In [54]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [55]:
```
df = pd.read_csv('bodyfat.csv')
df
```

Out[55]:

	Density	BodyFat	Age	Weight	Height	Neck	Chest	Abdomen	Hip	Thigh	Knee	Ankle
0	1.0708	12.3	23	154.25	67.75	36.2	93.1	85.2	94.5	59.0	37.3	21.9
1	1.0853	6.1	22	173.25	72.25	38.5	93.6	83.0	98.7	58.7	37.3	23.4
2	1.0414	25.3	22	154.00	66.25	34.0	95.8	87.9	99.2	59.6	38.9	24.0
3	1.0751	10.4	26	184.75	72.25	37.4	101.8	86.4	101.2	60.1	37.3	22.8
4	1.0340	28.7	24	184.25	71.25	34.4	97.3	100.0	101.9	63.2	42.2	24.0
...	...	...	...	...	...	...	...	...	...	...	...	...
247	1.0736	11.0	70	134.25	67.00	34.9	89.2	83.6	88.8	49.6	34.8	21.5
248	1.0236	33.6	72	201.00	69.75	40.9	108.5	105.0	104.5	59.6	40.8	23.2
249	1.0328	29.3	72	186.75	66.00	38.9	111.1	111.5	101.7	60.3	37.3	21.5
250	1.0399	26.0	72	190.75	70.50	38.9	108.3	101.3	97.8	56.0	41.6	22.7
251	1.0271	31.9	74	207.50	70.00	40.8	112.4	108.5	107.1	59.3	42.2	24.6

252 rows × 15 columns

In [56]:
```
df.shape
```

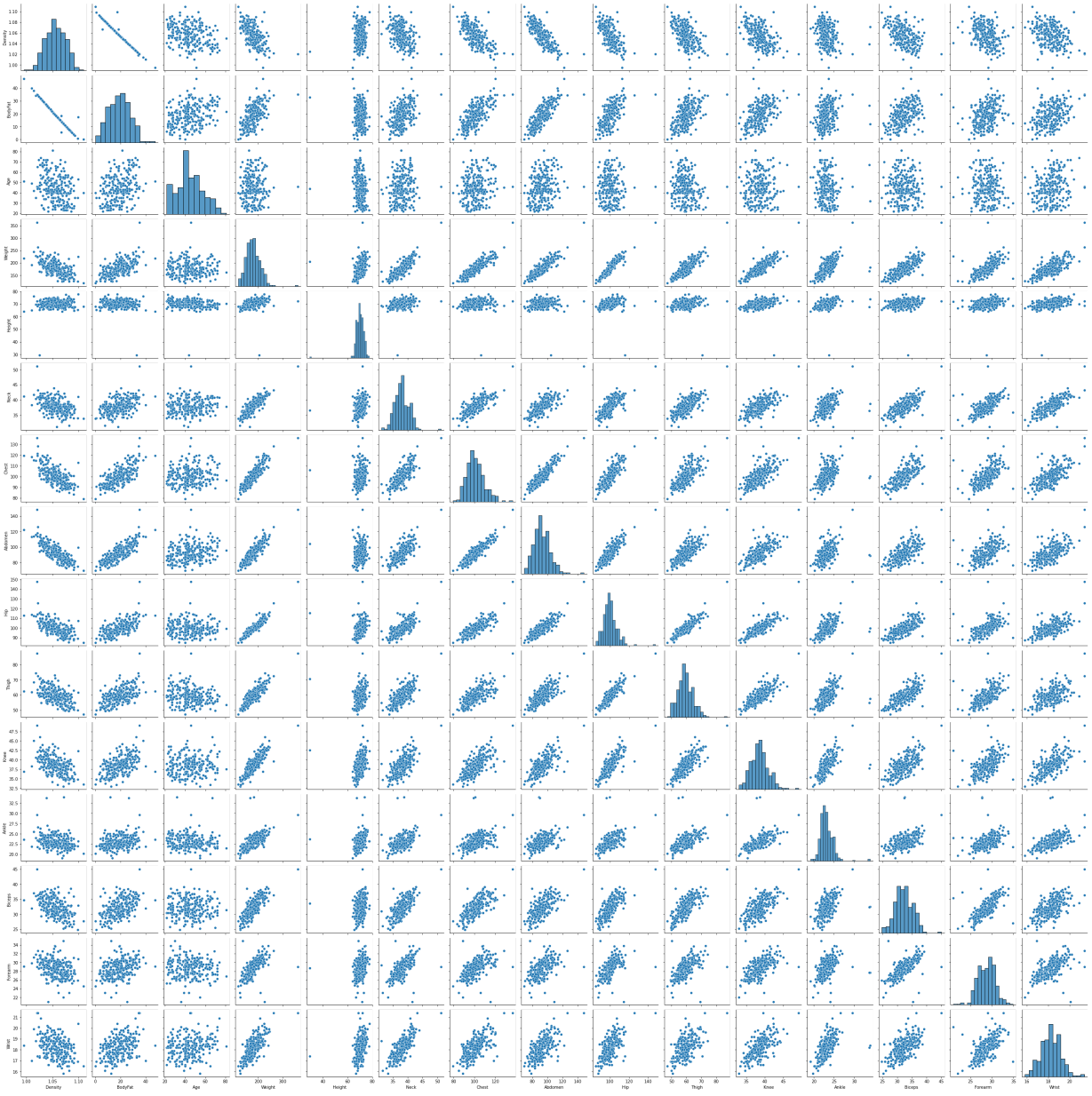
Out[56]:
```
(252, 15)
```

In [57]:
```
df.isnull().sum()
```

Out[57]:
```
Density    0
BodyFat    0
Age        0
Weight     0
Height     0
Neck       0
Chest      0
Abdomen    0
Hip        0
Thigh      0
Knee       0
Ankle      0
Biceps     0
Forearm    0
Wrist      0
dtype: int64
```

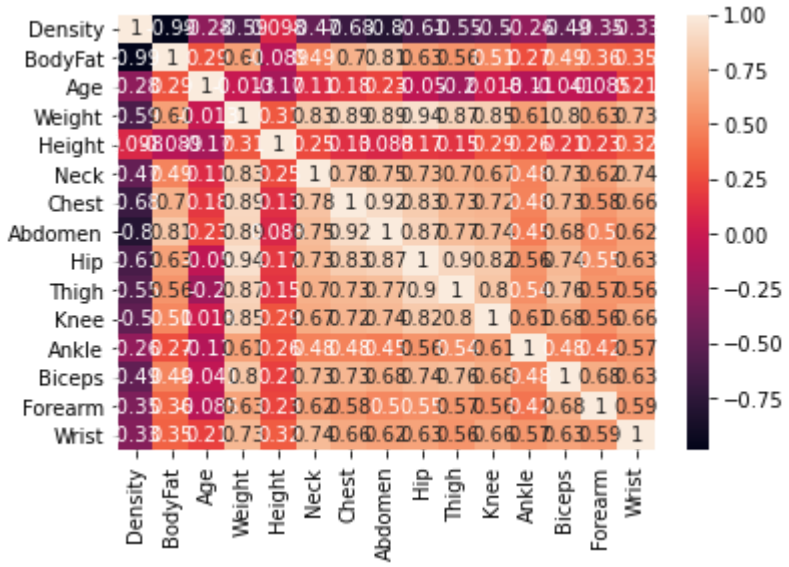
In [58]:
```
sns.pairplot(df)
```

Out[58]:
```
<seaborn.axisgrid.PairGrid at 0x1dfb42c3fa0>
```



In [59]:
```
sns.heatmap(df.corr(),annot=True)
```

Out[59]: <AxesSubplot:>



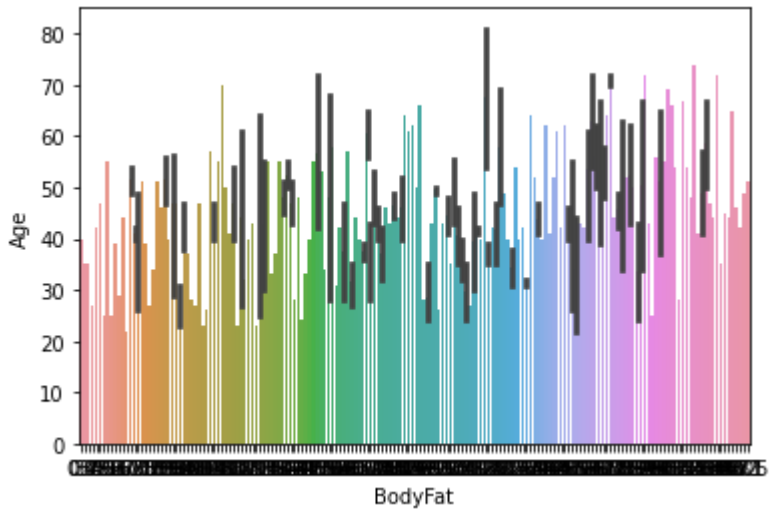
In [60]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 252 entries, 0 to 251
Data columns (total 15 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   Density  252 non-null    float64
 1   BodyFat  252 non-null    float64
 2   Age      252 non-null    int64  
 3   Weight   252 non-null    float64
 4   Height   252 non-null    float64
 5   Neck     252 non-null    float64
 6   Chest    252 non-null    float64
 7   Abdomen  252 non-null    float64
 8   Hip      252 non-null    float64
 9   Thigh    252 non-null    float64
 10  Knee     252 non-null    float64
 11  Ankle    252 non-null    float64
 12  Biceps   252 non-null    float64
 13  Forearm  252 non-null    float64
 14  Wrist    252 non-null    float64
dtypes: float64(14), int64(1)
memory usage: 29.7 KB
```

In [61]:
```
sns.barplot(df['BodyFat'],df['Age'])
```

```
C:\Users\Imran basha\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Fut
ureWarning: Pass the following variables as keyword args: x, y. From version 0.
12, the only valid positional argument will be `data`, and passing other argume
nts without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

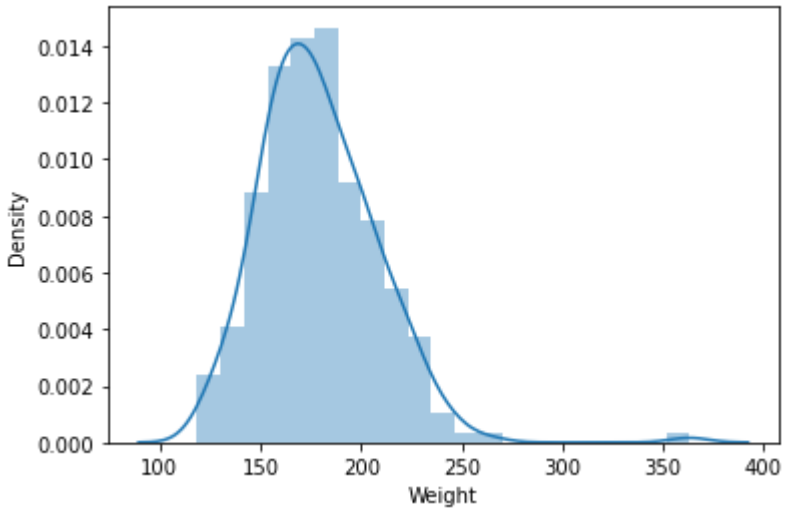
Out[61]: <AxesSubplot:xlabel='BodyFat', ylabel='Age'>



In [62]:
```
sns.distplot(df['Weight'])
```

```
C:\Users\Imran basha\anaconda3\lib\site-packages\seaborn\distributions.py:2557:
FutureWarning: `distplot` is a deprecated function and will be removed in a fut
ure version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histo
grams).
  warnings.warn(msg, FutureWarning)
```

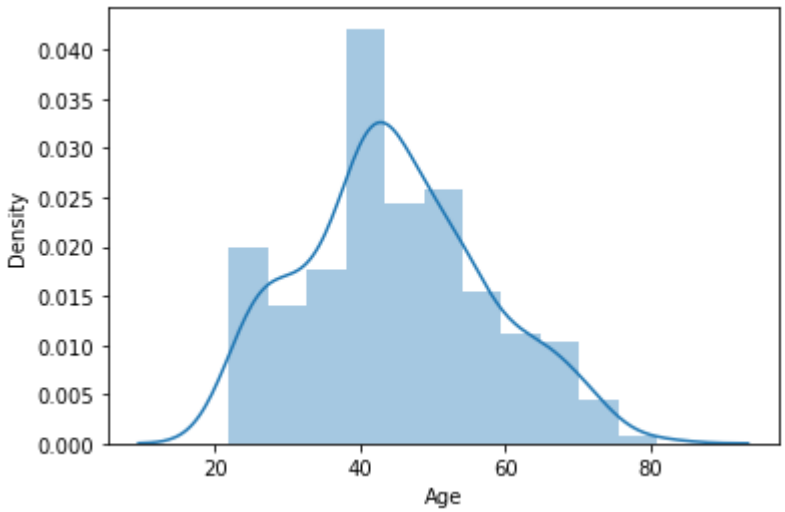
Out[62]: <AxesSubplot:xlabel='Weight', ylabel='Density'>



In [63]:
```
sns.distplot(df['Age'])
```

```
C:\Users\Imran basha\anaconda3\lib\site-packages\seaborn\distributions.py:2557:
FutureWarning: `distplot` is a deprecated function and will be removed in a fut
ure version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histo
grams).
  warnings.warn(msg, FutureWarning)
```

Out[63]: <AxesSubplot:xlabel='Age', ylabel='Density'>



In [*]:
```
sns.jointplot(df['BodyFat'],df['Age'])
```

```
C:\Users\Imran basha\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Fut
ureWarning: Pass the following variables as keyword args: x, y. From version 0.
12, the only valid positional argument will be `data`, and passing other argume
nts without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

In [*]:
```
sns.jointplot(df['BodyFat'],df['Age'],kind = "hex")
```

In [*]:
```
sns.jointplot(df['BodyFat'],df['Age'],kind = "kde")
```

In [*]:
```
sns.pairplot(df[['BodyFat','Age','Weight','Height']])
```

In [*]:
```
sns.stripplot(df['BodyFat'],df['Age'])
```

In [*]:
```
sns.countplot(df['Age'])
```

In [*]:
```
x = df.iloc[:,:-1].values
y = df.iloc[:,1].values
```

In [*]:
```
x
```

In [*]:
```
y
```

In [*]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=0)
```

In [*]:
```
x_train,x_test,y_train,y_test
```

In [*]:
```
from sklearn.linear_model import LinearRegression
reg = LinearRegression()
reg.fit(x_train,y_train)
```

In [*]:
```
y_predict=reg.predict(x_test)
y_predict
```

In [*]:
```
y_test
```

In [*]:
```
plt.figure(figsize=(15,15))
sns.heatmap(df.corr(),annot=True,cmap='RdYlGn',linewidths=2,linecolor='orange',ar
```

In [*]:
```
df.hist(figsize=(20,20),grid=False,color='red',bins=15);
```

In [*]:
```
X=df.drop(["BodyFat"], axis = 1)
y=df.BodyFat
```

In [41]:
```
from sklearn.preprocessing import StandardScaler
SC=StandardScaler()
X=SC.fit_transform(X)
```

In [42]:
```
from sklearn.ensemble import ExtraTreesRegressor
selection = ExtraTreesRegressor()
selection.fit(X,y)
```

Out[42]:
```
ExtraTreesRegressor()
```

In [43]:
```
print(selection.feature_importances_)
```

```
[0.70243704 0.00192995 0.02010906 0.00345007 0.0069074  0.0553786
 0.17300293 0.01793078 0.00633934 0.00338747 0.00107529 0.00473853
 0.0010223  0.00229122]
```

In [44]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random
```

In [46]:
```
X_train.shape
```

Out[46]:
```
(201, 14)
```

In [47]:
```
X_test.shape
```

Out[47]:
```
(51, 14)
```

In [48]:
```
from sklearn.linear_model import Lasso
LASSO=Lasso(alpha=0.5)
LASSO.fit(X_train,y_train)
```

Out[48]:
```
Lasso(alpha=0.5)
```

In [49]:
```
from sklearn.linear_model import Ridge
RIDGE=Ridge(alpha=0.5)
RIDGE.fit(X_train,y_train)
```

Out[49]:
```
Ridge(alpha=0.5)
```

In [ ]:
```

```