In [6]:

```
"""
CSV reader options.
"""
import csv

def dictparse(csvfilename, keyfield, separator, quote, quotestrategy):
    """
    Reads CSV file named csvfilename, parses
    it's content and returns the data within
    the file as a dictionary of dictionaries.
    """
    table = {}
    with open(csvfilename, "rt", newline='') as csvfile:
        csvreader = csv.DictReader(csvfile,
                                   skipinitialspace=True,
                                   delimiter=separator,
                                   quotechar=quote,
                                   quoting=quotestrategy)
        for row in csvreader:
            table[row[keyfield]] = row
    return table, csvreader.fieldnames


def print_table(table, fieldnames):
    """
    Print out table, which must be a dictionary
    of dictionaries, in a nicely formatted way.
    """
    print("{:<19}".format(fieldnames[0]), end='')
    for field in fieldnames[1:]:
        print("{:>6}".format(field), end='')
    print("")
    for name, row in table.items():
        # Header column left justified
        print("{:<19}".format(name), end='')
        # Remaining columns right justified
        for field in fieldnames[1:]:
            print("{:>6}".format(row[field]), end='')
        print("", end='\n')

table, fieldnames = dictparse("hightemp.csv", 'City', ',', '"', csv.QUOTE_MINIMAL)
print(fieldnames)
print_table(table, fieldnames)

print("")
print("")

table2, fieldnames2 = dictparse("hightemp2.csv", 'City', ',', '"', csv.QUOTE_NONNUMERIC)
print_table(table2, fieldnames2)

print("")
print("")

table3, fieldnames3 = dictparse("hightemp3.csv", 'City', ' ', "'", csv.QUOTE_NONNUMERIC)
print_table(table3, fieldnames3)
```

```
['City', 'Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']
City                 Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
Houston               62    65    72    78    84    90    92    93    88    81    71    63
Baghdad               61    66    75    86    97   108   111   111   104    91    75    64
Moscow                21    25    36    50    64    72    73    70    59    46    34    25
San Francisco         57    60    62    63    64    66    67    68    70    69    63    57
London                43    45    50    55    63    68    72    70    66    57    50    45
Chicago               32    36    46    59    70    81    84    82    75    63    48    36
Sydney                79    79    77    73    68    64    63    64    68    72    75    79
Paris                 45    46    54    61    68    73    77    77    70    61    52    46
Tokyo                 46    48    54    63    70    75    82    84    79    68    59    52
Shanghai              46    48    55    66    75    81    90    90    81    72    63    52


City                 Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
Houston, USA        62.0  65.0  72.0  78.0  84.0  90.0  92.0  93.0  88.0  81.0  71.0  63.0
Baghdad, Iraq       61.0  66.0  75.0  86.0  97.0 108.0 111.0 111.0 104.0  91.0  75.0  64.0
Moscow, Russia      21.0  25.0  36.0  50.0  64.0  72.0  73.0  70.0  59.0  46.0  34.0  25.0
San Francisco, USA  57.0  60.0  62.0  63.0  64.0  66.0  67.0  68.0  70.0  69.0  63.0  57.0
London, England     43.0  45.0  50.0  55.0  63.0  68.0  72.0  70.0  66.0  57.0  50.0  45.0
Chicago, USA        32.0  36.0  46.0  59.0  70.0  81.0  84.0  82.0  75.0  63.0  48.0  36.0
Sydney, Australia   79.0  79.0  77.0  73.0  68.0  64.0  63.0  64.0  68.0  72.0  75.0  79.0
Paris, France       45.0  46.0  54.0  61.0  68.0  73.0  77.0  77.0  70.0  61.0  52.0  46.0
Tokyo, Japan        46.0  48.0  54.0  63.0  70.0  75.0  82.0  84.0  79.0  68.0  59.0  52.0
Shanghai, China     46.0  48.0  55.0  66.0  75.0  81.0  90.0  90.0  81.0  72.0  63.0  52.0


City                 Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
Houston, USA        62.0  65.0  72.0  78.0  84.0  90.0  92.0  93.0  88.0  81.0  71.0  63.0
Baghdad, Iraq       61.0  66.0  75.0  86.0  97.0 108.0 111.0 111.0 104.0  91.0  75.0  64.0
Moscow, Russia      21.0  25.0  36.0  50.0  64.0  72.0  73.0  70.0  59.0  46.0  34.0  25.0
San Francisco, USA  57.0  60.0  62.0  63.0  64.0  66.0  67.0  68.0  70.0  69.0  63.0  57.0
London, England     43.0  45.0  50.0  55.0  63.0  68.0  72.0  70.0  66.0  57.0  50.0  45.0
Chicago, USA        32.0  36.0  46.0  59.0  70.0  81.0  84.0  82.0  75.0  63.0  48.0  36.0
Sydney, Australia   79.0  79.0  77.0  73.0  68.0  64.0  63.0  64.0  68.0  72.0  75.0  79.0
Paris, France       45.0  46.0  54.0  61.0  68.0  73.0  77.0  77.0  70.0  61.0  52.0  46.0
Tokyo, Japan        46.0  48.0  54.0  63.0  70.0  75.0  82.0  84.0  79.0  68.0  59.0  52.0
Shanghai, China     46.0  48.0  55.0  66.0  75.0  81.0  90.0  90.0  81.0  72.0  63.0  52.0
```

In [4]:

```
"""
Examples code for experimenting with options to the csv.read() and csv.write() methods
"""
import csv

# Function that prints 2D table to console

def print_table(table):
    """
    Echo a nested list to the console
    """
    for row in table:
        print(row)

# Options for reading a CSV file

def read_csv_file(file_name, file_delimeter):
    """
    Given a CSV file path and a delimiter as strings,
    read the data into a 2D table and return the table
    """
    with open(file_name, newline='') as csv_file: # don't need to explicitly close the file no
        csv_table = []
        csv_reader = csv.reader(csv_file, delimiter=file_delimeter)
        for row in csv_reader:
            csv_table.append(row)
    return csv_table


def csv_delimiter_examples():
    """
    Run some example of reading CSV files using different delimiter options
```

```
    """
    number_table = read_csv_file("number_table.csv", " ")
    print_table(number_table)
    print()
    name_table = read_csv_file("name_table.csv", ",")
    print_table(name_table)


csv_delimiter_examples()

# Options for writing a CSV file

def write_csv_file(csv_table, file_name, file_delimiter, quoting_value):
    """
    Given a nested list csv_table, write the data into a
    CSV file with the name file_name
    """

    with open(file_name, 'w', newline='') as csv_file:
        csv_writer = csv.writer(csv_file, delimiter=file_delimiter, quoting=quoting_value)
        for row in csv_table:
            csv_writer.writerow(row)

def csv_quoting_examples():
    """
    Run some example of writing 2D tables as CSV files using various quoting options
    """
    name_table = read_csv_file("name_table.csv", ",")
    name_table.append([1, 2, 3])
    write_csv_file(name_table, "name_table_minimal.csv", ",", csv.QUOTE_MINIMAL)
    write_csv_file(name_table, "name_table_all.csv", ",", csv.QUOTE_ALL)
    write_csv_file(name_table, "name_table_nonnumeric.csv", ",", csv.QUOTE_NONNUMERIC)
    #write_csv_file(name_table, "name_table_none.csv", ",", csv.QUOTE_NONE)
    # no escapechar is set for lots of quotes
#csv_quoting_examples()
```

```
['1', '2', '3']
['4', '5', '6']
['7', '8', '9']
['10', '11', '12']

['Joe', 'Scott', ' Stephen']
["Joe'S", " Scott's", "  Stephen's"]
["Joe's", "Scott's", ' "Stephen\'s"']
```

In [8]:

```
"""
Week 3 practice project template for Python Data Analysis
Reading and writing CSV files using lists
"""
import csv

###########################################################
# Part 1 - Week 3

def print_table(table):
    """
    Echo a nested listto the console
    """
    for row in table:
        print(row)

def read_csv_file(file_name):
    """
    Given a CSV file, read the data into a nested list
    Input: String corresponding to comma-separated  CSV file
    Output: Lists of lists consisting of the fields in the CSV file
    """

    return []

def write_csv_file(csv_table, file_name):
```

```
    """
    Input: Nested list csv_table and a string file_name
    Action: Write fields in csv_table into a comma-separated CSV file with the name file_name
    """
    pass

def test_part1_code():
    """
    Run examples that test the functions for part 1
    """
    # Simple test for reader
    test_table = read_csv_file("test_case.csv")  # create a small CSV for this test
    print_table(test_table)
    print()

    # Test the writer
    cancer_risk_table = read_csv_file("cancer_risk05_v4_county.csv")
    write_csv_file(cancer_risk_table, "cancer_risk05_v4_county_copy.csv")
    cancer_risk_copy = read_csv_file("cancer_risk05_v4_county_copy.csv")

    # Test whether two tables are the same

test_part1_code()
```

In [ ]: