LAB REPORT 1
TDDE31

Prepared by: Aswathi Karunakaran Beena(aswka169)
Nandana Krishna (nankri143)

Question 1

a. Maximum temperature

```
from pyspark import SparkContext

sc = SparkContext(appName = "exercise 1")
# This path is to the file on hdfs
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

# (key, value) = (year,temperature)
year_temperature = lines.map(lambda x: (x[1][0:4], (x[0],float(x[3]))))

#filter
year_temperature = year_temperature.filter(lambda x: int(x[0])>=1950 and int(x[0])<=2014)

#Get maximum temperatures
max_temperatures = year_temperature.reduceByKey(lambda a,b: (a[0], max(a[1], b[1])))
max_temperatures = max_temperatures.sortBy(ascending = False, keyfunc=lambda k: k[1])

# Following code will save the result into /user/ACCOUNT_NAME/BDA/output folder
max_temperatures.saveAsTextFile("BDA/output")
```

Output
(u'2011', (u'123340', 32.5))
(u'1982', (u'123250', 33.8))
(u'2002', (u'123250', 33.3))
(u'1986', (u'123250', 33.2))
(u'1991', (u'123250', 32.7))
(u'2006', (u'123250', 32.7))
(u'1999', (u'123250', 32.4))

(u'1996', (u'133260', 30.8))
(u'1985', (u'133260', 29.8))
(u'2009', (u'133250', 31.5))
(u'2012', (u'133250', 31.3))
(u'1964', (u'124020', 31.2))

b. Minimum temperature

```
from pyspark import SparkContext

sc = SparkContext(appName = "exercise 1")
# This path is to the file on hdfs
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

# (key, value) = (year,temperature)
year_temperature = lines.map(lambda x: (x[1][0:4], (x[0],float(x[3]))))

#filter
year_temperature = year_temperature.filter(lambda x: int(x[0])>=1950 and int(x[0])<=2014)

#Get min
min_temperatures = year_temperature.reduceByKey(lambda a,b: (a[0], min(a[1], b[1])))
min_temperatures = min_temperatures.sortBy(ascending = False, keyfunc=lambda k: k[1])
#print(max_temperatures.collect())

# Following code will save the result into /user/ACCOUNT_NAME/BDA/output folder

min_temperatures.saveAsTextFile("BDA/output")
```

Output
```
(u'1976', (u'140480', -42.2))
(u'1965', (u'140480', -44.0))
(u'1981', (u'140480', -44.0))
(u'1978', (u'140480', -47.7))
(u'2010', (u'140460', -41.7))
(u'2011', (u'140460', -42.0))
(u'2014', (u'140460', -42.5))
(u'2000', (u'140360', -37.6))
(u'1993', (u'140360', -39.0))
(u'2005', (u'140360', -39.4))
(u'1996', (u'140360', -41.7))
(u'1998', (u'140360', -42.7))
```

Question 2
a.Non distinct

```
from pyspark import SparkContext

sc = SparkContext(appName = "exercise 1")
# This path is to the file on hdfs
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

# (key, value) = (year,temperature)
year_temperature = lines.map(lambda x: (x[1][0:7], float(x[3])))

#filter by year and temperature reading > 10
year_temperature = year_temperature.filter(lambda x: int(x[0][0:4])>=1950 and int(x[0][0:4])<=2014)
year_temperature = year_temperature.filter(lambda x: int(x[1])>10)

year_temperature = year_temperature.map(lambda x: (x[0], 1))

count = year_temperature.reduceByKey(lambda a,b: a+b)
count = count.sortBy(ascending = False, keyfunc=lambda k: k[1])

#print(count.collect())

# Following code will save the result into /user/ACCOUNT_NAME/BDA/output folder
count.saveAsTextFile("BDA/output")
```

Output

```
(u'2014-07', 144824)
(u'2011-07', 142587)
(u'2010-07', 138997)
(u'2012-07', 130691)
(u'2013-07', 127454)
(u'2009-07', 127430)
(u'2003-07', 125604)
(u'2011-08', 124676)
(u'2002-07', 123771)
(u'2006-08', 123482)
(u'2002-08', 122292)
(u'2009-08', 121596)
(u'2005-07', 121246)
(u'2013-08', 121222)
(u'2006-07', 121122)
```

b. Distinct

```
from pyspark import SparkContext

sc = SparkContext(appName = "exercise 1")
# This path is to the file on hdfs
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

# (key, value) = (year,temperature)
year_temperature = lines.map(lambda x: (x[1][0:7], float(x[3])))

#filtering by year
year_temperature = year_temperature.filter(lambda x: int(x[0][0:4])>=1950 and int(x[0][0:4])<=2014)

# count distict readings from each station
year_temperature = year_temperature.filter(lambda x: int(x[1])>10).distinct()

year_temperature = year_temperature.map(lambda x: (x[0], 1))


count = year_temperature.reduceByKey(lambda a,b: a+b)
count = count.sortBy(ascending = False, keyfunc=lambda k: k[1])

#print(count.collect())

# Following code will save the result into /user/ACCOUNT_NAME/BDA/output folder
count.saveAsTextFile("BDA/output")
```

Output
(u'1998-10', 41)
(u'1978-03', 41)
(u'1978-11', 41)
(u'2005-11', 41)
(u'1957-03', 39)
(u'1990-02', 39)
(u'1999-11', 38)
(u'2008-03', 37)
(u'1956-04', 37)
(u'2011-11', 37)
(u'2011-03', 37)
(u'1959-03', 36)
(u'2007-11', 35)

Question 3

```
from pyspark import SparkContext

sc = SparkContext(appName = "exercise 1")
# This path is to the file on hdfs
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

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

#filter
year_temperature = year_temperature.filter(lambda x: int(x[0][0][0:4])>=1960 and
int(x[0][0][0:4])<=2014)

sumcount = year_temperature.combineByKey(lambda avg: (avg, 1), lambda x,avg: (x[0] + avg, x[1] + 1),
lambda x,y: (x[0] + y[0], x[1] + y[1]))
avg_temperature = sumcount.map(lambda  (k, (sum, count)): (k, sum/count))

# Following code will save the result into /user/ACCOUNT_NAME/BDA/output folder
avg_temperature.saveAsTextFile("BDA/output")
```

Output
```
((u'1970-11', u'115230'), -3.419166666666668)
((u'1978-06', u'62180'), 14.60700280112046)
((u'1963-11', u'94660'), -0.18111111111111103)
((u'1986-10', u'98040'), 6.6997311827956985)
((u'1985-12', u'124020'), -14.698387096774193)
((u'1960-06', u'117330'), 14.46111111111111)
((u'2012-01', u'148040'), -7.350134408602156)
((u'1992-09', u'105450'), 9.945555555555552)
((u'1997-08', u'98160'), 20.53442176870748)
((u'2008-01', u'108320'), 1.74260752688172)
((u'2006-11', u'66110'), 6.788472222222218)
((u'1984-09', u'72080'), 11.082222222222223)
((u'1980-03', u'192710'), -9.583225806451608)
((u'1989-10', u'123070'), 1.6496815286624207)
((u'1988-02', u'132180'), -6.873593073593074)
((u'1998-10', u'62260'), 9.280243572395124)
((u'1983-01', u'108110'), -0.2677419354838712)
((u'1990-02', u'151290'), -0.3294642857142857)
```

Question 4

```python
from pyspark import SparkContext

sc = SparkContext(appName = "exercise 1")
# This path is to the file on hdfs
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
precipitation_file = sc.textFile("BDA/input/precipitation-readings.csv")
tempslines = temperature_file.map(lambda line: line.split(";"))
precpslines = precipitation_file.map(lambda line: line.split(";"))

#(station number, temperature)
station_temperature = tempslines.map(lambda x: (x[0] ,float(x[3])))
#((station number, date), precipitation)
daily_precipitation = precpslines.map(lambda x: ((x[0], x[1]) ,float(x[3])))
#filter

station_temperature = station_temperature.reduceByKey(lambda x,y: x if x>=y else y)
station_temperature = station_temperature.filter(lambda x: int(x[1])>25 and int(x[1])<30)
daily_precipitation = daily_precipitation.reduceByKey(lambda a,b: a + b)
daily_precipitation = daily_precipitation.map(lambda x: (x[0][0], x[1]))
daily_precipitation = daily_precipitation.reduceByKey(lambda x,y: x if x>=y else y)
daily_precipitation = daily_precipitation.filter(lambda x: int(x[1])>100 and int(x[1])<200)

tempsprecps = station_temperature.join(daily_precipitation)
tempsprecps = tempsprecps.map(lambda x: (x[0], x[1], x[3]))

#print(max_temperatures.collect())

# Following code will save the result into /user/ACCOUNT_NAME/BDA/output folder
tempsprecps.saveAsTextFile("BDA/output")
```

Output
No output

Question 5

```
from pyspark import SparkContext

sc = SparkContext(appName = "exercise 1")
# This path is to the file on hdfs
precipitate_file = sc.textFile("BDA/input/precipitation-readings.csv")
ostergotland_file = sc.textFile("BDA/input/stations-Ostergotland.csv")
osterlines = ostergotland_file.map(lambda line: line.split(";"))
precpslines = precipitate_file.map(lambda line: line.split(";"))

#(station number, temperature)
station_oster = osterlines.map(lambda x: (x[0])).collect()

#((station number, date), precipitation)
monthly_precipitation = precpslines.map(lambda x: ((x[0], x[1][0:7]) ,(float(x[3]))))

#filter
monthly_precipitation = monthly_precipitation.filter(lambda x: x[0][0] in station_oster)
monthly_precipitation = monthly_precipitation.filter(lambda x: int (x[0][1][0:4]) >= 1993 and int
(x[0][1][0:4]) <= 2016)


monthly_precipitation = monthly_precipitation.reduceByKey(lambda a,b: a + b)
monthly_precipitation = monthly_precipitation.map(lambda x: (x[0][1][0:7], (x[1], 1)))
count = monthly_precipitation.reduceByKey(lambda a,b: (a[0] + b[0], a[1]+b[1]))
avg_monthly_precipitation = count.map(lambda a: (a[0], a[1][0]/a[1][1]))

#print(max_temperatures.collect())

# Following code will save the result into /user/ACCOUNT_NAME/BDA/output folder
avg_monthly_precipitation.saveAsTextFile("BDA/output")
```

Output
u'1996-11', 67.11666666666665)
(u'2008-10', 59.56666666666669)
(u'2014-05', 58.000000000000014)
(u'2004-12', 24.25)
(u'2001-11', 26.38333333333334)
(u'2011-05', 37.85)
(u'1999-07', 29.08333333333334)

(u'2010-02', 52.75000000000005)
(u'2013-08', 54.075)
(u'2010-09', 43.08333333333335)
(u'2013-05', 47.92500000000001)
(u'1998-11', 28.96666666666668)
(u'2002-03', 26.93333333333334)
(u'2013-02', 25.52500000000002)
(u'2000-01', 18.61666666666667)