```
import pandas as pd
import numpy as np

# Read filenya
df = pd.read_csv (r'Gandum.csv')
print(len(df))
print (df)


def describe(df):
    print( pd.concat([df.describe().T,
                      df.var().rename('variansi'),
                      df.median().rename('median'),
                      df.skew().rename('skewness'),
                      df.kurt().rename('kurtosis'),
                     ], axis=1).T)
    print("\nDATA MODUS ")
    print(df.mode())

    print("\nDATA RANGE ")
    minimum = df.min()
    maksimum = df.max()
    print(maksimum-minimum)

    print("\nDATA IQR ")
    Q1 = df.quantile(0.25)
    Q3 = df.quantile(0.75)
    IQR = Q3 - Q1
    print(IQR)


describe(df)
```

```
500
      id  Daerah  SumbuUtama  SumbuKecil  ...  Keliling   Bulatan    Ransum  Kelas
0      1    5781  128.288875   58.470846  ...   316.756  0.724041  2.194066      1
1      2    4176  109.348294   49.837688  ...   260.346  0.774227  2.194088      1
2      3    4555  114.427991   52.151207  ...   279.606  0.732159  2.194158      1
3      4    4141  108.701191   49.457349  ...   260.478  0.766960  2.197877      1
4      5    5273  122.747869   55.757848  ...   302.730  0.723031  2.201446      1
..   ...     ...         ...         ...  ...       ...       ...       ...    ...
495  496    5083  120.083450   54.821580  ...   286.377  0.778850  2.190441      2
496  497    4432  112.367050   51.294914  ...   270.823  0.759344  2.190608      2
497  498    5020  119.873742   54.718545  ...   285.799  0.772311  2.190733      2
498  499    4035  107.311728   48.930802  ...   258.503  0.758791  2.193132      2
499  500    3379   99.014789   44.631551  ...   237.593  0.752196  2.218493      2

[500 rows x 12 columns]
                    id         Daerah  ...      Ransum       Kelas
count       500.000000     500.000000  ...  500.000000  500.000000
mean        250.500000    4801.246000  ...    2.150915    1.502000
```

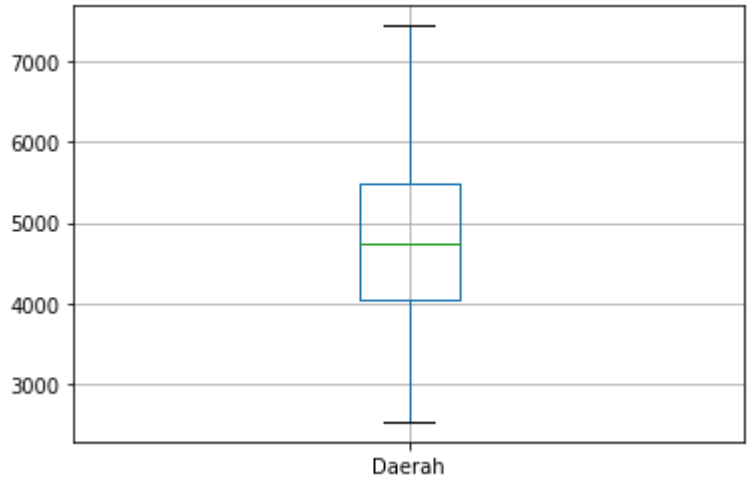
```
std         144.481833     986.395491  ...    0.249767    0.500497
min           1.000000    2522.000000  ...    1.440796    1.000000
25%         125.750000    4042.750000  ...    1.983939    1.000000
50%         250.500000    4735.000000  ...    2.193599    2.000000
75%         375.250000    5495.500000  ...    2.381612    2.000000
max         500.000000    7453.000000  ...    2.464809    2.000000
variansi  20875.000000  972976.065615  ...    0.062383    0.250497
median      250.500000    4735.000000  ...    2.193599    2.000000
skewness      0.000000       0.238144  ...   -0.658188   -0.008024
kurtosis     -1.200000      -0.434631  ...   -0.428656   -2.007984

[12 rows x 12 columns]

DATA MODUS
      id  Daerah  SumbuUtama  SumbuKecil  ...  Keliling   Bulatan    Ransum  Kelas
0      1  3992.0   74.133114   39.906517  ...   197.015  0.174590  1.440796    2.0
1      2  4881.0   74.364021   41.436419  ...   200.587  0.261297  1.453137    NaN
2      3  5642.0   74.691881   42.871879  ...   202.456  0.299298  1.465950    NaN
3      4  6083.0   76.293164   43.284979  ...   207.325  0.589146  1.483456    NaN
4      5     NaN   76.789043   44.119355  ...   207.697  0.603807  1.510000    NaN
..   ...     ...         ...         ...  ...       ...       ...       ...    ...
495  496     NaN  152.068440   63.322854  ...   375.651  0.872417  2.461017    NaN
496  497     NaN  152.113491   63.762307  ...   390.125  0.874243  2.461510    NaN
497  498     NaN  153.583387   64.012769  ...   434.235  0.874743  2.463297    NaN
498  499     NaN  227.105462   65.738475  ...   448.305  0.891706  2.463546    NaN
499  500     NaN  227.928583   68.977700  ...   488.837  0.904748  2.464809    NaN

[500 rows x 12 columns]

DATA RANGE
id              499.000000
Daerah         4931.000000
SumbuUtama      153.795469
SumbuKecil       29.071182
Keunikan          0.194085
AreaBulatan    5141.000000
Diameter         40.747172
KadarAir          0.468972
Keliling        291.822000
Bulatan           0.730158
Ransum            1.024013
```

```
df[["Daerah"]].plot(kind="hist",bins=[0,1000,2000,3000,4000,5000,6000,7000,8000,9000,10000],r
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89abb99b50>
```

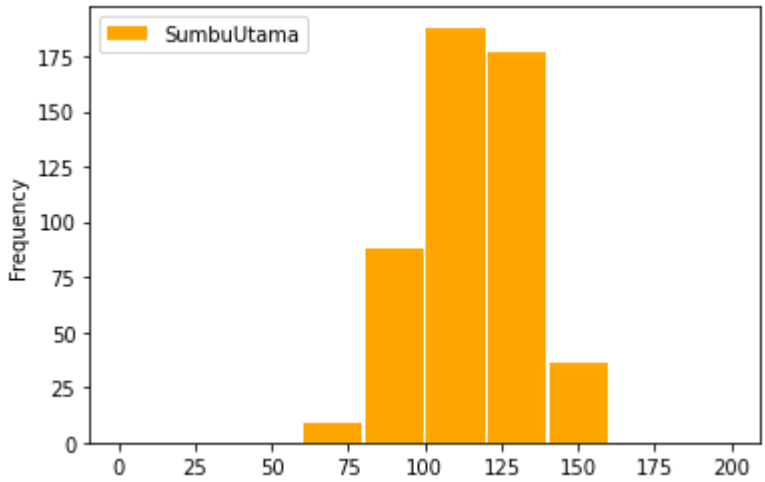

```
df.boxplot(column=['Daerah'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89ab9d8810>
```



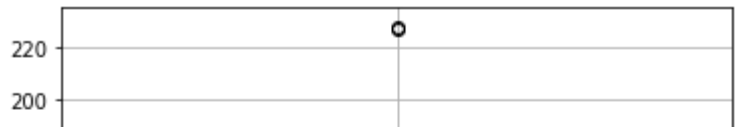

```
df[["SumbuUtama"]].plot(kind="hist",bins=[0,20,40,60,80,100,120,140,160,180,200],rwidth=0.95,
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89ab7771d0>
```



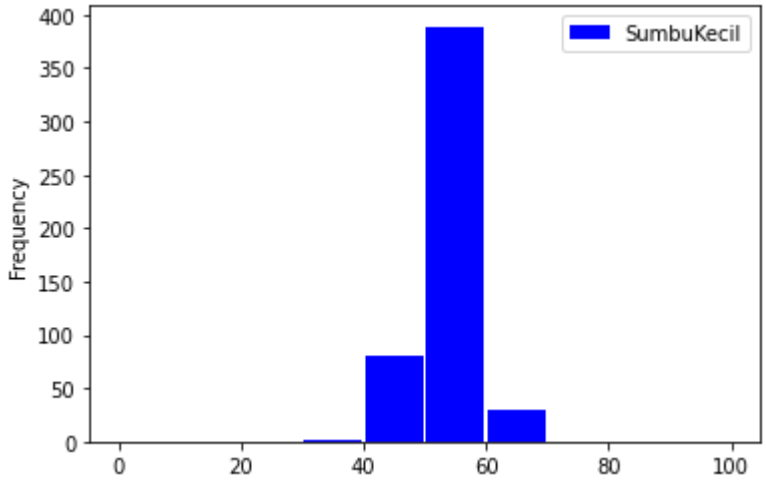

```
df.boxplot(column=['SumbuUtama'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89ab701590>
```



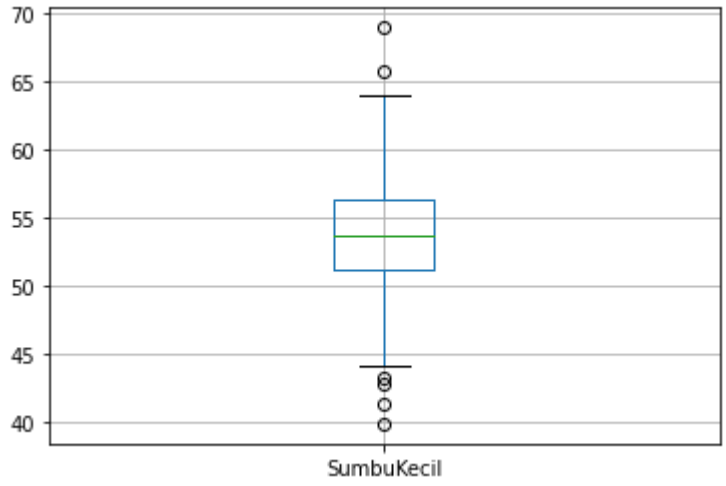

```
df[["SumbuKecil"]].plot(kind="hist",bins=[0,10,20,30,40,50,60,70,80,90,100],rwidth=0.95, colo
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89ab5edb10>
```




```
df.boxplot(column=['SumbuKecil'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89ab57a7d0>
```

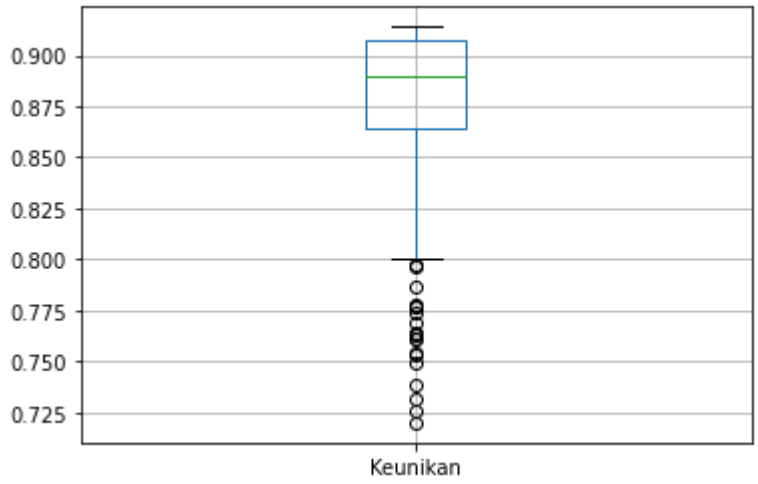



```
df[["Keunikan"]].plot(kind="hist",bins=[0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1],rwidth=0.95,
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89ab2d9f50>
```

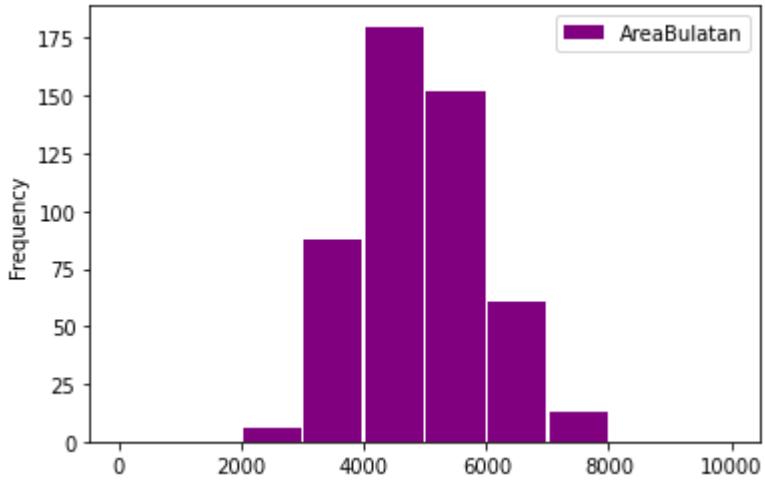

```
df.boxplot(column=['Keunikan'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89ab25f790>
```



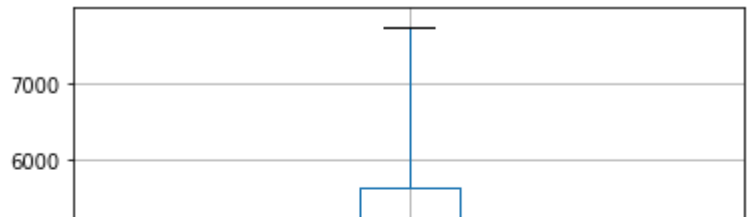

```
df[["AreaBulatan"]].plot(kind="hist",bins=[0,1000,2000,3000,4000,5000,6000,7000,8000,9000,100
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89ab2d33d0>
```



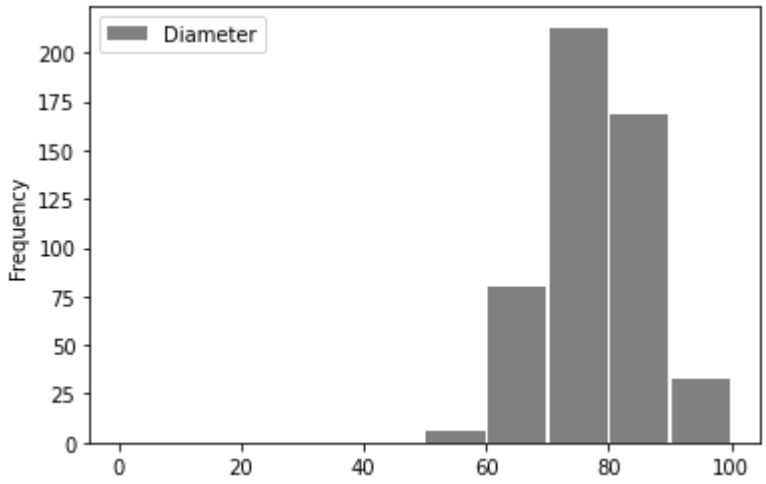

```
df.boxplot(column=['AreaBulatan'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89ab0f0390>
```



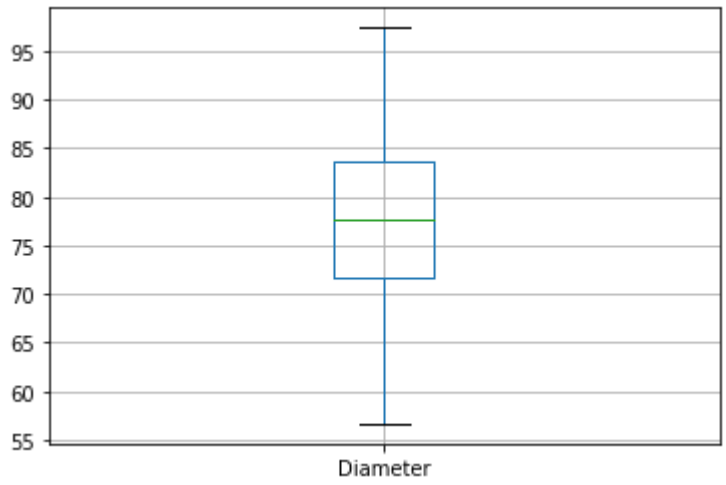

```
df[["Diameter"]].plot(kind="hist",bins=[0,10,20,30,40,50,60,70,80,90,100],rwidth=0.95, color=
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89ab0481d0>
```



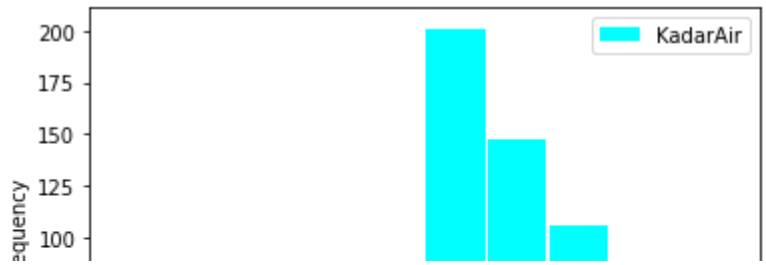

```
df.boxplot(column=['Diameter'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89aaf81790>
```



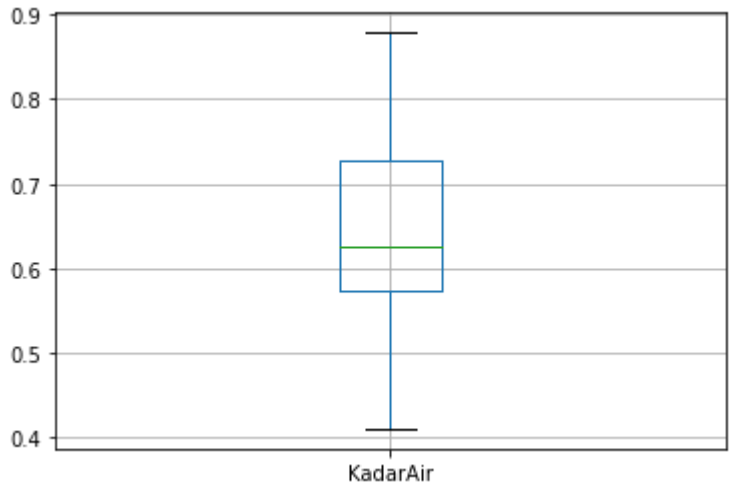

```
df[["KadarAir"]].plot(kind="hist",bins=[0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1],rwidth=0.95,
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89ab569fd0>
```



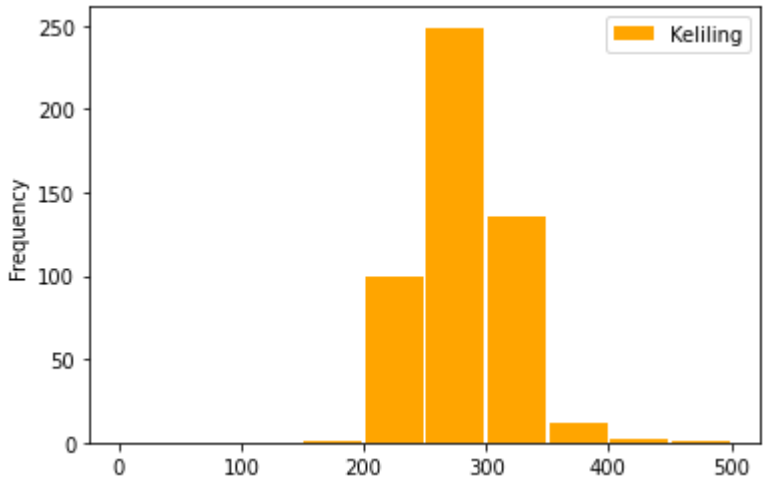

```
df.boxplot(column=['KadarAir'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89aaed2e10>
```



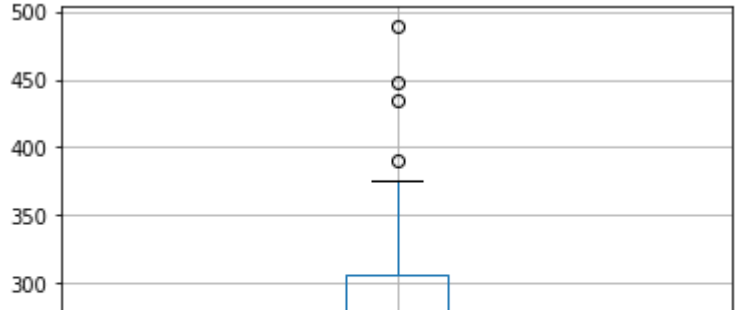

```
df[["Keliling"]].plot(kind="hist",bins=[0,50,100,150,200,250,300,350,400,450,500],rwidth=0.95
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89aae40150>
```



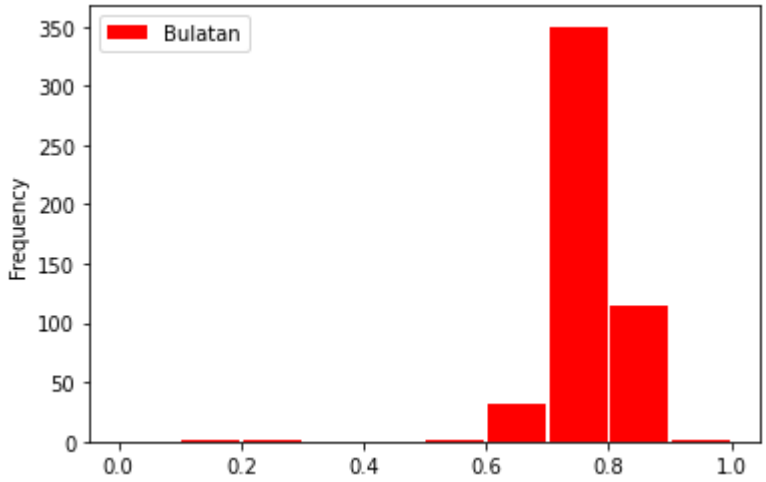

```
df.boxplot(column=['Keliling'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89aadd8210>
```



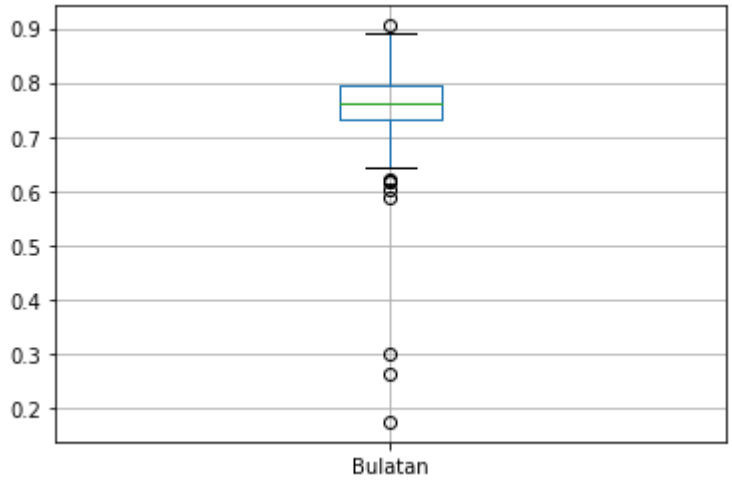

```
df[["Bulatan"]].plot(kind="hist",bins=[0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1],rwidth=0.95,
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89aad4b890>
```



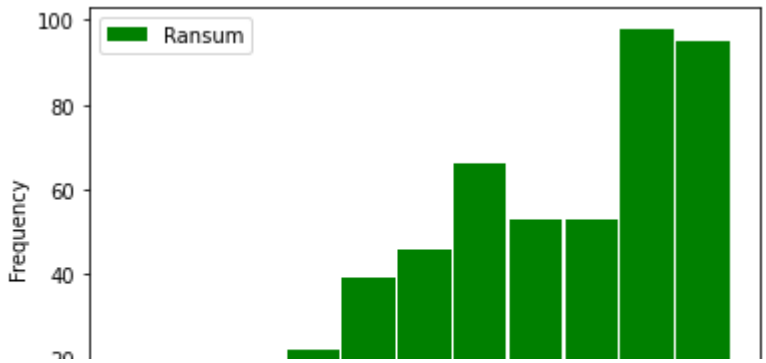

```
df.boxplot(column=['Bulatan'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89aac5e8d0>
```



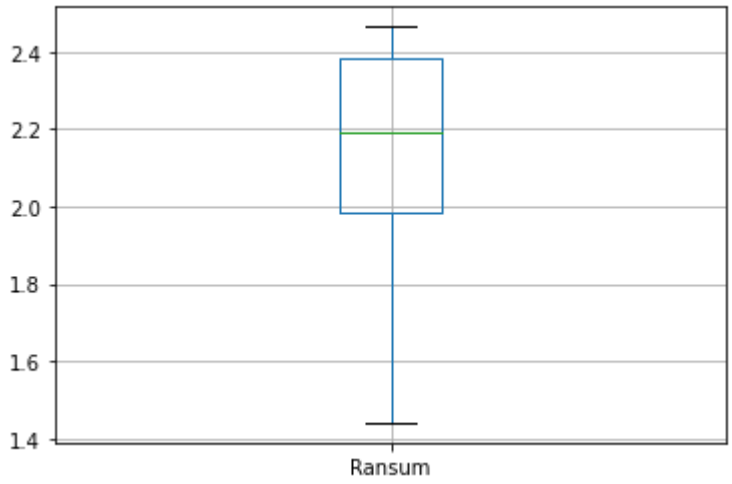

```
df[["Ransum"]].plot(kind="hist",bins=[1.4,1.5,1.6,1.7,1.8,1.9,2,2.1,2.2,2.3,2.4,2.5],rwidth=0
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89aa8ac350>
```




```
df.boxplot(column=['Ransum'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f89aa8acc10>
```




```
# NOMOR 3
import scipy.stats as stats
from scipy.stats import norm, normaltest

def normality_test(column):
  stat, pValue = normaltest(df[column])

  if pValue > 0.05:
    print("Kolom "+str(column) + " berdistribusi normal")
    print("Dengan nilai stat : "+str(stat))
    print("Dengan nilai pValue : "+str(pValue)+"\n")
  else:
    print("Kolom "+str(column) + " tidak berdistribusi normal")
    print("Dengan nilai stat : "+str(stat))
    print("Dengan nilai pValue : "+str(pValue)+"\n")

normality_test("Daerah")
normality_test("SumbuUtama")
normality_test("SumbuKecil")
normality_test("Keunikan")
normality_test("AreaBulatan")
normality_test("Diameter")
normality_test("Kelilling")
```

```
normality_test("Keliling")
normality_test("Bulatan")
normality_test("Ransum")
```

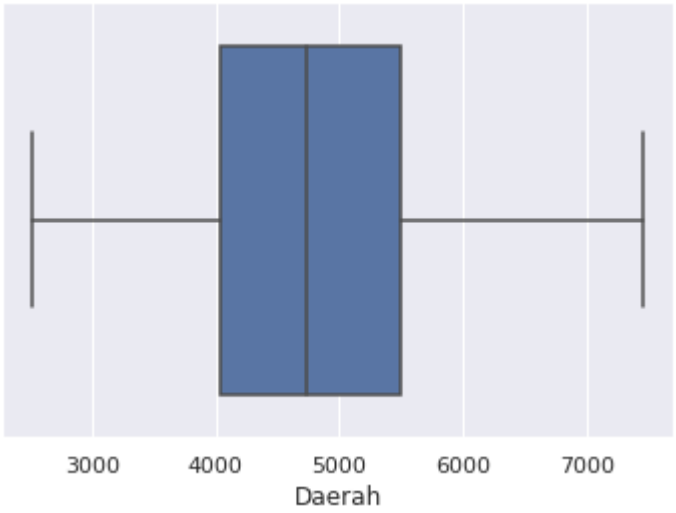
```
Kolom Daerah tidak berdistribusi normal
Dengan nilai stat : 10.858551334227265
Dengan nilai pValue : 0.004386271773193838

Kolom SumbuUtama tidak berdistribusi normal
Dengan nilai stat : 95.12010812035354
Dengan nilai pValue : 2.2127429343273333e-21

Kolom SumbuKecil berdistribusi normal
Dengan nilai stat : 3.698394471986242
Dengan nilai pValue : 0.1573634413290252

Kolom Keunikan tidak berdistribusi normal
Dengan nilai stat : 158.61743886606416
Dengan nilai pValue : 3.602971140062405e-35

Kolom AreaBulatan tidak berdistribusi normal
Dengan nilai stat : 10.738742339140217
Dengan nilai pValue : 0.004657058890055043

Kolom Diameter tidak berdistribusi normal
Dengan nilai stat : 7.446345772505218
Dengan nilai pValue : 0.024157198077543095

Kolom Keliling tidak berdistribusi normal
Dengan nilai stat : 67.46548246324498
Dengan nilai pValue : 2.2390130437166565e-15

Kolom Bulatan tidak berdistribusi normal
Dengan nilai stat : 442.11941526532235
Dengan nilai pValue : 9.885276103161724e-97

Kolom Ransum tidak berdistribusi normal
Dengan nilai stat : 37.060319894694054
Dengan nilai pValue : 8.963008041823752e-09
```

```
import seaborn as sns
sns.set()
# SOAL 4
# Melakukan test hipotesis 1 sampel, dengan menuliskan 6 langkah testing dan menampilkan
# juga boxplotnya untuk kolom/bagian yang bersesuaian.
# a. Nilai rata-rata Daerah di atas 4700?
# b. Nilai Rata-rata Sumbu Utama tidak sama dengan 116?
# c. Nilai Rata-rata 20 baris pertama kolom Sumbu Kecil bukan 50?
# d. Proporsi nilai Diameter yang lebih dari 85, adalah tidak sama dengan 15% ?
# e. Proporsi nilai Keliling yang kurang dari 100, adalah kurang dari 5% ?

sampel = df.sample(250) # ambil setengah data sebagai sampel yaitu 500/2 = 250
```

```
# Fungsi untuk menghitung nilai Z (Distribusi Sample) dari masing-masing kolom
def hitungNilaiZ(RataSampel, RataPopulasi, DeviasiPopulasi, jumlahSampel):
    return ((RataSampel - RataPopulasi) * (jumlahSampel) ** 1/2)/DeviasiPopulasi

def Soal4(Sampelnya, population, jumlahSampel, columns):
    MeanKolom = Sampelnya[columns].mean()
    print("Mean dari sample kolom Daerah: ", MeanKolom);
    NilaiZ = hitungNilaiZ(MeanKolom, population[columns].mean(), population[columns].std(), j
    print("Nilai Distribusi Z yang dihitung: ", NilaiZ);
    print("P valuenya adalah: ", norm.sf(abs(NilaiZ)));

    # Bikin BoxPlot
    sns.boxplot(data=Sampelnya, x=str(columns));
```

```
# 4A Nilai rata-rata Daerah di atas 4700 ?
# Langkah 1: null hypothesis Mean (H0) = 4700
# Langkah 2: alternative hypothesis (H1) Mean > 4700
# Langkah 3: alpha = 0.05
# Langkah 4: uji tes mean Z one tailed, alpha = 0.05, daerah kritis: Z > 1.645
# Langkah 5: uji tes statistik one tailed
# Langkah 6: nilai distribusi Z berada pada daerah krisis dan p-value lebih kecil dari alpha,

Soal4(sampel, df, 250, "Daerah")
```

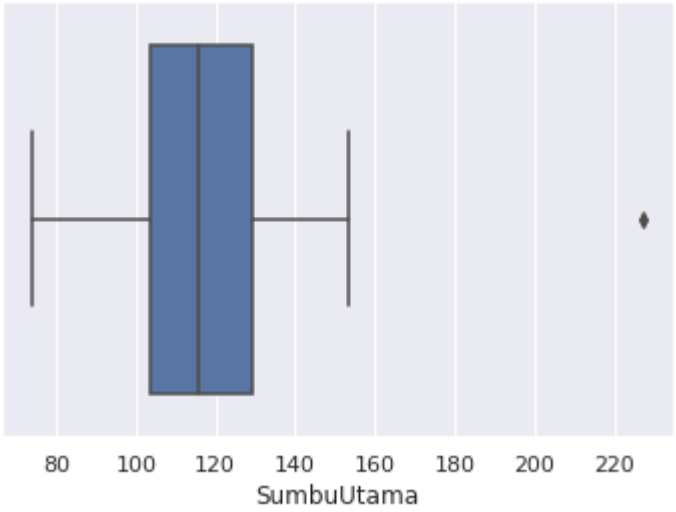
```
Mean dari sample kolom Daerah:  4783.096
Nilai Distribusi Z yang dihitung:  -2.3000409263756096
P valuenya adalah:  0.010722950753251815
```




```
# 4b Nilai Rata-rata Sumbu Utama tidak sama dengan 116 ?
# Langkah 1: null hypothesis Mean (H0) = 116
# Langkah 2: alternative hypothesis (H1) Mean != 116
# Langkah 3: alpha = 0.05
# Langkah 4: uji tes mean Z two tailed, alpha = 0.05 dan statistik two tailed maka daerah Z <
# Langkah 5: uji tes statistik two tailed
```

```
# Langkah 6: nilai distribusi Z tidak berada pada daerah krisis dan p-value lebih kecil dari

Soal4(sampel, df, 250, "SumbuUtama")
```

```
Mean dari sample kolom Daerah:  116.15149638995999
Nilai Distribusi Z yang dihitung:  0.7269539837082097
P valuenya adalah:  0.2336270727187293
```

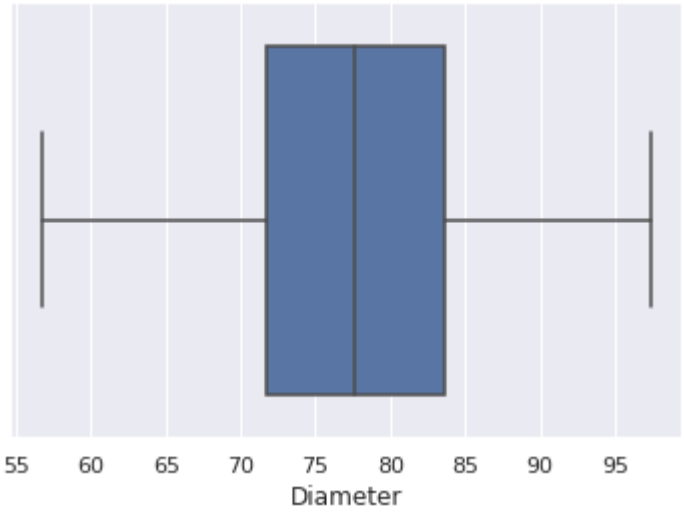



```
# 4c Nilai Rata-rata 20 baris pertama kolom Sumbu Kecil bukan 50 ?
# Langkah 1: null hypothesis Mean (H0) = 50
# Langkah 2: alternative hypothesis (H1) Mean != 50
# Langkah 3: alpha = 0.05
# Langkah 4: uji tes mean Z two tailed, alpha = 0.05 dan statistik two tailed maka daerah Z <
# Langkah 5: uji tes statistik two tailed

RataRataSampel20BarisPertama = sampel["SumbuKecil"].iloc[:19].mean()
print("Rata-rata 20 Baris Pertama pada kolom SumbuKecil adalah ", RataRataSampel20BarisPertam
zValue = hitungNilaiZ(RataRataSampel20BarisPertama, 50, df["SumbuKecil"].std(), 20)
print("zValue :  ", zValue);
print("pValue : ", norm.sf(abs(zValue)));

# Langkah 6: nilai distribusi Z  berada pada daerah krisis maka null hypothesis ditolak

# Boxplot "SumbuUtama"
sns.boxplot(data=sampel, x="SumbuKecil");
```

```
Rata-rata 20 Baris Pertama pada kolom SumbuKecil adalah  53.62848566315788
zValue :   8.912844601912363
pValue :  2.486983619008461e-19
```

```
# 4d Proporsi nilai Diameter yang lebih dari 85, adalah tidak sama dengan 15% ?
# Langkah 1: null hypothesis Mean (H0) P = 0.15
# Langkah 2: alternative hypothesis (H1) P != 0.15
# Langkah 3: alpha = 0.05
# Langkah 4: uji tes mean Z two tailed, alpha = 0.05 dan statistik two tailed maka daerah Z <
# Langkah 5: uji tes statistik two tailed
# Langkah 6: nilai Z tidak berada pada daerah kritis, maka null hypothesis di terima

countAbove85 = len(sampel["Diameter"].loc[sampel["Diameter"] > 85])
print("Banyaknya Gandum yang memiliki diameter lebih besar dari 85 adalah ", countAbove85);
zValue = ((countAbove85/250) - 0.15)/(0.15*0.85/250)**(1/2)
print("zValue : ", zValue);
print("pValue : ", norm.sf(abs(zValue)) * 2);

# Boxplot "Diameter"
sns.boxplot(data=sampel, x="Diameter");
```

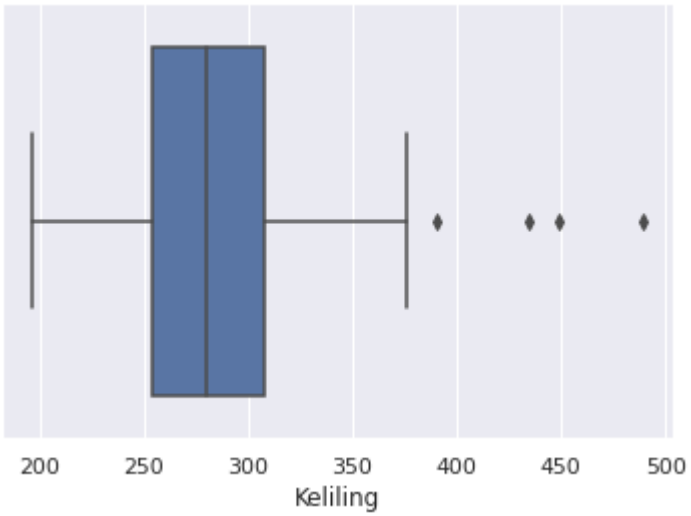
```
Banyaknya Gandum yang memiliki diameter lebih besar dari 85 adalah  49
zValue :  2.0369142367422195
pValue :  0.041658637191145986
```




```
# 4e Proporsi nilai Keliling yang kurang dari 100, adalah kurang dari 5% ?
# Langkah 1: null hypothesis Mean (H0) P = 0.05
# Langkah 2: alternative hypothesis (H1) P < 0.05
# Langkah 3: alpha = 0.05
# Langkah 4: uji tes mean Z one tailed, maka daerah kritis Z < -1.645
# Langkah 5: uji tes statistik one tailed
# Langkah 6: nilai Z berada pada daerah kritis, maka null hypothesis ditolak

countBelow100 = len(sampel["Keliling"].loc[sampel["Keliling"] < 100])
```

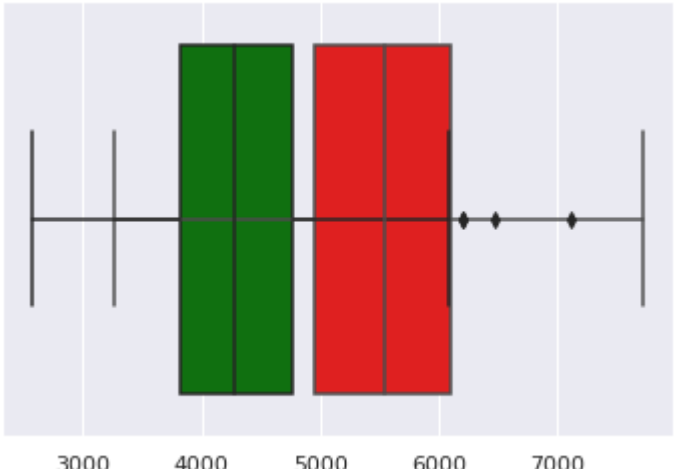
```
print("Jumlah keliling gandum yang lebih dari 100: ", countBelow100);
zValue = ((countBelow100/250) - 0.15)/(0.15*0.85/250)**(1/2)
print("zValue : ", zValue);
print("pValue : ", norm.sf(abs(zValue)));

# Boxplot "Keliling"
sns.boxplot(data=sampel, x="Keliling");
```

```
Jumlah keliling gandum yang lebih dari 100:  0
zValue :  -6.642111641550714
pValue :  1.546102975657358e-11
```




```
# SOAL 5
# Melakukan test hipotesis 2 sampel, dengan menuliskan 6 langkah testing dan menampilkan
# juga boxplotnya untuk kolom/bagian yang bersesuaian.
# a. Data kolom AreaBulatan dibagi 2 sama rata: bagian awal dan bagian akhir kolom.
# Benarkah rata-rata kedua bagian tersebut sama?
# b. Data kolom Kadar Air dibagi 2 sama rata: bagian awal dan bagian akhir kolom. Benarkah
# rata-rata bagian awal lebih besar dari pada bagian akhir sebesar 0.2?
# c. Rata-rata 20 baris pertama kolom Bulatan sama dengan 20 baris terakhirnya?
# d. Proporsi nilai bagian awal Ransum yang lebih dari 2, adalah lebih besar daripada,
# proporsi nilai yang sama di bagian akhir Ransum?
# e. Bagian awal kolom Diameter memiliki variansi yang sama dengan bagian akhirnya?


# 5A Data kolom AreaBulatan dibagi 2 sama rata: bagian awal dan bagian akhir kolom. Benarkah
# Kita bagi dahulu menjadi 2 bagian
data1 = df.iloc[0:250]
data2 = df.iloc[250:500]
# asumsi alpha 0.05
alpha = 0.05
sns.boxplot(data1['AreaBulatan'],color = "red")
sns.boxplot(data2['AreaBulatan'],color = "green")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f23a4e485d0>
```




```
def hitungDerajatKebebasan(var_data1,var_data2):
  return ((var_data1/250 + var_data2/250)**2)/ ((((var_data1/250)**2) / (249)) + (((var_data2

def tTest(mean_data1,mean_data2,var_data1,var_data2):
  return (mean_data1 - mean_data2)/(np.sqrt(var_data1/250 + var_data2/250))


print("Mean Comparison data 1 vs data 2")
mean_data1 = data1['AreaBulatan'].mean()
mean_data2 = data2['AreaBulatan'].mean()
print(mean_data1)
print(mean_data2)

print("\nVariance Comparison data 1 vs data 2")
var_data1 = data1['AreaBulatan'].var()
var_data2 = data2['AreaBulatan'].var()
print(var_data1)
print(var_data2)

print("\nDerajat Kebebasan : ")
print(hitungDerajatKebebasan(var_data1,var_data2))

print("\nNilai t-test : ")
print(tTest(mean_data1,mean_data2,var_data1,var_data2))
```

```
Mean Comparison data 1 vs data 2
5549.804
4324.292

Variance Comparison data 1 vs data 2
751733.1060080321
545480.4244337347

Derajat Kebebasan :
```

```
485.7209958986575

Nilai t-test :
17.013036648485464
```

```
# Untuk penentuannya disini ya kak

bound = stats.t.cdf(0.025,hitungDerajatKebebasan(var_data1,var_data2))
print("Daerah kritisnya terdapat pada :")
print("-∞ < "+str(-bound)+" atau "+str(bound)+ " < ∞")
print("Jadi t harus bernilai antara "+str(-bound)+" dan "+str(bound)+" agar H0 diterima")

tt = tTest(mean_data1,mean_data2,var_data1,var_data2)

if (tt < -bound or tt > bound):
  print("H0 tidak diterima karena t bernilai "+str(tt))
else:
  print("H0 diterima karena t bernilai "+str(tt))
```

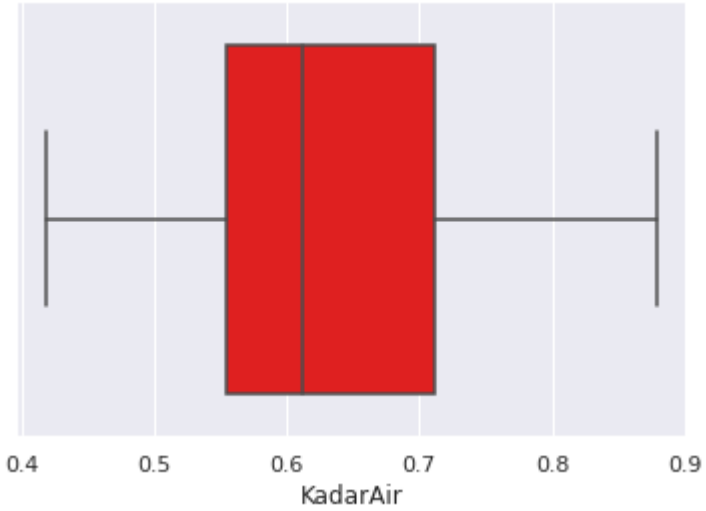
```
Daerah kritisnya terdapat pada :
-∞ < -0.5099673845396994 atau 0.5099673845396994 < ∞
Jadi t harus bernilai antara -0.5099673845396994 dan 0.5099673845396994 agar H0 diterima
H0 tidak diterima karena t bernilai 17.013036648485464
```

```
# 5B Data kolom Kadar Air dibagi 2 sama rata: bagian awal dan bagian akhir kolom. Benarkah ra
# Kita bagi dahulu menjadi 2 bagian
data1 = df.iloc[0:250]
data2 = df.iloc[250:500]
# asumsi alpha 0.05
alpha = 0.05
sns.boxplot(data1['KadarAir'],color = "red")
sns.boxplot(data2['KadarAir'],color = "green")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f23a51818d0>
```

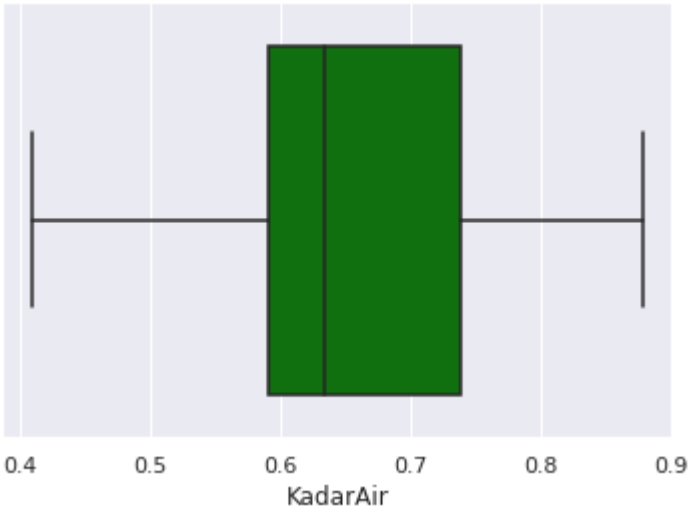
```
sns.boxplot(data1['KadarAir'],color = "red")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f23a52d1ad0>
```




```
sns.boxplot(data2['KadarAir'],color = "green")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f23a537c190>
```




```
print("Mean Comparison data 1 vs data 2")
mean_data1 = data1['KadarAir'].mean()
mean_data2 = data2['KadarAir'].mean()
print(mean_data1)
```

```
print(mean_data1)
print(mean_data2)

print("\nVariance Comparison data 1 vs data 2")
var_data1 = data1['KadarAir'].var()
var_data2 = data2['KadarAir'].var()
print(var_data1)
print(var_data2)

print("\nDerajat Kebebasan : ")
print(hitungDerajatKebebasan(var_data1,var_data2))

print("\nNilai t-test : ")
print(tTest(mean_data1,mean_data2,var_data1,var_data2))
```

```
Mean Comparison data 1 vs data 2
0.63574344072
0.6609999030760001

Variance Comparison data 1 vs data 2
0.009043200047076563
0.008482636662870607

Derajat Kebebasan :
497.4910475600773

Nilai t-test :
-3.016498704781028
```

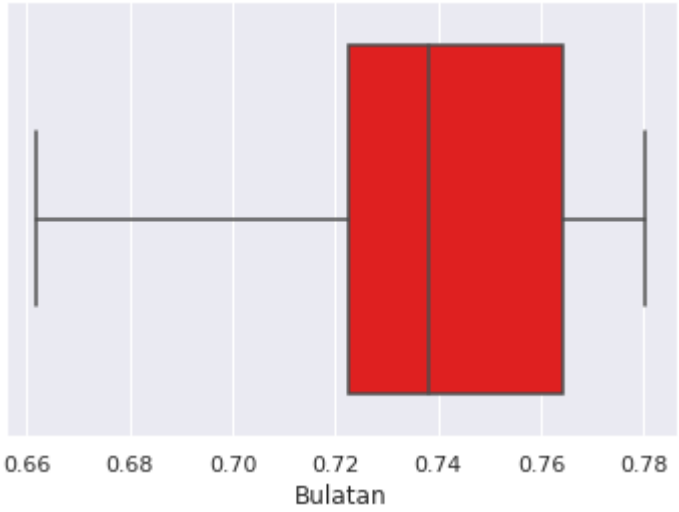
```
# Untuk penentuannya disini ya kak

bound = stats.t.cdf(0.025,hitungDerajatKebebasan(var_data1,var_data2))
print("Daerah kritisnya terdapat pada :")
print("-∞ < "+str(-bound)+" atau "+str(bound)+ " < ∞")
print("Jadi t harus bernilai antara "+str(-bound)+" dan "+str(bound)+" agar H0 diterima")

tt = tTest(mean_data1,mean_data2,var_data1,var_data2)

if (tt < -bound or tt > bound):
  print("H0 tidak diterima karena t bernilai "+str(tt))
else:
  print("H0 diterima karena t bernilai "+str(tt))
```

```
Daerah kritisnya terdapat pada :
-∞ < -0.5099675059652441 atau 0.5099675059652441 < ∞
Jadi t harus bernilai antara -0.5099675059652441 dan 0.5099675059652441 agar H0 diterima
H0 tidak diterima karena t bernilai -3.016498704781028
```

```
# 5C Rata-rata 20 baris pertama kolom Bulatan sama dengan 20 baris terakhirnya?
data1 = df.iloc[0:20]
data2 = df.iloc[480:500]
```

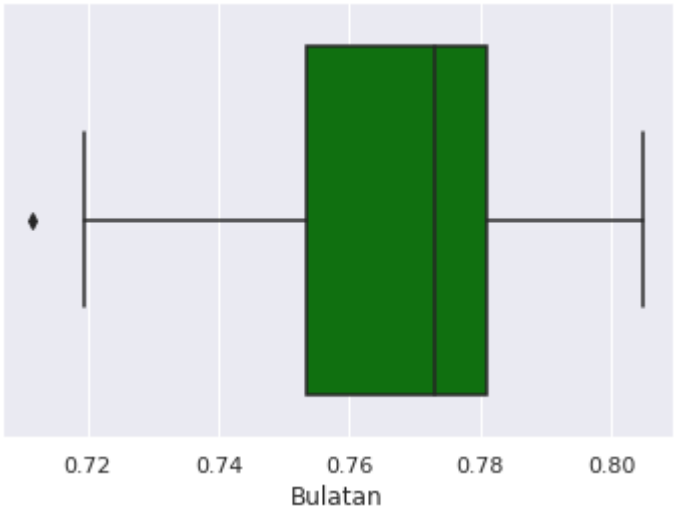
```
sns.boxplot(data1['Bulatan'],color = "red")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f23a50d6390>
```




```
sns.boxplot(data2['Bulatan'],color = "Green")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f23a5065810>
```




```
def hitungDerajatKebebasan(var_data1,var_data2):
  return ((var_data1/20 + var_data2/20)**2)/ ((((var_data1/20)**2) / (19)) + (((var_data2/20)

def tTest(mean_data1,mean_data2,var_data1,var_data2):
  return (mean_data1 - mean_data2)/(np.sqrt(var_data1/20 + var_data2/20))


print("Mean Comparison data 1 vs data 2")
mean data1 = data1['Bulatan'].mean()
```

```
mean_data2 = data2['Bulatan'].mean()
print(mean_data1)
print(mean_data2)

print("\nVariance Comparison data 1 vs data 2")
var_data1 = data1['Bulatan'].var()
var_data2 = data2['Bulatan'].var()
print(var_data1)
print(var_data2)

print("\nDerajat Kebebasan : ")
print(hitungDerajatKebebasan(var_data1,var_data2))

print("\nNilai t-test : ")
print(tTest(mean_data1,mean_data2,var_data1,var_data2))
```

```
Mean Comparison data 1 vs data 2
0.7375353552499999
0.767322437

Variance Comparison data 1 vs data 2
0.0009232346025806985
0.0006307661055759374

Derajat Kebebasan :
36.70006222999688

Nilai t-test :
-3.3792268633124025
```

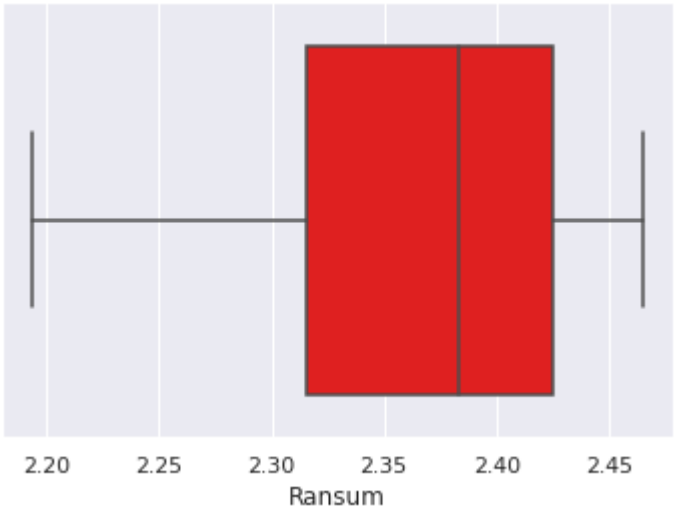
```
# Untuk penentuannya disini ya kak

bound = stats.t.cdf(0.025,hitungDerajatKebebasan(var_data1,var_data2))
print("Daerah kritisnya terdapat pada :")
print("-∞ < "+str(-bound)+" atau "+str(bound)+ " < ∞")
print("Jadi t harus bernilai antara "+str(-bound)+" dan "+str(bound)+" agar H0 diterima")

tt = tTest(mean_data1,mean_data2,var_data1,var_data2)

if (tt < -bound or tt > bound):
  print("H0 tidak diterima karena t bernilai "+str(tt))
else:
  print("H0 diterima karena t bernilai "+str(tt))
```

```
Daerah kritisnya terdapat pada :
-∞ < -0.5099047967145734 atau 0.5099047967145734 < ∞
Jadi t harus bernilai antara -0.5099047967145734 dan 0.5099047967145734 agar H0 diterima
H0 tidak diterima karena t bernilai -3.3792268633124025
```

```
# 5D Proporsi nilai bagian awal Ransum yang lebih dari 2, adalah lebih besar daripada propors
# untuk soal ini kita pakai z-test
```

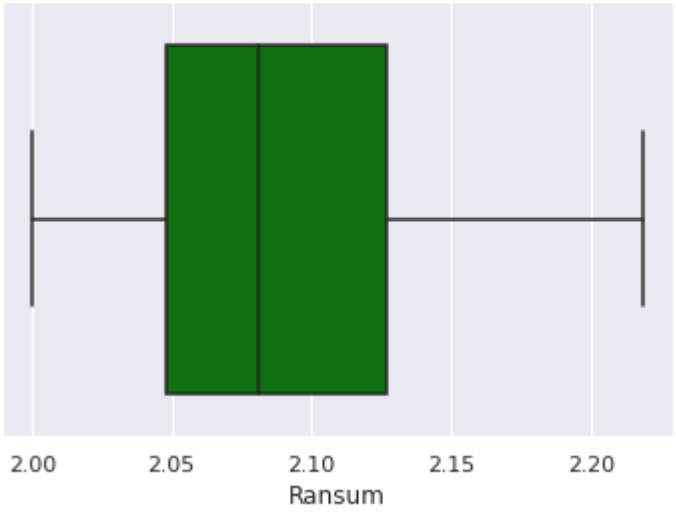
```
ddata1 = data1.Ransum[data1.Ransum>2]
sns.boxplot(ddata1,color = "Red")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f23a4cb4ed0>
```




```
ddata2 = data2.Ransum[data2.Ransum>2]
sns.boxplot(ddata2,color = "Green")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f23a4ce9490>
```



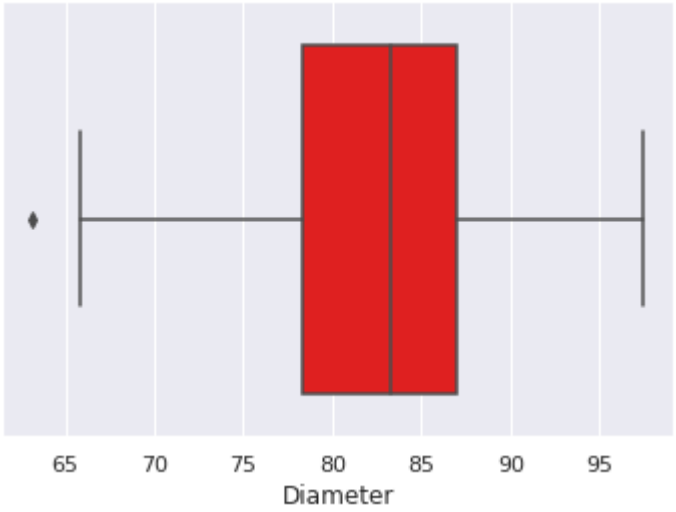

```
def zTest(ddata1,ddata2):
  temp1 = ddata1.shape[0] / 250
  temp2 = ddata2.shape[0] / 250
  temp3 = (ddata1.shape[0] + ddata2.shape[0]) / (500)
```

```
  return (temp1-temp2)/np.sqrt(temp3*(1-temp3)*(1/125))

def hitungNilaiKritis(alpha):
  return stats.norm.ppf(alpha)


print(" Nilai kritisnya diatas "+str(hitungNilaiKritis(alpha)))
print(" Nilai hasil Z test kita : "+str(zTest(ddata1,ddata2)))
print(" Karena nilai Z lebih besar daripada nilai kritis kita, H0 ditolak")
```

```
 Nilai kritisnya diatas -1.6448536269514729
 Nilai hasil Z test kita : 13.397486455610238
 Karena nilai Z lebih besar daripada nilai kritis kita, H0 ditolak
```

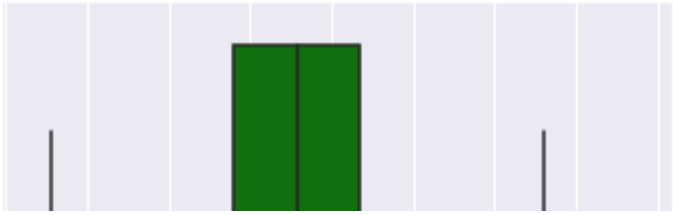
```
# 5E Bagian awal kolom Diameter memiliki variansi yang sama dengan bagian akhirnya?
# untuk soal ini kita pakai F-test
ddata1 = data1["Diameter"]
sns.boxplot(ddata1,color = "Red")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f23a496d490>
```




```
ddata2 = data2["Diameter"]
sns.boxplot(ddata2,color = "Green")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f23a496d6d0>
```



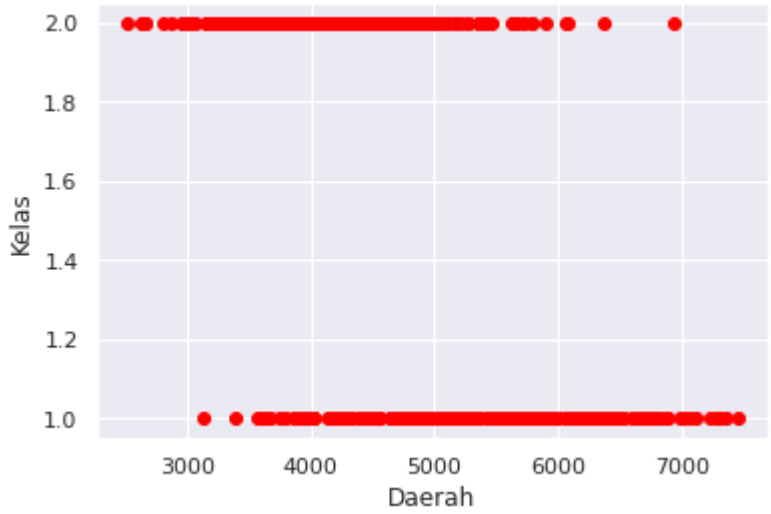
```
def pValue(var_data1,var_data2):
  return (1-stats.f.cdf((var_data1/var_data2),249,249))

def critBoundary(alpha):
  lower = stats.f(249,249).ppf(1-(alpha/2))
  upper = stats.f(249,249).ppf(alpha/2)
  return lower,upper

def BoundaryCheck(boundLow,boundUpp,num):
  if (num < boundLow or num > boundUpp):
    print ("H0 diterima karena f bernilai "+str(num))
  else:
    print ("H0 tidak diterima karena f bernilai "+str(num))


var_data1 = np.var(ddata1, ddof=1)
var_data2 = np.var(ddata2, ddof=1)
print("Variance Comparison antara ddata 1 dan ddata 2")
print(var_data1)
print(var_data2)
print("F value : "+str(var_data1/var_data2))
print("P value : "+str(pValue(var_data1,var_data2)))
temp1,temp2 = critBoundary(alpha)
print("Daerah kritisnya diatas "+str(temp1)+" dan dibawah "+str(temp2))
BoundaryCheck(temp1,temp2,var_data1/var_data2)
```

```
Variance Comparison antara ddata 1 dan ddata 2
43.16898042632247
39.82826438807723
F value : 1.0838780220421882
P value : 0.26278959201893937
Daerah kritisnya diatas 1.2827228078241388 dan dibawah 0.7795916576054985
H0 diterima karena f bernilai 1.0838780220421882
```

```
# Nomor 6 tes korelasi
import matplotlib.pyplot as plt

def cor(corr):
  if (corr > 0):
    print("Terdapat korelasi positif antara kedua kolom : "+str(corr))
  elif (corr == 0):
    print("Kedua kolom tidak berkorelasi")
  else:
```

```
    else:
      print("Terdapat korelasi negatif antara kedua kolom : "+str(corr))


print("KORELASI ANTARA KOLOM DAERAH - KELAS")
data = df["Daerah"].corr(df["Kelas"])
cor(data)
plt.scatter(df.Daerah, df.Kelas, color = "red")
plt.xlabel("Daerah")
plt.ylabel("Kelas")
plt.show()
```

```
KORELASI ANTARA KOLOM DAERAH - KELAS
Terdapat korelasi negatif antara kedua kolom : -0.6027466517416661
```

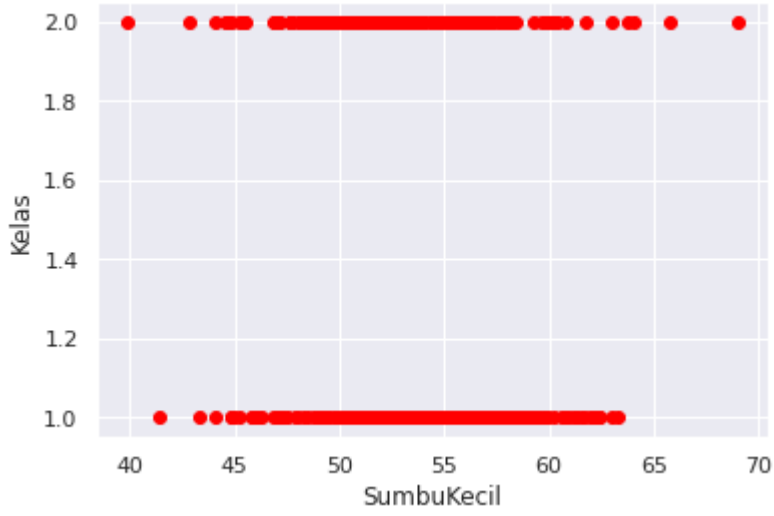



```
print("KORELASI ANTARA KOLOM SUMBU UTAMA - KELAS")
data = df["SumbuUtama"].corr(df["Kelas"])
cor(data)
plt.scatter(df.SumbuUtama, df.Kelas, color = "red")
plt.xlabel("SumbuUtama")
plt.ylabel("Kelas")
plt.show()
```

```
KORELASI ANTARA KOLOM SUMBU UTAMA - KELAS
Terdapat korelasi negatif antara kedua kolom : -0.7130906104204592
```

```
print("KORELASI ANTARA KOLOM SUMBU KECIL - KELAS")
data = df["SumbuKecil"].corr(df["Kelas"])
cor(data)
plt.scatter(df.SumbuKecil, df.Kelas, color = "red")
plt.xlabel("SumbuKecil")
plt.ylabel("Kelas")
plt.show()
```

```
KORELASI ANTARA KOLOM SUMBU KECIL - KELAS
Terdapat korelasi negatif antara kedua kolom : -0.1529751733553502
```

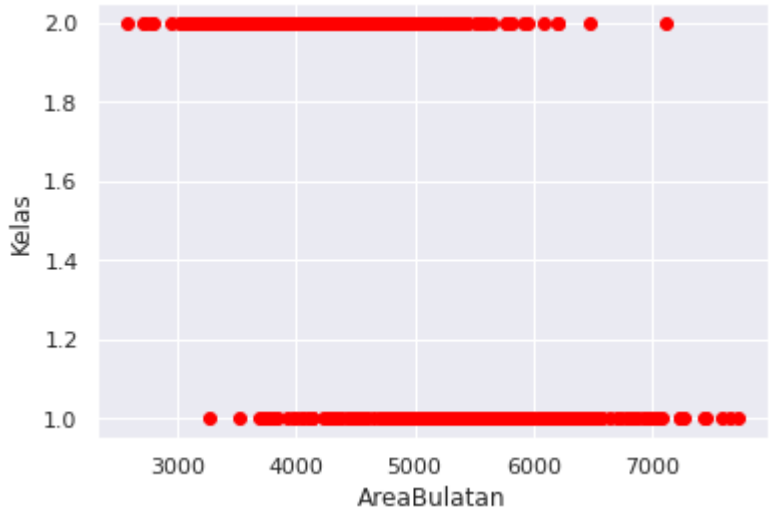



```
print("KORELASI ANTARA KOLOM KEUNIKAN - KELAS")
data = df["Keunikan"].corr(df["Kelas"])
cor(data)
plt.scatter(df.Keunikan, df.Kelas, color = "red")
plt.xlabel("Keunikan")
plt.ylabel("Kelas")
plt.show()
```

```
KORELASI ANTARA KOLOM KEUNIKAN - KELAS
```

```
print("KORELASI ANTARA KOLOM AREA BULATAN - KELAS")
data = df["AreaBulatan"].corr(df["Kelas"])
cor(data)
plt.scatter(df.AreaBulatan, df.Kelas, color = "red")
plt.xlabel("AreaBulatan")
plt.ylabel("Kelas")
plt.show()
```

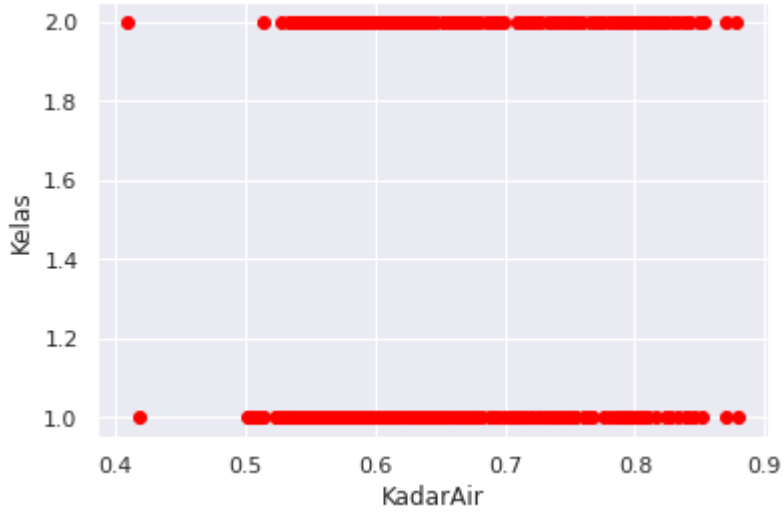
```
KORELASI ANTARA KOLOM AREA BULATAN - KELAS
Terdapat korelasi negatif antara kedua kolom : -0.6073125434153749
```




```
print("KORELASI ANTARA KOLOM DIAMETER - KELAS")
data = df["Diameter"].corr(df["Kelas"])
cor(data)
plt.scatter(df.Diameter, df.Kelas, color = "red")
plt.xlabel("Diameter")
plt.ylabel("Kelas")
plt.show()
```

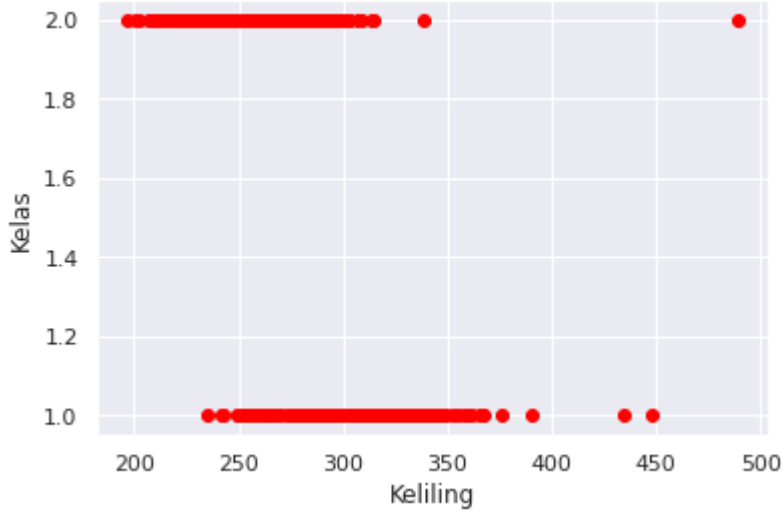
```
KORELASI ANTARA KOLOM DIAMETER - KELAS
print("KORELASI ANTARA KOLOM KADAR AIR - KELAS")
data = df["KadarAir"].corr(df["Kelas"])
cor(data)
plt.scatter(df.KadarAir, df.Kelas, color = "red")
plt.xlabel("KadarAir")
plt.ylabel("Kelas")
plt.show()
```

```
KORELASI ANTARA KOLOM KADAR AIR - KELAS
Terdapat korelasi positif antara kedua kolom : 0.13434422605727642
```



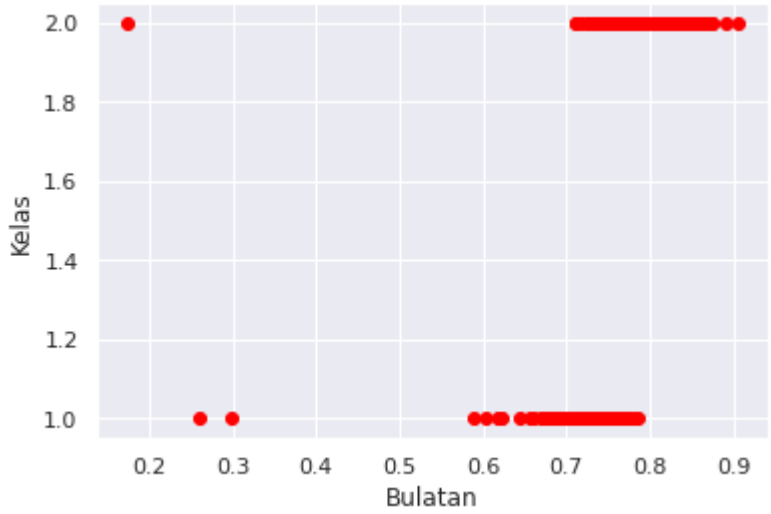

```
print("KORELASI ANTARA KOLOM KELILING - KELAS")
data = df["Keliling"].corr(df["Kelas"])
cor(data)
plt.scatter(df.Keliling, df.Kelas, color = "red")
plt.xlabel("Keliling")
plt.ylabel("Kelas")
plt.show()
```

```
KORELASI ANTARA KOLOM KELILING - KELAS
Terdapat korelasi negatif antara kedua kolom : -0.6348607454756854
```



```
print("KORELASI ANTARA KOLOM BULATAN - KELAS")
data = df["Bulatan"].corr(df["Kelas"])
cor(data)
plt.scatter(df.Bulatan, df.Kelas, color = "red")
plt.xlabel("Bulatan")
plt.ylabel("Kelas")
plt.show()
```

```
KORELASI ANTARA KOLOM BULATAN - KELAS
Terdapat korelasi positif antara kedua kolom : 0.5450045317240071
```




```
print("KORELASI ANTARA KOLOM RANSUM - KELAS")
data = df["Ransum"].corr(df["Kelas"])
cor(data)
plt.scatter(df.Ransum, df.Kelas, color = "red")
plt.xlabel("Ransum")
plt.ylabel("Kelas")
plt.show()
```

```
KORELASI ANTARA KOLOM RANSUM - KELAS
Terdapat korelasi negatif antara kedua kolom : -0.8399038681287484
```