# 732A54 Big Data Analytics

# Lab 1; pyspark

## Group G6:

Hoda Fakharzadehjahromy(hodfa840)
Seyda Aqsa Iftekhar(syeif776)

1) What are the lowest and highest temperatures measured each year for the period 1950-2014. Provide the lists sorted in the descending order with respect to the maximum temperature. In this exercise you will use the *temperature-readings.csv* file.

The output should at least contain the following information (You can also include a Station column so that you may find multiple stations that record the highest (lowest) temperature.):

Year, temperature

**Code:**

```
from pyspark import SparkContext
import os
import sys

os.environ['PYSPARK_PYTHON'] = sys.executable
os.environ['PYSPARK_DRIVER_PYTHON'] = sys.executable
sc = SparkContext(appName="ex1" )
# MAP
temperature1 = sc.textFile("data/temperature-readings.csv")
lines = temperature1.map(lambda x: x.split(";"))
#print(lines.take(3))
# filter
year_temperature = lines.map(lambda x: (x[1][0:4],float(x[3]) ))
year_temperature = year_temperature.filter(lambda x: (int(x[0] )>= 1950 and
int(x[0]) <= 2014))
 #reduce
max_temperature = year_temperature.reduceByKey(lambda a,b: a if a >= b else b  )
max_sorted_temperature = max_temperature.sortBy(ascending=False,keyfunc=lambda
k:k[1])

min_temperature = year_temperature.reduceByKey(lambda a,b: a if a < b else b  )

min_sorted_temperature = min_temperature.sortBy(ascending=False,keyfunc=lambda
k:k[1])

#print(min_sorted_temperature.take( 10))
#print("--------------------------------------------------")
#print(max_sorted_temperature.take( 10))
min_sorted_temperature.saveAsTextFile("res/ex1_min")
max_sorted_temperature.saveAsTextFile("res/ex1_max")
```

**Output:**

Max:

```
[('1992', 35.4), ('1994', 34.7), ('2014', 34.4), ('2010', 34.4), ('1989', 33.9),
('1982', 33.8), ('1968', 33.7), ('1966', 33.5), ('1983', 33.3), ('2002', 33.3),
('1986', 33.2), ('1970', 33.2), ('1956', 33.0), ('2000', 33.0), ('1959', 32.8),
('2006', 32.7), ('1991', 32.7), ('1988', 32.6), ('2011', 32.5)]
```

Min:

```
[('1999', -49.0), ('1978', -47.7), ('1987', -47.3), ('1967', -45.4), ('1956', -
45.0), ('1980', -45.0), ('1971', -44.3), ('1986', -44.2), ('2001', -44.0),
('1965', -44.0), ('1981', -44.0), ('1979', -44.0), ('1959', -43.6), ('1985', -
43.4), ('1958', -43.0), ('1998', -42.7), ('2012', -42.7), ('2014', -42.5),
('1977', -42.5)]
```

2) Count the number of readings for each month in the period of 1950-2014 which are higher than 10 degrees. Repeat the exercise, this time taking only distinct readings from each station. That is, if a station reported a reading above 10 degrees in some month, then it appears only once in the count for that month.

In this exercise you will use the *temperature-readings.csv* file.

The output should contain the following information:

Year, month, count

**Code: 2a)**

```
from pyspark import SparkContext
from operator import add
import os
import sys

os.environ['PYSPARK_PYTHON'] = sys.executable
os.environ['PYSPARK_DRIVER_PYTHON'] = sys.executable

sc = SparkContext(appName="ex2")

temperature= sc.textFile("data/temperature-readings.csv")

lines = temperature.map(lambda line: line.split(";"))
```

```
yearMontemp = lines.map(lambda x: ((x[1][0:4],x[1][5:7]), (x[0], float(x[3]))))
#print(year_month.take(5))
#print("-----------------------------------------")
#filtering
yearMontemp = yearMontemp.filter(lambda x: int(x[0][0])>=1950 and
int(x[0][0])<=2014 and x[1][1]>10)
#print(year_month.take(5))
#print("-----------------------------------------")
count_yearMontemp= yearMontemp.map(lambda x:(x[0],1)).reduceByKey(add).\
sortByKey(ascending=False)

count_yearMontemp.saveAsTextFile("output2/ex2a")
```

**Output 2a):**

```
(('2003', '09'), 70459)
(('2003', '08'), 108501)
(('2003', '07'), 128133)
(('2003', '06'), 99693)
(('2003', '05'), 48264)
(('2003', '04'), 11786)
(('2003', '03'), 3581)
(('2003', '02'), 3)
(('2002', '10'), 4939)
(('2002', '09'), 65928)
```

**Code 2b) distinct reading for each station:**

```
from pyspark import SparkContext
from operator import add
import os
import sys

os.environ['PYSPARK_PYTHON'] = sys.executable
os.environ['PYSPARK_DRIVER_PYTHON'] = sys.executable
```

```
sc = SparkContext(appName="ex2")

temperature= sc.textFile("./data/temperature-readings.csv")

lines = temperature.map(lambda line: line.split(";"))

year_month = lines.map(lambda x: ((x[1][0:4],x[1][5:7]), (x[0], float(x[3]))))
print(year_month.take(20))
print('--------------------------------------------------------------------------
-----')
#filtering
year_month = year_month.filter(lambda x: int(x[0][0])>=1950 and
int(x[0][0])<=2014 and x[1][1]>10)
print(year_month.take(20))

print('--------------------------------------------------------------------------
-----')

#selecting disctinct key
year_month_unique = year_month. map(lambda x: (x[0], (x[1][0], 1)))
print(year_month_unique.take(20))


year_month_unique = year_month. map(lambda x: (x[0], (x[1][0], 1))).distinct()
print(year_month_unique.take(20))
station_month_counts = year_month_unique. map(lambda x: (x[0],
1)).reduceByKey(add).sortByKey(ascending=False)
station_month_counts.saveAsTextFile("./res/ex2b_updated")
print(station_month_counts.collect())
```

**Output:**

```
(('1951', '02'), 1)
(('1950', '12'), 1)
(('1950', '11'), 2)
(('1950', '10'), 46)
(('1950', '09'), 50)
(('1950', '08'), 49)
(('1950', '07'), 49)
(('1950', '06'), 47)
(('1950', '05'), 46)
(('1950', '04'), 36)
(('1950', '03'), 26)
```

3) Find the average monthly temperature for each available station in Sweden. Your result should include average temperature for each station for each month in the period of 1960-2014. Bear in mind that not every station has the readings for each month in this timeframe.

In this exercise you will use the *temperature-readings.csv* file.

The output should contain the following information:

Year, month, station number, average monthly temperature

```
from pyspark import SparkContext
import os
import sys

os.environ['PYSPARK_PYTHON'] = sys.executable
os.environ['PYSPARK_DRIVER_PYTHON'] = sys.executable

sc = SparkContext(appName='ex')

temperature= sc.textFile("data/temperature-readings.csv").cache()

lines = temperature.map(lambda x: x.split(";"))

#temperature = temperature.sample(False,0.01)

temperature=lines.map(lambda x: ((int(x[1][0:4]) ,int(x[1][5:7]),int(x[1][8:10] ) ),  ( x[0]
, float(x[3]))))

# filter
select_temperature = temperature.filter(lambda x: x[0][0]>= 1960 and x[0][0]<=2014 )

select_temperature = select_temperature.map(lambda x:
((x[0][0],x[0][1],x[0][2],x[1][0]), (x[1][1],x[1][1])))
# #writing a function to calculate max and min together
#
def min_and_max(a,b):

    minimum = a[0] if a[0]<b[0] else b[0]
    maximum = b[1] if a[1]<b[1] else a[1]
    return (minimum,maximum)


select_temperature = select_temperature.reduceByKey(min_and_max)
```

```
##sum the daily values for each month
 select_temperature1 = select_temperature.map(lambda x:
((x[0][0],x[0][1],x[0][3]),(sum(x[1]),2)))

ave = select_temperature1.reduceByKey(lambda a, b: (a[0]+b[0],
a[1]+b[1])).map(lambda x: (x[0], round(x[1][0]/x[1][1],
ndigits=3))).sortByKey(ascending=False)

ave.saveAsTextFile("assignment3_updated")
print(ave.collect())
```

## Part of the Output:

```
 ((1969, 4, '75240'), 5.065), ((1969, 4, '75140'), 5.525), ((1969, 4, '75040'), 5.032),
((1969, 4, '74630'), 4.525), ((1969, 4, '74530'), 4.028), ((1969, 4, '74490'), 3.753),
((1969, 4, '74480'), 5.465), ((1969, 4, '74470'), 5.165), ((1969, 4, '74460'), 3.328),
((1969, 4, '74440'), 3.612), ((1969, 4, '74420'), 3.765), ((1969, 4, '74390'), 3.713),
((1969, 4, '74350'), 3.627), ((1969, 4, '74240'), 4.377), ((1969, 4, '74180'), 3.457),
((1969, 4, '73660'), 3.622), ((1969, 4, '73470'), 4.01), ((1969, 4, '73430'), 3.367), ((1969,
4, '73250'), 3.795), ((1969, 4, '73090'), 4.62), ((1969, 4, '72450'), 4.333), ((1969, 4,
'72400'), 4.943), ((1969, 4, '72300'), 4.795), ((1969, 4, '72290'), 5.017), ((1969, 4,
'72150'), 4.875), ((1969, 4, '72120'), 4.693), ((1969, 4, '72080'), 4.888), ((1969, 4,
'72070'), 5.735), ((1969, 4, '71470'), 5.218), ((1969, 4, '71430'), 5.313), ((1969, 4,
'71420'), 5.893), ((1969, 4, '71380'), 4.557), ((1969, 4, '68550'), 3.218), ((1969, 4,
'66600'), 3.182), ((1969, 4, '66470'), 5.638), ((1969, 4, '66410'), 4.335), ((1969, 4,
'66120'), 3.263), ((1969, 4, '65450'), 4.897), ((1969, 4, '65160'), 4.613), ((1969, 4,
'65130'), 5.487), ((1969, 4, '64520'), 4.58), ((1969, 4, '64400'), 4.363), ((1969, 4,
'64130'), 4.987), ((1969, 4, '64030'), 5.865), ((1969, 4, '64020'), 4.138), ((1969, 4,
'63630'), 4.832), ((1969, 4, '63530'), 4.393), ((1969, 4, '63500'), 4.812), ((1969, 4,
'63450'), 4.375), ((1969, 4, '63440'), 4.79), ((1969, 4, '63340'), 5.315), ((1969, 4,
'63230'), 4.918), ((1969, 4, '63220'), 4.397), ((1969, 4, '63050'), 4.57), ((1969, 4,
'62560'), 5.28), ((1969, 4, '62410'), 4.918), ((1969, 4, '62190'), 4.537), ((1969, 4,
'62180'), 4.922), ((1969, 4, '62080'), 6.68), ((1969, 4, '62030'), 5.938), ((1969, 4,
'55570'), 3.273), ((1969, 4, '54550'), 4.748), ((1969, 4, '54390'), 5.522), ((1969, 4,
'54300'), 5.403), ((1969, 4, '54230'), 4.052), ((1969, 4, '53650'), 5.44), ((1969, 4,
'53560'), 5.568), ((1969, 4, '53540'), 6.233), ((1969, 4, '53470'), 6.108), ((1969, 4,
'53430'), 5.843), ((1969, 4, '53360'), 6.03), ((1969, 4, '53260'), 4.392), ((1969, 4,
'53250'), 5.63), ((1969, 4, '53200'), 4.093), ((1969, 4, '52550'), 6.018), ((1969, 4,
'52230'), 4.722), ((1969, 4, '192830'), -5.253), ((1969, 4, '192710'), -4.08), ((1969, 4,
'191900'), -5.92), ((1969, 4, '189780'), -4.877), ((1969, 4, '188830'), -4.685), ((1969, 4,
'188800'), -3.218), ((1969, 4, '183980'), -2.537), ((1969, 4, '183760'), -3.212), ((1969, 4,
'182930'), -3.923), ((1969, 4, '182720'), -1.622), ((1969, 4, '181900'), -2.053), ((1969, 4,
'180940'), -5.477), ((1969, 4, '180750'), -2.433), ((1969, 4, '179950'), -5.763), ((1969, 4,
'178740'), -1.913), ((1969, 4, '173810'), -0.668), ((1969, 4, '172790'), -1.13), ((1969, 4,
'171920'), -2.575), ((1969, 4, '171810'), -0.342), ((1969, 4, '170670'), 0.092), ((1969, 4,
'169880'), -1.152), ((1969, 4, '167980'), -1.878), ((1969, 4, '167860'), -3.747), ((1969, 4,
'167710'), -2.167), ((1969, 4, '166870'), -3.23), ((1969, 4, '166810'), -1.553), ((1969, 4,
'163950'), -1.493), ((1969, 4, '163690'), 0.028), ((1969, 4, '162980'), 0.54), ((1969, 4,
'162970'), 0.445), ((1969, 4, '162880'), 0.782), ((1969, 4, '162860'), -1.048), ((1969, 4,
'161940'), 0.038), ((1969, 4, '161790'), 0.5), ((1969, 4, '161770'), -0.273), ((1969, 4,
'160960'), 0.143), ((1969, 4, '160790'), -0.67), ((1969, 4, '160740'), -0.977), ((1969, 4,
'159970'), -2.117), ((1969, 4, '159770'), -0.832), ((1969, 4, '158850'), -2.028), ((1969, 4,
'158750'), -0.062), ((1969, 4, '157930'), -0.993), ((1969, 4, '157720'), -0.083), ((1969, 4,
'156770'), -0.107), ((1969, 4, '156730'), 0.257), ((1969, 4, '155940'), -1.028), ((1969, 4,
'155910'), -0.72), ((1969, 4, '154720'), -1.047), ((1969, 4, '151290'), -0.735), ((1969, 4,
'151220'), 0.895), ((1969, 4, '149570'), -0.827), ((1969, 4, '149160'), -0.69), ((1969, 4,
'148350'), -0.01), ((1969, 4, '148050'), -0.083), ((1969, 4, '147570'), -0.798), ((1969, 4,
'147460'), -0.485), ((1969, 4, '147100'), 0.742), ((1969, 4, '146380'), -0.27), ((1969, 4,
'145340'), 0.18), ((1969, 4, '145280'), 0.74), ((1969, 4, '144300'), 0.758), ((1969, 4,
'144160'), 0.458), ((1969, 4, '143440'), 1.037), ((1969, 4, '142030'), 1.005), ((1969, 4,
'140500'), 2.443), ((1969, 4, '140490'), 1.103), ((1969, 4, '140480'), 0.662), ((1969, 4,
'140360'), -1.21), ((1969, 4, '140200'), 0.55), ((1969, 4, '139540'), 1.31), ((1969, 4,
'139340'), 1.252), ((1969, 4, '139200'), 2.002), ((1969, 4, '139110'), 0.908), ((1969, 4,
'138270'), 2.363), ((1969, 4, '138240'), 1.228), ((1969, 4, '137560'), 0.903), ((1969, 4,
'137110'), 3.012), ((1969, 4, '137100'), 3.993), ((1969, 4, '137080'), 3.107), ((1969, 4,
'137030'), 2.133), ((1969, 4, '136420'), 2.123), ((1969, 4, '136360'), 2.343), ((1969, 4,
'136090'), 2.538), ((1969, 4, '136010'), 2.625), ((1969, 4, '135440'), 1.047), ((1969, 4,
'135380'), 0.933), ((1969, 4, '134520'), 0.788), ((1969, 4, '134150'), 1.812), ((1969, 4,
'134110'), 1.578), ((1969, 4, '134100'), 2.418), ((1969, 4, '134090'), 2.373), ((1969, 4,
'133470'), 1.022), ((1969, 4, '133230'), 1.243), ((1969, 4, '133200'), 2.17), ((1969, 4,
'133050'), 0.595), ((1969, 4, '132240'), 1.583), ((1969, 4, '132180'), -0.303), ((1969, 4,
'128590'), 2.145), ((1969, 4, '128510'), 2.528), ((1969, 4, '127620'), 1.518), ((1969, 4,
'127380'), 2.647), ((1969, 4, '127310'), 2.002), ((1969, 4, '127240'), 3.435), ((1969, 4,
'127220'), 2.798), ((1969, 4, '127140'), 1.32), ((1969, 4, '126460'), 2.4), ((1969, 4,
'126300'), 3.272), ((1969, 4, '125450'), 1.042), ((1969, 4, '124390'), 0.867), ((1969, 4,
'124110'), 2.018), ((1969, 4, '124020'), 1.653), ((1969, 4, '123480'), 0.462), ((1969, 4,
'123250'), 2.18), ((1969, 4, '123070'), 0.722), ((1969, 4, '122610'), -0.635), ((1969, 4,
'122360'), -0.803), ((1969, 4, '122330'), 0.595), ((1969, 4, '117440'), 3.93), ((1969, 4,
'117330'), 1.557), ((1969, 4, '117160'), 2.167), ((1969, 4, '116480'), 3.097), ((1969, 4,
'116430'), 3.36), ((1969, 4, '116240'), 3.238), ((1969, 4, '115230'), 2.695), ((1969, 4,
'114510'), 2.133), ((1969, 4, '114330'), 0.495), ((1969, 4, '114270'), 2.15), ((1969, 4,
'114010'), 3.177), ((1969, 4, '113410'), 0.955), ((1969, 4, '113100'), 1.783), ((1969, 4,
'112170'), 1.987), ((1969, 4, '112080'), 1.178), ((1969, 4, '108320'), 2.507), ((1969, 4,
'108300'), 2.49), ((1969, 4, '108110'), 4.427), ((1969, 4, '108100'), 3.393), ((1969, 4,
'107440'), 2.27), ((1969, 4, '107400'), 3.665), ((1969, 4, '107330'), 3.473), ((1969, 4,
'107270'), 3.338), ((1969, 4, '107180'), 3.71), ((1969, 4, '107120'), 3.655), ((1969, 4,
'106580'), 2.772), ((1969, 4, '106270'), 3.245), ((1969, 4, '106100'), 3.207), ((1969, 4,
'105450'), 3.355), ((1969, 4, '105370'), 3.012), ((1969, 4, '105360'), 3.365), ((1969, 4,
'105260'), 2.88), ((1969, 4, '105230'), 3.118), ((1969, 4, '105200'), 2.672), ((1969, 4,
'104530'), 2.64),
```

4) Provide a list of stations with their associated maximum measured temperatures and maximum measured daily precipitation. Show only those stations where the maximum temperature is between 25 and 30 degrees andmaximumdaily precipitation is between 100 mm and 200mm.

In this exercise you will use the *temperature-readings.csv* and *precipitation-readings.csv* files.

The output should contain the following information:

Station number, maximum measured temperature, maximum daily precipitation

**Code:**

```
 from pyspark import SparkContext
import os
import sys
from operator import add

os.environ['PYSPARK_PYTHON'] = sys.executable
os.environ['PYSPARK_DRIVER_PYTHON'] = sys.executable
sc = SparkContext(appName='ex')

temperature = sc.textFile("data/temperature-readings.csv").cache()
temperature = temperature.map(lambda a: a.split(';'))
temperature = temperature.map(lambda x: (x[0],float(x[3])))
temperature = temperature.reduceByKey(max)
temperature = temperature.filter(lambda x: x[1] >= 25 and x[1] <= 30 )

#print(temperature.collect())


precipitation = sc.textFile("data/precipitation-readings.csv").cache()
precipitation = precipitation.map(lambda x: x.split(';') )

precipitation = precipitation.map(lambda x: ((x[0],x[1]),float(x[3])))
#calculating daily precipitation
precipitation1 = precipitation.reduceByKey(add)
precipitation1=precipitation1.map(lambda x: (x[0][0],x[1]))
#finding daily precipitation
precipitation1 = precipitation1.reduceByKey(max)
precipitation1 = precipitation1.filter(lambda x: x[1] >= 100 and x[1] <= 200)

#print(precipitation1.collect())


# join the RDDs
stations = temperature.join(precipitation1)
# #the results
stations.saveAsTextFile("assignment4_updated")
print(stations.collect())
```

**Output:**

```
[]
```

5) Calculate the average monthly precipitation for the Östergotland region (list of stations is provided in the separate file) for the period 1993-2016. In order to dothis, you will firstneed to calculate the total monthly precipitation for each station before calculating the monthly average (by averaging over stations).
In this exercise you will use the *precipitation-readings.csv* and *stations-Ostergotland.csv* files

The output should contain the following information:

Year,month, averagemonthly precipitation

**Code:**

```
from pyspark import SparkContext
import os
import sys
from operator import add

os.environ['PYSPARK_PYTHON'] = sys.executable
os.environ['PYSPARK_DRIVER_PYTHON'] = sys.executable

# create the spark application
sc = SparkContext(appName="exe")


ostergotland = sc.textFile('data/stations-Ostergotland.csv')
stations = ostergotland.map(lambda line: line.split(";")[0]).collect()
stations = sc.broadcast(stations)
#print(stations.value)
precipitation = sc.textFile("data/precipitation-readings.csv").cache()
precipitation = precipitation.map(lambda x: x.split(';') )
precipitation1 = precipitation.filter(lambda x: x[0] in stations.value)

precipitation1 = precipitation1.filter(lambda x: int(x[1][0:4]) >= 1993 and int(x[1][0:4])
<= 2016)
precipitation1 = precipitation1.map(lambda x: ((x[0],x[1][0:7]),float(x[3])))

#calculate the monthly precipitation for each station
precipitation1 = precipitation1.reduceByKey(add)

tmp = precipitation1.map(lambda x: (x[0][1],  (x[1], 1)))
tmp = tmp.reduceByKey(lambda x,y: (x[0]+y[0],x[1]+y[1]))
#print(tmp.collect())
```

```
precip_ave = tmp.map(lambda x: (x[0],x[1][0]/x[1][1]))
# #precip_ave.saveAsTextFile("assignment5_updated")
print(precip_ave.collect())
```

**Output:**

```
 [('2014-09', 48.45), ('2009-05', 54.167), ('2009-08', 61.567), ('2016-04', 26.9), ('1998-
05', 38.367), ('2002-02', 47.583), ('2016-02', 21.562), ('1997-03', 9.55), ('1999-01',
61.933), ('2009-03', 34.483), ('2011-12', 42.133), ('2015-09', 101.3), ('2015-10', 2.263),
('2006-12', 29.733), ('2008-11', 46.75), ('1994-06', 45.1), ('1999-05', 27.383), ('2004-01',
26.4), ('2004-09', 37.2), ('2006-08', 148.083), ('1995-02', 31.8), ('1996-02', 15.733),
('1998-10', 53.417), ('2007-06', 108.95), ('2013-01', 21.525), ('2014-08', 90.812), ('2008-
02', 28.25), ('2004-11', 54.133), ('2012-02', 28.667), ('2000-02', 24.05), ('2008-12',
43.483), ('2003-07', 113.467), ('1997-01', 5.783), ('2010-06', 48.65), ('2003-08', 55.283),
('1997-09', 43.567), ('2001-03', 30.267), ('2004-02', 27.35), ('2007-08', 54.167), ('2006-
06', 31.133), ('2006-07', 28.983), ('1994-09', 94.6), ('2008-10', 59.567), ('1996-06',
51.667), ('2010-03', 23.883), ('2016-03', 19.963), ('2010-09', 43.083), ('1995-03', 34.4),
('2010-08', 108.05), ('1997-11', 64.45), ('2007-04', 21.25), ('2015-11', 63.888), ('2000-
05', 25.317), ('2007-10', 28.117), ('2000-09', 27.517), ('2004-10', 78.183), ('2005-07',
104.35), ('2012-12', 66.933), ('2005-10', 38.05), ('2005-08', 76.967), ('2016-05', 29.25),
('2010-05', 67.167), ('2012-09', 72.75), ('1998-03', 33.9), ('1994-12', 50.9), ('2015-12',
28.925), ('2002-05', 72.133), ('2008-08', 138.517), ('2010-01', 35.983), ('2002-11',
53.683), ('2015-02', 24.825), ('1995-01', 26.0), ('2005-03', 23.467), ('2013-02', 25.525),
('2011-02', 24.517), ('1993-10', 43.2), ('1993-09', 40.6), ('2001-04', 46.067), ('2002-06',
98.783), ('2013-04', 38.288), ('2015-01', 59.113), ('2013-10', 53.875), ('2014-07',
22.988), ('2000-06', 62.017), ('1999-02', 25.15), ('2007-12', 54.7), ('2008-03', 42.2),
('2000-11', 108.117), ('2007-03', 40.517), ('1997-02', 35.433), ('2011-01', 35.133),
('1995-05', 26.0), ('2007-01', 68.633), ('1998-11', 28.967), ('2006-01', 17.683), ('2008-
06', 42.933), ('2009-10', 56.833), ('1995-11', 31.6), ('2001-01', 36.417), ('2001-08',
69.967), ('2003-09', 8.883), ('2007-11', 50.683), ('2005-02', 33.5), ('2012-01', 43.55),
('2000-08', 54.85), ('1999-04', 54.55), ('2009-06', 49.767), ('2008-01', 44.967), ('2008-
07', 85.2), ('2003-11', 54.45), ('1994-02', 22.5), ('1994-05', 25.1), ('1993-07', 95.4),
('1995-07', 43.6), ('1999-12', 66.0), ('2002-08', 8.25), ('2007-05', 40.517), ('2000-12',
63.267), ('2001-02', 36.767), ('2016-06', 47.663), ('2011-09', 52.567), ('2015-08',
26.987), ('1998-06', 88.883), ('2007-09', 61.867), ('2015-05', 93.225), ('2012-04',
62.783), ('1996-04', 8.1), ('2000-04', 36.167), ('2006-11', 71.717), ('2009-02', 24.783),
('1994-08', 58.8), ('1999-11', 18.45), ('2008-04', 20.25), ('2011-05', 37.85), ('2012-06',
132.2), ('2012-10', 65.583), ('2013-08', 54.075), ('2015-04', 15.337), ('1997-04', 25.95),
('2014-12', 35.463), ('2012-05', 22.967), ('1997-06', 86.983), ('2006-04', 44.367), ('2007-
02', 33.067), ('1999-08', 54.8), ('1995-06', 97.2), ('2012-08', 68.817), ('1998-07',
85.167), ('2010-07', 92.4), ('2013-11', 46.375), ('2003-12', 52.117), ('2001-11', 26.383),
('2002-07', 80.517), ('1995-10', 20.867), ('2010-04', 23.783), ('2005-11', 32.6), ('1994-
04', 23.1), ('2014-06', 75.138), ('2001-10', 60.483), ('2009-01', 15.883), ('2014-05',
58.0), ('2016-07', 0.0), ('2005-01', 18.05), ('2013-03', 7.387), ('2001-07', 40.283), ('2002-
01', 55.0), ('2003-06', 66.667), ('2004-06', 56.85), ('2006-09', 19.267), ('2011-06',
88.35), ('2014-03', 36.563), ('2013-09', 26.188), ('1993-06', 56.5), ('1996-01', 10.417),
('1999-09', 55.517), ('1997-12', 69.6), ('2010-12', 37.183), ('2007-07', 95.967), ('1996-
11', 67.117), ('2010-10', 52.533), ('2011-08', 86.267), ('1994-01', 22.1), ('2010-02',
52.75), ('1998-04', 44.417), ('2015-07', 119.1), ('2002-09', 16.067), ('2012-03', 8.55),
('2005-12', 56.633), ('1997-10', 58.15), ('1993-05', 21.1), ('2016-01', 22.325), ('1993-08',
80.7), ('1995-08', 16.05), ('1997-08', 24.617), ('1998-12', 59.383), ('2004-05', 39.7),
('2009-11', 64.217), ('2001-06', 26.767), ('2009-04', 2.8), ('2009-09', 29.95), ('2011-03',
19.833), ('2011-10', 43.75), ('1995-12', 5.117), ('2004-08', 75.483), ('1997-05', 60.8),
('2004-12', 24.25), ('2006-02', 34.75), ('2003-03', 6.9), ('1995-04', 61.7), ('2011-07',
94.917), ('1994-03', 37.6), ('2002-10', 60.5), ('2013-12', 42.263), ('1996-05', 63.233),
('2003-02', 9.117), ('2008-05', 23.133), ('2009-12', 53.45), ('2013-07', 54.562), ('2008-
09', 47.367), ('1999-10', 18.55), ('1993-04', 0.0), ('2014-10', 72.137), ('1996-10', 22.45),
('2014-01', 62.575), ('2003-01', 17.717), ('1996-08', 37.717), ('2000-01', 18.617), ('2000-
07', 135.867), ('2002-12', 20.917), ('2003-04', 51.417), ('2003-05', 68.45), ('2005-05',
55.383), ('2014-02', 43.713), ('2001-12', 35.183), ('1995-09', 134.55), ('1994-11', 15.7),
('1996-07', 84.033), ('1998-09', 56.833), ('2006-03', 27.867), ('1994-10', 33.2), ('1998-
01', 44.317), ('1998-08', 86.467), ('2004-03', 28.483), ('2006-05', 52.333), ('2012-07',
59.067), ('2013-06', 61.325), ('2014-04', 31.763), ('2005-04', 11.65), ('2005-09', 13.95),
('1993-12', 37.1), ('1999-06', 50.25), ('1996-12', 39.55), ('2002-04', 29.917), ('1996-03',
10.033), ('2011-04', 14.917), ('2000-10', 110.3), ('1998-02', 50.033), ('2006-10',
118.167), ('2015-06', 78.663), ('2001-09', 110.633), ('1993-11', 42.8), ('1999-03',
42.233), ('2004-07', 96.0), ('2012-11', 68.65), ('2015-03', 42.613), ('2000-03', 21.683),
('2009-07', 113.167), ('2013-05', 47.925), ('2014-11', 52.425), ('2002-03', 26.933),
('2010-11', 93.55), ('1996-09', 57.467), ('1997-07', 41.967), ('2001-05', 33.983), ('2011-
11', 13.467), ('1999-07', 29.083), ('2003-10', 45.583), ('2004-04', 20.617), ('1994-07',
0.0), ('2005-06', 67.967)]
```