In [ ]:

```
from pyspark.sql import *
from pyspark.sql.functions import *
from pyspark import SparkContext
from pyspark.sql import SQLContext
import pandas as pd

# create the Spark Session
spark = SparkSession.builder.getOrCreate()

# create the Spark Context
sc = spark.sparkContext

sqlContext = SQLContext(sc)
rddCovid = sqlContext.read.csv('tests.csv', header=True).rdd.cache()
rddLocalidades = sqlContext.read.csv('localidades.csv', header=True).rdd.cache()
```

In [ ]:

```
#Ejercicio A
covidEnero = rddCovid.filter(lambda x: x[0] > "2020-12-31" and x[0] < "2021-02-01").map(lambda x: (x
[2], 1))
covidFebrero = rddCovid.filter(lambda x: x[0] > "2021-01-31" and x[0] < "2021-03-01").map(lambda x:
(x[2], 1))
covidMarzo = rddCovid.filter(lambda x: x[0] > "2021-02-28" and x[0] < "2021-04-01").map(lambda x: (x
[2], 1))

rddLocalidades = rddLocalidades.map(lambda x: (x[0],x[2]))

covidEnero = covidEnero.join(rddLocalidades).map(lambda x: (x[1][1], x[1][0])).reduceByKey(lambda a,
b: a+b)
covidFebrero = covidFebrero.join(rddLocalidades).map(lambda x: (x[1][1], x[1]
[0])).reduceByKey(lambda a,b: a+b)
covidMarzo = covidMarzo.join(rddLocalidades).map(lambda x: (x[1][1], x[1][0])).reduceByKey(lambda a,
b: a+b)

#Deberia antes del map llenar los "none" pero desconozco la funcion para eso.
eneroFebrero = covidEnero.fullOuterJoin(covidFebrero).map(lambda x: (x[0], 1 if ((x[1][1] * 100) / x
[1][0]) > 119 else 0))
febreroMarzo = covidFebrero.fullOuterJoin(covidMarzo).map(lambda x: (x[0], 1 if ((x[1][1] * 100) / x
[1][0]) > 119 else 0))

eneroFebrero.fullOuterJoin(febreroMarzo).filter(lambda x: (x[1][0] == 1 or x[1][1] == 1)).map(lambda
x: x[0]).collect()
```

In [ ]:

```
#Ejercicio B
promedioDeTest = rddCovid.filter(lambda x: x[0] > "2020/12/31" and x[0] < "2021/04/01").count()
promedioDeTest = promedioDeTest / rddLocalidades.count()

EjercicioB = rddCovid.map(lambda x: (x[2], (1, 1 if x[3] == 'positivo' else 0, 1 if x[3] == 'negativ
o' else 0))).reduceByKey(lambda a,b: a+b)
EjercicioB = EjercicioB.filter(lambda x: x[1][0] > (promedioDeTest * 0.3)).map(lambda x: (x[0], (x[1
][1] / x[1][2]))).reduce(lambda a,b: a if a[1] > b[1] else b)
EjercicioB.join(rddLocalidades).map(lambda x: x[1][1]).collect() #(id localidad, (proporcion, nombr
e, provincia))
```