In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import f1_score
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn import svm
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
from sklearn.metrics import precision_score
```

In [2]:
```
df=pd.read_csv('D:\AI & EC\W15\Heart_Disease_Prediction.csv')
df
```

Out[2]:

	Age	Sex	Chest pain type	BP	Cholesterol	FBS over 120	EKG results	Max HR	Exercise angina	ST depression	Slope of ST	Number of vessels fluro	Thallium	Heart Disease
0	70	1	4	130	322	0	2	109	0	2.4	2	3	3	Presence
1	67	0	3	115	564	0	2	160	0	1.6	2	0	7	Absence
2	57	1	2	124	261	0	0	141	0	0.3	1	0	7	Presence
3	64	1	4	128	263	0	0	105	1	0.2	2	1	7	Absence
4	74	0	2	120	269	0	2	121	1	0.2	1	1	3	Absence
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
265	52	1	3	172	199	1	0	162	0	0.5	1	0	7	Absence
266	44	1	2	120	263	0	0	173	0	0.0	1	0	7	Absence
267	56	0	2	140	294	0	2	153	0	1.3	2	0	3	Absence
268	57	1	4	140	192	0	0	148	0	0.4	2	0	6	Absence
269	67	1	4	160	286	0	2	108	1	1.5	2	3	3	Presence

270 rows × 14 columns

In [3]:
```
df = pd.read_csv('D:\AI & EC\W15\Heart_Disease_Prediction.csv')
encoder = LabelEncoder()
encoder.fit(df['Heart Disease'])
labels = encoder.transform(df['Heart Disease'])
df['Heart Disease'] = labels
df
```

Out[3]:

	Age	Sex	Chest pain type	BP	Cholesterol	FBS over 120	EKG results	Max HR	Exercise angina	ST depression	Slope of ST	Number of vessels fluro	Thallium	Heart Disease
0	70	1	4	130	322	0	2	109	0	2.4	2	3	3	1
1	67	0	3	115	564	0	2	160	0	1.6	2	0	7	0
2	57	1	2	124	261	0	0	141	0	0.3	1	0	7	1
3	64	1	4	128	263	0	0	105	1	0.2	2	1	7	0
4	74	0	2	120	269	0	2	121	1	0.2	1	1	3	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
265	52	1	3	172	199	1	0	162	0	0.5	1	0	7	0
266	44	1	2	120	263	0	0	173	0	0.0	1	0	7	0
267	56	0	2	140	294	0	2	153	0	1.3	2	0	3	0
268	57	1	4	140	192	0	0	148	0	0.4	2	0	6	0
269	67	1	4	160	286	0	2	108	1	1.5	2	3	3	1

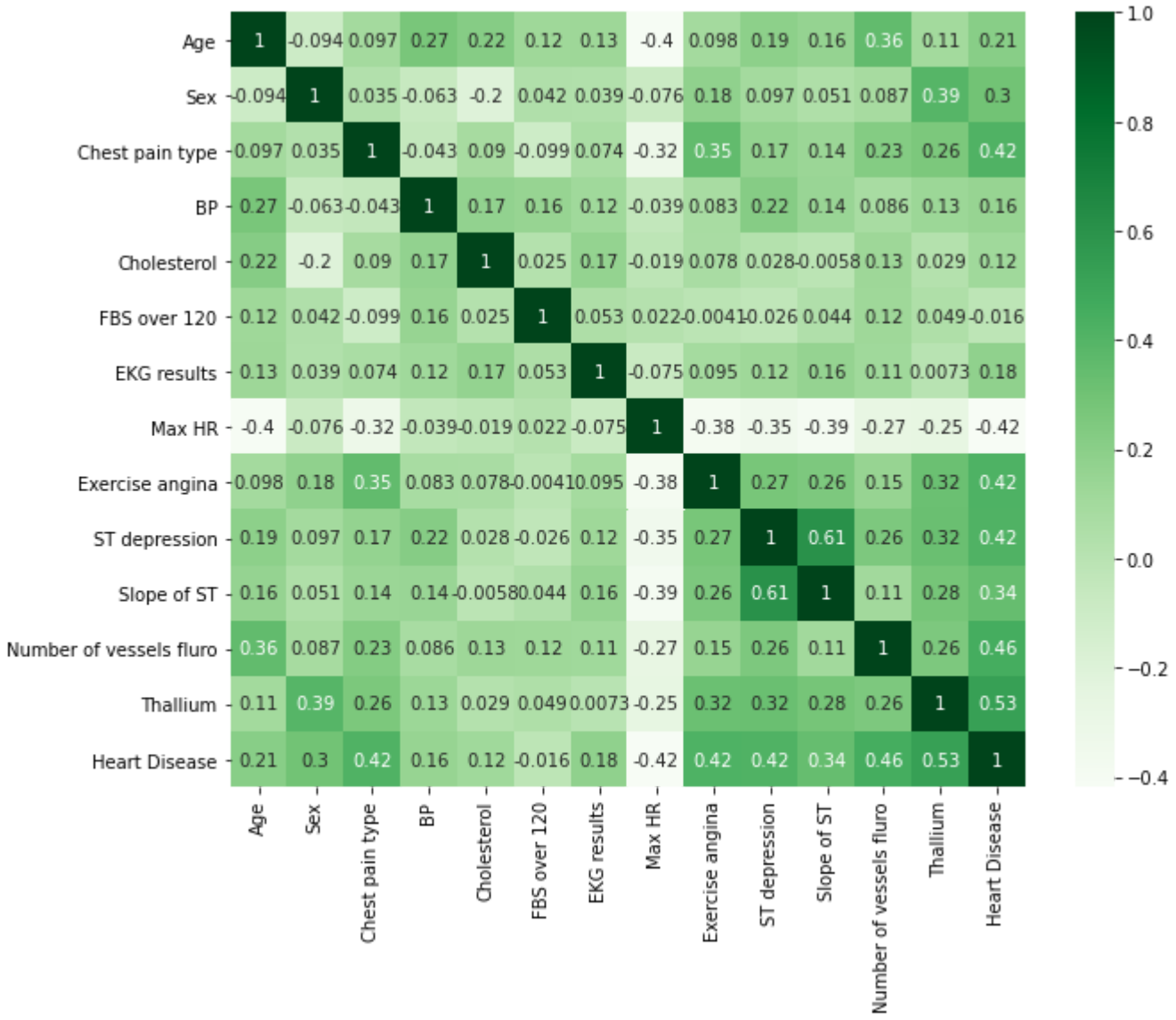
270 rows × 14 columns

In [4]:
```
df.isnull().sum()
```

Out[4]:
```
Age                       0
Sex                       0
Chest pain type           0
BP                        0
Cholesterol               0
FBS over 120              0
EKG results               0
Max HR                    0
Exercise angina           0
ST depression             0
Slope of ST               0
Number of vessels fluro   0
Thallium                  0
Heart Disease             0
dtype: int64
```

In [5]:
```
fig, ax = plt.subplots(figsize=(10,8))
sns.heatmap(df.corr(), annot=True, cmap='Greens')
```

Out[5]: <AxesSubplot:>



In [6]:
```
x=df.drop(['Age','Sex','BP','Cholesterol','FBS over 120','EKG results','Max HR','Slope of ST','Heart Disease'],axis=1)
y=df['Heart Disease']
x
```

Out[6]:

	Chest pain type	Exercise angina	ST depression	Number of vessels fluro	Thallium
0	4	0	2.4	3	3
1	3	0	1.6	0	7
2	2	0	0.3	0	7
3	4	1	0.2	1	7
4	2	1	0.2	1	3
...	...	...	...	...	...
265	3	0	0.5	0	7
266	2	0	0.0	0	7
267	2	0	1.3	0	3
268	4	0	0.4	0	6
269	4	1	1.5	3	3

270 rows × 5 columns

In [7]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=0,test_size=0.2)
```

In [8]:
```
lr=LogisticRegression(max_iter=10000)
lr.fit(x_train,y_train)
pred_1=lr.predict(x_test)
score_1=accuracy_score(y_test,pred_1)
```

In [9]:
```
score_1
```

Out[9]:
```
0.7592592592592593
```

In [10]:
```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
pred_2=rfc.predict(x_test)
score_2=accuracy_score(y_test,pred_2)
```

In [11]:
```
score_2
```

Out[11]:
```
0.7592592592592593
```

In [12]:
```

list_1=[]
for i in range(1,21):
    knn=KNeighborsClassifier(n_neighbors=i)
    knn.fit(x_train,y_train)
    preds=knn.predict(x_test)
    scores=accuracy_score(y_test,preds)
    list_1.append(scores)
max(list_1)

```

Out[12]:
```
0.7777777777777778
```

In [13]:
```
cm = confusion_matrix(y_test, preds)
print("Confusion matrix:\n", cm)
```

```
Confusion matrix:
 [[26  4]
 [ 8 16]]
```

In [14]:
```
f1 = f1_score(y_test, preds)
print("F1 score:", f1)
```

```
F1 score: 0.7272727272727272
```

In [15]:
```
precision = precision_score(y_test, preds)
print("Precision:", precision)
```

```
Precision: 0.8
```

In [16]:
```
svm = svm.SVC(max_iter=10000)
svm.fit(x_train, y_train)
pred_2 = svm.predict(x_test)
score_2 = accuracy_score(y_test, pred_2)
score_2
```

Out[16]:
```
0.7407407407407407
```

In [ ]:
```

```