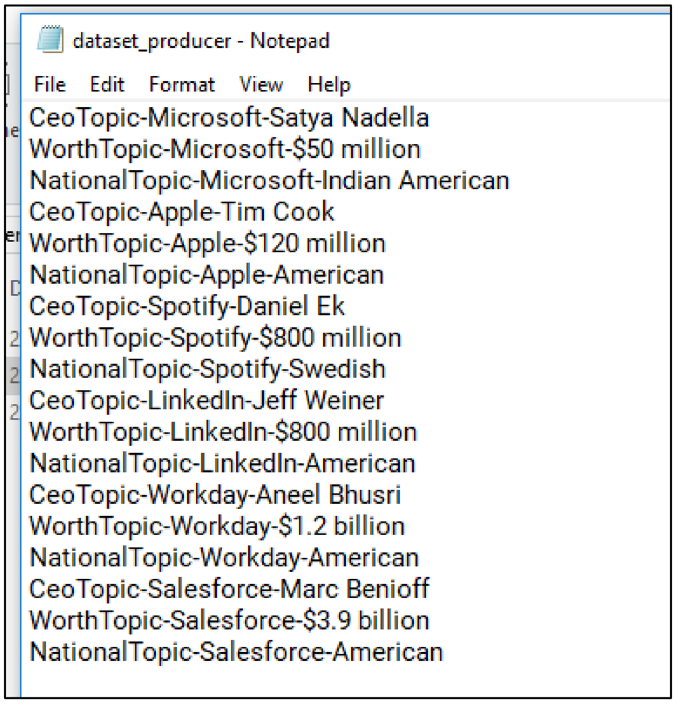
Session 23: More Kafka Assignment 1

TASK-1:

Create a java program MyKafkaProducer.java that takes a file name and delimiter as input arguments.
It should read the content of file line by line.
Fields in the file are in following order
1. Kafka Topic Name
2. Key
3. value
For every line, insert the key and value to the repsective Kafka broker in a fire and forget mode.
After record is sent, it should print appropriate message on screen.
Pass **dataset_producer.txt** as the input file and -as delimiter.
LINK: https://drive.google.com/file/d/0B_Qjau8wv1KoSnR5eHpKOF9rTFU/view?usp=sharing

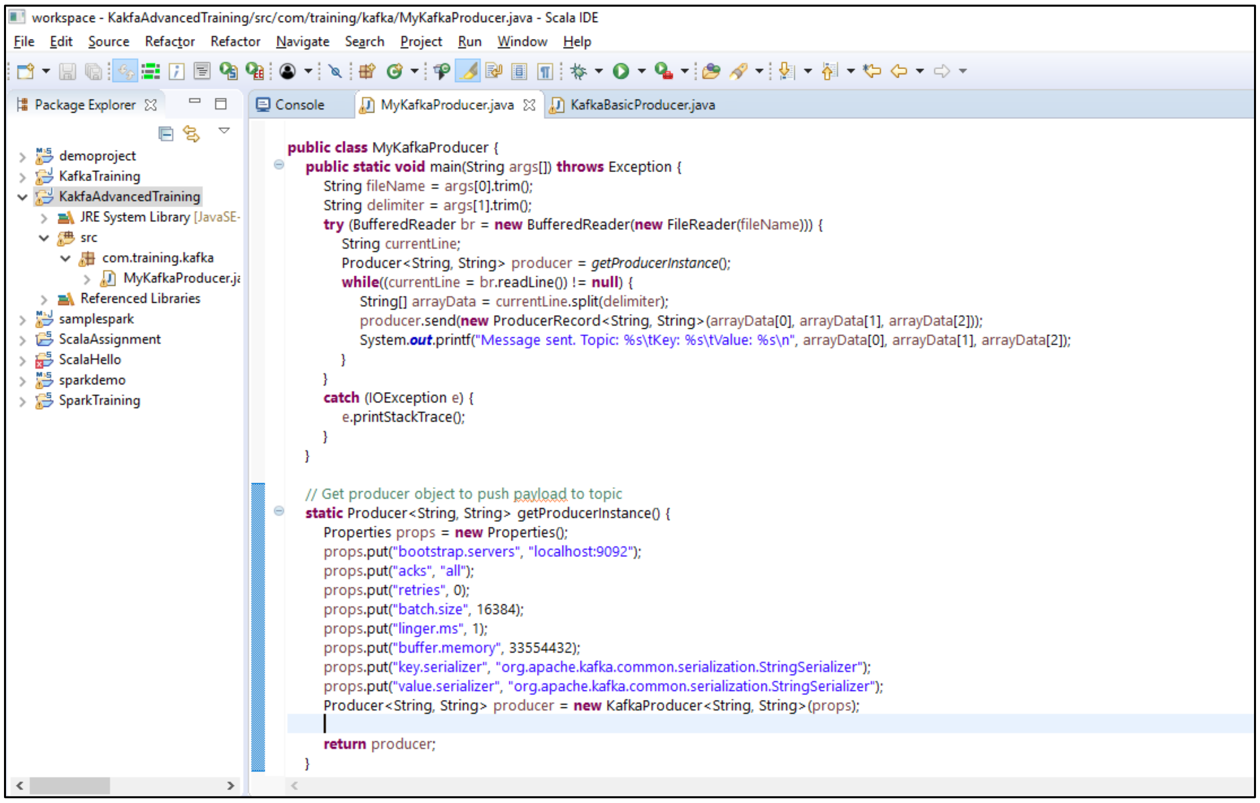
Note: The above input data file path (google drive) given in assignment PDF is invalid. No file exist in the path. So, I have created a sample file for my own.

Input: dataset_producer.txt



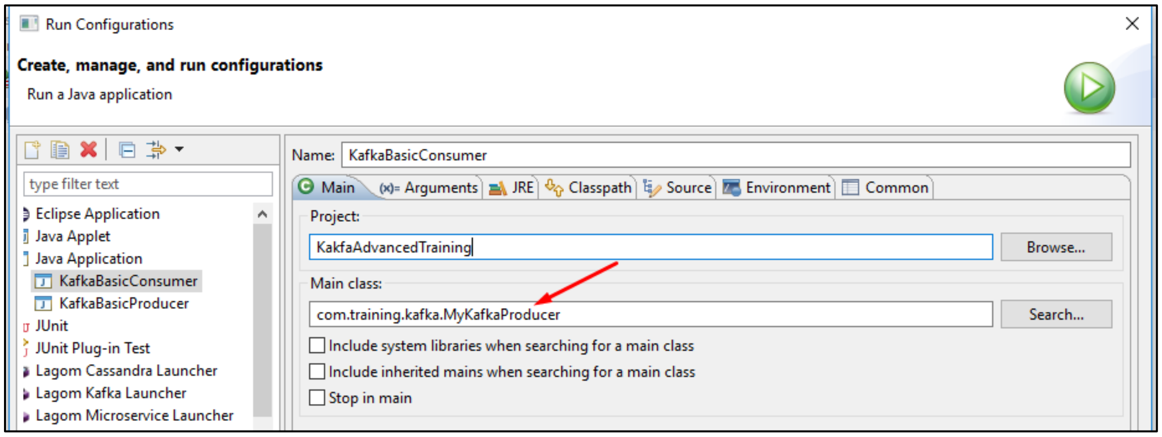
```
CeoTopic-Microsoft-Satya Nadella
WorthTopic-Microsoft-$50 million
NationalTopic-Microsoft-Indian American
CeoTopic-Apple-Tim Cook
WorthTopic-Apple-$120 million
NationalTopic-Apple-American
CeoTopic-Spotify-Daniel Ek
WorthTopic-Spotify-$800 million
NationalTopic-Spotify-Swedish
CeoTopic-LinkedIn-Jeff Weiner
WorthTopic-LinkedIn-$800 million
NationalTopic-LinkedIn-American
CeoTopic-Workday-Aneel Bhusri
WorthTopic-Workday-$1.2 billion
NationalTopic-Workday-American
CeoTopic-Salesforce-Marc Benioff
WorthTopic-Salesforce-$3.9 billion
NationalTopic-Salesforce-American
```

Program:



```
public class MyKafkaProducer {
    public static void main(String args[]) throws Exception {
        String fileName = args[0].trim();
        String delimiter = args[1].trim();
        try (BufferedReader br = new BufferedReader(new FileReader(fileName))) {
            String currentLine;
            Producer<String, String> producer = getProducerInstance();
            while((currentLine = br.readLine()) != null) {
                String[] arrayData = currentLine.split(delimiter);
                producer.send(new ProducerRecord<String, String>(arrayData[0], arrayData[1], arrayData[2]));
                System.out.printf("Message sent. Topic: %s\tKey: %s\tValue: %s\n", arrayData[0], arrayData[1], arrayData[2]);
            }
        }
        catch (IOException e) {
            e.printStackTrace();
        }
    }

    // Get producer object to push payload to topic
    static Producer<String, String> getProducerInstance() {
        Properties props = new Properties();
        props.put("bootstrap.servers", "localhost:9092");
        props.put("acks", "all");
        props.put("retries", 0);
        props.put("batch.size", 16384);
        props.put("linger.ms", 1);
        props.put("buffer.memory", 33554432);
        props.put("key.serializer", "org.apache.kafka.common.serialization.StringSerializer");
        props.put("value.serializer", "org.apache.kafka.common.serialization.StringSerializer");
        Producer<String, String> producer = new KafkaProducer<String, String>(props);

        return producer;
    }
```

Execution:



Create, manage, and run configurations

Run a Java application

Name: KafkaBasicConsumer

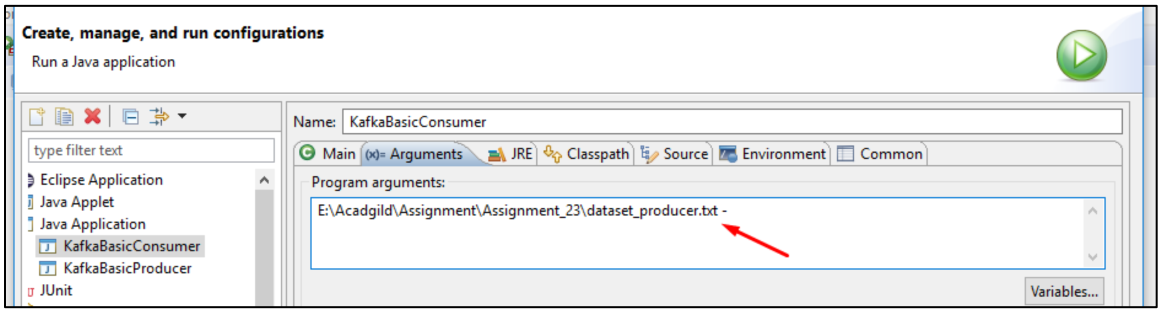
Project: KafkaAdvancedTraining

Main class: com.training.kafka.MyKafkaProducer

☐ Include system libraries when searching for a main class

☐ Include inherited mains when searching for a main class

☐ Stop in main



Create, manage, and run configurations

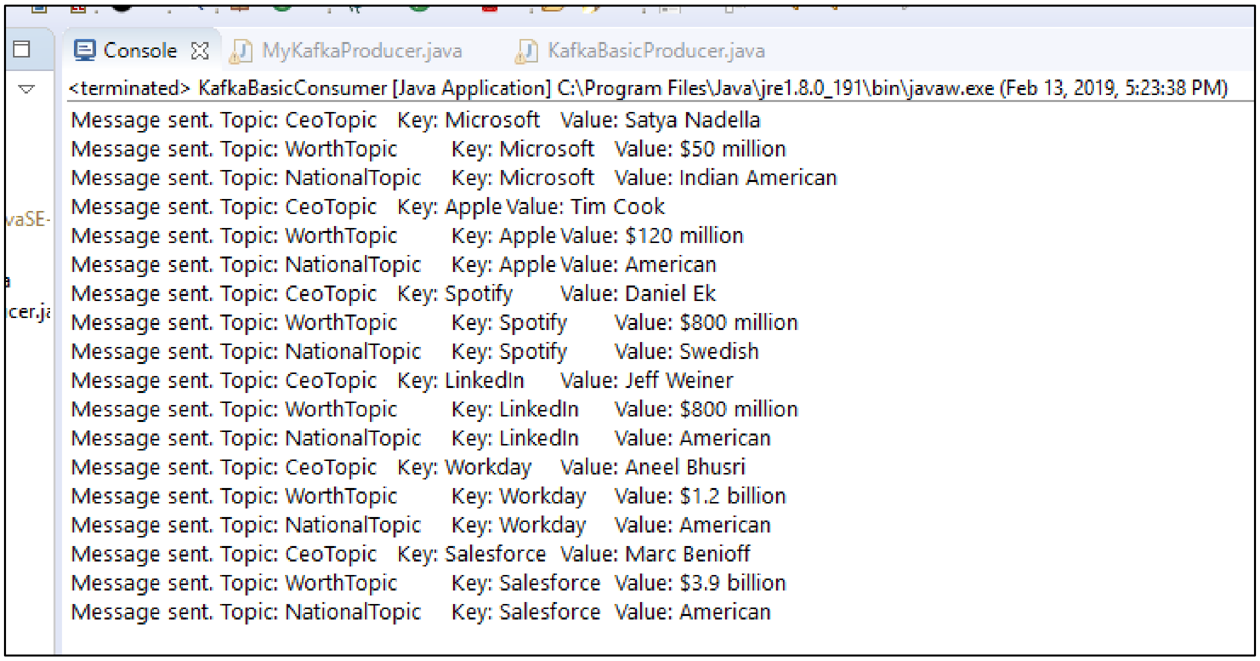
Run a Java application

Name: KafkaBasicConsumer

Program arguments:

E:\Acadgild\Assignment\Assignment_23\dataset_producer.txt -

Execution Output:



```
<terminated> KafkaBasicConsumer [Java Application] C:\Program Files\Java\jre1.8.0_191\bin\javaw.exe (Feb 13, 2019, 5:23:38 PM)
Message sent. Topic: CeoTopic   Key: Microsoft   Value: Satya Nadella
Message sent. Topic: WorthTopic      Key: Microsoft   Value: $50 million
Message sent. Topic: NationalTopic   Key: Microsoft   Value: Indian American
Message sent. Topic: CeoTopic   Key: Apple Value: Tim Cook
Message sent. Topic: WorthTopic      Key: Apple Value: $120 million
Message sent. Topic: NationalTopic   Key: Apple Value: American
Message sent. Topic: CeoTopic   Key: Spotify     Value: Daniel Ek
Message sent. Topic: WorthTopic      Key: Spotify     Value: $800 million
Message sent. Topic: NationalTopic   Key: Spotify     Value: Swedish
Message sent. Topic: CeoTopic   Key: LinkedIn    Value: Jeff Weiner
Message sent. Topic: WorthTopic      Key: LinkedIn    Value: $800 million
Message sent. Topic: NationalTopic   Key: LinkedIn    Value: American
Message sent. Topic: CeoTopic   Key: Workday     Value: Aneel Bhusri
Message sent. Topic: WorthTopic      Key: Workday     Value: $1.2 billion
Message sent. Topic: NationalTopic   Key: Workday     Value: American
Message sent. Topic: CeoTopic   Key: Salesforce  Value: Marc Benioff
Message sent. Topic: WorthTopic      Key: Salesforce  Value: $3.9 billion
Message sent. Topic: NationalTopic   Key: Salesforce  Value: American
```

As per the input file, there are 3 topics involved in publishing messages,

- CeoTopic
- WorthTopic
- NationalTopic

All 3 topics were subscribed and queried to see the messages. All the topics were displaying messages accordingly,

CeoTopic:

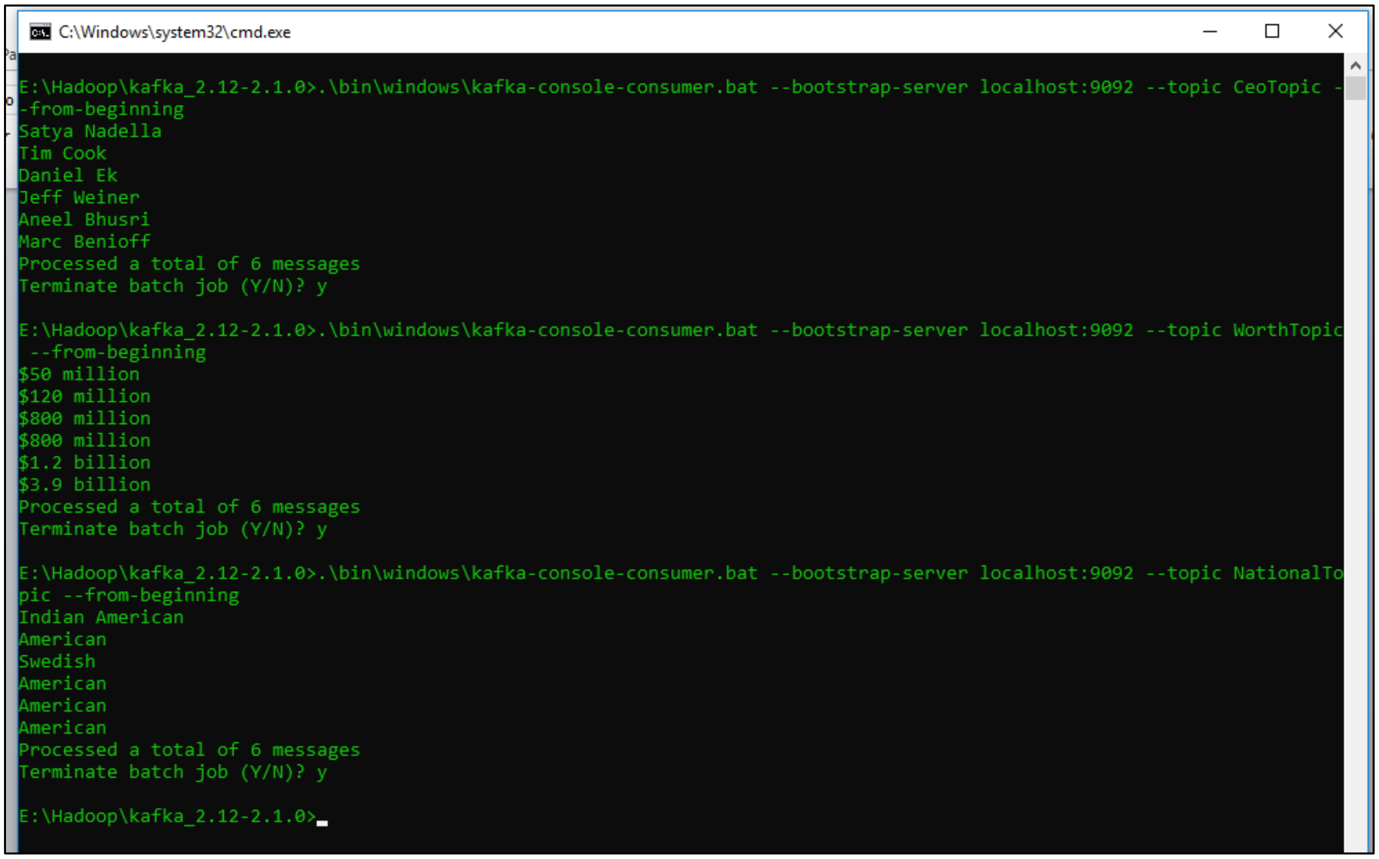
.\bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic CeoTopic --from-beginning

WorthTopic:

.\bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic WorthTopic --from-beginning

NationalTopic:

.\bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic NationalTopic --from-beginning


C:\Windows\system32\cmd.exe

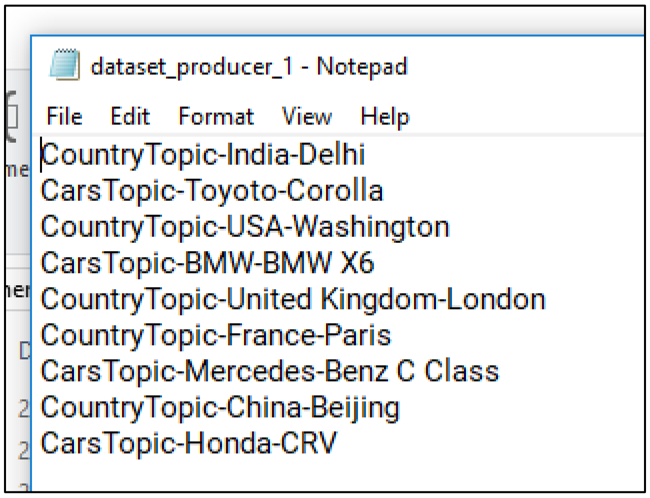
```
E:\Hadoop\kafka_2.12-2.1.0>.\bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic CeoTopic -
-from-beginning
Satya Nadella
Tim Cook
Daniel Ek
Jeff Weiner
Aneel Bhusri
Marc Benioff
Processed a total of 6 messages
Terminate batch job (Y/N)? y

E:\Hadoop\kafka_2.12-2.1.0>.\bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic WorthTopic
 --from-beginning
$50 million
$120 million
$800 million
$800 million
$1.2 billion
$3.9 billion
Processed a total of 6 messages
Terminate batch job (Y/N)? y

E:\Hadoop\kafka_2.12-2.1.0>.\bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic NationalTo
pic --from-beginning
Indian American
American
Swedish
American
American
American
Processed a total of 6 messages
Terminate batch job (Y/N)? y

E:\Hadoop\kafka_2.12-2.1.0>_
```

TASK-2:

Modify the previous program MyKafkaProducer.java and create a new Java program KafkaProducerWithAck.java
This should perform the same task as of KafkaProducer.java with some modification.
When passing any data to a topic, it should wait for acknowledgement.
After acknowledgement is received from the broker, it should print the key and value which has been written to a specified topic.
The application should attempt for 3 retries before giving any exception.
Pass **dataset_producer.txt** as the input file and -as delimiter.

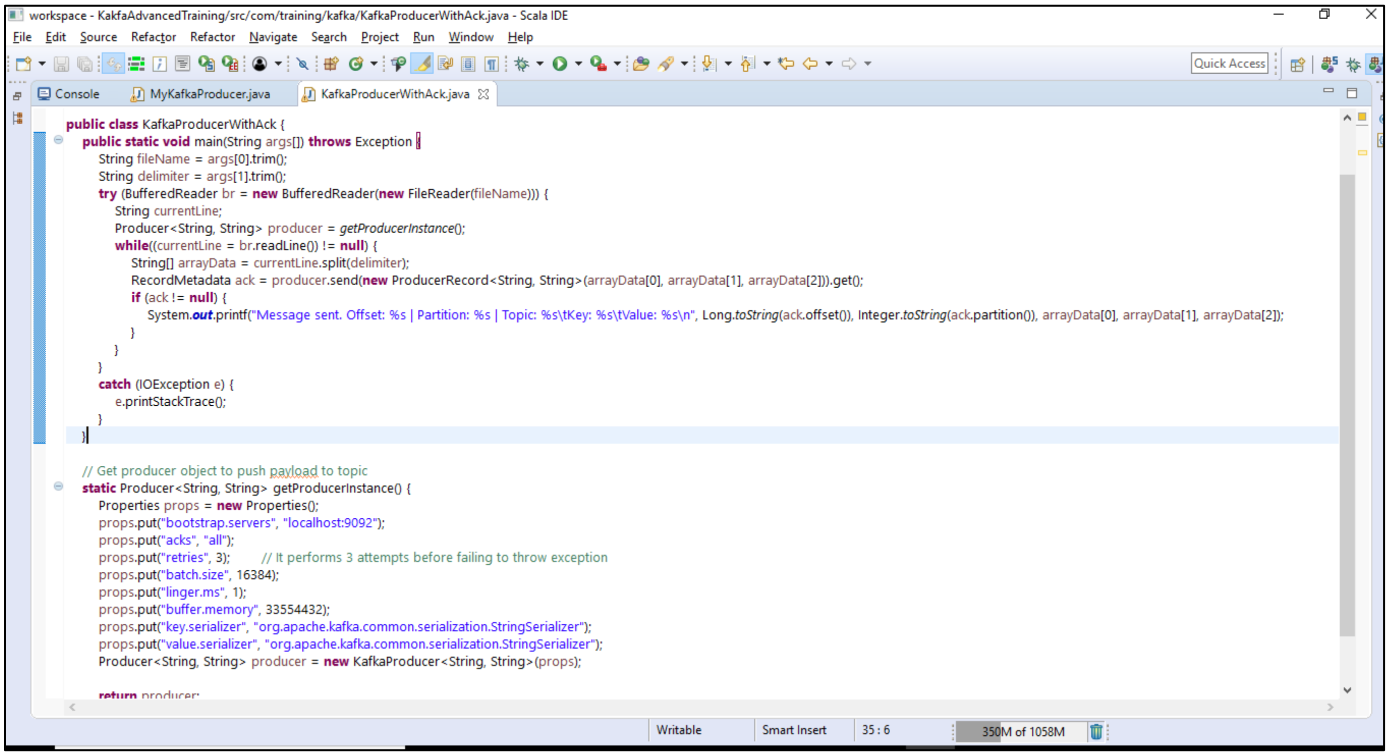
Note: The input data file path (google drive) given in assignment PDF is invalid. No file exist in the path. So, I have created a sample file for my own.

Input: dataset_producer_1.txt



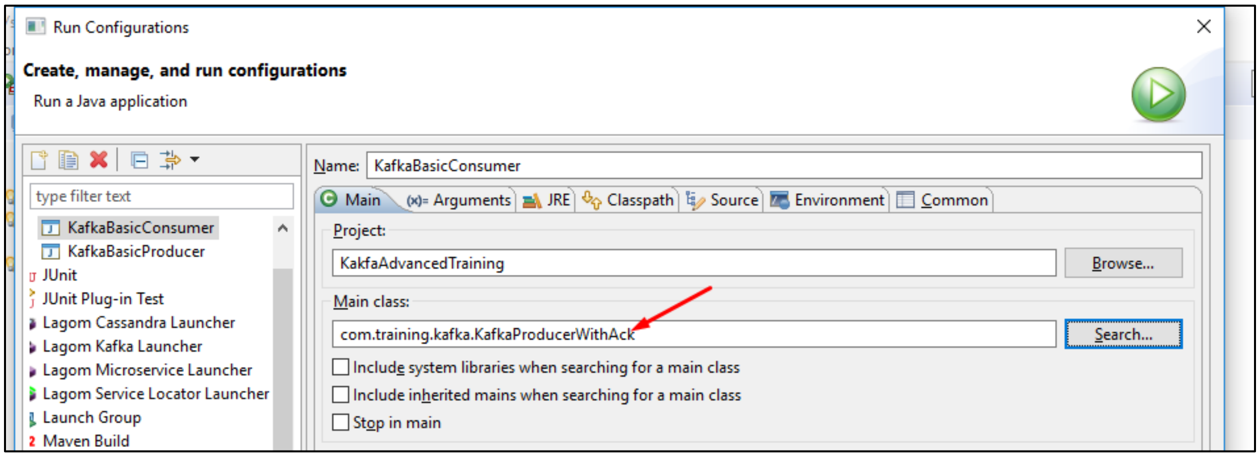
```
CountryTopic-India-Delhi
CarsTopic-Toyoto-Corolla
CountryTopic-USA-Washington
CarsTopic-BMW-BMW X6
CountryTopic-United Kingdom-London
CountryTopic-France-Paris
CarsTopic-Mercedes-Benz C Class
CountryTopic-China-Beijing
CarsTopic-Honda-CRV
```

Program:


workspace - KakfaAdvancedTraining/src/com/training/kafka/KafkaProducerWithAck.java - Scala IDE
File Edit Source Refactor Refactor Navigate Search Project Run Window Help
Quick Access
Console
MyKafkaProducer.java
KafkaProducerWithAck.java
Writable
Smart Insert
35 : 6
350M of 1058M

```java
public class KafkaProducerWithAck {
  public static void main(String args[]) throws Exception {
    String fileName = args[0].trim();
    String delimiter = args[1].trim();
    try (BufferedReader br = new BufferedReader(new FileReader(fileName))) {
      String currentLine;
      Producer<String, String> producer = getProducerInstance();
      while((currentLine = br.readLine()) != null) {
        String[] arrayData = currentLine.split(delimiter);
        RecordMetadata ack = producer.send(new ProducerRecord<String, String>(arrayData[0], arrayData[1], arrayData[2])).get();
        if (ack != null) {
          System.out.printf("Message sent. Offset: %s | Partition: %s | Topic: %s\tKey: %s\tValue: %s\n", Long.toString(ack.offset()), Integer.toString(ack.partition()), arrayData[0], arrayData[1], arrayData[2]);
        }
      }
    }
    catch (IOException e) {
      e.printStackTrace();
    }
  }

  // Get producer object to push payload to topic
  static Producer<String, String> getProducerInstance() {
    Properties props = new Properties();
    props.put("bootstrap.servers", "localhost:9092");
    props.put("acks", "all");
    props.put("retries", 3);       // It performs 3 attempts before failing to throw exception
    props.put("batch.size", 16384);
    props.put("linger.ms", 1);
    props.put("buffer.memory", 33554432);
    props.put("key.serializer", "org.apache.kafka.common.serialization.StringSerializer");
    props.put("value.serializer", "org.apache.kafka.common.serialization.StringSerializer");
    Producer<String, String> producer = new KafkaProducer<String, String>(props);

    return producer;
```

Console application argument settings:



Run Configurations

Create, manage, and run configurations

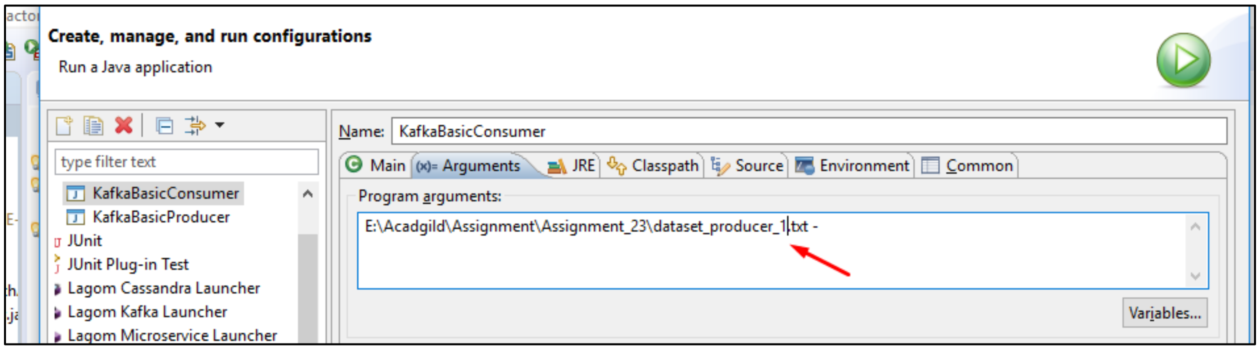
Run a Java application

Name: KafkaBasicConsumer

Project: KakfaAdvancedTraining

Main class: com.training.kafka.KafkaProducerWithAck

☐ Include system libraries when searching for a main class
☐ Include inherited mains when searching for a main class
☐ Stop in main

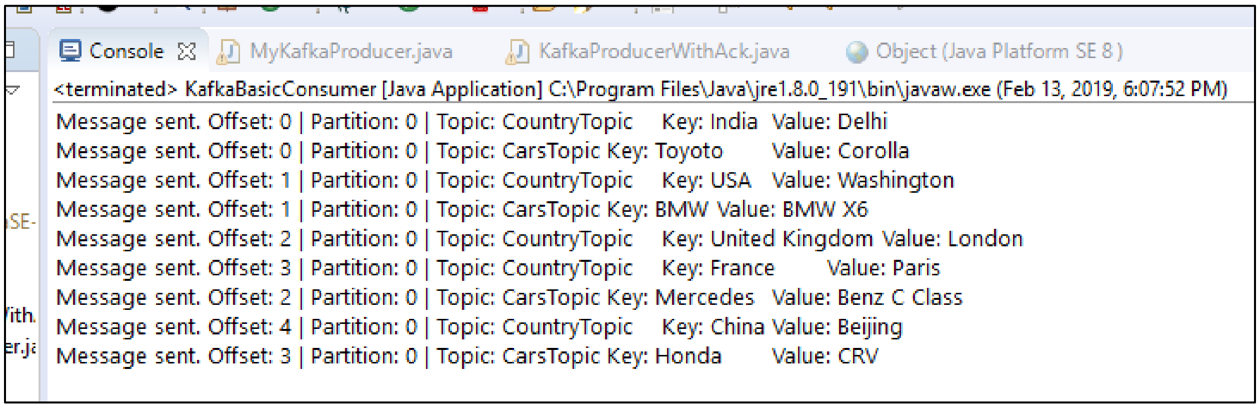


Name: KafkaBasicConsumer

Program arguments:

```
E:\Acadgild\Assignment\Assignment_23\dataset_producer_1.txt -
```

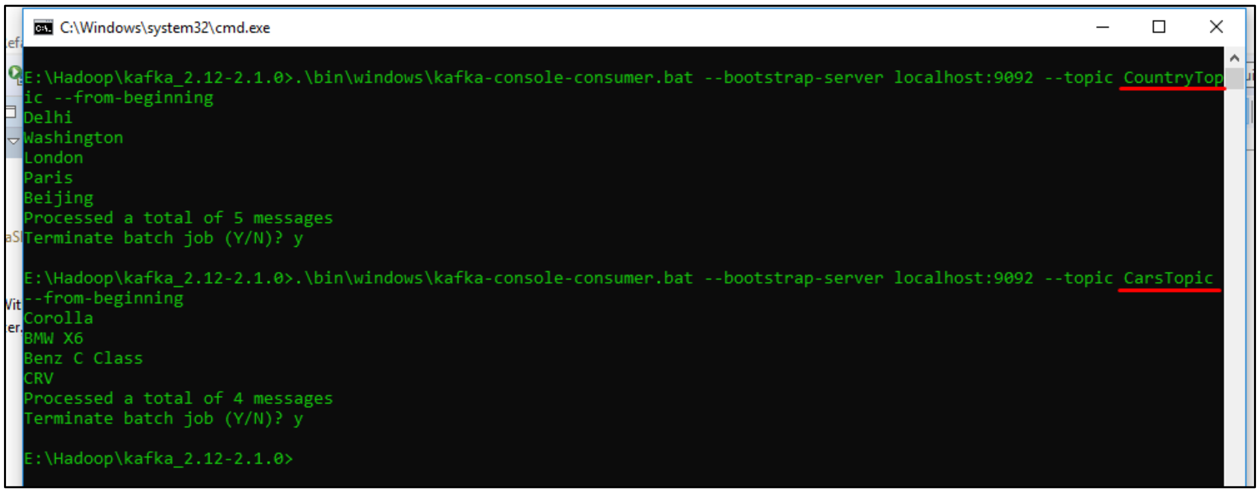
Execution Result:



```
<terminated> KafkaBasicConsumer [Java Application] C:\Program Files\Java\jre1.8.0_191\bin\javaw.exe (Feb 13, 2019, 6:07:52 PM)
Message sent. Offset: 0 | Partition: 0 | Topic: CountryTopic    Key: India  Value: Delhi
Message sent. Offset: 0 | Partition: 0 | Topic: CarsTopic Key: Toyoto      Value: Corolla
Message sent. Offset: 1 | Partition: 0 | Topic: CountryTopic    Key: USA   Value: Washington
Message sent. Offset: 1 | Partition: 0 | Topic: CarsTopic Key: BMW  Value: BMW X6
Message sent. Offset: 2 | Partition: 0 | Topic: CountryTopic    Key: United Kingdom  Value: London
Message sent. Offset: 3 | Partition: 0 | Topic: CountryTopic    Key: France        Value: Paris
Message sent. Offset: 2 | Partition: 0 | Topic: CarsTopic Key: Mercedes   Value: Benz C Class
Message sent. Offset: 4 | Partition: 0 | Topic: CountryTopic    Key: China Value: Beijing
Message sent. Offset: 3 | Partition: 0 | Topic: CarsTopic Key: Honda      Value: CRV
```

Final Result:

Both topics (CountryTopic and CarsTopic) are subscribed and queried. The published messages are listed in it.



```
E:\Hadoop\kafka_2.12-2.1.0>.\bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic CountryTop
ic --from-beginning
Delhi
Washington
London
Paris
Beijing
Processed a total of 5 messages
Terminate batch job (Y/N)? y

E:\Hadoop\kafka_2.12-2.1.0>.\bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic CarsTopic
--from-beginning
Corolla
BMW X6
Benz C Class
CRV
Processed a total of 4 messages
Terminate batch job (Y/N)? y

E:\Hadoop\kafka_2.12-2.1.0>
```