Ahmet Burak Şişci

In [ ]:
```
"""
<b> - Bold text
<strong> - Important text
<i> - Italic text
<em> - Emphasized text
<mark> - Marked text
<small> - Smaller text
<del> - Deleted text
<ins> - Inserted text
<sub> - Subscript text
<sup> - Superscript text
    """
```

Out[ ]:
```
'\n<b> - Bold text\n<strong> - Important text\n<i> - Italic text\n<em> - Emphas
ized text\n<mark> - Marked text\n<small> - Smaller text\n<del> - Deleted text\n
<ins> - Inserted text\n<sub> - Subscript text\n<sup> - Superscript text\n    '
```

In [ ]:
```
print("hello world")
```

```
hello world
```

tip deneme

escape + m --> markdown satırı

In [ ]:
```
# escape + y --> kod satırı
```

In [ ]:
```
a=3
b=4
```

In [ ]:
```
a+b
```

Out[ ]:
```
7
```

In [ ]:
```
c=a+b
c
```

Out[ ]:
```
7
```

In [ ]:
```
c
```

Out[ ]:
```
7
```

In [ ]:
```
a,c
```

Out[ ]:
```
(3, 7)
```

In [ ]:
```
a,b,c
```

Out[ ]:
```
(3, 4, 7)
```

In [ ]:
```
d = "Ahmet Burak"
```

In [ ]:
```
d
```

Out[ ]:
```
'Ahmet Burak'
```

In [ ]:
```
a,b,c,d
```

Out[ ]:
```
(3, 4, 7, 'Ahmet Burak')
```

nicee

html kuralları markdown yazarken geçerli.

In [ ]:
```
list1 = [1,2,3,5,6,4,3,35,78,5,3,56,78,5,3,5,7,4,3]

sonuc = {}
```

In [ ]:
```
for i in list1:
    if i in sonuc:
        sonuc[i] += 1
    else:
        sonuc[i] = 1

sonuc
```

Out[ ]:
```
{1: 1, 2: 1, 3: 5, 5: 4, 6: 1, 4: 2, 35: 1, 78: 2, 56: 1, 7: 1}
```

In [ ]:
```
import pandas as pd # pandas ile okuma yap
```

In [ ]:
```
df = pd.read_csv("diabetes_clean.csv")

df
```

Out[ ]:

	pregnancies	glucose	diastolic	triceps	insulin	bmi	dpf	age	diabetes
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

768 rows × 9 columns

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   pregnancies  768 non-null    int64  
 1   glucose      768 non-null    int64  
 2   diastolic    768 non-null    int64  
 3   triceps      768 non-null    int64  
 4   insulin      768 non-null    int64  
 5   bmi          768 non-null    float64
 6   dpf          768 non-null    float64
 7   age          768 non-null    int64  
 8   diabetes     768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [ ]:
```
"""ders 1 son"""
```

Out[ ]:
```
'ders 1 son'
```

In [ ]:
```
df.head()
```

Out[ ]:

	pregnancies	glucose	diastolic	triceps	insulin	bmi	dpf	age	diabetes
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

In [ ]:
```
df.head(30)  # ilk satırları verir.
```

Out[ ]:

	pregnancies	glucose	diastolic	triceps	insulin	bmi	dpf	age	diabetes
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
5	5	116	74	0	0	25.6	0.201	30	0
6	3	78	50	32	88	31.0	0.248	26	1
7	10	115	0	0	0	35.3	0.134	29	0
8	2	197	70	45	543	30.5	0.158	53	1
9	8	125	96	0	0	0.0	0.232	54	1
10	4	110	92	0	0	37.6	0.191	30	0
11	10	168	74	0	0	38.0	0.537	34	1
12	10	139	80	0	0	27.1	1.441	57	0
13	1	189	60	23	846	30.1	0.398	59	1
14	5	166	72	19	175	25.8	0.587	51	1
15	7	100	0	0	0	30.0	0.484	32	1
16	0	118	84	47	230	45.8	0.551	31	1
17	7	107	74	0	0	29.6	0.254	31	1
18	1	103	30	38	83	43.3	0.183	33	0
19	1	115	70	30	96	34.6	0.529	32	1
20	3	126	88	41	235	39.3	0.704	27	0
21	8	99	84	0	0	35.4	0.388	50	0
22	7	196	90	0	0	39.8	0.451	41	1
23	9	119	80	35	0	29.0	0.263	29	1
24	11	143	94	33	146	36.6	0.254	51	1
25	10	125	70	26	115	31.1	0.205	41	1
26	7	147	76	0	0	39.4	0.257	43	1
27	1	97	66	15	140	23.2	0.487	22	0
28	13	145	82	19	110	22.2	0.245	57	0
29	5	117	92	0	0	34.1	0.337	38	0

In [ ]:
```
df.tail()
```

Out[ ]:

	pregnancies	glucose	diastolic	triceps	insulin	bmi	dpf	age	diabetes
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

In [ ]:
```
df.info() # bilgi verir işte

# boş cell var mı?
# satır sayısı.
# sütunların başlıkları.
# veri tipleri
# memory usage = bellekte kapladığı alan.
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   pregnancies  768 non-null    int64  
 1   glucose      768 non-null    int64  
 2   diastolic    768 non-null    int64  
 3   triceps      768 non-null    int64  
 4   insulin      768 non-null    int64  
 5   bmi          768 non-null    float64
 6   dpf          768 non-null    float64
 7   age          768 non-null    int64  
 8   diabetes     768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [ ]:
```
df.describe()
# özet istatistikleri verir. okunması zor olduğu için transpozunu alarak okunur
```

Out[ ]:

	pregnancies	glucose	diastolic	triceps	insulin	bmi	
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.47
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.33
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420

In [ ]:
```
df.describe().T # sayısal veriler.
```

Out[ ]:

	count	mean	std	min	25%	50%	75%	ma
pregnancies	768.0	3.845052	3.369578	0.000	1.00000	3.0000	6.00000	17.(
glucose	768.0	120.894531	31.972618	0.000	99.00000	117.0000	140.25000	199.(
diastolic	768.0	69.105469	19.355807	0.000	62.00000	72.0000	80.00000	122.(
triceps	768.0	20.536458	15.952218	0.000	0.00000	23.0000	32.00000	99.(
insulin	768.0	79.799479	115.244002	0.000	0.00000	30.5000	127.25000	846.(
bmi	768.0	31.992578	7.884160	0.000	27.30000	32.0000	36.60000	67.
dpf	768.0	0.471876	0.331329	0.078	0.24375	0.3725	0.62625	2.4
age	768.0	33.240885	11.760232	21.000	24.00000	29.0000	41.00000	81.(
diabetes	768.0	0.348958	0.476951	0.000	0.00000	0.0000	1.00000	1.(

In [ ]:
```
df.glucose   # sadece belirli bir sütundaki bilgileri getirir.
```

Out[ ]:
```
0      148
1       85
2      183
3       89
4      137
      ...
763    101
764    122
765    121
766    126
767     93
Name: glucose, Length: 768, dtype: int64
```

In [ ]:
```
df["glucose"]   # isimde boşluk karakteri varsa bu şekil yaz
```

Out[ ]:
```
0      148
1       85
2      183
3       89
4      137
      ...
763    101
764    122
765    121
766    126
767     93
Name: glucose, Length: 768, dtype: int64
```

In [ ]:
```
df[["glucose","age"]]
```

Out[ ]:

	glucose	age
0	148	50
1	85	31
2	183	32
3	89	21
4	137	33
...	...	...
763	101	63
764	122	27
765	121	30
766	126	47
767	93	23

768 rows × 2 columns

In [ ]:
```
df[(df.diabetes==1) & (df.age>40)]
```

Out[ ]:

	pregnancies	glucose	diastolic	triceps	insulin	bmi	dpf	age	diabetes
0	6	148	72	35	0	33.6	0.627	50	1
8	2	197	70	45	543	30.5	0.158	53	1
9	8	125	96	0	0	0.0	0.232	54	1
13	1	189	60	23	846	30.1	0.398	59	1
14	5	166	72	19	175	25.8	0.587	51	1
...	...	...	...	...	...	...	...	...	...
754	8	154	78	32	0	32.4	0.443	45	1
757	0	123	72	0	0	36.3	0.258	52	1
759	6	190	92	0	0	35.5	0.278	66	1
761	9	170	74	31	0	44.0	0.403	43	1
766	1	126	60	0	0	30.1	0.349	47	1

102 rows × 9 columns

hocanın slayt notları:

istatistik 2 ye ayrılır. bunları bil.

(sınavda yorum soruları olacak. bunlar önemli.)

veri türleri:

1. sürekli veriler(continuous): tam sayı olmayan, analog veriler. sürekli akar. hisse senedi fiyatalrı.
2. ayrık veriler: tma sayıdır. satış adeti, sınıftaki öğrenci sayısı vb.

bunları bil:

1. static type programing language
2. dynemaic type programing language

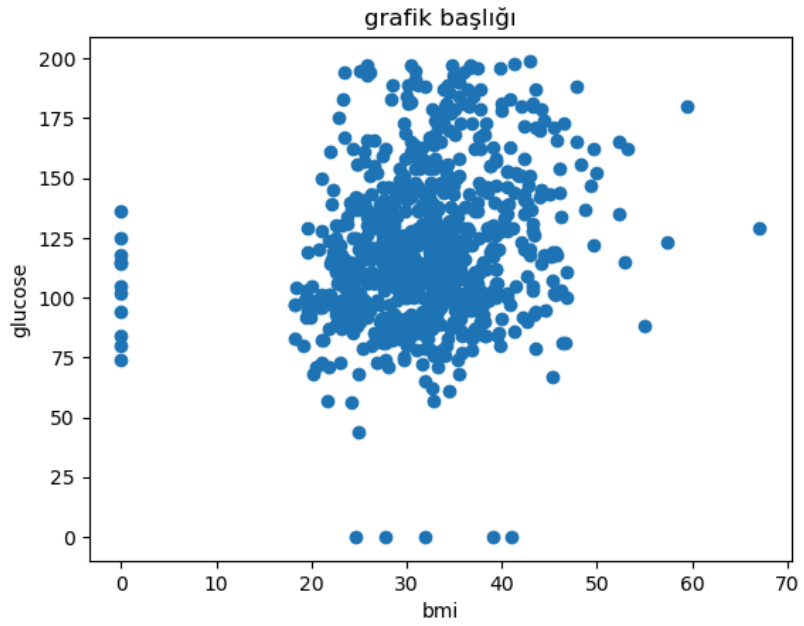
numerik verileri görselleştirme:

matplotlib kütüphanesi kullanılır. iki veri arasındaki ilişkiyi verir.

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

In [ ]:
```
plt.title("grafik başlığı")
plt.xlabel('bmi')
plt.ylabel('glucose')
plt.scatter(x=df.bmi, y=df.glucose)
```

Out[ ]:
```
<matplotlib.collections.PathCollection at 0x746fc00d7e00>
```

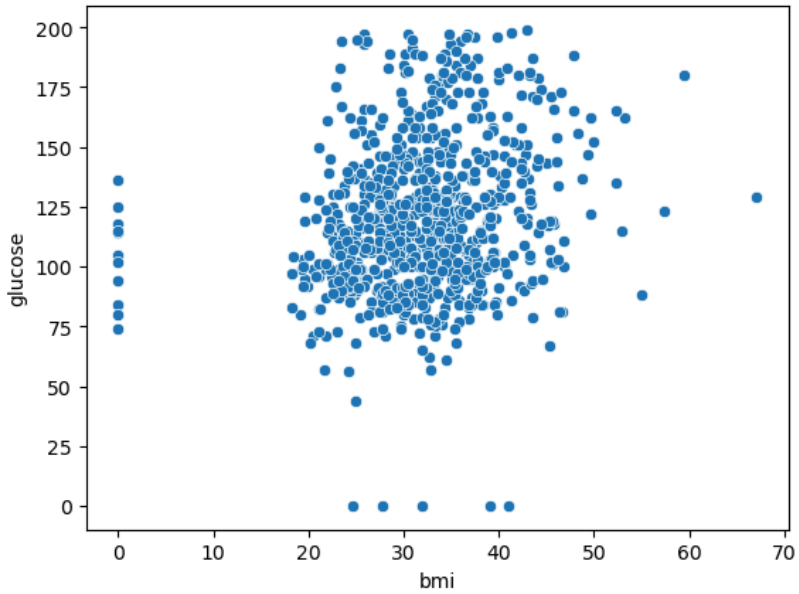


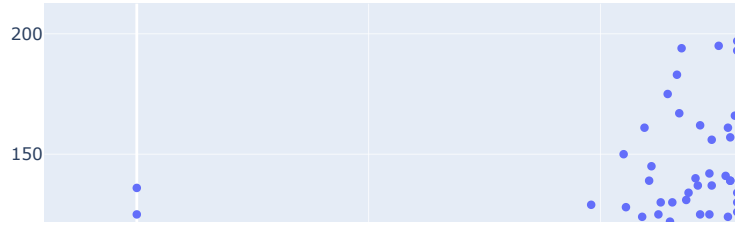
In [ ]:
```
# daha iyi görselliği olan seaborn kütüphanesi de grafik çizmek içim kullanılır.
```

In [ ]:
```
import seaborn as sns
```

In [ ]:
```
import seaborn as sns
sns.scatterplot(data=df, x='bmi', y='glucose')
```

Out[ ]:
```
<Axes: xlabel='bmi', ylabel='glucose'>
```





import plotly.express as px

In [ ]:
```
import plotly.express as px

px.scatter(data_frame=df, x='bmi', y='glucose')
```

descriptive statistic: ...kaçırıdm bu kısmı...

inferential statistic: bir çıkarım yapmamız gerekir

"facebooktaki insanların %25 i reklam görünce kyafet alır" çıkarımı

merkez ölçümleri:

çünürde ev fiyatları.
türkiyede en yaygın saç tipi.

(bu tür verilerde tam netlik olmaz. yaklaşım olur.)

In [ ]:
```
import pandas as pd
df_animal = pd.read_csv("msleep.csv")
df_animal
```

Out[ ]:

	name	genus	vore	order	conservation	sleep_total	sleep_rem	sle
0	Cheetah	Acinonyx	carni	Carnivora	lc	12.1	NaN	
1	Owl monkey	Aotus	omni	Primates	NaN	17.0	1.8	
2	Mountain beaver	Aplodontia	herbi	Rodentia	nt	14.4	2.4	
3	Greater short-tailed shrew	Blarina	omni	Soricomorpha	lc	14.9	2.3	
4	Cow	Bos	herbi	Artiodactyla	domesticated	4.0	0.7	
...	...	...	...	...	...	...	...	
78	Tree shrew	Tupaia	omni	Scandentia	NaN	8.9	2.6	
79	Bottle-nosed dolphin	Tursiops	carni	Cetacea	NaN	5.2	NaN	
80	Genet	Genetta	carni	Carnivora	NaN	6.3	1.3	
81	Arctic fox	Vulpes	carni	Carnivora	NaN	12.5	NaN	
82	Red fox	Vulpes	carni	Carnivora	NaN	9.8	2.4	

83 rows × 11 columns

In [ ]:
```
import numpy as np
# ortalama
# np.mean(df_animal.sleep_total)
df_animal['sleep_total'].mean()
```

Out[ ]:
```
np.float64(10.433734939759034)
```

In [ ]:
```
import numpy as np
# medyan
np.median(df_animal.sleep_total), # numpy
df_animal.sleep_total.median() # pandas
```

Out[ ]:
```
10.1
```

In [ ]:
```
# mod yöntem1
df_animal.sleep_total.value_counts()
```

Out[ ]:
```
sleep_total
12.5    4
10.1    3
9.1     2
8.7     2
14.4    2
       ..
8.6     1
4.4     1
15.6    1
8.9     1
5.2     1
Name: count, Length: 65, dtype: int64
```

In [ ]:
```
#mode yöntem2
df_animal.vore.value_counts(dropna=False)
```

Out[ ]:
```
vore
herbi      32
omni       20
carni      19
NaN         7
insecti     5
Name: count, dtype: int64
```

In [ ]:
```
import statistics as stat

stat.mode(df_animal.vore), df_animal.vore.mode()
```

Out[ ]:
```
('herbi',
 0    herbi
 Name: vore, dtype: object)
```

In [ ]:
```
df_animal[df_animal.vore == "insecti"] # df_animal içinden "insecti" olanları çe
```

Out[ ]:

	name	genus	vore	order	conservation	sleep_total	sleep_rem
21	Big brown bat	Eptesicus	insecti	Chiroptera	lc	19.7	3.9
42	Little brown bat	Myotis	insecti	Chiroptera	NaN	19.9	2.0
61	Giant armadillo	Priodontes	insecti	Cingulata	en	18.1	6.1
66	Eastern american mole	Scalopus	insecti	Soricomorpha	lc	8.4	2.1
74	Short-nosed echidna	Tachyglossus	insecti	Monotremata	NaN	8.6	NaN

In [ ]:
```
df_animal[df_animal.vore == "insecti"]["sleep_rem"].agg(["mean", "median"])
# beslenme türü (vore) "böcekçil" (insecti) olan hayvanların
```

```
# REM uykusu (sleep_rem) sürelerinin ortalamasını ve medyanını (ortanca değerini
```

Out[ ]:
```
mean      3.525
median    3.000
Name: sleep_rem, dtype: float64
```

In [ ]:
```
# add outlier # en son satıra ekleme
df_animal.loc[len(df_animal.index)] = ["New Insect", "", "insecti", "", "", 0.0,
df_animal
```

Out[ ]:

	name	genus	vore	order	conservation	sleep_total	sleep_rem	sl
0	Cheetah	Acinonyx	carni	Carnivora	lc	12.1	NaN	
1	Owl monkey	Aotus	omni	Primates	NaN	17.0	1.8	
2	Mountain beaver	Aplodontia	herbi	Rodentia	nt	14.4	2.4	
3	Greater short-tailed shrew	Blarina	omni	Soricomorpha	lc	14.9	2.3	
4	Cow	Bos	herbi	Artiodactyla	domesticated	4.0	0.7	
...	...	...	...	...	...	...	...	
79	Bottle-nosed dolphin	Tursiops	carni	Cetacea	NaN	5.2	NaN	
80	Genet	Genetta	carni	Carnivora	NaN	6.3	1.3	
81	Arctic fox	Vulpes	carni	Carnivora	NaN	12.5	NaN	
82	Red fox	Vulpes	carni	Carnivora	NaN	9.8	2.4	
83	New Insect		insecti			0.0	0.0	

84 rows × 11 columns

In [ ]:
```
df_animal[df_animal.vore == "insecti"]["sleep_rem"].agg(["mean", "median"])
```

Out[ ]:
```
mean      2.82
median    2.10
Name: sleep_rem, dtype: float64
```

ödev

Örneğin, kazakların maliyeti 30 TL ise ancak 10-200 TL aralığında başka yerlerden alınabiliyorsa, 30 TL'den bir tane bulma olasılığımız nedir?

Kazak fiyatları 20-50 TL arasında olursa bu olasılık değişir mi?

Soru 1: Örneğin, kazakların maliyeti 30 TL ise ancak 10-200 TL aralığında başka yerlerden alınabiliyorsa, 30 TL'den bir tane bulma olasılığımız nedir? Bu soruyu cevaplamak için kazak fiyatlarının dağılımı hakkında daha fazla bilgiye ihtiyaç vardır. Eğer fiyatların 10 TL ile 200 TL arasında düzgün bir dağılıma sahip olduğunu varsayarsak (yani her fiyatın gelme olasılığı eşitse), bu durumda belirli bir fiyata (30 TL) rastlama olasılığı teorik olarak sıfıra çok yakındır. Sürekli bir dağılımda tek bir noktanın olasılığı sıfır kabul edilir. Ancak, eğer kazak fiyatları tamsayı değerler alıyorsa (10, 11, 12, ..., 200 TL gibi), bu aralıkta toplam 191 farklı fiyat bulunur. Bu durumda 30 TL'lik bir kazak bulma olasılığı 1/191 olur.
Soru 2: Kazak fiyatları 20-50 TL arasında olursa bu olasılık değişir mi? Evet, olasılık değişir. Fiyat aralığı daraldığı için, belirli bir fiyattaki kazağı bulma olasılığı artar. Yine fiyatların tamsayı olduğunu varsayarsak, 20 TL ile 50 TL arasında 31 farklı fiyat değeri vardır (50 - 20 + 1 = 31). Bu durumda 30 TL'lik bir kazak bulma olasılığı 1/31'e yükselir.

varyans nedir?

```
Varyans, bir veri setindeki her bir değerin, o setin
ortalamasından ne kadar uzakta olduğunun karelerinin
ortalamasıdır. Standart
sapma gibi, verilerin ne kadar yayıldığını ölçer.
```

standart sapma nedir?

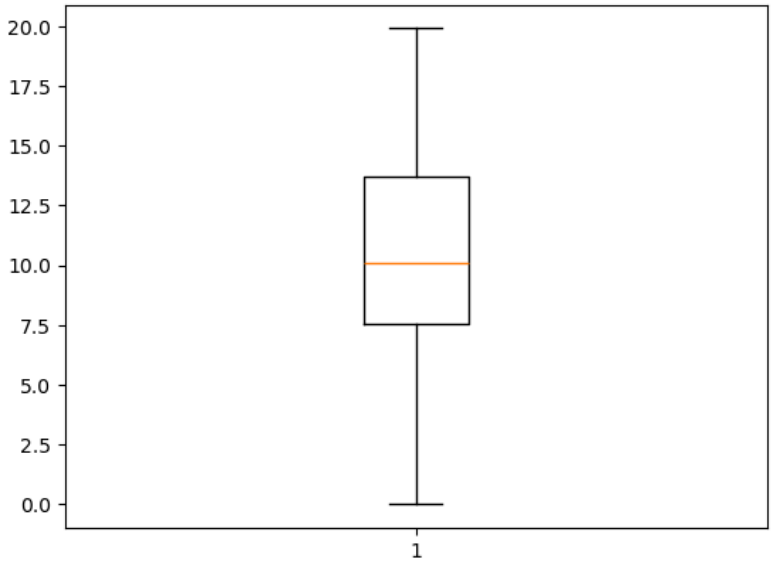
```
Standart sapma, bir veri setindeki değerlerin, o setin
ortalamasından ne kadar uzakta yayıldığını gösteren bir ölçüdür
```

quartiles() nedir?

```
Kuartiller (Quartiles), sıralanmış bir veri setini dört eşit
parçaya bölen üç değerdir. (çeyreklikler)
```

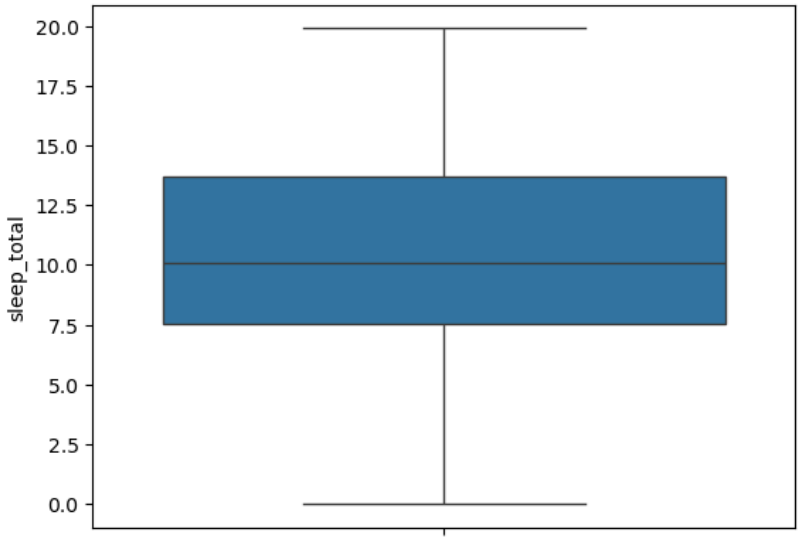
In [ ]:
```
plt.boxplot(df_animal.sleep_total)
```

Out[ ]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x746fba7a2990>,
  <matplotlib.lines.Line2D at 0x746fba7a2ad0>],
 'caps': [<matplotlib.lines.Line2D at 0x746fba7a2c10>,
  <matplotlib.lines.Line2D at 0x746fba7a2d50>],
 'boxes': [<matplotlib.lines.Line2D at 0x746fba7a2850>],
 'medians': [<matplotlib.lines.Line2D at 0x746fba7a2e90>],
 'fliers': [<matplotlib.lines.Line2D at 0x746fba7a2fd0>],
 'means': []}
```



In [ ]:
```
sns.boxplot(data=df_animal, y='sleep_total')
```

Out[ ]:
```
<Axes: ylabel='sleep_total'>
```



In [ ]:
```
# daha görsel olmasını istersem:
px.box(data_frame=df_animal, y='sleep_total')
```

In [ ]:
```
import numpy as np
np.var(df_animal.sleep_total, ddof=1)
```

Out[ ]:
```
20.863040734366034
```

In [ ]:
```
np.sqrt(np.var(df_animal.sleep_total, ddof=1))
```

Out[ ]:
```
np.float64(4.56760776932149)
```

In [ ]:
```
np.std(df_animal.sleep_total, ddof=1)
```

Out[ ]:
```
4.56760776932149
```

ödev 2

yukarıdaki 3 cell de "ddof=1" nedir

ddof=1'in anlamı şudur: Hesaplama yapılırken, veri sayısının (n) kendisi yerine (n-1) değerinin kullanılmasını sağlar. Bu, istatistikte "Örneklem Standart Sapması" veya "Örneklem Varyansı" olarak bilinen hesaplama yöntemidir.

In [ ]:
```
# Mean Absolute Deviation
# Ortalama Mutlak Sapma (translate)
```

```
dists = df_animal.sleep_total - \
np.mean(df_animal.sleep_total)
np.mean(np.abs(dists))
```

Out[ ]:
```
np.float64(3.6387755102040815)
```

Mean Absolute Deviation (Ortalama Mutlak Sapma), bir veri setindeki her bir değerin, o setin ortalamasından mutlak olarak ne kadar saptığının ortalamasıdır.

In [ ]: