```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

df=pd.read_csv('C:\\Users\\HP\\Downloads\\heart.csv')

if df['sex'].dtype != 'object':
    df['sex'] = df['sex'].astype(str)
df['sex'] = df['sex'].map({'1': 'male', '0': 'female'})

print(df.head())
print(df.dtypes)
```

```
   age     sex  cp  trestbps  chol  fbs  restecg  thalach  exang
oldpeak  \
0   52    male   0       125   212    0        1      168      0
1.0
1   53    male   0       140   203    1        0      155      1
3.1
2   70    male   0       145   174    0        1      125      1
2.6
3   61    male   0       148   203    0        1      161      0
0.0
4   62  female   0       138   294    1        1      106      0
1.9

   slope  ca  thal  target
0      2   2     3       0
1      0   0     3       0
2      0   0     3       0
3      2   1     3       0
4      1   3     2       0
age           int64
sex          object
cp            int64
trestbps      int64
chol          int64
fbs           int64
restecg       int64
thalach       int64
exang         int64
oldpeak     float64
slope         int64
ca            int64
thal          int64
target        int64
dtype: object
```

```
df.head()
```

```
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak
slope  \
0   52    1   0       125   212    0        1      168      0      1.0
2
1   53    1   0       140   203    1        0      155      1      3.1
0
2   70    1   0       145   174    0        1      125      1      2.6
0
3   61    1   0       148   203    0        1      161      0      0.0
2
4   62    0   0       138   294    1        1      106      0      1.9
1

   ca  thal  target
0   2     3       0
1   0     3       0
2   0     3       0
3   1     3       0
4   3     2       0
```

```
df.tail()
```

```
      age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang
oldpeak  \
1020   59    1   1       140   221    0        1      164      1
0.0
1021   60    1   0       125   258    0        0      141      1
2.8
1022   47    1   0       110   275    0        0      118      1
1.0
1023   50    0   0       110   254    0        0      159      0
0.0
1024   54    1   0       120   188    0        1      113      0
1.4

      slope  ca  thal  target
1020      2   0     2       1
1021      1   1     3       0
1022      1   1     2       0
1023      2   0     2       1
1024      1   1     3       0
```

```
df.columns.values
```

```
array(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg',
       'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal',
'target'],
      dtype=object)
```

```
df.isna().sum()
```

```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

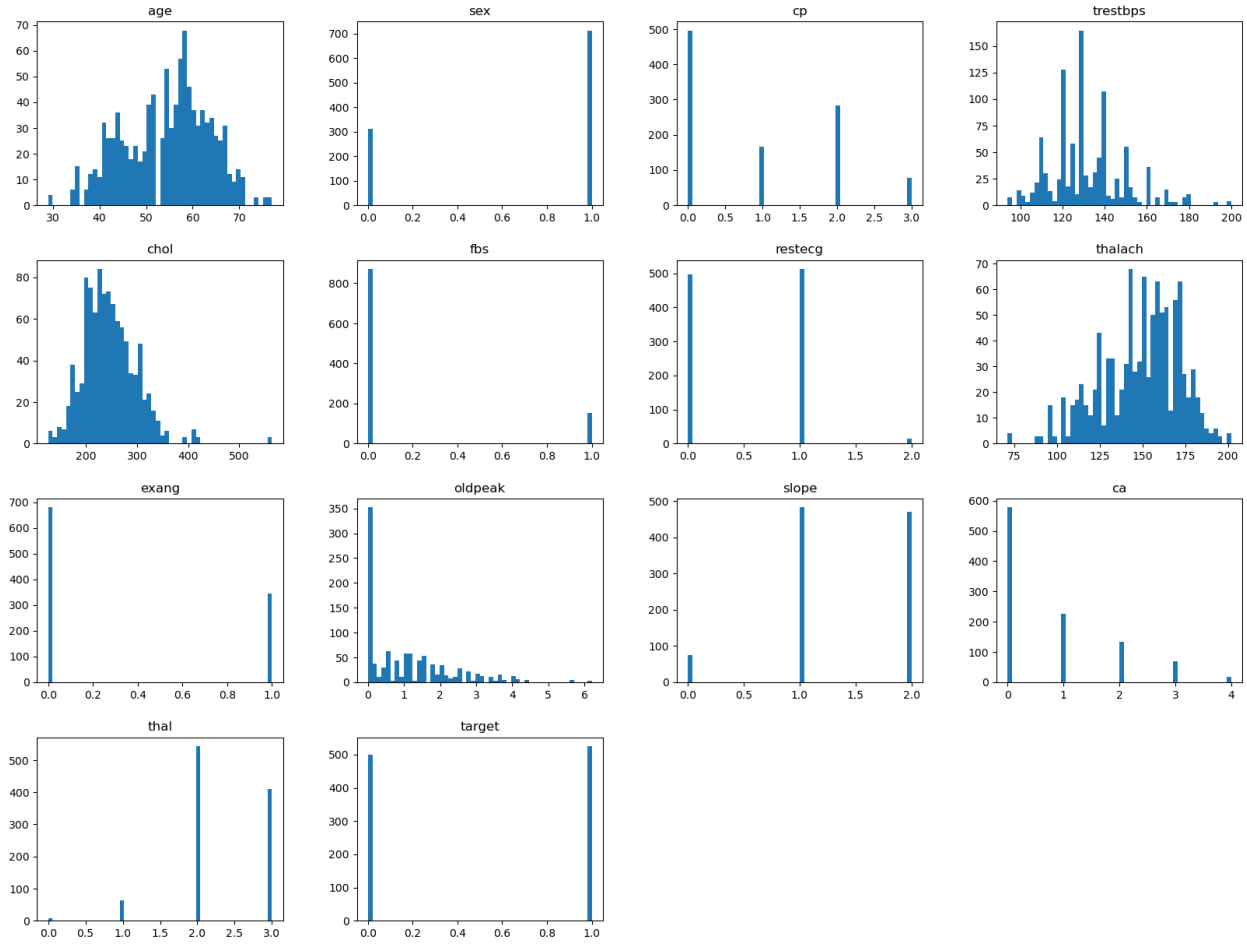
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1025 non-null   int64  
 1   sex       1025 non-null   int64  
 2   cp        1025 non-null   int64  
 3   trestbps  1025 non-null   int64  
 4   chol      1025 non-null   int64  
 5   fbs       1025 non-null   int64  
 6   restecg   1025 non-null   int64  
 7   thalach   1025 non-null   int64  
 8   exang     1025 non-null   int64  
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64  
 11  ca        1025 non-null   int64  
 12  thal      1025 non-null   int64  
 13  target    1025 non-null   int64  
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

```
df.hist(bins=50,grid=False ,figsize=(20,15))
```

```
array([[<Axes: title={'center': 'age'}>, <Axes: title={'center':
'sex'}>,
        <Axes: title={'center': 'cp'}>,
        <Axes: title={'center': 'trestbps'}>],
       [<Axes: title={'center': 'chol'}>,
        <Axes: title={'center': 'fbs'}>,
```

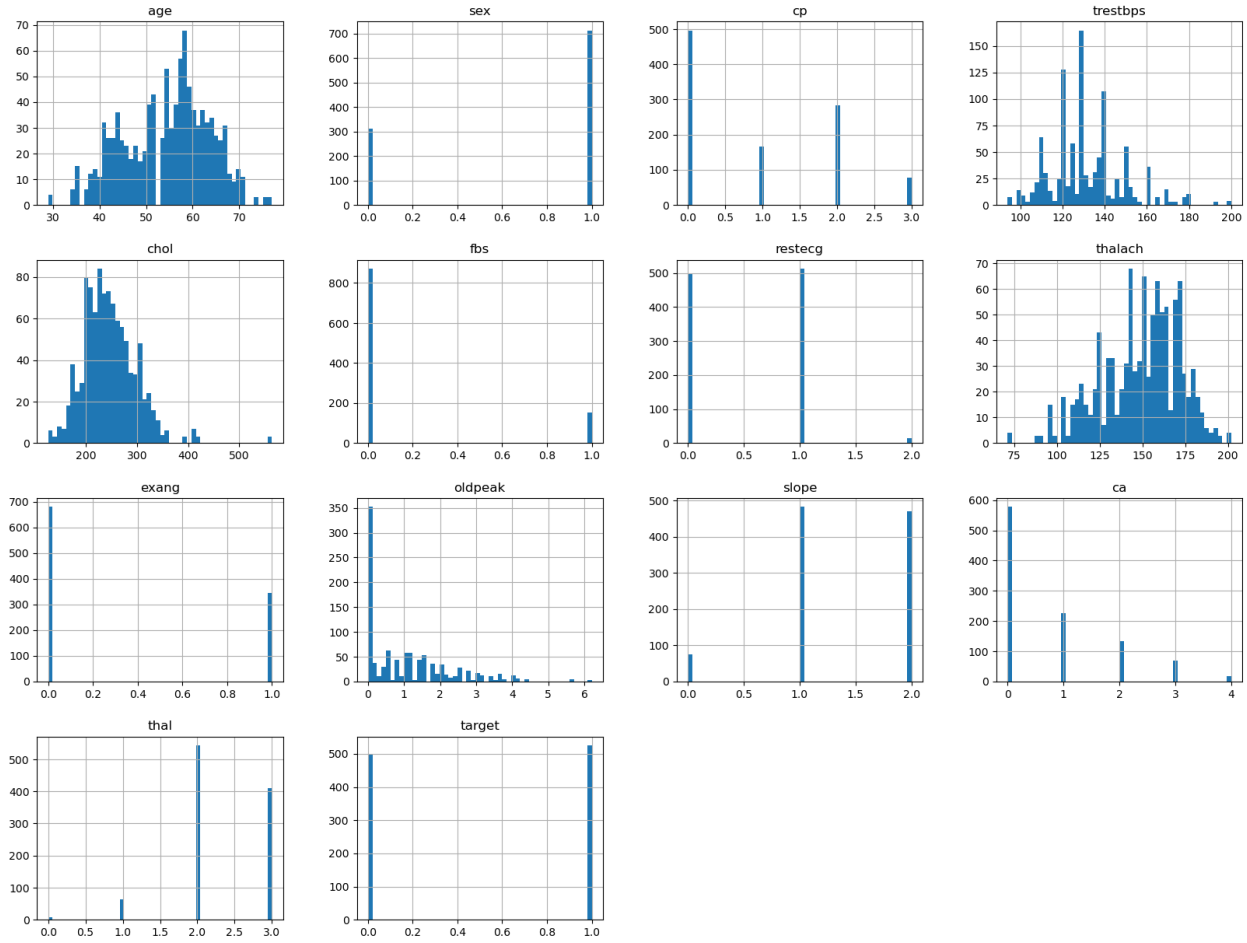
```
       <Axes: title={'center': 'restecg'}>,
       <Axes: title={'center': 'thalach'}>],
      [<Axes: title={'center': 'exang'}>,
       <Axes: title={'center': 'oldpeak'}>,
       <Axes: title={'center': 'slope'}>,
       <Axes: title={'center': 'ca'}>],
      [<Axes: title={'center': 'thal'}>,
       <Axes: title={'center': 'target'}>, <Axes: >, <Axes: >]],
     dtype=object)
```




```
df.hist(bins=50,grid=True ,figsize=(20,15))
```

```
array([[<Axes: title={'center': 'age'}>, <Axes: title={'center':
'sex'}>,
       <Axes: title={'center': 'cp'}>,
       <Axes: title={'center': 'trestbps'}>],
      [<Axes: title={'center': 'chol'}>,
       <Axes: title={'center': 'fbs'}>,
       <Axes: title={'center': 'restecg'}>,
       <Axes: title={'center': 'thalach'}>],
```

```
       [<Axes: title={'center': 'exang'}>,
        <Axes: title={'center': 'oldpeak'}>,
        <Axes: title={'center': 'slope'}>,
        <Axes: title={'center': 'ca'}>],
       [<Axes: title={'center': 'thal'}>,
        <Axes: title={'center': 'target'}>, <Axes: >, <Axes: >]],
      dtype=object)
```




```
df.describe()
```

	age	sex	cp	trestbps	chol
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.00000
mean	54.434146	0.695610	0.942439	131.611707	246.00000
std	9.072290	0.460373	1.029641	17.516718	51.59251
min	29.000000	0.000000	0.000000	94.000000	126.00000
25%	48.000000	0.000000	0.000000	120.000000	211.00000

```
50%      56.000000     1.000000     1.000000   130.000000   240.00000

75%      61.000000     1.000000     2.000000   140.000000   275.00000

max      77.000000     1.000000     3.000000   200.000000   564.00000


                fbs      restecg      thalach        exang      oldpeak
\
count  1025.000000  1025.000000  1025.000000  1025.000000  1025.000000

mean      0.149268     0.529756   149.114146     0.336585     1.071512

std       0.356527     0.527878    23.005724     0.472772     1.175053

min       0.000000     0.000000    71.000000     0.000000     0.000000

25%       0.000000     0.000000   132.000000     0.000000     0.000000

50%       0.000000     1.000000   152.000000     0.000000     0.800000

75%       0.000000     1.000000   166.000000     1.000000     1.800000

max       1.000000     2.000000   202.000000     1.000000     6.200000


              slope           ca         thal       target
count  1025.000000  1025.000000  1025.000000  1025.000000
mean      1.385366     0.754146     2.323902     0.513171
std       0.617755     1.030798     0.620660     0.500070
min       0.000000     0.000000     0.000000     0.000000
25%       1.000000     0.000000     2.000000     0.000000
50%       1.000000     0.000000     2.000000     1.000000
75%       2.000000     1.000000     3.000000     1.000000
max       2.000000     4.000000     3.000000     1.000000
```

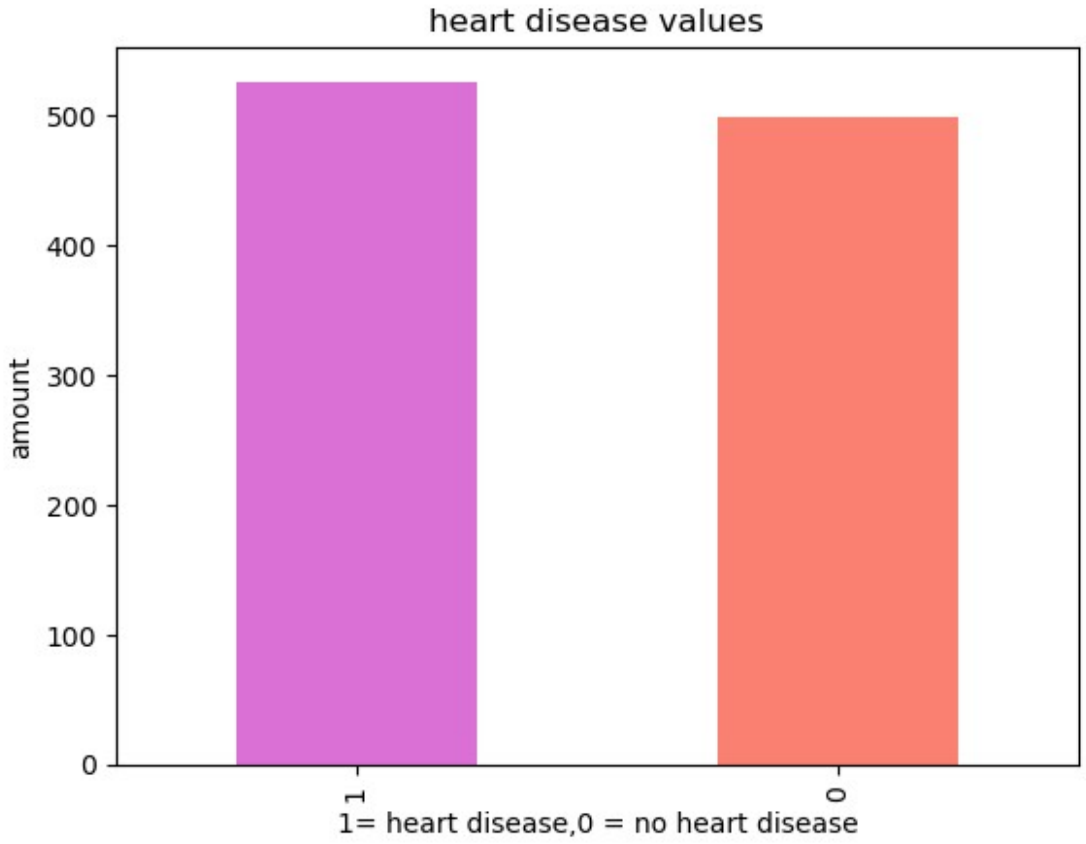
```
questions=["1.how many people have heart disease and how many people
don't?",
           "2.people of which sex have the most heart disease?",
           "3.people of which sex has which type of chest pain most?",
           "4. people with which chest pain are prone to have most
heart disease?"]
questions
```

```
["1.how many people have heart disease and how many people don't?",
 '2.people of which sex have the most heart disease?',
 '3.people of which sex has which type of chest pain most?',
 '4. people with which chest pain are prone to have most heart
disease?']
```

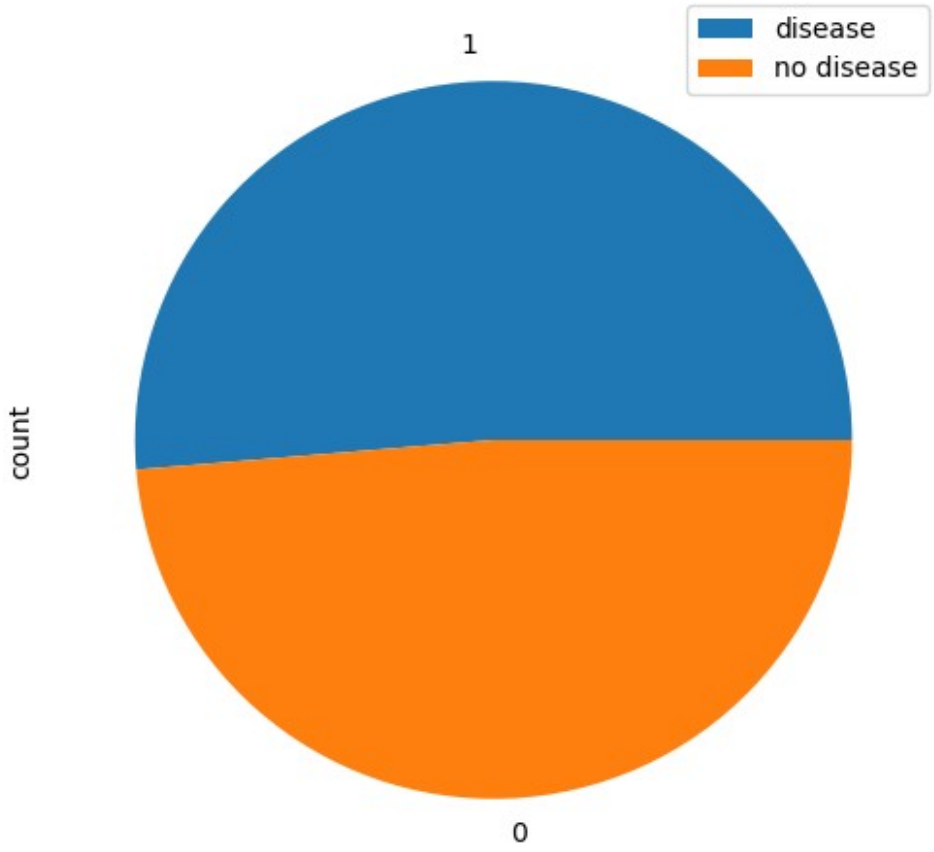
```
df.target.value_counts()
```

```
target
1    526
0    499
Name: count, dtype: int64
```

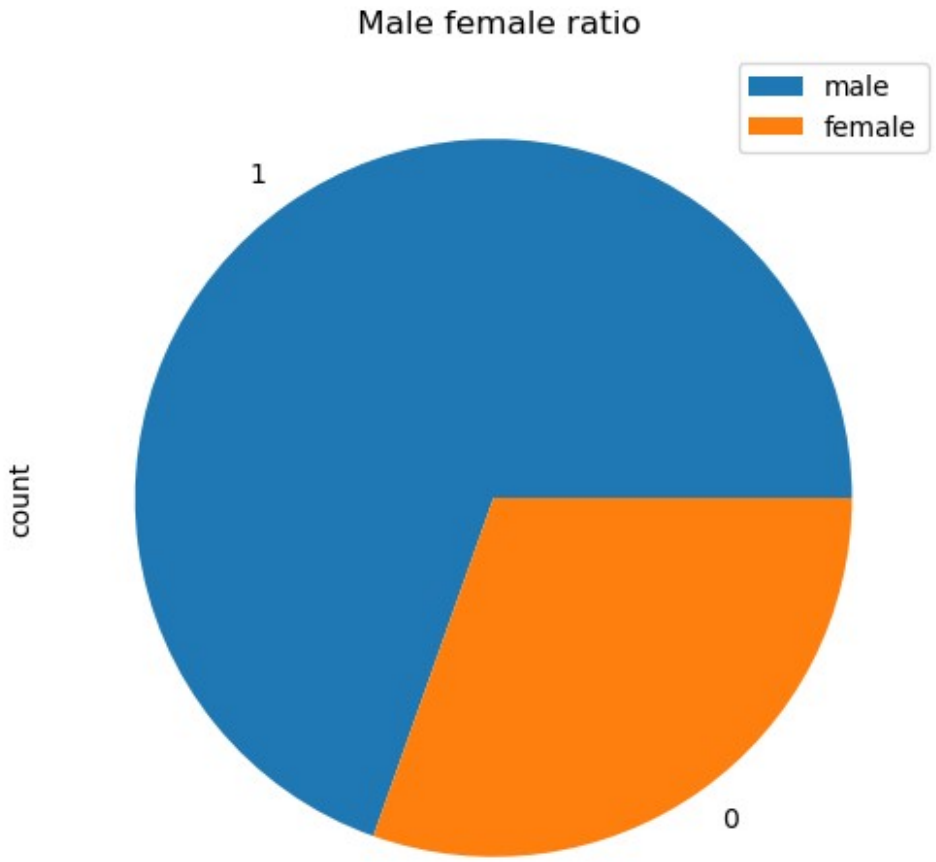
```
df.target.value_counts().plot(kind='bar', color=["orchid","salmon"])
plt.title("heart disease values")
plt.xlabel("1= heart disease,0 = no heart disease")
plt.ylabel("amount");
```




```
df.target.value_counts().plot(kind='pie',figsize=(8,6))
plt.legend(["disease","no disease"]);
```



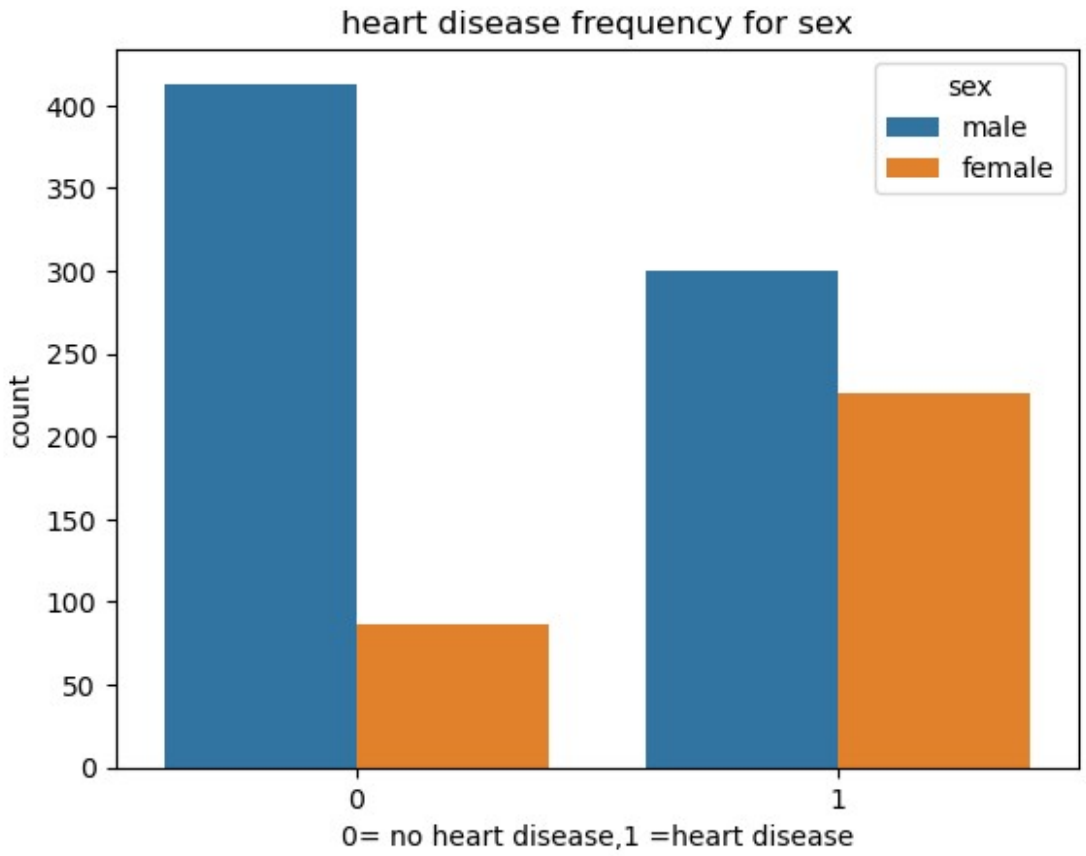

```
df.sex.value_counts()

sex
1    713
0    312
Name: count, dtype: int64

df.sex.value_counts().plot(kind='pie',figsize=(8,6))
plt.title(' Male female ratio')
plt.legend(['male','female']);
```




```
pd.crosstab(df.target,df.sex)
```

sex	0	1
target		
0	86	413
1	226	300

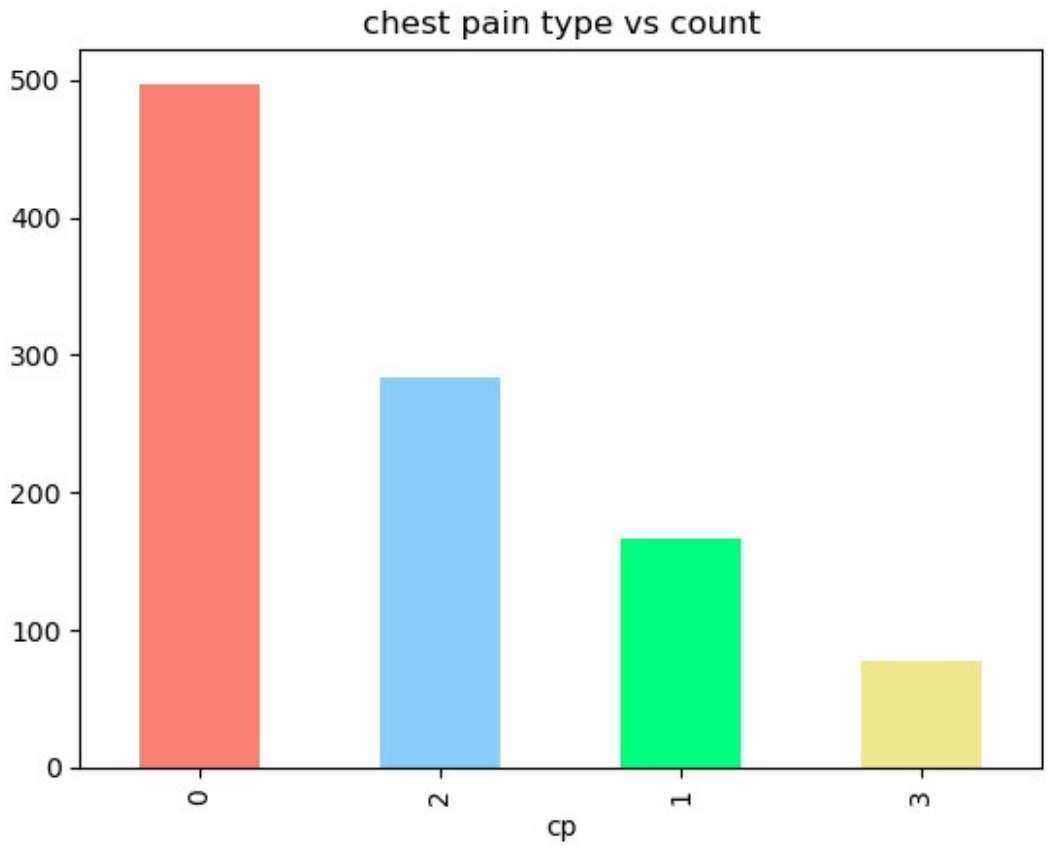
```
sns.countplot(x= 'target',data= df, hue='sex')
plt.title("heart disease frequency for sex")
plt.xlabel("0= no heart disease,1 =heart disease");
```




```
df.cp.value_counts()
```

```
cp
0    497
2    284
1    167
3     77
Name: count, dtype: int64
```

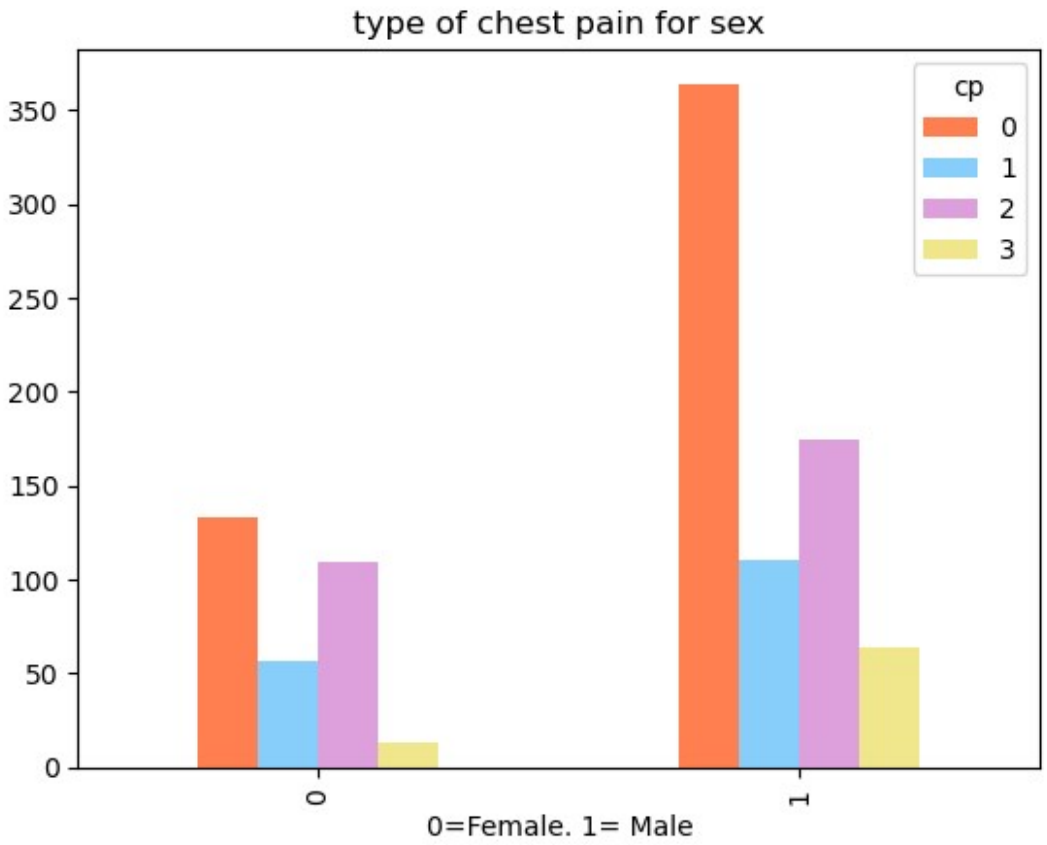
```
df.cp.value_counts().plot(kind='bar',color=['salmon','lightskyblue','s
pringgreen','khaki'])
plt.title('chest pain type vs count');
```




```
pd.crosstab(df.sex,df.cp)
```

sex	cp 0	1	2	3
0	133	57	109	13
1	364	110	175	64

```
pd.crosstab(df.sex,df.cp).plot(kind='bar',color=['coral','lightskyblue
','plum','khaki'])
plt.title('type of chest pain for sex')
plt.xlabel('0=Female. 1= Male');
```



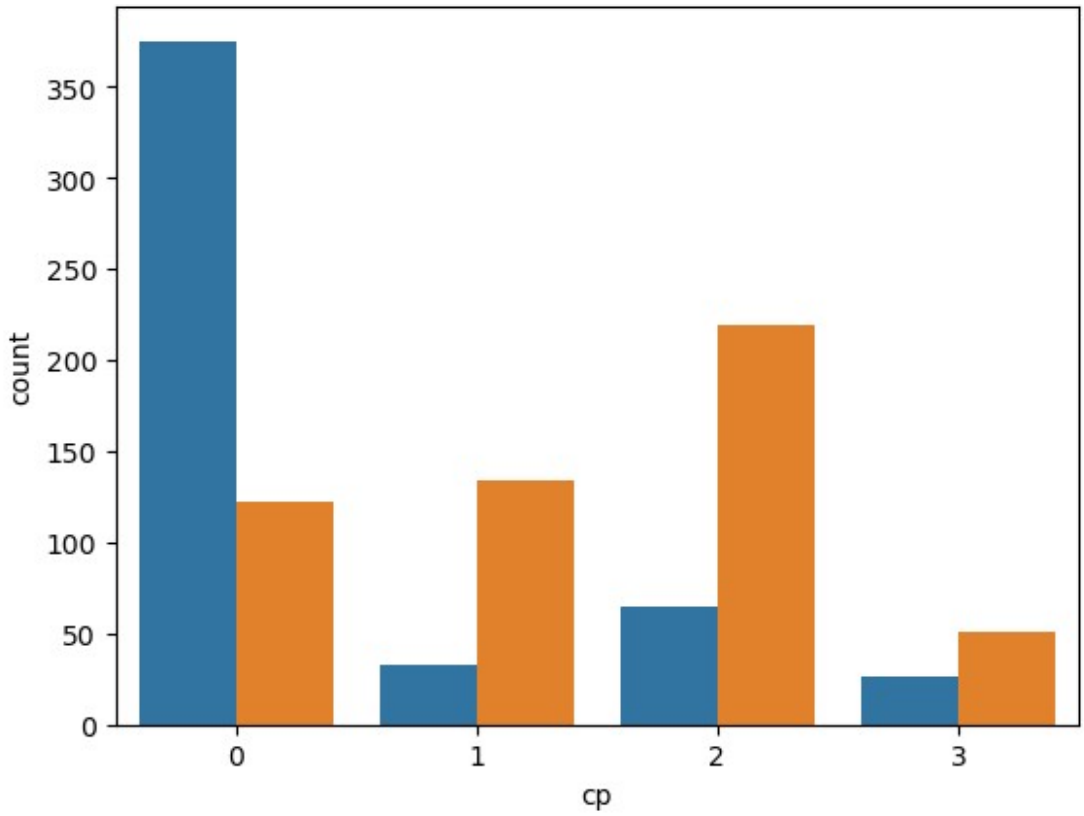

```
pd.crosstab(df.cp,df.target)

target    0    1
cp
0       375  122
1        33  134
2        65  219
3        26   51

if df['target'].dtype != 'object':
    df['target'] = df['target'].astype(str

print(df.head())
print(df.dtypes)

   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak
slope  \
0   52  NaN   0       125   212    0        1      168      0      1.0
2
1   53  NaN   0       140   203    1        0      155      1      3.1
0
2   70  NaN   0       145   174    0        1      125      1      2.6
0
3   61  NaN   0       148   203    0        1      161      0      0.0
```

```
2
4   62  NaN   0       138   294    1        1      106      0      1.9
1

   ca  thal  target
0   2     3       0
1   0     3       0
2   0     3       0
3   1     3       0
4   3     2       0
age           int64
sex          object
cp            int64
trestbps      int64
chol          int64
fbs           int64
restecg       int64
thalach       int64
exang         int64
oldpeak     float64
slope         int64
ca            int64
thal          int64
target        int64
dtype: object
```

```
sns.countplot(x='cp',data=df,hue='target');
```

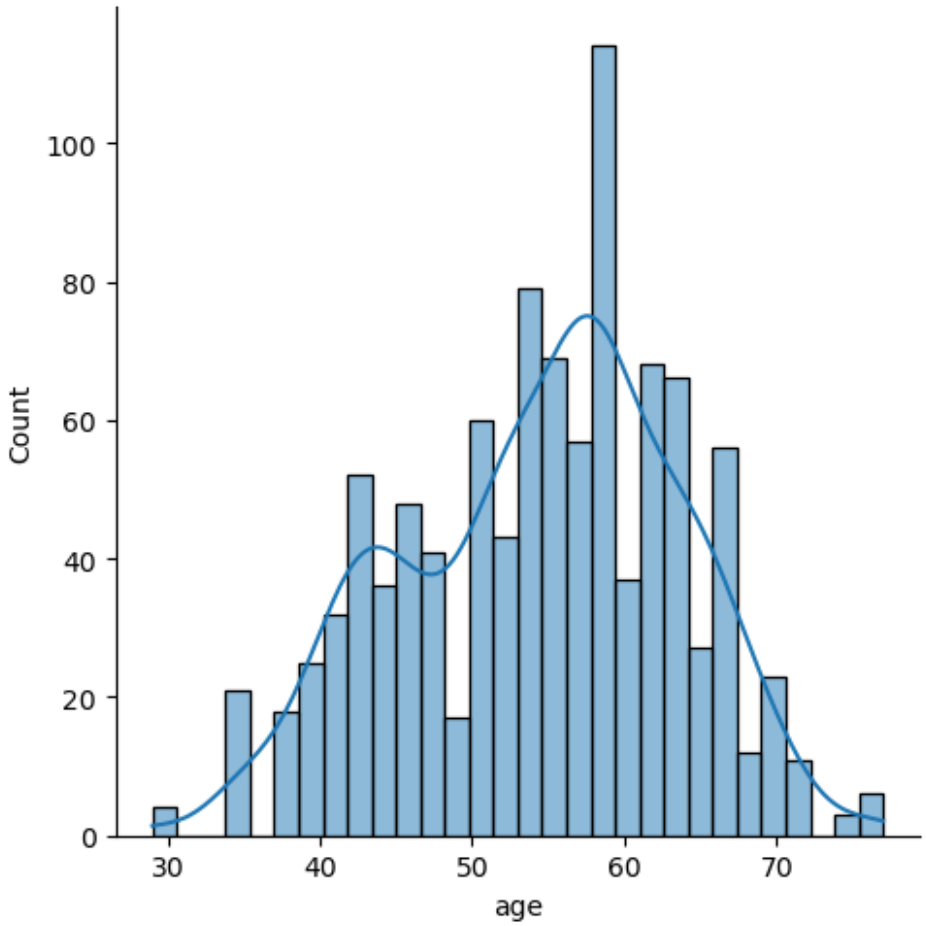
```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call
last)
Cell In[41], line 1
----> 1 sns.countplot(x='cp',data=df,hue='target')

File ~\anaconda3\Lib\site-packages\seaborn\categorical.py:2955, in
countplot(data, x, y, hue, order, hue_order, orient, color, palette,
saturation, width, dodge, ax, **kwargs)
   2952 if ax is None:
   2953     ax = plt.gca()
-> 2955 plotter.plot(ax, kwargs)
   2956 return ax

File ~\anaconda3\Lib\site-packages\seaborn\categorical.py:1587, in
_BarPlotter.plot(self, ax, bar_kws)
   1585 """Make the plot."""
   1586 self.draw_bars(ax, bar_kws)
-> 1587 self.annotate_axes(ax)
   1588 if self.orient == "h":
   1589     ax.invert_yaxis()
```

```
File ~\anaconda3\Lib\site-packages\seaborn\categorical.py:767, in
_CategoricalPlotter.annotate_axes(self, ax)
    764     ax.set_ylim(-.5, len(self.plot_data) - .5, auto=None)
    766 if self.hue_names is not None:
--> 767     ax.legend(loc="best", title=self.hue_title)

File ~\anaconda3\Lib\site-packages\matplotlib\axes\_axes.py:322, in
Axes.legend(self, *args, **kwargs)
    204 @_docstring.dedent_interpd
    205 def legend(self, *args, **kwargs):
    206     """
    207     Place a legend on the Axes.
    208
   (...)
    320     .. plot:: gallery/text_labels_and_annotations/legend.py
    321     """
--> 322     handles, labels, kwargs =
mlegend._parse_legend_args([self], *args, **kwargs)
    323     self.legend_ = mlegend.Legend(self, handles, labels,
**kwargs)
    324     self.legend_._remove_method = self._remove_legend

File ~\anaconda3\Lib\site-packages\matplotlib\legend.py:1361, in
_parse_legend_args(axs, handles, labels, *args, **kwargs)
   1357     handles = [handle for handle, label
   1358                in zip(_get_legend_handles(axs, handlers),
labels)]
   1360 elif len(args) == 0:  # 0 args: automatically detect labels
and handles.
-> 1361     handles, labels = _get_legend_handles_labels(axs,
handlers)
   1362     if not handles:
   1363         log.warning(
   1364             "No artists with labels found to put in legend.
Note that "
   1365             "artists whose label start with an underscore are
ignored "
   1366             "when legend() is called with no argument.")

File ~\anaconda3\Lib\site-packages\matplotlib\legend.py:1291, in
_get_legend_handles_labels(axs, legend_handler_map)
   1289 for handle in _get_legend_handles(axs, legend_handler_map):
   1290     label = handle.get_label()
-> 1291     if label and not label.startswith('_'):
   1292         handles.append(handle)
   1293         labels.append(label)

AttributeError: 'numpy.int64' object has no attribute 'startswith'
```



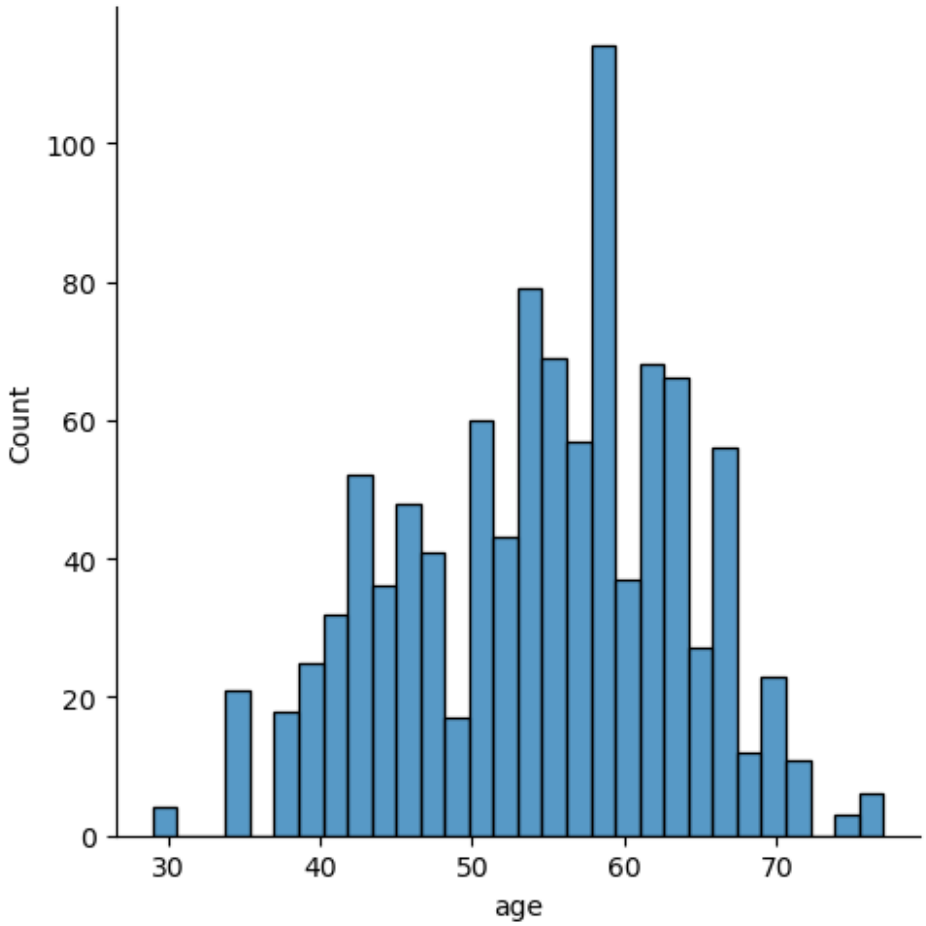

```
sns.displot(x='age',data=df,bins=30,kde=True);
```

```
C:\Users\HP\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



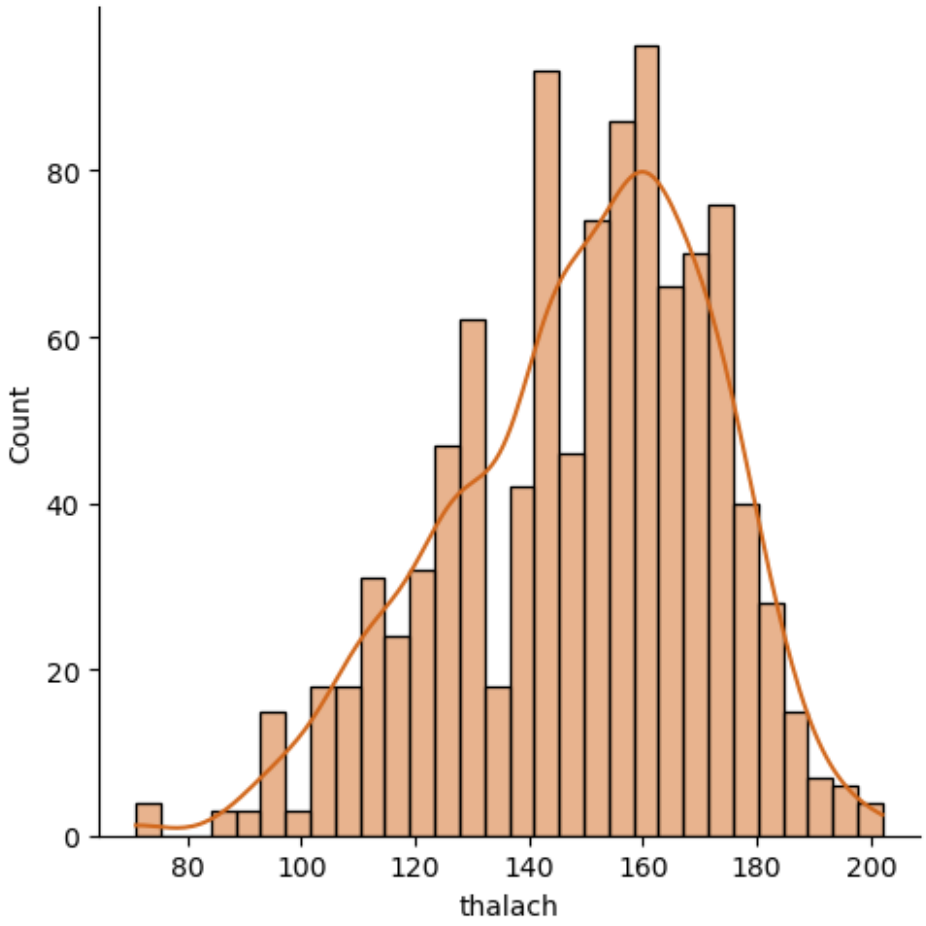

```
sns.displot(x='age',data=df,bins=30,kde=False);
```

```
C:\Users\HP\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```




```
sns.displot(x='thalach',data=df,bins=30,kde=True,color='chocolate');
```

```
C:\Users\HP\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```


Count
0
20
40
60
80
80
100
120
140
160
180
200
thalach