   1   0   0   0   0
```

```
a=len(data2.index)


AB_n10=np.sum(data2.iloc[0,:]-data2.iloc[1,:]==1)
AB_n11=np.sum((data2.iloc[0,:]-data2.iloc[1,:]==0)&(data2.iloc[0,:]==1)&(data2.iloc[1,:]==1))
AB_n00=np.sum((data2.iloc[0,:]-data2.iloc[1,:]==0)&(data2.iloc[0,:]==0)&(data2.iloc[1,:]==0))
AB_n01=np.sum(data2.iloc[1,:]-data2.iloc[0,:]==1)
print("AB_n00 "+ str(AB_n00)+" AB_n01 "+str(AB_n01)+" AB_n10 "+ str(AB_n10) +" AB_n11 "+str(AB_n11))
```

```
AB_n00 4 AB_n01 1 AB_n10 2 AB_n11 3
```

```
n10 = []

n10.append(np.sum(data2.iloc[0,:]-data2.iloc[1,:]==1))
n10.append(np.sum(data2.iloc[1,:]-data2.iloc[2,:]==1))


for i in range(a):
    for j in range(a):
        if(str(data2.iloc[i,:].name)!=str(data2.iloc[j,:].name)):
            n10=np.sum(data2.iloc[i,:]-data2.iloc[j,:]==1)
            n11=np.sum((data2.iloc[i,:]-data2.iloc[j,:]==0)&(data2.iloc[i,:]==1)&(data2.iloc[j,:]==1))
            n00=np.sum((data2.iloc[i,:]-data2.iloc[j,:]==0)&(data2.iloc[i,:]==0)&(data2.iloc[j,:]==0))
            n01=np.sum(data2.iloc[j,:]-data2.iloc[i,:]==1)
            jacard=n11/(n11++n10+n01)
            dice=2*n11/(2*n11+n10+n01)
            print(str(data2.iloc[i,:].name)+" "+str(data2.iloc[j,:].name))
            print("Jacard: "+str(jacard))
            print("Dice: "+str(dice))
            print(str(data2.iloc[i,:].name)+" "+str(data2.iloc[j,:].name)+" n00 "+ str(n00)+" n01 "+str(n01)+" n10 "+ str(n10) +" n
```

```
Dice: 0.444444444444444
D A n00 3 n01 3 n10 2 n11 2
D B
Jacard: 0.14285714285714285
```

```
Jacard: 0.14285714285714285
Dice: 0.25
D B n00 3 n01 3 n10 3 n11 1
D C
Jacard: 0.42857142857142855
Dice: 0.6
D C n00 3 n01 3 n10 1 n11 3
D E
Jacard: 0.4
Dice: 0.5714285714285714
D E n00 5 n01 1 n10 2 n11 2
D F
Jacard: 0.2
Dice: 0.3333333333333333
D F n00 5 n01 1 n10 3 n11 1
E A
Jacard: 0.14285714285714285
Dice: 0.25
E A n00 3 n01 4 n10 2 n11 1
E B
Jacard: 0.16666666666666666
Dice: 0.2857142857142857
E B n00 4 n01 3 n10 2 n11 1
E C
Jacard: 0.2857142857142857
Dice: 0.4444444444444444
E C n00 3 n01 4 n10 1 n11 2
E D
Jacard: 0.4
Dice: 0.5714285714285714
E D n00 5 n01 2 n10 1 n11 2
E F
Jacard: 0.0
Dice: 0.0
E F n00 5 n01 2 n10 3 n11 0
F A
Jacard: 0.4
Dice: 0.5714285714285714
F A n00 5 n01 3 n10 0 n11 2
F B
Jacard: 0.5
Dice: 0.6666666666666666
F B n00 6 n01 2 n10 0 n11 2
F C
Jacard: 0.3333333333333333
```

```
Jacard: 0.5555555555555556
Dice: 0.5
F C n00 4 n01 4 n10 0 n11 2
F D
Jacard: 0.2
Dice: 0.3333333333333333
F D n00 5 n01 3 n10 1 n11 1
F E
Jacard: 0.0
Dice: 0.0
F E n00 5 n01 3 n10 2 n11 0
```

Zad 3

1.Jest małe zróżnicowanie wartości cechy w dwóch środkowych ćwiartkach rozkładu

2.Jest duże zróżnicowanie wartości cechy w dwóch środkowych ćwiartkach rozkładu

3.Pudełko nr 3 ma większe zróżnicowanie wartości cechy w dwóch środkowych ćwiartkach rozkładu niż pudełko nr 4

4.Mediana ma tą samą wartość

5.Mediana w przypadku kategorii reprezentowanej przez wykres pudełkowy nr 2 jest większa niż w wypadku nr 4

6.Wartość maksimum w wypadku kategorii nr 1 jest większa niż w wypadku kategorii nr 2, zaś w przypadku minimum wartość ta w wypadku kategorii nr 1 jest mniejsza niż w wypadku kategorii nr 2.

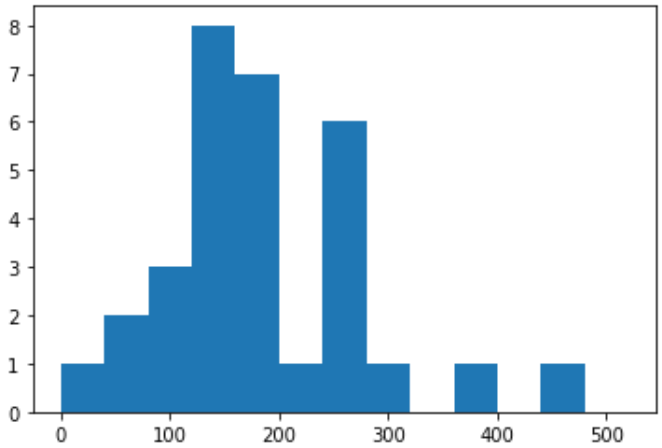
Zad 4

```
XD=np.array([30, 75, 79, 80, 85, 105, 126, 130, 138, 140, 149, 149, 152, 156, 161, 166, 173, 179, 182, 184, 198, 223,
240, 242, 245, 247, 254, 274, 291, 384, 470])
```

```
import matplotlib.pyplot as plt


#plt.hist(XD, bins='auto')
plt.hist(XD, bins = [0,40,80,120,160,200,240,280,320,360,400,440,480,520])
```

```
(array([1., 2., 3., 8., 7., 1., 6., 1., 0., 1., 0., 1., 0.]),
 array([  0,  40,  80, 120, 160, 200, 240, 280, 320, 360, 400, 440, 480,
        520]),
 <a list of 13 Patch objects>)
```




```
df = pd.DataFrame(XD, columns = ['Column_A'])


df.mean()
```

```
Column_A    184.096774
dtype: float64
```

```
df.median()
```

```
Column_A    166.0
dtype: float64
```

```
dominant=df['Column_A'].value_counts()
print(dominant)
```

```
149    2
30     1
179    1
384    1
291    1
274    1
254    1
247    1
245    1
242    1
240    1
223    1
198    1
184    1
182    1
173    1
75     1
166    1
161    1
156    1
152    1
140    1
138    1
130    1
126    1
105    1
85     1
80     1
79     1
470    1
Name: Column_A, dtype: int64
```

```
max(dominant)
```

```
2
```

```
from scipy.stats import skew
```

```
skew(df)
```

```
array([1.11158529])
```

```
np.quantile(XD,0.0)
```

```
30.0
```

```
Q1=np.quantile(XD,0.25)
```

```
Q2=np.quantile(XD,0.5)
```

```
Q3=np.quantile(XD,0.75)
```

```
Q4=np.quantile(XD,1)
```

```
df.describe()
```

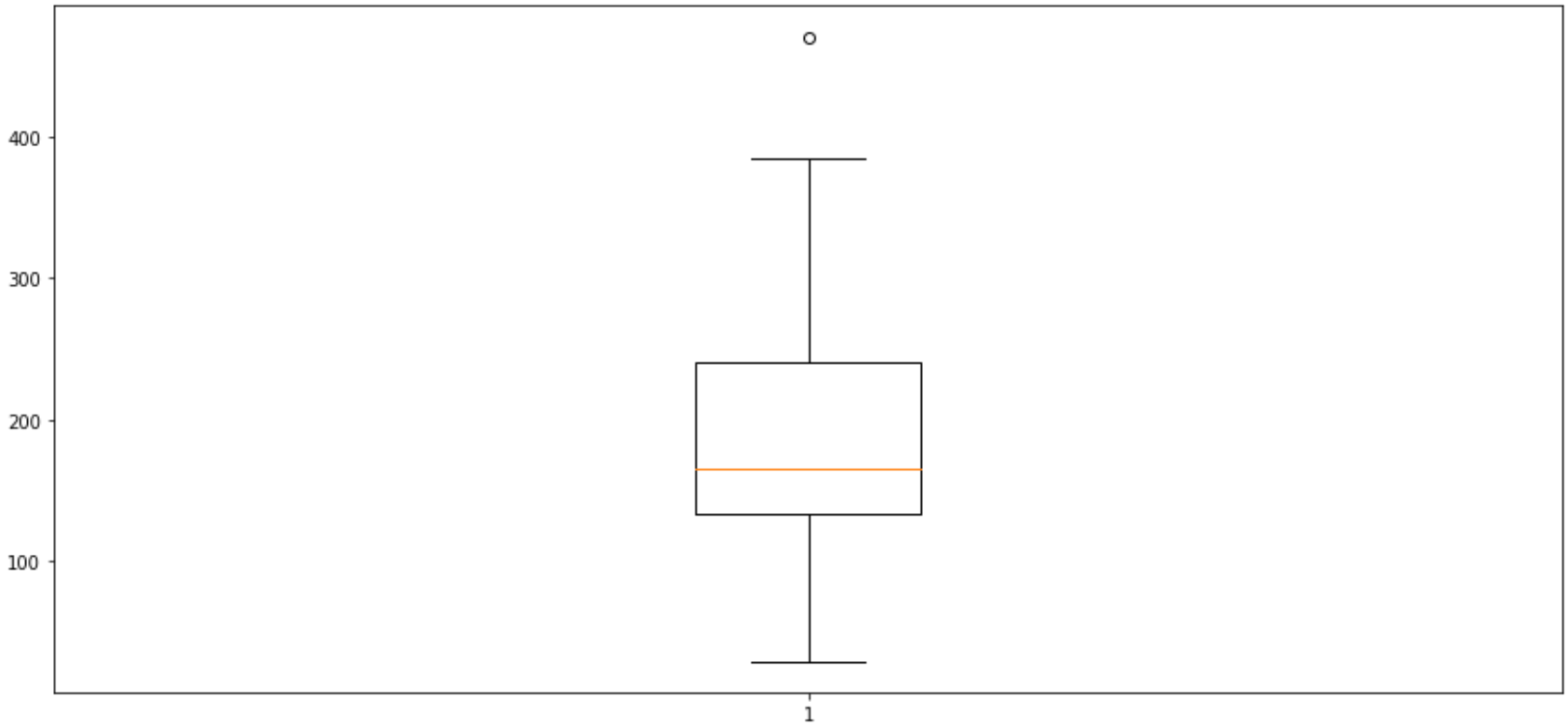
	Column_A
count	31.000000
mean	184.096774
std	91.510420
min	30.000000
25%	134.000000
50%	166.000000
75%	241.000000
max	470.000000

```
IQR=Q3-Q1
```

```
print(IQR)
```

```
     107.0

fig·=·plt.figure(figsize·=(15,·7))
·
#·Creating·plot
plt.boxplot(XD)
·
#·show·plot
plt.show()
```



Zad 5

```
dataEx5 = pd.read_csv('SkokiNarciarskie.csv', delimiter=",")
print(dataEx5)
```

```
        x0    x1  n
```

```
1  105  110   1
2  110  115   6
3  115  120  10
4  120  125  10
5  125  130  15
6  130  135   2
7  135  140   6
```

```
dataEx5.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7 entries, 1 to 7
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   x0      7 non-null      int64
 1   x1      7 non-null      int64
 2   n       7 non-null      int64
dtypes: int64(3)
memory usage: 224.0 bytes
```

```
dataEx5.head()
```

	x0	x1	n
1	105	110	1
2	110	115	6
3	115	120	10
4	120	125	10
5	125	130	15

```
dataEx5.describe()
```

	x0	x1	n
count	7.000000	7.000000	7.000000
mean	120.000000	125.000000	7.142857
std	10.801234	10.801234	4.913538
min	105.000000	110.000000	1.000000
25%	112.500000	117.500000	4.000000
50%	120.000000	125.000000	6.000000

```
for zjebane_gowno_Twojej_Strej in dataEx5:
  print(zjebane_gowno_Twojej_Strej)
```

```
    x0
    x1
    n
```

```
test=[]
for index, row in dataEx5.iterrows():
  print((row[0]+row[1])/2)
  for i in range(row[2]):
    print("test",i)
    test.append((row[0]+row[1])/2)
```

```
    107.5
    test 0
    112.5
    test 0
    test 1
    test 2
    test 3
    test 4
    test 5
    117.5
    test 0
    test 1
    test 2
    test 3
    test 4
```

```
test 5
test 6
test 7
test 8
test 9
122.5
test 0
test 1
test 2
test 3
test 4
test 5
test 6
test 7
test 8
test 9
127.5
test 0
test 1
test 2
test 3
test 4
test 5
test 6
test 7
test 8
test 9
test 10
test 11
test 12
test 13
test 14
132.5
test 0
test 1
137.5
test 0
test 1
test 2
test 3
test 4
test 5
```

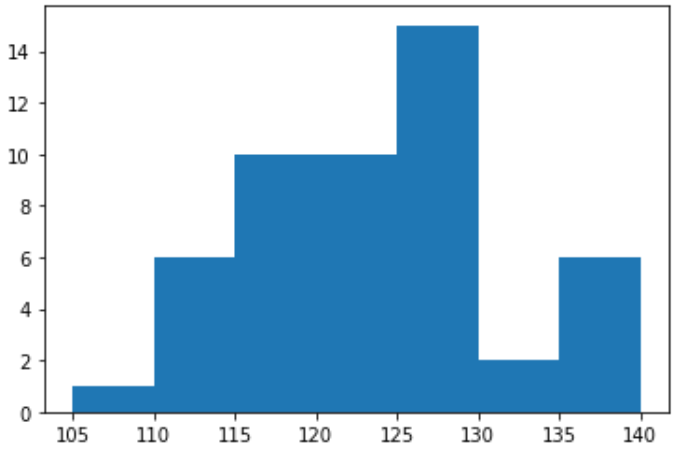
```
test
```

```
[107.5,
 112.5,
 112.5,
 112.5,
 112.5,
 112.5,
 112.5,
 117.5,
 117.5,
 117.5,
 117.5,
 117.5,
 117.5,
 117.5,
 117.5,
 117.5,
 117.5,
 122.5,
 122.5,
 122.5,
 122.5,
 122.5,
 122.5,
 122.5,
 122.5,
 122.5,
 122.5,
 122.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
 127.5,
```

```
 132.5,
 132.5,
 137.5,
 137.5,
 137.5,
 137.5,
 137.5,
 137.5]
```

```
plt.hist(test, bins = [105,110,115,120,125,130,135,140])
```

```
(array([ 1.,  6., 10., 10., 15.,  2.,  6.]),
 array([105, 110, 115, 120, 125, 130, 135, 140]),
 <a list of 7 Patch objects>)
```




```
df = pd.DataFrame(test, columns = ['Column_A'])
```

```
df.mean()
```

```
Column_A    123.7
dtype: float64
```

```
df.median()
```

```
Column_A    122.5
dtype: float64
```

```
dominant=df['Column_A'].value_counts()
print(dominant)
```

```
127.5    15
117.5    10
122.5    10
112.5     6
137.5     6
132.5     2
107.5     1
Name: Column_A, dtype: int64
```

```
max(dominant)
```

```
15
```

```
skew(df)
```

```
array([0.12821351])
```

```
np.quantile(test,0.0)
```

```
107.5
```

```
Q1=np.quantile(test,0.25)
```

```
Q2=np.quantile(test,0.5)
```

```
Q3=np.quantile(test,0.75)
```

```
Q4=np.quantile(test,1)
```

```
df.describe()
```

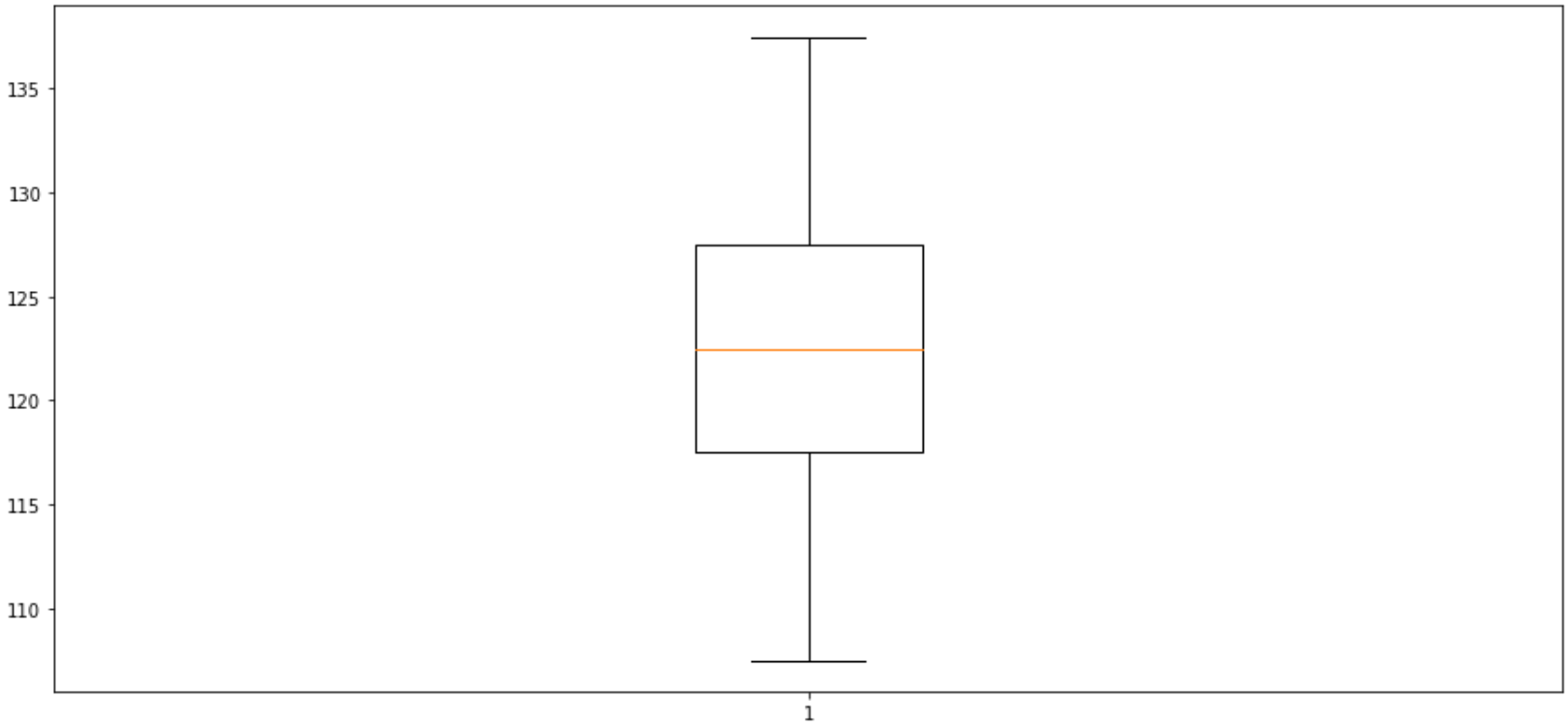
	Column_A
count	50.000000
mean	123.700000
std	7.730142
min	107.500000
25%	117.500000
50%	122.500000
75%	127.500000
max	137.500000

```
IQR=Q3-Q1


print(IQR)
```

```
    10.0
```

```
fig = plt.figure(figsize =(15, 7))
plt.boxplot(test)
plt.show()
```



✓ 0 s ukończono o 21:07