CS4830: Big Data Laboratory Assignment-1

Shania Mitra CH18B067

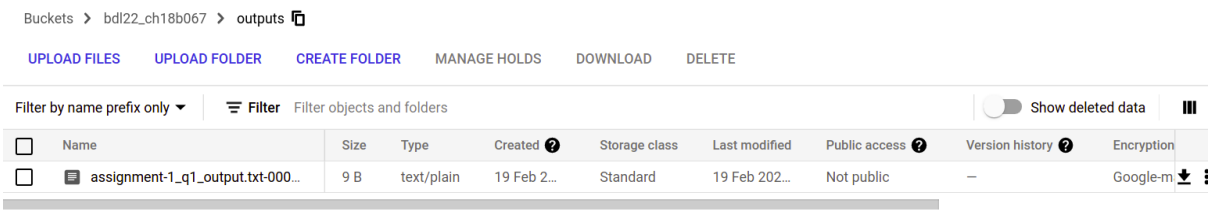
Please Note: Screenshots have been pasted in the file instead of being attached separately in the zip file.

Q1. Write a Python code to count lines of the file placed in the BDL2022 bucket (gs://bdl2022/lines_big.txt) using Dataflow and provide the screenshot of the file that is generated in your bucket.

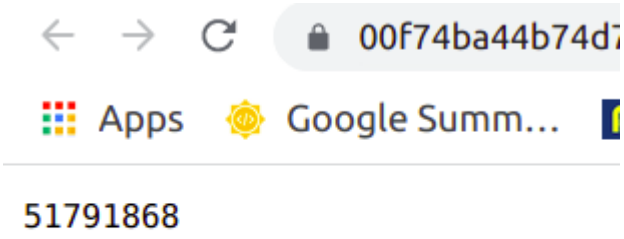
Code Screenshot:

```
import apache_beam as beam
from apache_beam.io import ReadFromText
from apache_beam.io import WriteToText
from apache_beam.options.pipeline_options import PipelineOptions
from apache_beam.options.pipeline_options import GoogleCloudOptions
from apache_beam.options.pipeline_options import StandardOptions
options = PipelineOptions()
google_cloud_options = options.view_as(GoogleCloudOptions)
google_cloud_options.project = 'graphite-byte-268703' # Enter your project ID
google_cloud_options.job_name = 'lab3'
google_cloud_options.temp_location = "gs://bdl22_ch18b067/temp"
google_cloud_options.region = "us-central1"
options.view_as(StandardOptions).runner = 'DataflowRunner'
p = beam.Pipeline(options=options)
lines = p | 'Read' >> beam.io.ReadFromText('gs://bdl2022/lines_big.txt')|beam.combiners.Count.Globally() |'Write' >> beam.io.WriteToText('gs://bdl22_ch18b067/outputs/assignment-1_q1_output.txt')
result = p.run()
```

File generated in bucket:



Content of file:



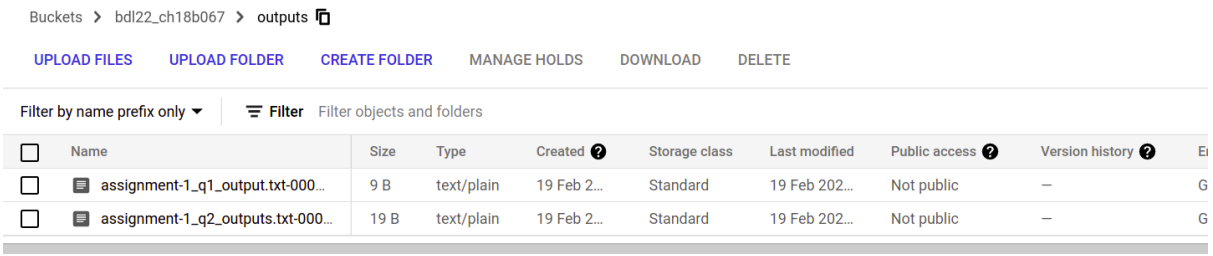
Q2. Write a Python code to get the average number of words in a line of the file placed in the BDL2022 bucket (gs://bdl2022/lines_big.txt) using Dataflow provide the screenshot of the file that is generated in your bucket.

Screenshot of Code

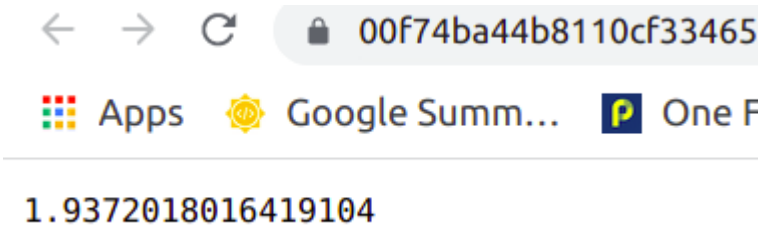
```
 GNU nano 3.2                                  assignment-1_q2.py

import apache_beam as beam
from apache_beam.io import ReadFromText
from apache_beam.io import WriteToText
from apache_beam.options.pipeline_options import PipelineOptions
from apache_beam.options.pipeline_options import GoogleCloudOptions
import re
from apache_beam.options.pipeline_options import StandardOptions
options = PipelineOptions()
google_cloud_options = options.view_as(GoogleCloudOptions)
google_cloud_options.project = 'graphite-byte-260703' # Enter your project ID
google_cloud_options.job_name = 'lab3q2'
google_cloud_options.temp_location = "gs://bdl22_ch18b067/temp"
google_cloud_options.region = "us-central1"
options.view_as(StandardOptions).runner = 'DataflowRunner'
p = beam.Pipeline(options=options)
lines = p | 'Read' >> beam.io.ReadFromText( 'gs://bdl2022/lines_big.txt' ) |'Find Words' >> beam.Map(lambda line: $
result = p.run()
```

Screenshot of file in bucket:

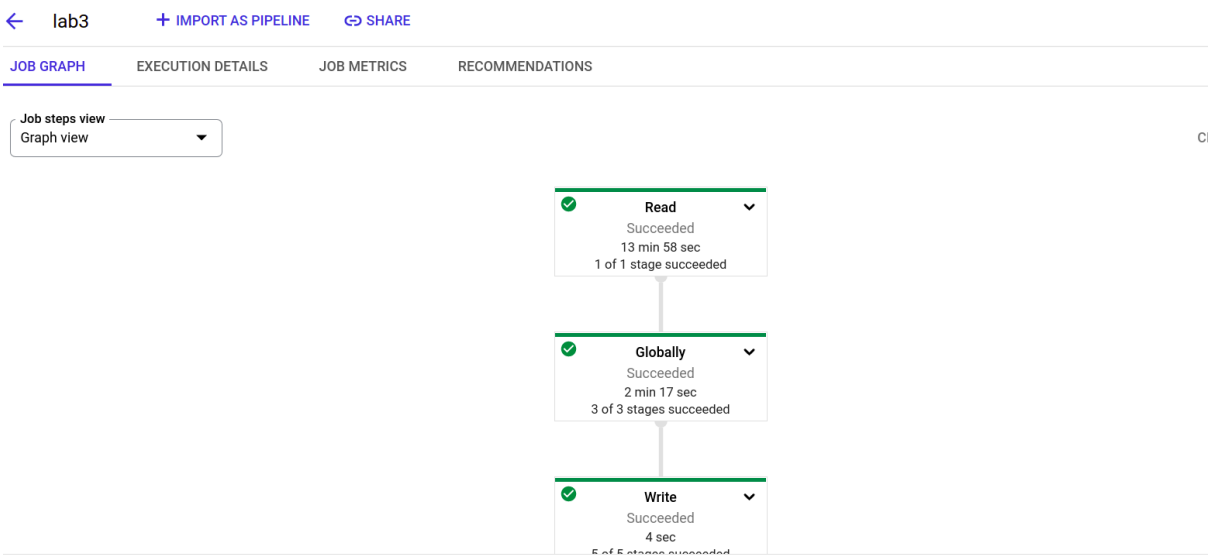


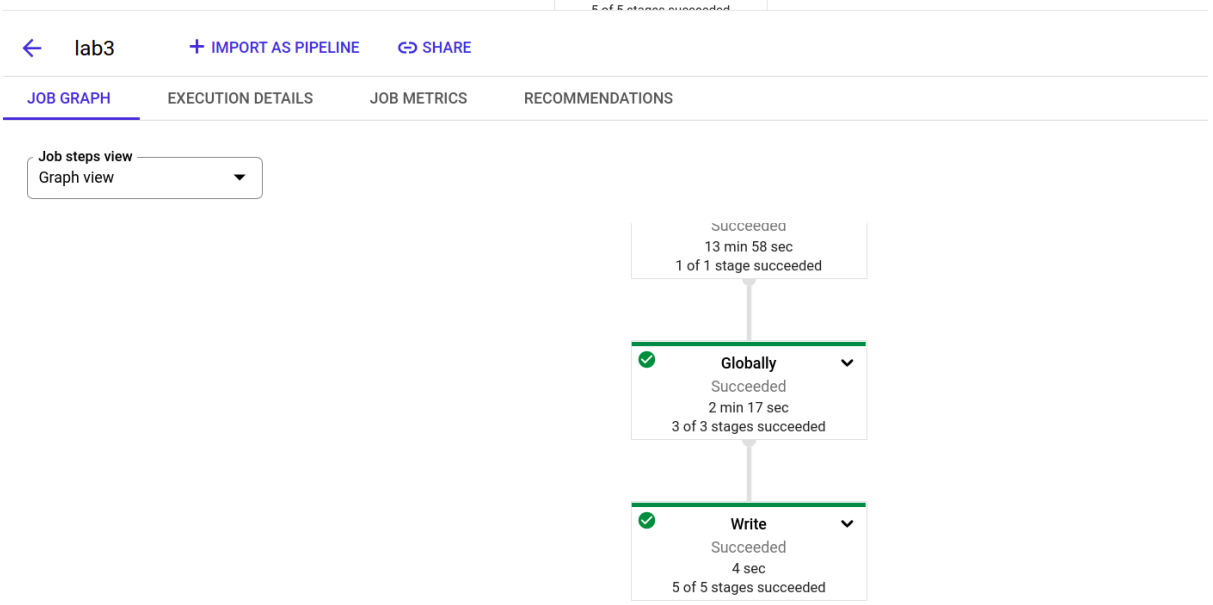
Content of the file:



Q3. Provide the screenshot for the execution graph created by Dataflow in the background for the pipeline object created for questions 1 and 2.

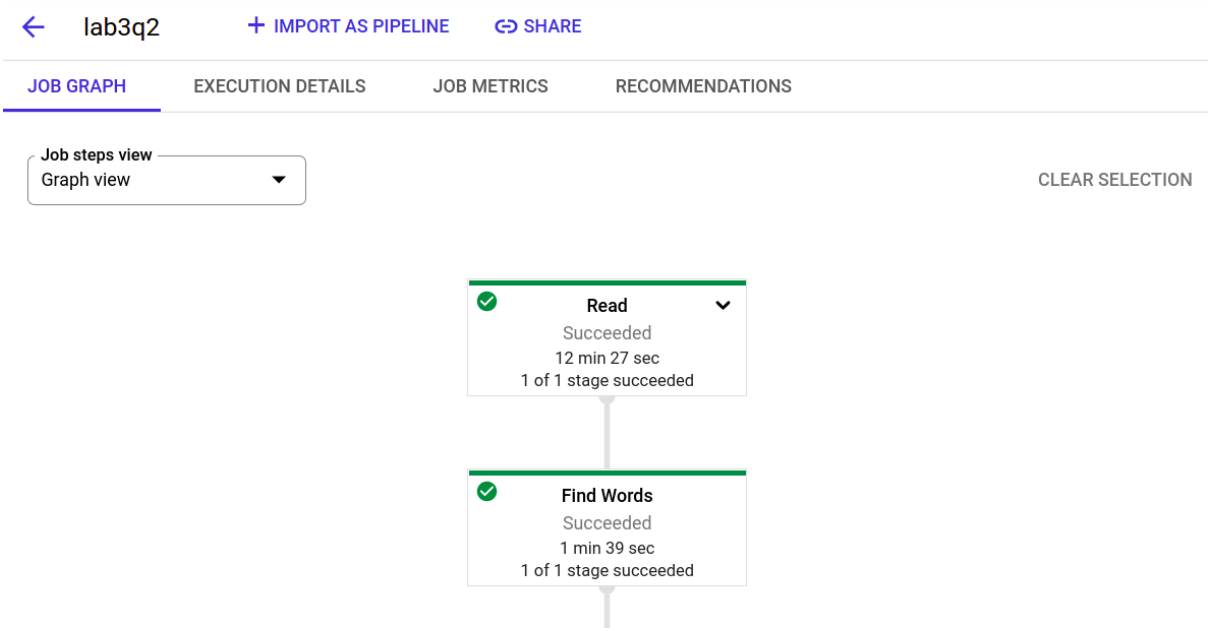
Graph for Question-1



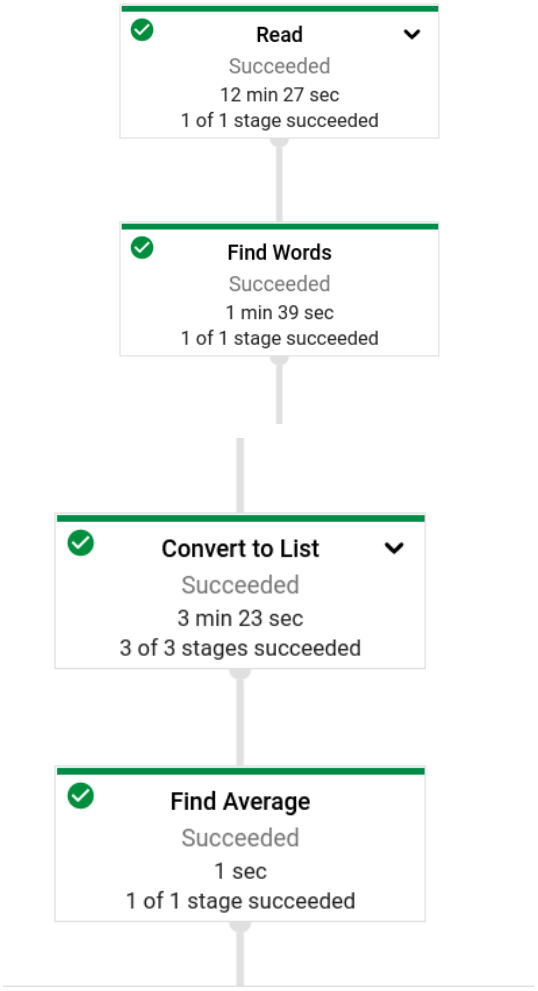


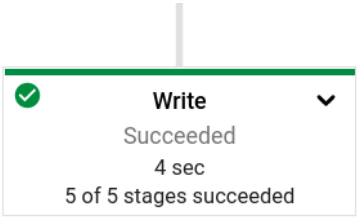
Graph for Question-2

Full Screen View



Full Pipeline View:





Q4. Explain the pipeline used in the first two questions. What issues did you face while trying to make the code work for the first two questions, and how did you resolve them?

In question-1, we first read the big text file using beam.io.ReadFromText, which returns each line in the file separately. Then we count the total number of elements in the file using beam.combiners.Count.Globally(), which returns the count of the number of lines. Finally, we write the output to the output file.
In question-2, we first read the big text file using beam.io.ReadFromText, which returns a list of lines in the file. Then, we split each line by space using beam.Map(lambda line: len(line.split())). These individual outputs are then collated into a list using beam.combiners.ToList(). Finally, we sum the number of words and divide it by the number of lines (length of list) and find the average using beam.Map(lambda x : sum(x) /len(x). Finally we write the output to a file using beam.io.WriteToText()

In the second question it was difficult to find the function to collate individual outputs into a list so that we could apply an anonymous function to find the average. Upon spending sufficient time reading through the documentation I was able to discover the correct function.

Q5. Trigger a dataflow using GCF for any one of the first two questions.

Here, the function for counting lines in a file is deployed.

```
 GNU nano 3.2                                  main.py

def get_line_count(data, context):
        from google.cloud import storage
        file = data['name']
        bucket = client.get_bucket('bdl22_ch18b067')
        blob = bucket.get_blob(file)
        x = blob.download_as_string()
        x = x.decode('utf-8')
        print(len(x.split('\n')))
        return
```

```
way2shania@instance-1:~$ gcloud functions deploy get_line_count --runtime python37 --trigger-resource bdl22_ch18b06
7 --trigger-event google.storage.object.finalize
API [cloudfunctions.googleapis.com] not enabled on project [1075303201840]. Would you like to enable and retry
(this will take a few minutes)? (y/N)?  y
```
