Mathias Fredholm, matfr953, 961121-8158
David Gumpert Harryson, davha130, 960302-4937

Lab 1 - Spark

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.

(u'1975', ((u'102190', 36.1), (u'123480', -37.0)))
(u'1992', ((u'112080', 35.4), (u'112080', -36.1)))
(u'1994', ((u'102210', 34.7), (u'112080', -40.5)))
(u'2014', ((u'108320', 34.4), (u'108320', -42.5)))
(u'2010', ((u'133250', 34.4), (u'108320', -41.7)))
(u'1989', ((u'102210', 33.9), (u'102210', -38.2)))
(u'1982', ((u'123250', 33.8), (u'123250', -42.2)))
(u'1968', ((u'123480', 33.7), (u'123480', -42.0)))
(u'1966', ((u'102190', 33.5), (u'123480', -49.4)))
(u'2002', ((u'102190', 33.3), (u'102190', -42.2)))
(u'1983', ((u'102200', 33.3), (u'112080', -38.2)))
(u'1986', ((u'102200', 33.2), (u'102200', -44.2)))
(u'1970', ((u'102190', 33.2), (u'102190', -39.6)))
(u'2000', ((u'102190', 33.0), (u'133470', -37.6)))
(u'1956', ((u'108640', 33.0), (u'108640', -45.0)))
(u'1959', ((u'123480', 32.8), (u'108640', -43.6)))
(u'2006', ((u'123250', 32.7), (u'123250', -40.6)))
(u'1991', ((u'102210', 32.7), (u'102210', -39.3)))
(u'1988', ((u'102200', 32.6), (u'133260', -39.9)))
(u'2011', ((u'123340', 32.5), (u'108320', -42.0)))
(u'1999', ((u'102210', 32.4), (u'123250', -49.0)))
(u'1973', ((u'123480', 32.2), (u'112080', -39.3)))
(u'2003', ((u'108320', 32.2), (u'133470', -41.5)))

Mathias Fredholm, matfr953, 961121-8158
David Gumpert Harryson, davha130, 960302-4937

```
from pyspark import SparkContext
sc = SparkContext(appName = "exercise 1")

# Importing the temperature readings file and splitting the data to seperate each reading
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

# Mapping the temperature readings as key-value pairs with the value being the year of the reading and the key a tuple
 of the station number and the temperature
year_temperature = lines.map(lambda x: (x[1][0:4], (x[0],float(x[3]))))

#Filtering to only look at readings between the year 1950 and 2014
year_temperature = year_temperature.filter(lambda x: int(x[0])>=1950 and int(x[0])<=2014)

#Getting the max temperate for each year
max_temperatures = year_temperature.reduceByKey(lambda a,b: (a[0], max(a[1], b[1])))
#Getting the min temperature for each year
min_temperatures = year_temperature.reduceByKey(lambda a,b: (a[0], min(a[1], b[1])))
#Join max and min temperature
min_max_temp = max_temperatures.join(min_temperatures)

#coalesce and sort the results
min_max_temp = min_max_temp.coalesce(1, shuffle = True)
min_max_temp = min_max_temp.sortBy(ascending = False, keyfunc=lambda k: k[1][0][1])

#Saving the file
min_max_temp = min_max_temp.saveAsTextFile("BDA/output")
```

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

(u'2014-09', 76817)
(u'2014-05', 48799)
(u'2014-07', 144824)
(u'2014-10', 32624)
(u'2014-06', 91123)
(u'2014-11', 5089)
(u'2014-02', 4)
(u'2014-08', 115047)
(u'2014-03', 2725)
(u'2014-04', 15572)
(u'2013-08', 121222)
(u'2013-04', 4501)
(u'2013-06', 113111)
(u'2013-07', 127454)
(u'2013-10', 28038)
(u'2013-03', 16)
(u'2013-09', 72138)
(u'2013-02', 3)
(u'2013-11', 268)
(u'2013-05', 71797)
(u'2013-12', 1)

Mathias Fredholm, matfr953, 961121-8158
David Gumpert Harryson, davha130, 960302-4937

(u'2012-03', 5065)
(u'2012-10', 11675)

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

# Mapping the temperature readings as key-value pairs with the value being the year
 and month of the reading and the key a tuple of the station number and the tempera
ture(key, value) = (year,temperature)
year_and_month_temperature = lines.map(lambda x: (x[1][0:7], (x[0],float(x[3]))))

#Filtering for the focal years
year_and_month_temperature = year_and_month_temperature.filter(lambda x: int(x[0][0
:4])>=1950 and int(x[0][0:4])<=2014)

#Getting all temperature > 10
high_temperatures = year_and_month_temperature.filter(lambda x: int(x[1][1])>=10)

#Map to prep for count
high_temperatures = high_temperatures.map(lambda x: (x[0], 1))

#Count number of temperatures above 10 degrees for each month
count_of_high_temperatures = high_temperatures.reduceByKey(lambda a,b: a + b)

#Coalesce, sort and save
count_of_high_temperatures = count_of_high_temperatures.coalesce(1, shuffle=True)
count_of_high_temperatures = count_of_high_temperatures.sortBy(ascending = False, k
eyfunc=lambda k:int(k[0][0:4]))
count_of_high_temperatures.saveAsTextFile("BDA/output")
```

(b)
(u'2014-08', 296)
(u'2014-04', 254)
(u'2014-09', 296)
(u'2014-07', 297)
(u'2014-10', 272)
(u'2014-05', 296)
(u'2014-12', 3)
(u'2014-03', 170)
(u'2014-06', 298)
(u'2014-11', 160)
(u'2014-02', 16)
(u'2013-09', 299)
(u'2013-05', 301)
(u'2013-12', 13)
(u'2013-08', 300)
(u'2013-06', 302)
(u'2013-04', 209)
(u'2013-02', 6)
(u'2013-11', 123)

Mathias Fredholm, matfr953, 961121-8158
David Gumpert Harryson, davha130, 960302-4937

(u'2013-07', 301)
(u'2013-03', 9)
(u'2013-10', 271)
(u'2012-02', 62)

```
from pyspark import SparkContext
sc = SparkContext(appName = "exercise 2b")

# Read the file and split
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

# Create key value pairs
year_and_month_temperature = lines.map(lambda x: (x[1][0:7], (x[0],float(x[3]))))

# Filter on year
year_and_month_temperature = year_and_month_temperature.filter(lambda x: int(x[0][0:4])>=1950 and int(x[0][0:4])<=2014)

#Get all temperature > 10
high_temperatures = year_and_month_temperature.filter(lambda x: int(x[1][1])>=10)

#Map to prep for count, now we only take into account distinct values on year and mont + station
high_temperatures = high_temperatures.map(lambda x: (x[1][0],(x[0],1))).distinct().map(lambda x: (x[1][0], x[1][1]))

#Count number of temperatures above 10 degrees for each month
count_of_high_temperatures = high_temperatures.reduceByKey(lambda a,b: a + b)

#Coalesce, sort and save
count_of_high_temperatures = count_of_high_temperatures.coalesce(1, shuffle=True)
count_of_high_temperatures = count_of_high_temperatures.sortBy(ascending = False, keyfunc=lambda k:int(k[0][0:4]))
count_of_high_temperatures.saveAsTextFile("BDA/output")
```

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.

((u'2014-12', u'137110'), -5.43010752688172)
((u'2014-02', u'96190'), 2.0915178571428577)
((u'2014-07', u'146350'), 18.14704301075268)
((u'2014-09', u'97530'), 11.312777777777784)
((u'2014-11', u'167990'), -9.238055555555556)
((u'2014-12', u'134090'), -1.7266666666666668)
((u'2014-07', u'162860'), 19.537634408602134)
((u'2014-12', u'172940'), -7.420161290322579)
((u'2014-12', u'149120'), -4.243548387096775)
((u'2014-05', u'52230'), 12.974838709677414)
((u'2014-01', u'162790'), -7.758064516129036)
((u'2014-06', u'138240'), 11.723643949930453)
((u'2014-08', u'133180'), 13.300268817204303)
((u'2014-10', u'162860'), 2.9601615074024257)
((u'2014-09', u'87140'), 14.054971590909094)
((u'2014-12', u'123340'), -8.793548387096775)
((u'2014-01', u'171790'), -15.527553763440858)
((u'2014-10', u'66110'), 11.928494623655899)
((u'2014-11', u'162870'), -3.6733333333333333)
((u'2014-10', u'78290'), 9.234051144010758)

Mathias Fredholm, matfr953, 961121-8158
David Gumpert Harryson, davha130, 960302-4937

((u'2014-08', u'74440'), 13.583870967741934)
((u'2014-04', u'81350'), 7.905555555555556)
((u'2014-08', u'192840'), 11.748118279569885)

```
from pyspark import SparkContext
sc = SparkContext(appName = "exercise 3")

#Read file and split
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

#Map for the key-value pairs: (key, value) = ((year-month, station), temperature)
monthly_temperature = lines.map(lambda x: ((x[1][0:7], x[0]), (float(x[3]),1)))

#Filter over year span
filtered_monthly_temperature = monthly_temperature.filter(lambda x: int(x[0][0][0:4])>=1950 and int(x[0][0][0:4])<=2014)

#Sum temperature and count on each month.
sum_monthly_temperature = filtered_monthly_temperature.reduceByKey(lambda a,b: (a[0] + b[0],a[1]+b[1]))

# Divide the sum of the temperature by the count of readings that month
avg_monthly_temperature = sum_monthly_temperature.map(lambda x: (x[0], x[1][0]/x[1][1]))

# Coalesce and sort
avg_monthly_temperature = avg_monthly_temperature.coalesce(1, shuffle=True)
avg_monthly_temperature = avg_monthly_temperature.sortBy(ascending = False, keyfunc=lambda k:int(k[0][0][0:4]))

#Save to file
avg_monthly_temperature.saveAsTextFile("BDA/output")
```

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 and maximum daily 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

Output: EMPTY

Mathias Fredholm, matfr953, 961121-8158
David Gumpert Harryson, davha130, 960302-4937

```
from pyspark import SparkContext
sc = SparkContext(appName = "exercise 4")

#Import temperature and precipitation files
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
rain_file = sc.textFile("BDA/input/precipitation-readings.csv")

#Split the two imported files on ";"
temp_lines = temperature_file.map(lambda line: line.split(";"))
rain_lines = rain_file.map(lambda line: line.split(";"))

#Map the temperature file to key-value pair. (Station, temperature)
#Map the precipitation file to key-value pair. (Station, precipitation)
temperatures = temp_lines.map(lambda x: (x[0], float(x[3])))
rain = rain_lines.map(lambda x: (x[0], float(x[3])))

#Reduce temperatures to get maximum temperature for each station
#Reduce rain to get maximum precipitation for each station
temperatures = temperatures.reduceByKey(lambda a,b: max(a,b))
rain = rain.reduceByKey(lambda a,b: max(a,b))

#Filter to get temperatures and precipitation within the scope of the assignment
temperatures = temperatures.filter(lambda x: x[1] >= 25 and x[1] <= 30)
rain = rain.filter(lambda x: x[1] >= 100 and x[1] <= 200)

#Join and save
temp_rain = temperatures.join(rain).coalesce(1, shuffle=True)
temp_rain.saveAsTextFile("BDA/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 do this, you will first need 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, average monthly precipitation

(u'2016-04', 26.900000000000006)
(u'2016-06', 47.6625)
(u'2016-01', 22.325000000000003)
(u'2016-03', 19.962500000000006)
(u'2016-07', 0.0)
(u'2016-02', 21.5625)
(u'2016-05', 29.250000000000007)
(u'2015-10', 2.2625)
(u'2015-06', 78.66250000000002)
(u'2015-03', 42.61250000000001)
(u'2015-09', 101.29999999999998)
(u'2015-05', 93.22499999999997)
(u'2015-07', 119.09999999999995)
(u'2015-12', 28.924999999999997)
(u'2015-01', 59.112500000000026)

Mathias Fredholm, matfr953, 961121-8158
David Gumpert Harryson, davha130, 960302-4937

(u'2015-04', 15.337499999999999)
(u'2015-11', 63.88750000000002)
(u'2015-02', 24.824999999999996)
(u'2015-08', 26.987499999999997)
(u'2014-08', 90.81249999999997)
(u'2014-07', 22.9875)
(u'2014-02', 43.71250000000002)
(u'2014-10', 72.13749999999999)

```
from pyspark import SparkContext
sc = SparkContext(appName = "Exercise 4")

#Import stations and precipitation file
stations_file = sc.textFile("BDA/input/stations-Ostergotland.csv")
rain_file = sc.textFile("BDA/input/precipitation-readings.csv")

#Split the two imported files on ";"
station_lines = stations_file.map(lambda line: line.split(";"))
rain_lines = rain_file.map(lambda line: line.split(";"))

#Map rain to a key-value pair. ((Station, year-month), precipitation)
rain = rain_lines.map(lambda x: ((x[0],x[1][0:7]), float(x[3])))
rain = rain.filter(lambda x: (int(x[0][1][0:4]) <= 2016 and int(x[0][1][0:4]) >= 1993))

# List of all stations relevent
stations = station_lines.map(lambda x: (x[0])).collect()

#Filter away stations not i Region ostergotland
rain_ost = rain.filter(lambda x: (x[0][0] in stations))

#Sum the precipitation for each station, each month
rain_ost_month_station = rain_ost.reduceByKey(lambda a,b: a+b)

#Map rain_ost_mont_station to get key-vale pair (Year-month, (Sum of precipitation that month, count))
rain_ost_month = rain_ost_month_station.map(lambda x: ((x[0][1]),(x[1], 1)))

#Sum total precipitation of each month and count number of measurements made
rain_ost_month = rain_ost_month.reduceByKey(lambda a,b: (a[0]+b[0], a[1]+b[1]))

#Get average precipitation of each month for region Ostergotland
rain_ost_month_avg = rain_ost_month.mapValues(lambda x: (x[0]/x[1]))

#Coalesce and sort
output = rain_ost_month_avg.coalesce(1, shuffle=True)
output_sorted = output.sortBy(ascending = False, keyfunc=lambda x: (int(x[0][0:4])))
output_sorted.saveAsTextFile("BDA/output")
```