In [10]:

```
# importing required libraries
import pandas as pd
import numpy as np
from sklearn.datasets import load_digits
import matplotlib.pyplot as plt
```

In [11]:

```
digits = load_digits()
```

In [12]:

```
dir(digits)
```

Out[12]:

```
['DESCR', 'data', 'feature_names', 'frame', 'images', 'target', 'target_names']
```

In [9]:

```
digits.images[1]
```

Out[9]:

```
array([[ 0.,  0.,  0., 12., 13.,  5.,  0.,  0.],
       [ 0.,  0.,  0., 11., 16.,  9.,  0.,  0.],
       [ 0.,  0.,  3., 15., 16.,  6.,  0.,  0.],
       [ 0.,  7., 15., 16., 16.,  2.,  0.,  0.],
       [ 0.,  0.,  1., 16., 16.,  3.,  0.,  0.],
       [ 0.,  0.,  1., 16., 16.,  6.,  0.,  0.],
       [ 0.,  0.,  1., 16., 16.,  6.,  0.,  0.],
       [ 0.,  0.,  0., 11., 16., 10.,  0.,  0.]])
```

In [15]:

```
## plotting different images
%matplotlib inline
plt.gray()
```

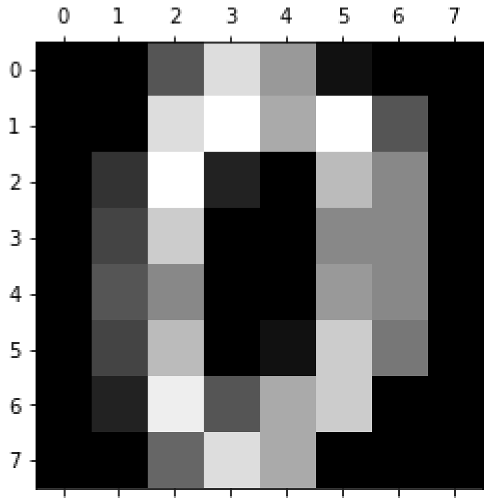
```
<Figure size 432x288 with 0 Axes>
```

In [16]:

```
plt.matshow(digits.images[0])
```

Out[16]:

```
<matplotlib.image.AxesImage at 0x1bec80f3b80>
```



Splitting Data in X and Y

In [17]:

```
X = digits.data
Y = digits.target
```

In [18]:

```
## Splitting Data for testing and training
from sklearn.model_selection import train_test_split
```

In [19]:

```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.2,random_state=1)
```

In [20]:

```
## Importing Algorithm
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
```

In [21]:

```
model.fit(X_train,Y_train)
```

```
C:\Users\hp\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:763: Convergence
Warning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[21]:

```
LogisticRegression()
```

In [22]:

```
model.score(X_test,Y_test)
```

Out[22]:

```
0.9694444444444444
```

In [23]:

```
Y_pred = model.predict(X_test)
```

In [24]:

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test,Y_pred)
cm
```

Out[24]:

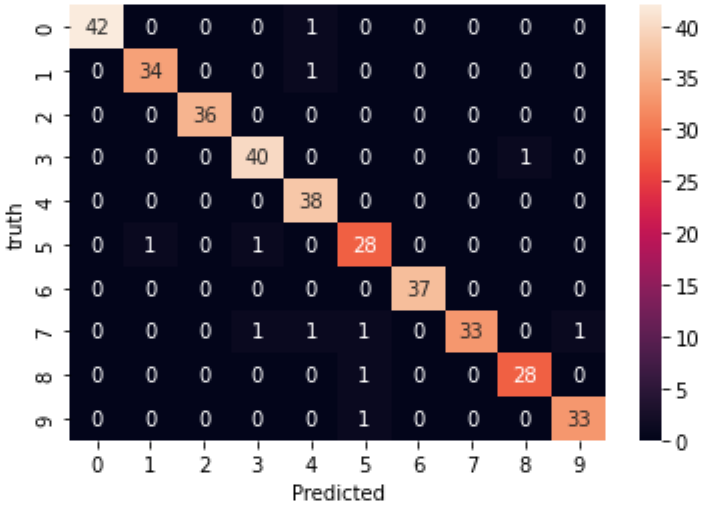
```
array([[42,  0,  0,  0,  1,  0,  0,  0,  0,  0],
       [ 0, 34,  0,  0,  1,  0,  0,  0,  0,  0],
       [ 0,  0, 36,  0,  0,  0,  0,  0,  0,  0],
       [ 0,  0,  0, 40,  0,  0,  0,  0,  1,  0],
       [ 0,  0,  0,  0, 38,  0,  0,  0,  0,  0],
       [ 0,  1,  0,  1,  0, 28,  0,  0,  0,  0],
       [ 0,  0,  0,  0,  0,  0, 37,  0,  0,  0],
       [ 0,  0,  0,  1,  1,  1,  0, 33,  0,  1],
       [ 0,  0,  0,  0,  0,  1,  0,  0, 28,  0],
       [ 0,  0,  0,  0,  0,  1,  0,  0,  0, 33]], dtype=int64)
```

In [25]:

```
import seaborn as sn
sn.heatmap(cm,annot=True)
plt.xlabel("Predicted")
plt.ylabel("truth")
```

Out[25]:

```
Text(33.0, 0.5, 'truth')
```


truth
Predicted
0 1 2 3 4 5 6 7 8 9
42 0 0 0 1 0 0 0 0 0
0 34 0 0 1 0 0 0 0 0
0 0 36 0 0 0 0 0 0 0
0 0 0 40 0 0 0 0 1 0
0 0 0 0 38 0 0 0 0 0
0 1 0 1 0 28 0 0 0 0
0 0 0 0 0 0 37 0 0 0
0 0 0 1 1 1 0 33 0 1
0 0 0 0 0 1 0 0 28 0
0 0 0 0 0 1 0 0 0 33
40
35
30
25
20
15
10
5
0

In [ ]: