```
Problem Set 2.1

import pandas
import pandasql


def num_rainy_days(filename):
    weather_data = pandas.read_csv(filename)

    q = """
    SELECT COUNT(*)
    FROM weather_data
    WHERE cast(rain as integer) = 1;
    """

    #Execute your SQL command against the pandas frame
    rainy_days = pandasql.sqldf(q.lower(), locals())
    return rainy_days


Problem Set 2.2

import pandas
import pandasql


def max_temp_aggregate_by_fog(filename):
        weather_data = pandas.read_csv(filename)

    q = """
    SELECT fog, MAX(cast (maxtempi as integer))
    FROM weather_data
    GROUP BY fog;
    """

    #Execute your SQL command against the pandas frame
    foggy_days = pandasql.sqldf(q.lower(), locals())
    return foggy_days

Problem Set 2.3

import pandas
import pandasql

def avg_weekend_temperature(filename):
    weather_data = pandas.read_csv(filename)

    q = """
    SELECT avg(cast (meantempi as integer))
    FROM weather_data
    WHERE cast (strftime('%w', date) as integer) IN (6,0);
    """

    #Execute your SQL command against the pandas frame
    mean_temp_weekends = pandasql.sqldf(q.lower(), locals())
    return mean_temp_weekends
```

```
Problem Set 2.4

import pandas
import pandasql

def avg_min_temperature(filename):
    weather_data = pandas.read_csv(filename)

    q = """
    SELECT avg(cast (mintempi as integer))
    FROM weather_data
    WHERE cast(rain as integer) = 1
    AND cast(mintempi as integer) > 55;
    """

    #Execute your SQL command against the pandas frame
    avg_min_temp_rainy = pandasql.sqldf(q.lower(), locals())
    return avg_min_temp_rainy



Problem Set 2.5

import csv

def fix_turnstile_data(filenames):
    for name in filenames:
        f_in = open(name, 'r')
        f_out = open('updated_'+name, 'w')
        reader_in = csv.reader(f_in, delimiter=',')
        writer_out = csv.writer(f_out, delimiter=',')
        for x in reader_in:
            c1 = x[0]
            c2 = x[1]
            c3 = x[2]

            for k in range(len(x)/5):
                        out = [c1, c2, c3,
                               x[k*5 + 3],
                               x[k*5 + 4],
                               x[k*5 + 5],
                               x[k*5 + 6],
                               x[k*5 + 7]]
                        writer_out.writerow(out)



      Problem Set 2.6

      def create_master_turnstile_file(filenames, output_file):
          with open(output_file, 'w') as master_file:
             master_file.write('C/A,UNIT,SCP,DATEn,TIMEn,DESCn,ENTRIESn,EXITSn\n')
             for filename in filenames:
                  f_in = open(filename, 'r')
                  f_in.next()
                  for line in f_in:
                      master_file.write(line)
                  f_in.close


```

```
123.        Problem Set 2.7
124.
125.        import pandas
126.
127.        def filter_by_regular(filename):
128.
129.            turnstile_data = pandas.read_csv(filename)
130.            turnstile_data = turnstile_data[turnstile_data['DESCn'] == 'REGULAR']
131.            # your code here
132.            # more of your code here
133.
134.            return turnstile_data
135.
136.
137.        Problem Set 2.8
138.
139.        import pandas
140.
141.        def get_hourly_entries(df):
142.
143.            df['ENTRIESn_hourly'] = (df['ENTRIESn'] -  df['ENTRIESn'].shift(1)).fillna(1
    )
144.            return df
145.
146.
147.        Problem Set 2.9
148.
149.        import pandas
150.
151.        def get_hourly_exits(df):
152.
153.            df['EXITSn_hourly'] = (df['EXITSn'] - df['EXITSn'].shift(1)).fillna(0)
154.            return df
155.
156.
157.        Problem Set 2.10
158.
159.        import pandas
160.
161.        def time_to_hour(time):
162.
163.            hour = int(time[0:2])# your code here
164.            return hour
165.
166.
167.        Problem Set 2.11
168.
169.        import datetime
170.
171.        def reformat_subway_dates(date):
172.
173.            date_formatted = datetime.datetime.strptime(date, "%m-%d-%y").strftime("%Y-
    %m-%d")# your code here
174.            return date_formatted
```