Lab 10- W205 Section 3 Spring 2017
Sanjay Dorairaj

SUBMISSION 1: *Print only words with a length > 5 characters. Submit the pyspark code*

```
from pyspark import SparkContext
from pyspark.streaming import StreamingContext
ssc = StreamingContext(sc, 1)
lines= ssc.textFileStream("file:///tmp/datastreams")
uclines = lines.map(lambda word: word.upper())
uclines = uclines.filter(lambda word: len(word) > 5)
uclines.pprint()
ssc.start()
```

Output:

```
-------------------------------------------
Time: 2017-04-11 04:20:32
-------------------------------------------
COLORADO
DORADO
ELDORADE
GATRORADE

-------------------------------------------
Time: 2017-04-11 04:20:33
-------------------------------------------

-------------------------------------------
Time: 2017-04-11 04:20:34
-------------------------------------------

-------------------------------------------
Time: 2017-04-11 04:20:35
-------------------------------------------

-------------------------------------------
Time: 2017-04-11 04:20:36
-------------------------------------------

-------------------------------------------
```

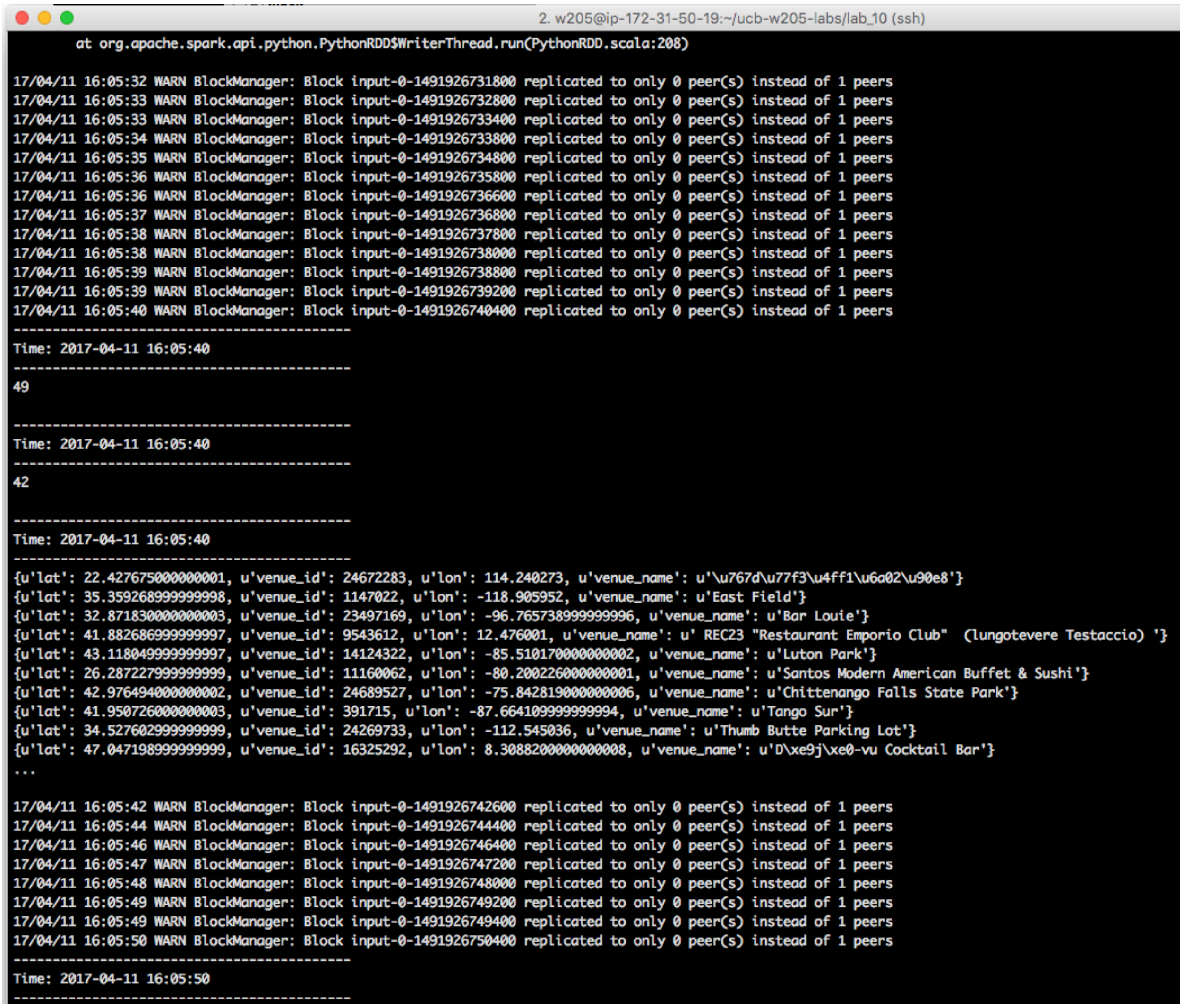
```
Time: 2017-04-11 04:20:37
-------------------------------------------
COLORADO
DORADO
ELDORADE
GATRORADE
```

SUBMISSION 2: *Change the code so that you save the venue components to a text file. Submit you code.*

```
MASTER=local[4] pyspark

from pyspark import SparkContext
from pyspark.streaming import StreamingContext
import json
ssc = StreamingContext(sc, 10)
lines = ssc.textFileStream("file:///tmp/datastreams")
slines = lines.flatMap(lambda x: [ j['venue'] for j in
json.loads('['+x+']') if 'venue' in j] )
cnt=slines.count()
cnt.pprint()
slines.pprint()
slines.saveAsTextFiles("file:///tmp/venues.txt")

ssc.start()
```

SUBMISSION 3: *In a previous module in this class you learnt about streams, burstiness and kafka. Describe how you would solve a situation where (1) you have a very busty stream where you spark streaming process may not always be able to keep up with the data it receives, meaning it the time it takes to process takes longer than the batch interval. (2) Like other programs stream processing programs need to be updated. Describe the implications of updating this simple processing program. What side effects can it have? How could you potentially handle them?.*

Output with Spark Submit



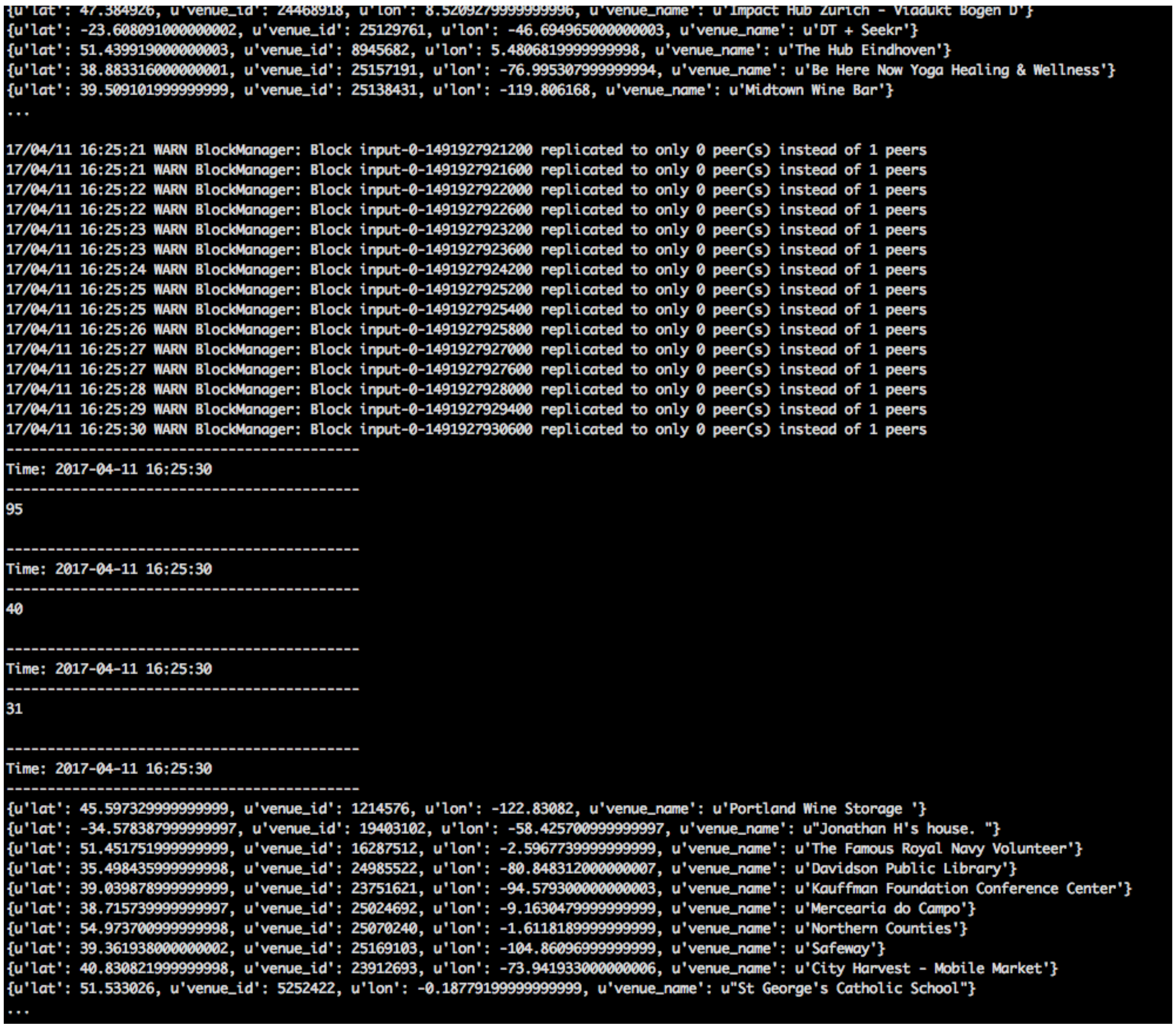
```
        at org.apache.spark.api.python.PythonRDD$WriterThread.run(PythonRDD.scala:208)

17/04/11 16:05:32 WARN BlockManager: Block input-0-1491926731800 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:33 WARN BlockManager: Block input-0-1491926732800 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:33 WARN BlockManager: Block input-0-1491926733400 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:34 WARN BlockManager: Block input-0-1491926733800 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:35 WARN BlockManager: Block input-0-1491926734800 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:36 WARN BlockManager: Block input-0-1491926735800 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:36 WARN BlockManager: Block input-0-1491926736600 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:37 WARN BlockManager: Block input-0-1491926736800 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:38 WARN BlockManager: Block input-0-1491926737800 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:38 WARN BlockManager: Block input-0-1491926738000 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:39 WARN BlockManager: Block input-0-1491926738800 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:39 WARN BlockManager: Block input-0-1491926739200 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:40 WARN BlockManager: Block input-0-1491926740400 replicated to only 0 peer(s) instead of 1 peers
-------------------------------------------
Time: 2017-04-11 16:05:40
-------------------------------------------
49

-------------------------------------------
Time: 2017-04-11 16:05:40
-------------------------------------------
42

-------------------------------------------
Time: 2017-04-11 16:05:40
-------------------------------------------
{u'lat': 22.427675000000001, u'venue_id': 24672283, u'lon': 114.240273, u'venue_name': u'\u767d\u77f3\u4ff1\u6a02\u90e8'}
{u'lat': 35.359268999999998, u'venue_id': 1147022, u'lon': -118.905952, u'venue_name': u'East Field'}
{u'lat': 32.871830000000003, u'venue_id': 23497169, u'lon': -96.765738999999996, u'venue_name': u'Bar Louie'}
{u'lat': 41.882686999999997, u'venue_id': 9543612, u'lon': 12.476001, u'venue_name': u' REC23 "Restaurant Emporio Club"  (lungotevere Testaccio) '}
{u'lat': 43.118049999999997, u'venue_id': 14124322, u'lon': -85.510170000000002, u'venue_name': u'Luton Park'}
{u'lat': 26.287227999999999, u'venue_id': 11160062, u'lon': -80.200226000000001, u'venue_name': u'Santos Modern American Buffet & Sushi'}
{u'lat': 42.976494000000002, u'venue_id': 24689527, u'lon': -75.842819000000006, u'venue_name': u'Chittenango Falls State Park'}
{u'lat': 41.950726000000003, u'venue_id': 391715, u'lon': -87.664109999999994, u'venue_name': u'Tango Sur'}
{u'lat': 34.527602999999999, u'venue_id': 24269733, u'lon': -112.545036, u'venue_name': u'Thumb Butte Parking Lot'}
{u'lat': 47.047198999999999, u'venue_id': 16325292, u'lon': 8.3088200000000008, u'venue_name': u'D\xe9j\xe0-vu Cocktail Bar'}
...

17/04/11 16:05:42 WARN BlockManager: Block input-0-1491926742600 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:44 WARN BlockManager: Block input-0-1491926744400 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:46 WARN BlockManager: Block input-0-1491926746400 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:47 WARN BlockManager: Block input-0-1491926747200 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:48 WARN BlockManager: Block input-0-1491926748000 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:49 WARN BlockManager: Block input-0-1491926749200 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:49 WARN BlockManager: Block input-0-1491926749400 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:05:50 WARN BlockManager: Block input-0-1491926750400 replicated to only 0 peer(s) instead of 1 peers
-------------------------------------------
Time: 2017-04-11 16:05:50
-------------------------------------------
```

Describe how you would solve a situation where (1) you have a very busty stream where you spark streaming process may not always be able to keep up with the data it receives, meaning it the time it takes to process takes longer than the batch interval.

1. Kafka queues have tunable parameters that allow us to adjust the retention period to ensure that programs such as Spark Streaming are able to deal with bursty traffic. A longer retention period ensures that even though there is bursty traffic, Kafka is able to buffer this bursty traffic till the stream processing program has had the time to process all the events.
2. In case, the streaming processing is unable to cope with bursty traffic despite tuning Kafka, then, the alternative would be for the streaming program to apply a filter to the incoming events so as to reduce the number of events that it processes.

3. A third alternative, is to send the events from Kafka directly to a file system and then read off that filesystem or database, although this may defeat the process of stream processing.
4. Yet another alternative could be to increase the size of the min-batches by increasing the time between consecutive processing. This may reduce transport overhead related to processing the increasing stream, giving a bit more time for the streaming program to process the incoming dstreams.

SUBMISSION 4a: *Provide a screenshot of the running Spark Streaming application that shows that the CountByWindow indeed provides an sum of the counts from the 3 latest batches. See example screenshot below.*



```
{u'lat': 47.384926, u'venue_id': 24468918, u'lon': 8.520927999999996, u'venue_name': u'Impact Hub Zurich - Viadukt Bogen D'}
{u'lat': -23.608091000000002, u'venue_id': 25129761, u'lon': -46.694965000000003, u'venue_name': u'DT + Seekr'}
{u'lat': 51.439919000000003, u'venue_id': 8945682, u'lon': 5.4806819999999998, u'venue_name': u'The Hub Eindhoven'}
{u'lat': 38.883316000000001, u'venue_id': 25157191, u'lon': -76.995307999999994, u'venue_name': u'Be Here Now Yoga Healing & Wellness'}
{u'lat': 39.509101999999999, u'venue_id': 25138431, u'lon': -119.806168, u'venue_name': u'Midtown Wine Bar'}
...

17/04/11 16:25:21 WARN BlockManager: Block input-0-1491927921200 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:21 WARN BlockManager: Block input-0-1491927921600 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:22 WARN BlockManager: Block input-0-1491927922000 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:22 WARN BlockManager: Block input-0-1491927922600 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:23 WARN BlockManager: Block input-0-1491927923200 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:23 WARN BlockManager: Block input-0-1491927923600 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:24 WARN BlockManager: Block input-0-1491927924200 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:25 WARN BlockManager: Block input-0-1491927925200 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:25 WARN BlockManager: Block input-0-1491927925400 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:26 WARN BlockManager: Block input-0-1491927925800 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:27 WARN BlockManager: Block input-0-1491927927000 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:27 WARN BlockManager: Block input-0-1491927927600 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:28 WARN BlockManager: Block input-0-1491927928000 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:29 WARN BlockManager: Block input-0-1491927929400 replicated to only 0 peer(s) instead of 1 peers
17/04/11 16:25:30 WARN BlockManager: Block input-0-1491927930600 replicated to only 0 peer(s) instead of 1 peers
-------------------------------------------
Time: 2017-04-11 16:25:30
-------------------------------------------
95

-------------------------------------------
Time: 2017-04-11 16:25:30
-------------------------------------------
40

-------------------------------------------
Time: 2017-04-11 16:25:30
-------------------------------------------
31

-------------------------------------------
Time: 2017-04-11 16:25:30
-------------------------------------------
{u'lat': 45.597329999999999, u'venue_id': 1214576, u'lon': -122.83082, u'venue_name': u'Portland Wine Storage '}
{u'lat': -34.578387999999997, u'venue_id': 19403102, u'lon': -58.425700999999997, u'venue_name': u"Jonathan H's house. "}
{u'lat': 51.451751999999999, u'venue_id': 16287512, u'lon': -2.5967739999999999, u'venue_name': u'The Famous Royal Navy Volunteer'}
{u'lat': 35.498435999999998, u'venue_id': 24985522, u'lon': -80.848312000000007, u'venue_name': u'Davidson Public Library'}
{u'lat': 39.039878999999999, u'venue_id': 23751621, u'lon': -94.579300000000003, u'venue_name': u'Kauffman Foundation Conference Center'}
{u'lat': 38.715739999999997, u'venue_id': 25024692, u'lon': -9.1630479999999999, u'venue_name': u'Mercearia do Campo'}
{u'lat': 54.973700999999998, u'venue_id': 25070240, u'lon': -1.6118189999999999, u'venue_name': u'Northern Counties'}
{u'lat': 39.361938000000002, u'venue_id': 25169103, u'lon': -104.86096999999999, u'venue_name': u'Safeway'}
{u'lat': 40.830821999999998, u'venue_id': 23912693, u'lon': -73.941933000000006, u'venue_name': u'City Harvest - Mobile Market'}
{u'lat': 51.533026, u'venue_id': 5252422, u'lon': -0.18779199999999999, u'venue_name': u"St George's Catholic School"}
...
```

SUBMISSION 4b: *Also explain what the difference is between having 10 sec batches with a 30 sec sliding window and a 30 second batch length.*

Batch size refers to the dstream duration. If the mini-batch size is set to 10 seconds, then the dstream that is sent to the streaming program would contain RDDs for the past 10 seconds. Similarly, if the batch length is 30 seconds, then the dstream would contain RDDs for the past 30 seconds.

The window size refers to dstreams that were accumulated during the window size. A window size of 30 seconds with 10 second batch size would operate on 10-second mini batches over a 30 second duration, which in this case would involve 3 mini batches.