In [2]:
```
import numpy as np
```

In [3]:
```
import pandas as pd
```

In [ ]:
```
import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
import seaborn as sns
```

In [6]:
```
h_data = pd.read_csv('haberman.csv')
```

In [8]:
```
df_new=h_data.drop(['status'], axis=1)
```

In [10]:
```
h_data.head()
```

Out[10]:

	age	operation_year	axil_nodes	status
0	30	64	1	1
1	30	62	3	1
2	30	65	0	1
3	31	59	2	1
4	31	65	4	1

In [11]:
```
h_data.shape
```

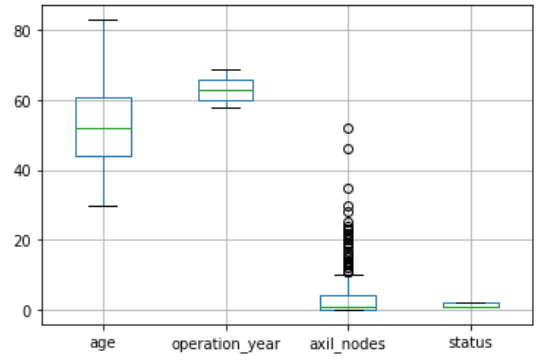
Out[11]:
```
(306, 4)
```

In [12]:
```
h_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 306 entries, 0 to 305
Data columns (total 4 columns):
age               306 non-null int64
operation_year    306 non-null int64
axil_nodes        306 non-null int64
status            306 non-null int64
dtypes: int64(4)
memory usage: 9.7 KB
```

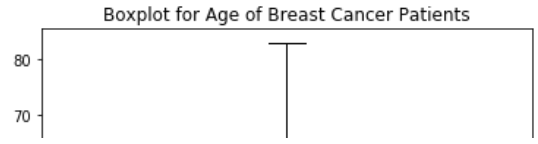
In [13]:
```
#Figure 1
```

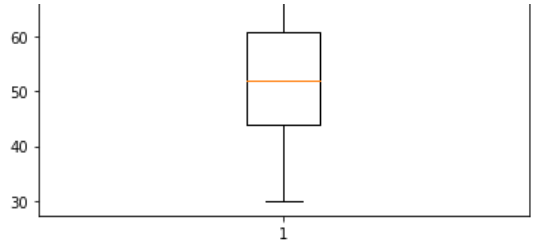
In [13]:
```
h_data.boxplot()
plt.show()
```



In [15]:
```
#Figure 2
```

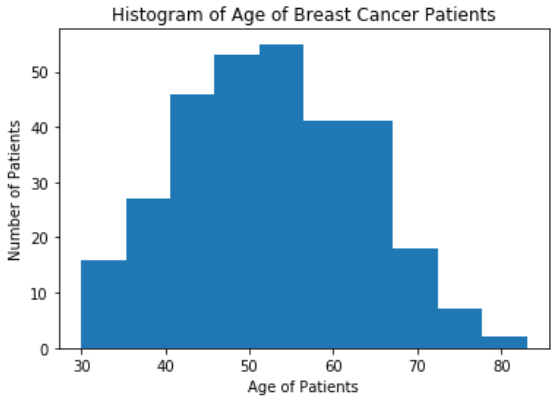
In [14]:
```
plt.boxplot(h_data.age)
plt.title('Boxplot for Age of Breast Cancer Patients')
plt.show()
```





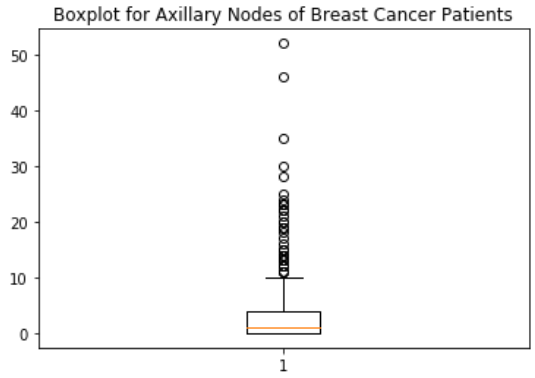
In [17]:
```
#Figure 3
#Histogram
```

In [15]:
```
plt.hist(h_data.age, bins=10)
plt.xlabel('Age of Patients')
plt.ylabel('Number of Patients')
plt.title('Histogram of Age of Breast Cancer Patients')
plt.show()
```



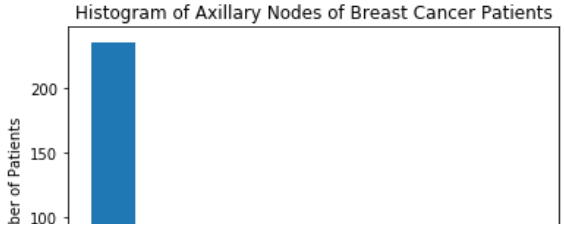
In [19]:
```
#Figure 4
```

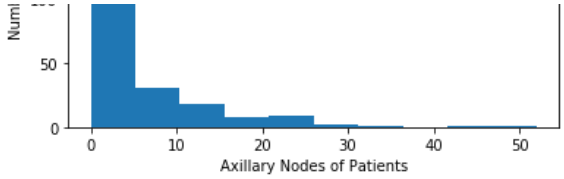
In [16]:
```
plt.boxplot(h_data.axil_nodes)
plt.title('Boxplot for Axillary Nodes of Breast Cancer Patients')
plt.show()
```



In [26]:
```
#Figure 5
#Histogram
```

In [17]:
```
plt.hist(h_data.axil_nodes, bins=10)
plt.xlabel('Axillary Nodes of Patients')
plt.ylabel('Number of Patients')
plt.title('Histogram of Axillary Nodes of Breast Cancer Patients')
plt.show()
```

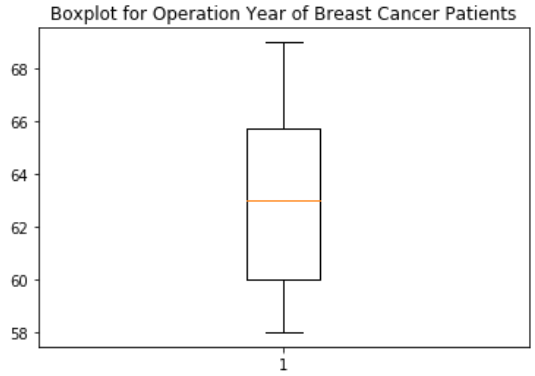




In [21]:
```
#Figure 6
```

In [22]:
```
plt.boxplot(h_data.operation_year)
plt.title('Boxplot for Operation Year of Breast Cancer Patients')
```

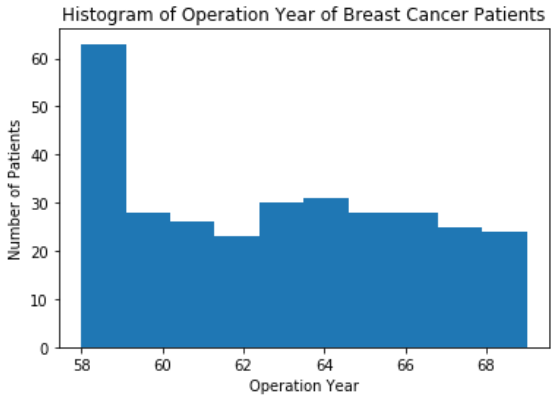
Out[22]: Text(0.5, 1.0, 'Boxplot for Operation Year of Breast Cancer Patients')



In [23]:
```
#Figure 7
#Histogram
```

In [24]:
```
plt.hist(h_data.operation_year, bins=10)
plt.xlabel('Operation Year')
plt.ylabel('Number of Patients')
plt.title('Histogram of Operation Year of Breast Cancer Patients')
```

Out[24]: Text(0.5, 1.0, 'Histogram of Operation Year of Breast Cancer Patients')



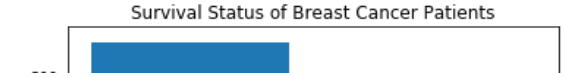
In [25]:
```
#Figure 8
# Bar diagram
```

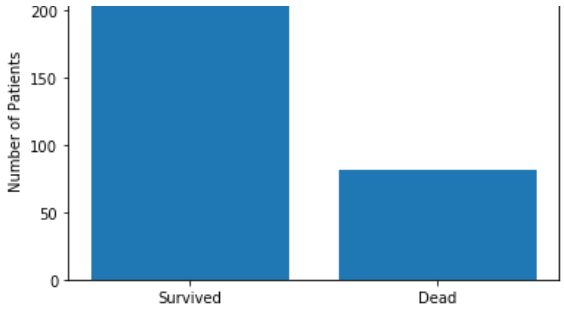
In [19]:
```
objects = ('Survived', 'Dead')
```

In [20]:
```
x_pos = np.arange(len(objects))
```

In [21]:
```
status_fre=[225, 81]
```

In [22]:
```
plt.bar(x_pos, status_fre)
plt.xticks(x_pos, objects)
plt.ylabel('Number of Patients')
plt.title('Survival Status of Breast Cancer Patients')
plt.show()
```

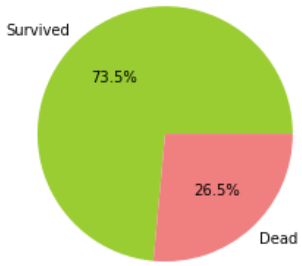




In [31]:
```
#Figure 9
#Pie Chart
```

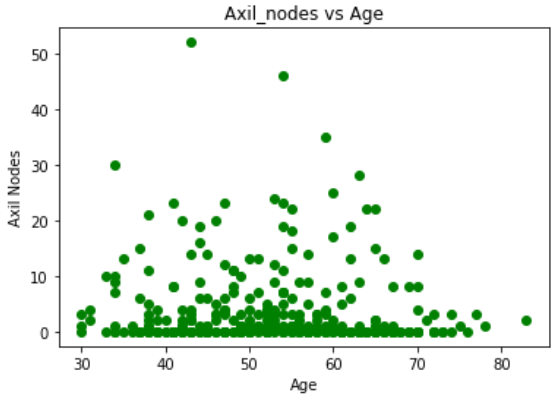
In [24]:
```
status_fre=[225, 81]
```

In [25]:
```
plt.pie(status_fre, labels=['Survived', 'Dead'], colors=['yellowgreen', 'lightcoral'],  autopct='%.1f%%')
plt.show()
```



In [35]:
```
#Figure 10
#Scatter Diagram
```

In [26]:
```
plt.scatter(h_data['age'],h_data['axil_nodes'], color = 'g')
plt.xlabel('Age')
plt.ylabel('Axil Nodes')
plt.title('Axil_nodes vs Age')
plt.show()
```



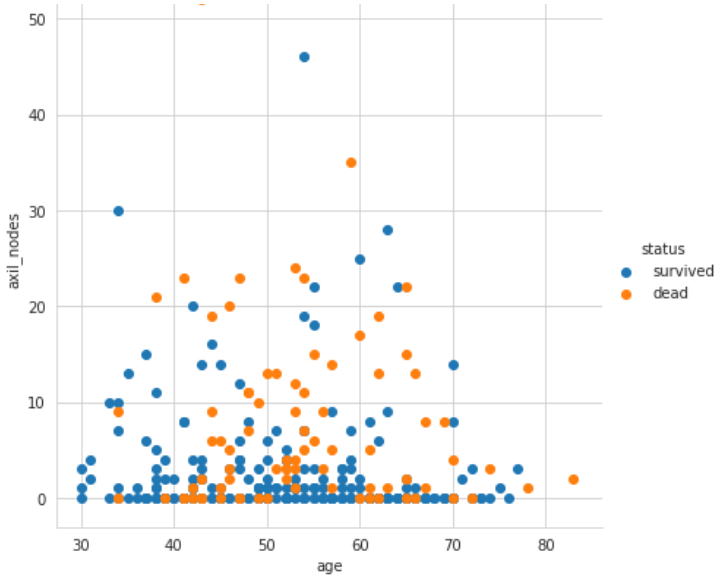
In [37]:
```
#to label status variable in readable format, 1 labeled as 'survived' and 2 labeled as 'dead'
```

In [27]:
```
h_data['status'] = h_data['status'].map({1:'survived', 2:'dead'})
```

In [39]:
```
#Figure 11
```

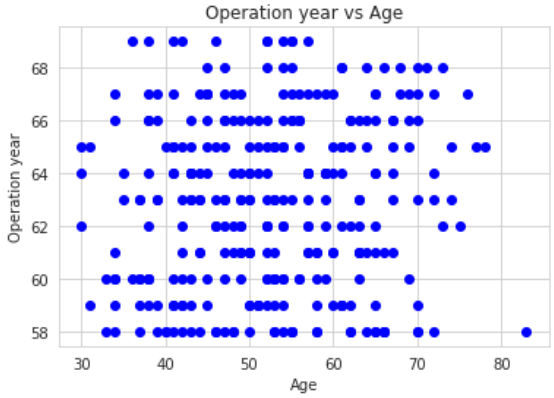
In [28]:
```
sns.set_style('whitegrid');
```

In [29]:
```
sns.FacetGrid(h_data, hue = 'status', height = 6)\
.map(plt.scatter, 'age', 'axil_nodes')\
.add_legend();
```



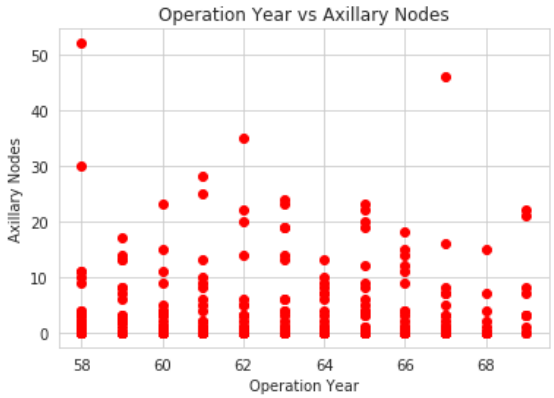
In [44]:
```
#Figure 12
#Scatter Diagram
```

In [30]:
```
plt.scatter(h_data['age'],h_data['operation_year'], c = 'b')
plt.xlabel('Age')
plt.ylabel('Operation year')
plt.title('Operation year vs Age')
plt.show()
```



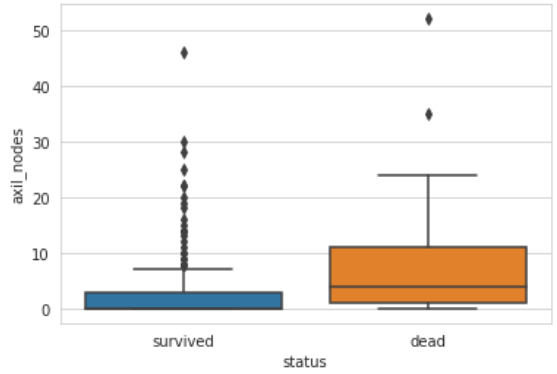
In [46]:
```
#Figure 13
#Scatter Diagram
```

In [22]:
```
plt.scatter(h_data['operation_year'],h_data['axil_nodes'], color = 'r')
plt.xlabel('Operation Year')
plt.ylabel('Axillary Nodes')
plt.title('Operation Year vs Axillary Nodes')
plt.show()
```

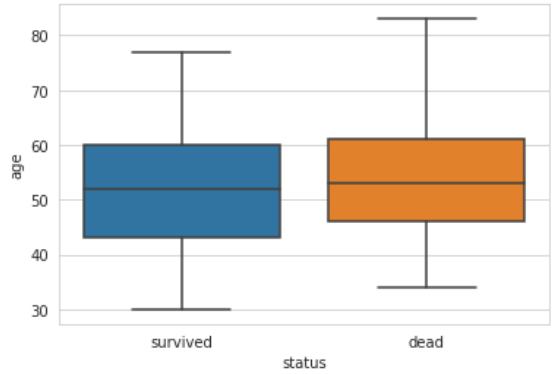


In [48]:
```
#Figure 14
```

In [32]:
```
sns.boxplot(x='status',y='axil_nodes', data=h_data)
plt.show()
```

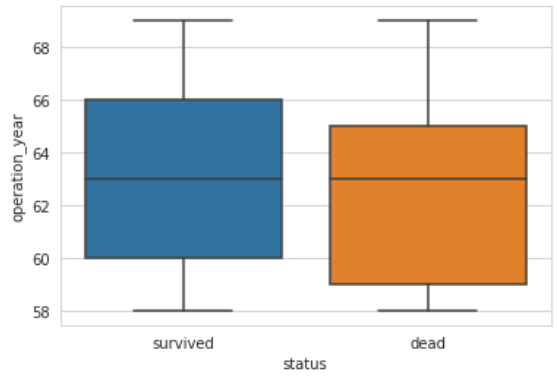


In [33]:
```
#Figure 15
sns.boxplot(x='status',y='age', data=h_data)
plt.show()
```



In [56]:
```
#Figure 16
```

In [34]:
```
sns.boxplot(x='status',y='operation_year', data=h_data)
plt.show()
```



In [58]:
```
#Figure 17
#Pairplots
```

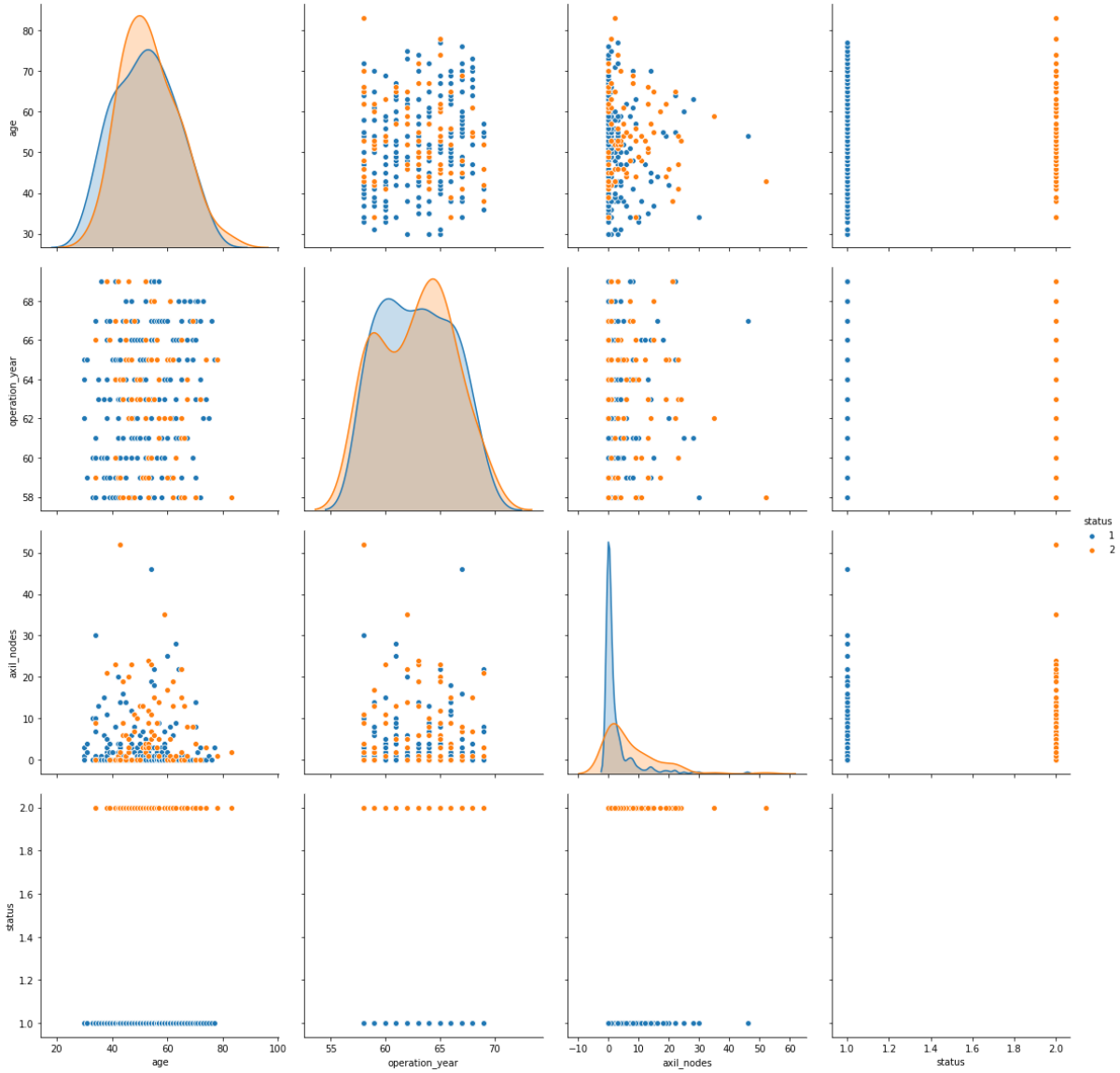
In [24]:
```
sns.set_style('whitegrid')
```

In [10]:
```
h_data['status'] = h_data['status'].map({1:'survived', 2:'dead'})
```

In [7]:
```
sns.pairplot(h_data, hue = 'status', height = 4)
plt.show()
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages/scipy/stats/stats.py:1713: FutureWarning: U
sing a non-tuple sequence for multidimensional indexing is deprecated; use `arr[tuple(seq)]` instead of `a
rr[seq]`. In the future this will be interpreted as an array index, `arr[np.array(seq)]`, which will resul
t either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages/statsmodels/nonparametric/kde.py:487: Runti
meWarning: invalid value encountered in true_divide
  binned = fast_linbin(X, a, b, gridsize) / (delta * nobs)
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages/statsmodels/nonparametric/kdetools.py:34: R
untimeWarning: invalid value encountered in double_scalars
  FAC1 = 2*(np.pi*bw/RANGE)**2
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages/numpy/core/fromnumeric.py:83: RuntimeWarnin
g: invalid value encountered in reduce
  return ufunc.reduce(obj, axis, dtype, out, **passkwargs)
```



In [12]:
```
#Summary Measures
```

In [13]:
```
h_data.describe()
```

Out[13]:

	age	operation_year	axil_nodes
count	306.000000	306.000000	306.000000
mean	52.457516	62.852941	4.026144
std	10.803452	3.249405	7.189654
min	30.000000	58.000000	0.000000
25%	44.000000	60.000000	0.000000
50%	52.000000	63.000000	1.000000
75%	60.750000	65.750000	4.000000
max	83.000000	69.000000	52.000000

In [15]:
```
h_data.mode()
```

In [15]:
```
h_data.mode()
```

Out[15]:

	age	operation_year	axil_nodes	status
0	52	58	0	NaN

In [19]:
```
h_data.median()
```

Out[19]:
```
age               52.0
operation_year    63.0
axil_nodes         1.0
status             NaN
dtype: float64
```

In [17]:
```
h_data.corr()
```

Out[17]:

	age	operation_year	axil_nodes
age	1.000000	0.089529	-0.063176
operation_year	0.089529	1.000000	-0.003764
axil_nodes	-0.063176	-0.003764	1.000000

In [18]:
```
h_data.cov()
```

Out[18]:

	age	operation_year	axil_nodes
age	116.714583	3.142912	-4.907082
operation_year	3.142912	10.558631	-0.087946
axil_nodes	-4.907082	-0.087946	51.691118

In [ ]:
```

```