In [37]:
```
import sys

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import Perceptron
from sklearn.pipeline import Pipeline
from sklearn.datasets import load_files
from sklearn.model_selection import train_test_split
from sklearn import metrics
```

In [38]:
```
languages_data_folder = 'data\languages\paragraphs'
dataset = load_files(languages_data_folder)
```

In [62]:
```
docs_train, docs_test, y_train, y_test = train_test_split(
    dataset.data, dataset.target, test_size=0.25)
```

In [63]:
```
# TASK: Build a vectorizer that splits strings into sequence of 1 to 3 charact
ers instead of word tokens
vectorizer = TfidfVectorizer(ngram_range=(1, 9), analyzer='char',
                             use_idf=False)
```

In [64]:
```
# TASK: Build a vectorizer / classifier pipeline using the previous analyzer
# the pipeline instance should stored in a variable named clf
clf = Pipeline([
    ('vec', vectorizer),
    ('clf', Perceptron(tol=1e-3)),
])
```

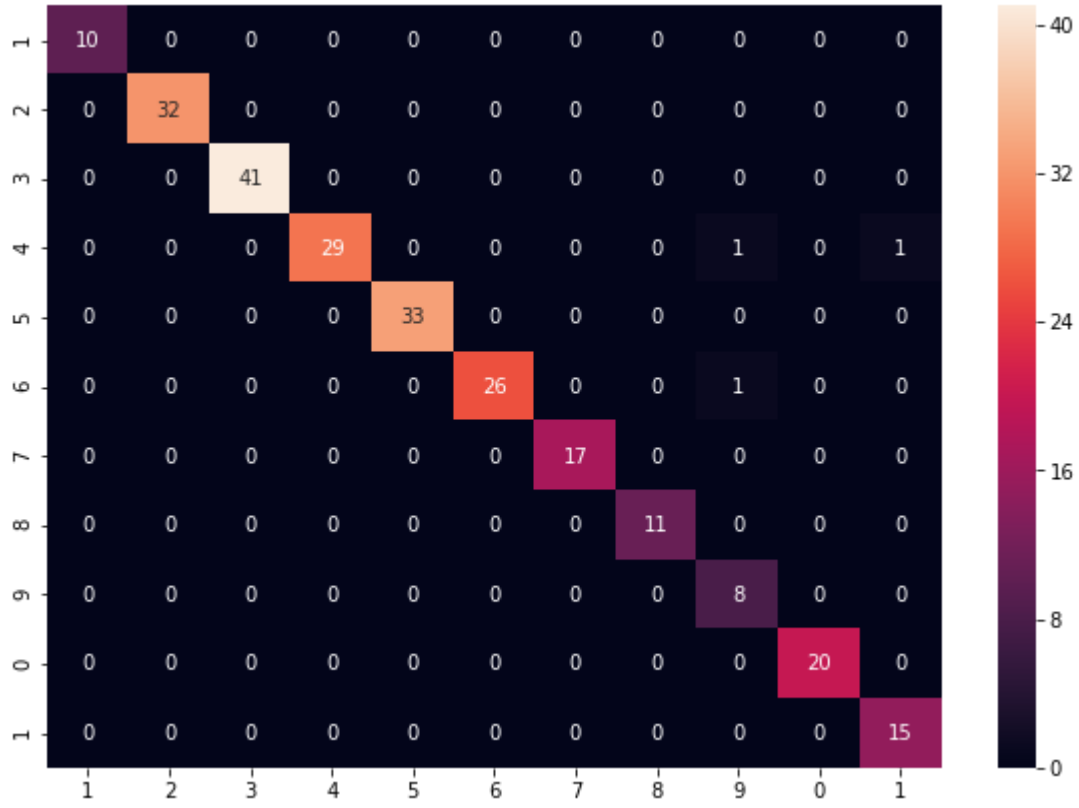
In [65]:
```
# TASK: Fit the pipeline on the training set
clf.fit(docs_train, y_train)

# TASK: Predict the outcome on the testing set in a variable named y_predicted
y_predicted = clf.predict(docs_test)

# Print the classification report
print(metrics.classification_report(y_test, y_predicted,
                                    target_names=dataset.target_names))
```

```
             precision    recall  f1-score   support

         ar       1.00      1.00      1.00        10
         de       1.00      1.00      1.00        32
         en       1.00      1.00      1.00        41
         es       1.00      0.94      0.97        31
         fr       1.00      1.00      1.00        33
         it       1.00      0.96      0.98        27
         ja       1.00      1.00      1.00        17
         nl       1.00      1.00      1.00        11
         pl       0.80      1.00      0.89         8
         pt       1.00      1.00      1.00        20
         ru       0.94      1.00      0.97        15

avg / total       0.99      0.99      0.99       245
```

In [66]:
```
# Plot the confusion matrix
cm = metrics.confusion_matrix(y_test, y_predicted)
#print(cm)

import matplotlib.pyplot as plt
#plt.matshow(cm, cmap=plt.cm.jet)
#plt.show()
```

In [72]:
```
import seaborn as sn
import pandas as pd
import matplotlib.pyplot as plt
array = cm
df_cm = pd.DataFrame(array, index = [i for i in "12345678901"],
                  columns = [i for i in "12345678901"])
plt.figure(figsize = (10,7))
sn.heatmap(df_cm, annot=True)
```

Out[72]: <matplotlib.axes._subplots.AxesSubplot at 0xd007883898>



In [68]:
```
# Predict the result on some short new sentences:

sentences = [
    u'This is a language detection test.',
    u'Ceci est un test de d\xe9tection de la langue.',
    u'Dies ist ein Test, um die Sprache zu erkennen.',
]
predicted = clf.predict(sentences)

for s, p in zip(sentences, predicted):
    print(u'The language of "%s" is "%s"' % (s, dataset.target_names[p]))
```

```
The language of "This is a language detection test." is "en"
The language of "Ceci est un test de détection de la langue." is "fr"
The language of "Dies ist ein Test, um die Sprache zu erkennen." is "de"
```