Homework 4
CSCE 587
Fall 2016
Due: 12/1/2016 via Dropbox
Name: Sendurr Selvaraj
VIP ID: 00323540

Problem 1:
create a set of map, reduce, mapreduce functions to process the data in test_25K.csv. In the case of this problem you are to determine how many flight cancellations there were for each origin airport Look at the map function we saw in class for counting flights for hints. To figure out which columns correspond to origin airport (Origin) and cancelled (Cancelled), look at testData.csv which has header information. Start by testing your code on the small data set testDataNoHdr.csv. When your code works, try your code with test_25K.csv.

Assuming that you used the line out = mr(hdfs.data, hdfs.out) to invoke you map reduce job, output your results using:
results = from.dfs(out)
results.df = as.data.frame(results, stringsAsFactors=F)
colnames(results.df) = c('Origin', 'Cancelled')
results.df

R-code

```
#**********************************
#*  Problem 1                     *
#**********************************
# Set environmental variables
Sys.setenv(HADOOP_CMD="/usr/bin/hadoop")
Sys.setenv(HADOOP_STREAMING="/usr/hdp/2.3.0.0-2557/hadoop-mapreduce/hadoop-
streaming-2.7.1.2.3.0.0-2557.jar")

# Load the following packages in the following order
library(rhdfs)
library(rmr2)

# initialize the connection from rstudio to hadoop
hdfs.init()

# Doing simple mapreduce on airline data
# Our map function which returns the keyval <origin airport, cancellations>
map1 = function(k,flights) {
  return ( keyval(as.character(flights[[17]]),flights[[22]]))
}
```

```
# Our reduce function which sums up the cancelled flights at each origin airport
reduce1 = function(origin, counts) {
  keyval(origin, sum(counts,na.rm=TRUE))
}

# Our mapreduce function which invokes map1 and reduce1 and parses
# the input file expected it to be comma delimited
mr1 = function(input, output = NULL) {
  mapreduce(input = input,
            output = output,
            input.format = make.input.format("csv", sep=","),
            map = map1,
            reduce = reduce1)}

# Set up the input definition (small dataset) and output definition
hdfs.root = '/user/share/student'
hdfs.data = file.path(hdfs.root,'test_25K.csv')
hdfs.out = file.path(hdfs.root,'out1')

# Invoke out mapreduce job
out = mr1(hdfs.data, hdfs.out)

# Fetch the results from HDFS and coerce into a dataframe
results = from.dfs(out)
results.df = as.data.frame(results, stringsAsFactors=F)

# add column heading to dataframe
colnames(results.df) = c('Origin', 'Canceled')

# Display results
results.df
```

Output

	Origin	Canceled
1	ABE	0
2	ABI	1
3	ABQ	1
4	ABY	1
5	ACK	0
6	ACT	0
7	ACV	1
8	ACY	0
9	ADQ	1
10	AEX	0
11	AGS	2

12	ALB	1
13	AMA	0
14	ANC	1
15	ATL	26
16	ATW	0
17	AUS	4
18	AVL	1
19	AVP	0
20	AZO	1
21	BDL	0
22	BET	1
23	BFL	0
24	BGM	0
25	BGR	1
26	BHM	0
27	BIL	0
28	BIS	0
29	BMI	0
30	BNA	4
31	BOI	0
32	BOS	22
33	BPT	0
34	BQK	0
35	BQN	0
36	BRO	0
37	BRW	0
38	BTM	0
39	BTR	0
40	BTV	1
41	BUF	1
42	BUR	1
43	BWI	6
44	BZN	0
45	CAE	1
46	CAK	0
47	CDC	0
48	CDV	0
49	CEC	1
50	CHA	1
51	CHO	0
52	CHS	4
53	CIC	0
54	CID	2
55	CLD	0
56	CLE	3
57	CLL	0

58	CLT	5
59	CMH	5
60	CMI	0
61	COD	1
62	COS	1
63	CPR	0
64	CRP	0
65	CRW	0
66	CSG	1
67	CVG	24
68	DAB	2
69	DAL	7
70	DAY	2
71	DBQ	1
72	DCA	6
73	DEN	6
74	DFW	11
75	DHN	1
76	DLH	0
77	DRO	0
78	DSM	1
79	DTW	6
80	EFD	0
81	EGE	0
82	EKO	0
83	ELP	1
84	ERI	0
85	EUG	0
86	EVV	0
87	EWR	16
88	EYW	0
89	FAI	2
90	FAR	0
91	FAT	1
92	FAY	0
93	FCA	0
94	FLL	2
95	FNT	0
96	FSD	2
97	FSM	0
98	FWA	0
99	GEG	0
100	GFK	0
101	GGG	0
102	GJT	0
103	GNV	0

104	GPT	0
105	GRB	0
106	GRK	0
107	GRR	0
108	GSO	4
109	GSP	5
110	GST	0
111	GTF	0
112	GTR	1
113	GUC	0
114	HDN	0
115	HLN	0
116	HNL	0
117	HOU	12
118	HPN	0
119	HRL	0
120	HSV	0
121	HTS	1
122	HVN	1
123	IAD	12
124	IAH	7
125	ICT	2
126	IDA	0
127	ILE	0
128	ILM	0
129	IND	3
130	IPL	0
131	ISP	1
132	ITO	0
133	IYK	0
134	JAC	1
135	JAN	0
136	JAX	4
137	JFK	6
138	JNU	1
139	KOA	0
140	KTN	0
141	LAN	0
142	LAS	4
143	LAW	0
144	LAX	9
145	LBB	1
146	LCH	0
147	LEX	2
148	LFT	0
149	LGA	7

150	LGB	0
151	LIH	0
152	LIT	0
153	LNK	0
154	LNY	0
155	LRD	0
156	LSE	0
157	LWB	0
158	LYH	0
159	MAF	0
160	MBS	0
161	MCI	2
162	MCN	0
163	MCO	8
164	MDT	0
165	MDW	1
166	MEI	0
167	MEM	0
168	MFE	0
169	MFR	0
170	MGM	0
171	MHT	0
172	MIA	5
173	MKE	1
174	MKK	0
175	MLB	1
176	MLI	0
177	MLU	0
178	MOB	0
179	MOD	0
180	MOT	0
181	MQT	0
182	MRY	1
183	MSN	0
184	MSO	1
185	MSP	11
186	MSY	4
187	MTJ	0
188	MYR	2
189	OAK	1
190	OGG	0
191	OKC	2
192	OMA	2
193	OME	1
194	ONT	1
195	ORD	45

196	ORF	1
197	OTZ	0
198	OXR	0
199	PBI	0
200	PDX	3
201	PFN	0
202	PHF	0
203	PHL	6
204	PHX	7
205	PIA	0
206	PIE	0
207	PIH	0
208	PIT	2
209	PNS	0
210	PSC	0
211	PSG	0
212	PSP	1
213	PVD	1
214	PWM	2
215	RAP	0
216	RDD	0
217	RDM	0
218	RDU	6
219	RIC	3
220	RNO	1
221	ROA	2
222	ROC	2
223	RST	0
224	RSW	2
225	SAN	6
226	SAT	3
227	SAV	1
228	SBA	0
229	SBN	0
230	SBP	0
231	SCC	0
232	SCE	1
233	SDF	0
234	SEA	5
235	SFO	4
236	SGF	0
237	SGU	0
238	SHV	0
239	SIT	0
240	SJC	3
241	SJT	0

242 SJU 3
243 SLC 6
244 SMF 2
245 SMX 0
246 SNA 4
247 SPS 0
248 SRQ 1
249 STL 3
250 STT 1
251 STX 0
252 SUN 0
253 SWF 0
254 SYR 0
255 TLH 2
256 TOL 0
257 TPA 8
258 TRI 0
259 TUL 2
260 TUS 0
261 TVC 1
262 TWF 0
263 TXK 0
264 TYR 0
265 TYS 3
266 VCT 0
267 VIS 0
268 VLD 0
269 VPS 0
270 WRG 0
271 XNA 3
272 YAK 0
273 YUM 0

Problem 2:

create another set of map, reduce, mapreduce functions (with different names) to process the data in test_25K.csv. In the case of this problem you should find the maximal taxi in time by destination airport. In other words, for each destination airport you report the taxi in time that was largest. If the data set has n destination airports then your output should list each of the n airports with the largest taxi in time, one airport per row. To figure out which columns correspond to destination airport (Dest) and taxi in time (TaxiIn), look at testData.csv which has header information. Define the column names for output using:
colnames(results.df) = c('Airport', 'Max Taxi In')

R-code:

```
#**********************************
#*  Problem 2                     *
#**********************************

map2 = function(k,flights) {
  return ( keyval(as.character(flights[[18]]),flights[[20]]))
}

# Our reduce function which finds the largest taxin time for each destination airports
reduce2 = function(origin, counts) {
  keyval(origin, max(counts,na.rm=TRUE))
}

# Our mapreduce function which invokes map1 and reduce1 and parses
# the input file expected it to be comma delimited
mr2 = function(input, output = NULL) {
  mapreduce(input = input,
            output = output,
            input.format = make.input.format("csv", sep=","),
            map = map2,
            reduce = reduce2)}

# Set up the input definition (small dataset) and output definition
hdfs.root = '/user/share/student'
hdfs.data = file.path(hdfs.root,'test_25K.csv')
hdfs.out = file.path(hdfs.root,'out2')

# Invoke out mapreduce job
out = mr2(hdfs.data, hdfs.out)

# Fetch the results from HDFS and coerce into a dataframe
results = from.dfs(out)
results.df = as.data.frame(results, stringsAsFactors=F)

# add column heading to dataframe
colnames(results.df) = c('Airport', 'Max Taxi In')

# Display results
results.df
```

Output

	Airport	Max Taxi In
1	ABE	7
2	ABI	4
3	ABQ	37
4	ABY	6
5	ACK	6
6	ACT	8
7	ACV	5
8	ACY	4
9	ADQ	3
10	AEX	5
11	AGS	5
12	AKN	3
13	ALB	8
14	AMA	9
15	ANC	11
16	ATL	1469
17	ATW	8
18	AUS	13
19	AVL	4
20	AVP	6
21	AZO	10
22	BDL	12
23	BET	3
24	BFL	10
25	BGM	63
26	BGR	10
27	BHM	9
28	BIL	19
29	BIS	6
30	BMI	7
31	BNA	28
32	BOI	9
33	BOS	63
34	BPT	4
35	BQK	3
36	BQN	6
37	BRO	5
38	BRW	5
39	BTM	5
40	BTR	7
41	BTV	16
42	BUF	16
43	BUR	12

44	BWI	22
45	BZN	6
46	CAE	1446
47	CAK	8
48	CDC	3
49	CDV	4
50	CEC	3
51	CHA	6
52	CHO	4
53	CHS	14
54	CIC	3
55	CID	6
56	CLD	5
57	CLE	19
58	CLL	34
59	CLT	26
60	CMH	13
61	CMI	8
62	COD	6
63	COS	18
64	CPR	6
65	CRP	7
66	CRW	8
67	CSG	3
68	CVG	1326
69	DAB	6
70	DAL	10
71	DAY	8
72	DCA	1078
73	DEN	545
74	DFW	1451
75	DHN	8
76	DLG	4
77	DLH	9
78	DRO	2
79	DSM	7
80	DTW	40
81	EFD	5
82	EGE	8
83	EKO	6
84	ELP	8
85	ERI	6
86	EUG	5
87	EVV	8
88	EWR	33
89	EYW	4

90	FAI	6
91	FAR	8
92	FAT	7
93	FAY	1442
94	FCA	6
95	FLL	31
96	FLO	7
97	FNT	15
98	FSD	8
99	FSM	9
100	FWA	6
101	GEG	20
102	GFK	6
103	GGG	7
104	GJT	4
105	GNV	1443
106	GPT	11
107	GRB	7
108	GRK	6
109	GRR	19
110	GSO	8
111	GSP	8
112	GTF	5
113	GTR	4
114	HDN	3
115	HLN	6
116	HNL	23
117	HOU	24
118	HPN	21
119	HRL	7
120	HSV	14
121	HTS	2
122	HVN	4
123	IAD	1444
124	IAH	50
125	ICT	1351
126	IDA	4
127	ILE	20
128	ILM	6
129	IND	22
130	IPL	3
131	ISP	7
132	ITO	6
133	IYK	4
134	JAC	3
135	JAN	1446

136	JAX	22
137	JFK	39
138	JNU	5
139	KOA	7
140	KTN	5
141	LAN	5
142	LAS	42
143	LAW	6
144	LAX	48
145	LBB	8
146	LCH	7
147	LEX	7
148	LFT	5
149	LGA	90
150	LGB	27
151	LIH	13
152	LIT	1357
153	LNK	6
154	LRD	6
155	LSE	8
156	LWB	8
157	LYH	6
158	MAF	6
159	MBS	5
160	MCI	10
161	MCN	5
162	MCO	278
163	MDT	10
164	MDW	30
165	MEI	7
166	MEM	27
167	MFE	6
168	MFR	6
169	MGM	6
170	MHT	10
171	MIA	98
172	MKE	11
173	MKK	3
174	MLB	15
175	MLI	7
176	MLU	5
177	MOB	8
178	MOD	5
179	MOT	6
180	MQT	8
181	MRY	6

182	MSN	7
183	MSO	6
184	MSP	65
185	MSY	13
186	MTJ	4
187	MYR	1442
188	OAK	21
189	OGG	7
190	OKC	20
191	OMA	21
192	OME	4
193	ONT	9
194	ORD	448
195	ORF	1443
196	OTZ	4
197	OXR	4
198	PBI	1445
199	PDX	10
200	PFN	6
201	PHF	9
202	PHL	51
203	PHX	27
204	PIA	5
205	PIE	7
206	PIH	4
207	PIT	27
208	PNS	6
209	PSC	7
210	PSG	4
211	PSP	9
212	PVD	13
213	PWM	21
214	RAP	7
215	RDD	4
216	RDM	4
217	RDU	31
218	RIC	14
219	RNO	22
220	ROA	7
221	ROC	31
222	RST	9
223	RSW	7
224	SAN	20
225	SAT	8
226	SAV	11
227	SBA	12

228	SBN	5
229	SBP	6
230	SCE	5
231	SDF	10
232	SEA	63
233	SFO	29
234	SGF	8
235	SGU	5
236	SHV	9
237	SIT	8
238	SJC	19
239	SJT	7
240	SJU	21
241	SLC	125
242	SMF	14
243	SMX	4
244	SNA	25
245	SPS	9
246	SRQ	7
247	STL	28
248	STT	6
249	STX	3
250	SUN	3
251	SWF	8
252	SYR	30
253	TLH	8
254	TOL	8
255	TPA	59
256	TRI	6
257	TUL	7
258	TUS	20
259	TVC	7
260	TWF	7
261	TXK	5
262	TYR	12
263	TYS	10
264	VCT	4
265	VIS	3
266	VLD	3
267	VPS	18
268	WRG	5
269	XNA	1444
270	YAK	3
271	YUM	4