Reading Data into Dictionaries

Consider CSV data of the form:

```
Alabama,10,20,30,...
Alaska,32,43,56,...
.
.
.
Wyoming,2,0,78,...
```

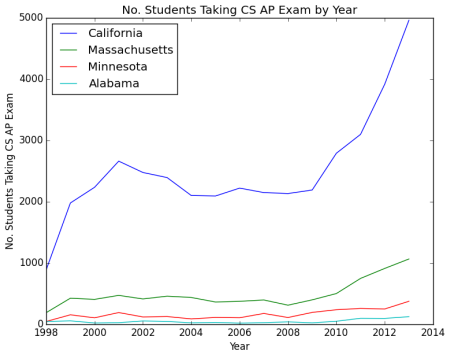
Write a function `to_data` that takes a filename and returns a dictionary `data` where each key is a state name and each value is a list of integers.

```
>>> data["Minnesota"]
[47, 156, 107, 193, 121, 128]
>>> data["Iowa"]
[15, 36, 52, 57, 62, 45]
```

```
1 import csv
2
3 def data_from_file(filename):
4     with open(filename) as fin:
5         return {state: [int(x) for x in nums]
6                 for state, *nums in csv.reader(fin)}
```

Plotting with Matplotlib

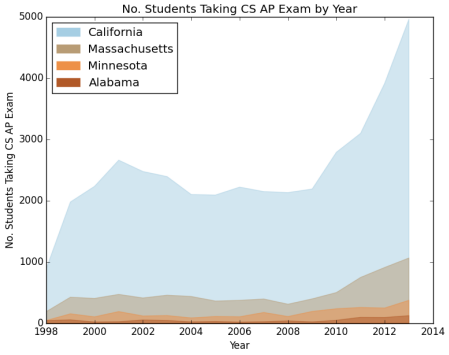
Suppose we want to plot test takers, by state and year. Here is a nice line plot with legend and axis labels.



And here is the code.

```
1  import matplotlib.pyplot as plt
2
3  def plot1(data, states, years):
4      for state in states:
5          plt.plot(years, data[state], label=state)
6      plt.legend(loc="best")
7      plt.xlabel("Year")
8      plt.ylabel("No. Students Taking CS AP Exam")
9      plt.title("No. Students Taking CS AP Exam by Year")
10     plt.savefig("out.png")
```

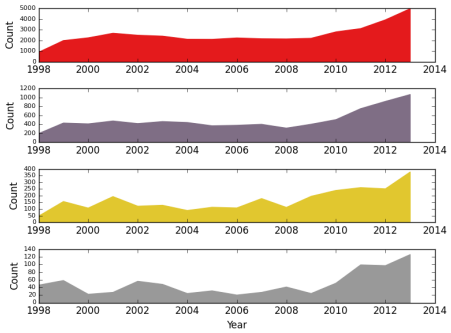
A more aesthetically pleasing plot might fill the areas under the curves in. However, this requires us to choose our own colors using colormaps as well as create and place our own legend.



Here is the code.

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3  import matplotlib.patches as mpatches
4  from itertools import count
5
6  def plot2(data, states, years):
7      colors = plt.cm.Paired(np.linspace(0,1,len(states)))
8      patches = []
9      for state,c in zip(states,colors):
10         plt.fill_between(years, data[state], color=c, alpha=0.5)
11         patches.append(mpatches.Patch(color=c, label=state))
12     plt.legend(handles=patches, loc="upper left")
13     plt.xlabel("Year")
14     plt.ylabel("No. Students Taking CS AP Exam")
15     plt.title("No. Students Taking CS AP Exam by Year")
16     plt.savefig("out2.png")
```

Suppose we wanted subplots for each of the states.



We can use the `subplot2grid` command to help out.

```
1
2  def plot3(data, states, years):
3      colors = plt.cm.Set1(np.linspace(0,1,len(states)))
4      for i, state, c in zip(count(), states, colors):
5          ax = plt.subplot2grid((len(states),1),(i,0))
6          ax.fill_between(years, data[state], color=c)
7          ax.set_ylabel("Count")
8          for tick in ax.yaxis.get_major_ticks():
9              tick.label.set_fontsize(8)
10
11     plt.tight_layout()
12     plt.xlabel("Year")
13     plt.savefig("out3.png")
```