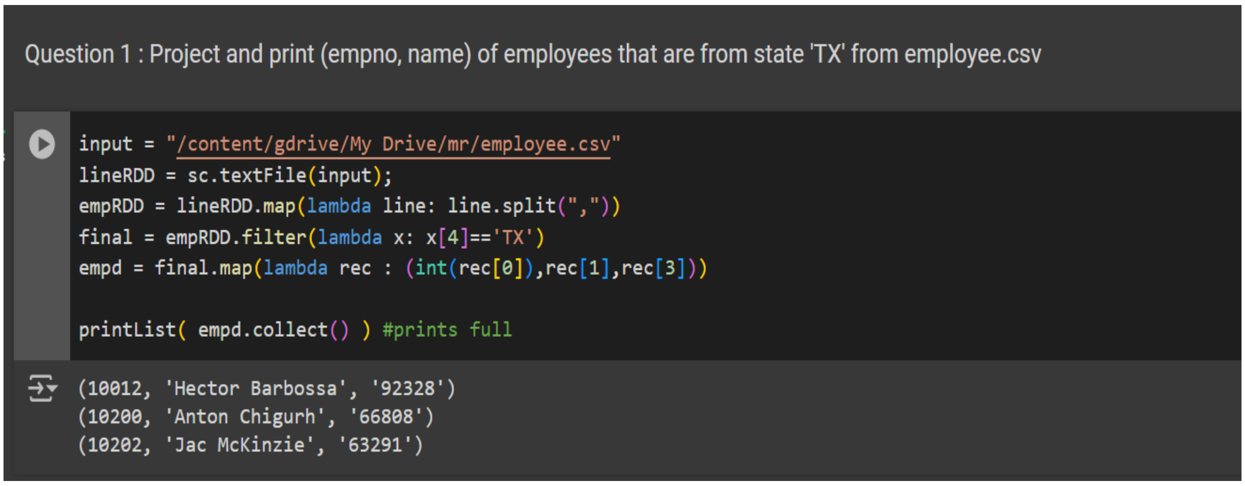
Big Data Processing Lab 3

Roll Number: 202201320
Google Colab Link:
https://colab.research.google.com/drive/1Tw2AvdEKIfNESvqe8mp7eF1PHnuu_Vwa?usp=sharing

Question 1: Project and print (empno, name) of employees that are from state 'TX' from employee.csv.
Code and Output:

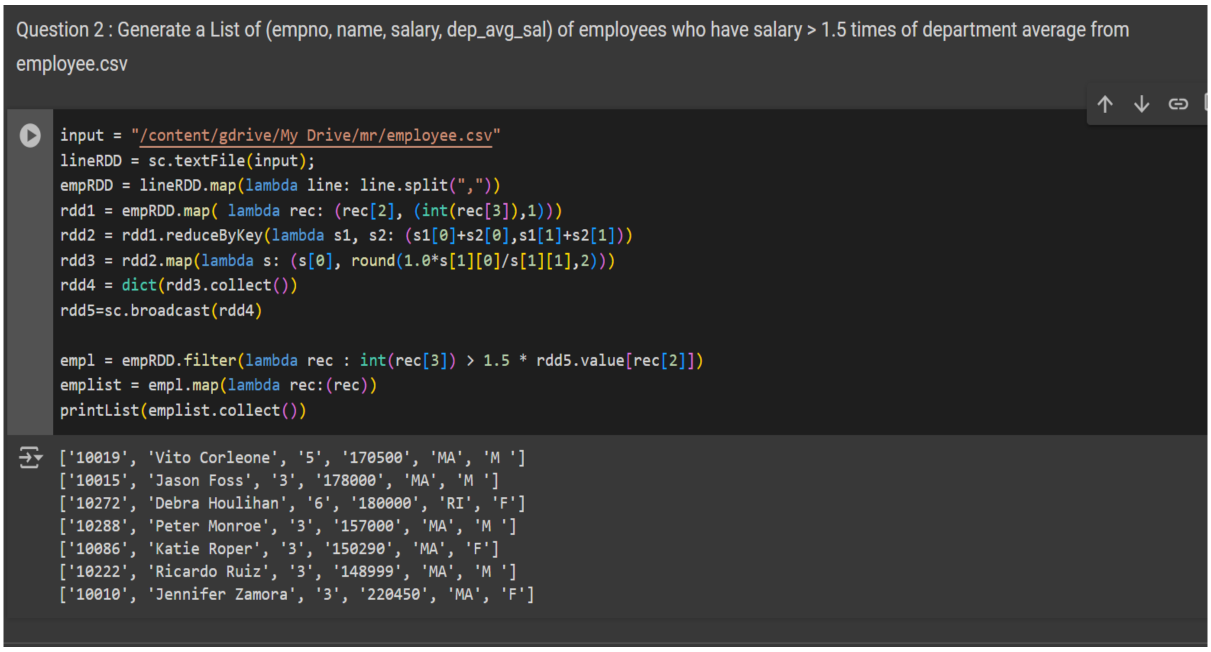


Question 1 : Project and print (empno, name) of employees that are from state 'TX' from employee.csv

```
input = "/content/gdrive/My Drive/mr/employee.csv"
lineRDD = sc.textFile(input);
empRDD = lineRDD.map(lambda line: line.split(","))
final = empRDD.filter(lambda x: x[4]=='TX')
empd = final.map(lambda rec : (int(rec[0]),rec[1],rec[3]))

printList( empd.collect() ) #prints full
```

```
(10012, 'Hector Barbossa', '92328')
(10200, 'Anton Chigurh', '66808')
(10202, 'Jac McKinzie', '63291')
```

Question 2: Generate a List of (empno, name, salary, dep_avg_sal) of employees who have salary > 1.5 times of department average.
Code and Output:



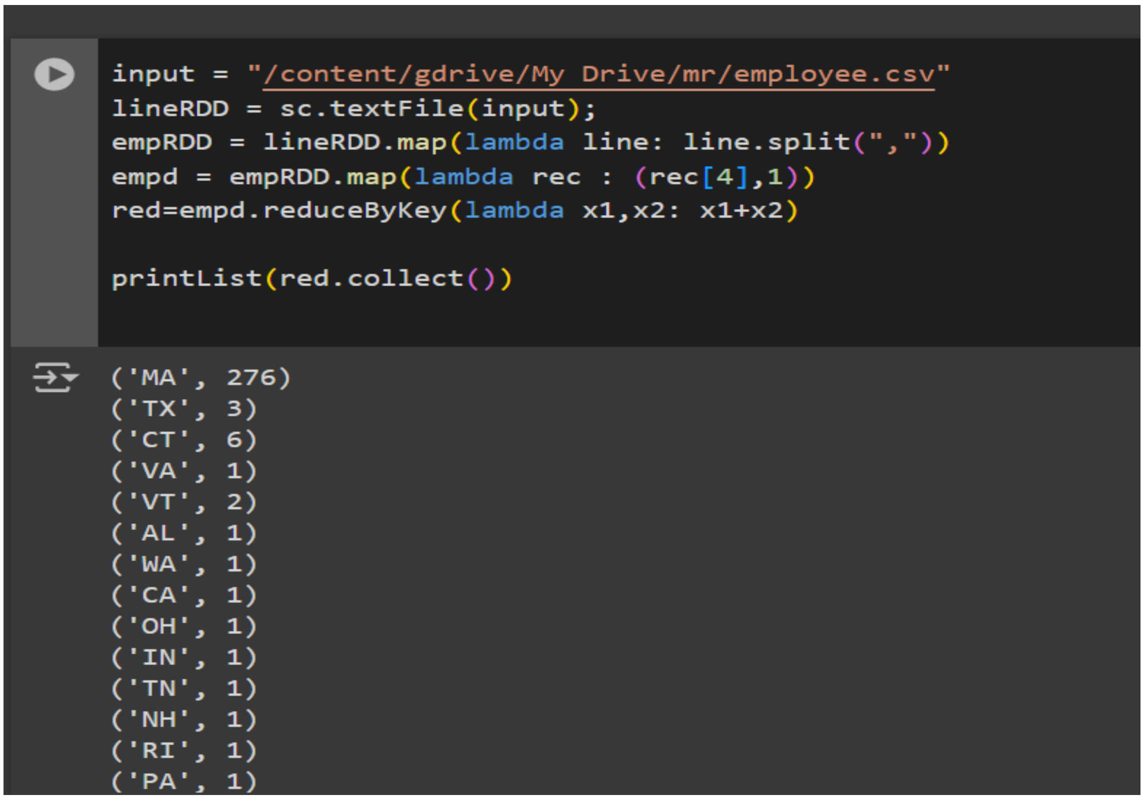
Question 2 : Generate a List of (empno, name, salary, dep_avg_sal) of employees who have salary > 1.5 times of department average from employee.csv

```
input = "/content/gdrive/My Drive/mr/employee.csv"
lineRDD = sc.textFile(input);
empRDD = lineRDD.map(lambda line: line.split(","))
rdd1 = empRDD.map( lambda rec: (rec[2], (int(rec[3]),1)))
rdd2 = rdd1.reduceByKey(lambda s1, s2: (s1[0]+s2[0],s1[1]+s2[1]))
rdd3 = rdd2.map(lambda s: (s[0], round(1.0*s[1][0]/s[1][1],2)))
rdd4 = dict(rdd3.collect())
rdd5=sc.broadcast(rdd4)

empl = empRDD.filter(lambda rec : int(rec[3]) > 1.5 * rdd5.value[rec[2]])
emplist = empl.map(lambda rec:(rec))
printList(emplist.collect())
```

```
['10019', 'Vito Corleone', '5', '170500', 'MA', 'M ']
['10015', 'Jason Foss', '3', '178000', 'MA', 'M ']
['10272', 'Debra Houlihan', '6', '180000', 'RI', 'F']
['10288', 'Peter Monroe', '3', '157000', 'MA', 'M ']
['10086', 'Katie Roper', '3', '150290', 'MA', 'F']
['10222', 'Ricardo Ruiz', '3', '148999', 'MA', 'M ']
['10010', 'Jennifer Zamora', '3', '220450', 'MA', 'F']
```

Question 3: Compute state-wise count of employees from employee.csv

Code and Output:

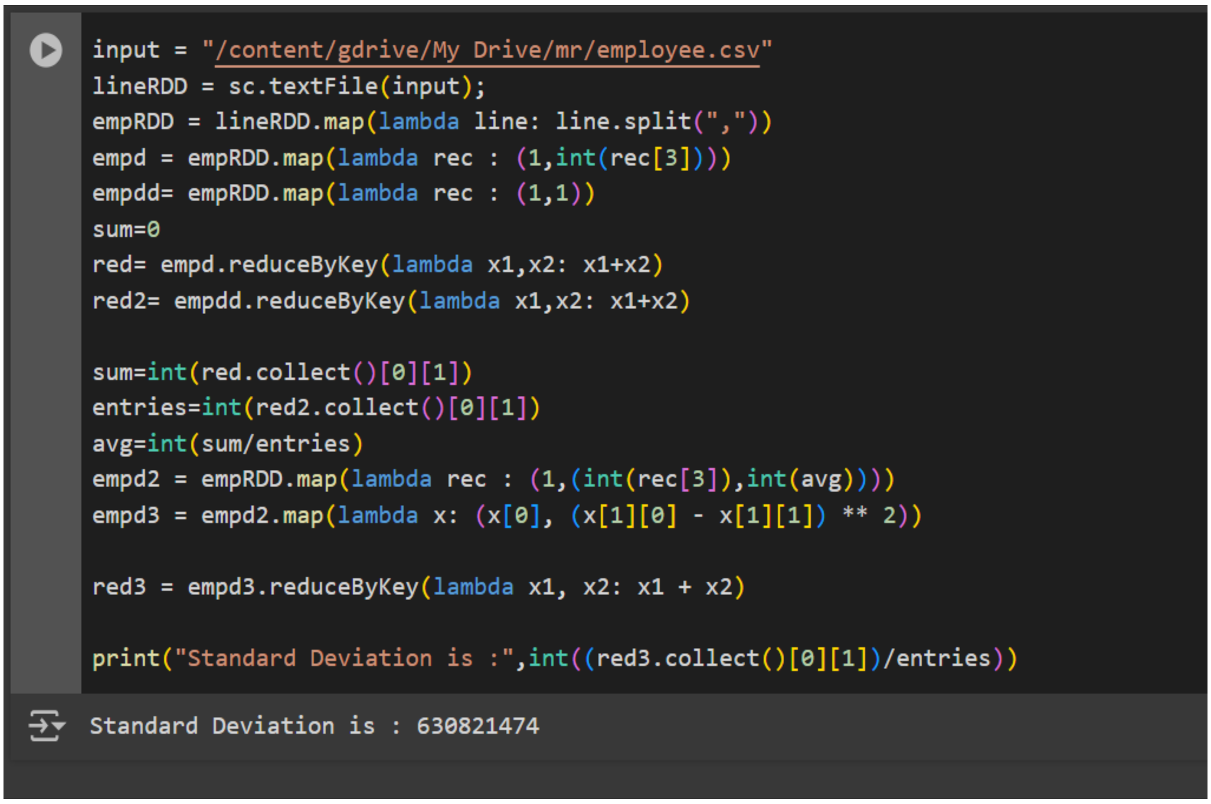


```
input = "/content/gdrive/My Drive/mr/employee.csv"
lineRDD = sc.textFile(input);
empRDD = lineRDD.map(lambda line: line.split(","))
empd = empRDD.map(lambda rec : (rec[4],1))
red=empd.reduceByKey(lambda x1,x2: x1+x2)

printList(red.collect())
```

```
('MA', 276)
('TX', 3)
('CT', 6)
('VA', 1)
('VT', 2)
('AL', 1)
('WA', 1)
('CA', 1)
('OH', 1)
('IN', 1)
('TN', 1)
('NH', 1)
('RI', 1)
('PA', 1)
```

Question 4: Compute the Standard Deviation of salary

Code and Output:

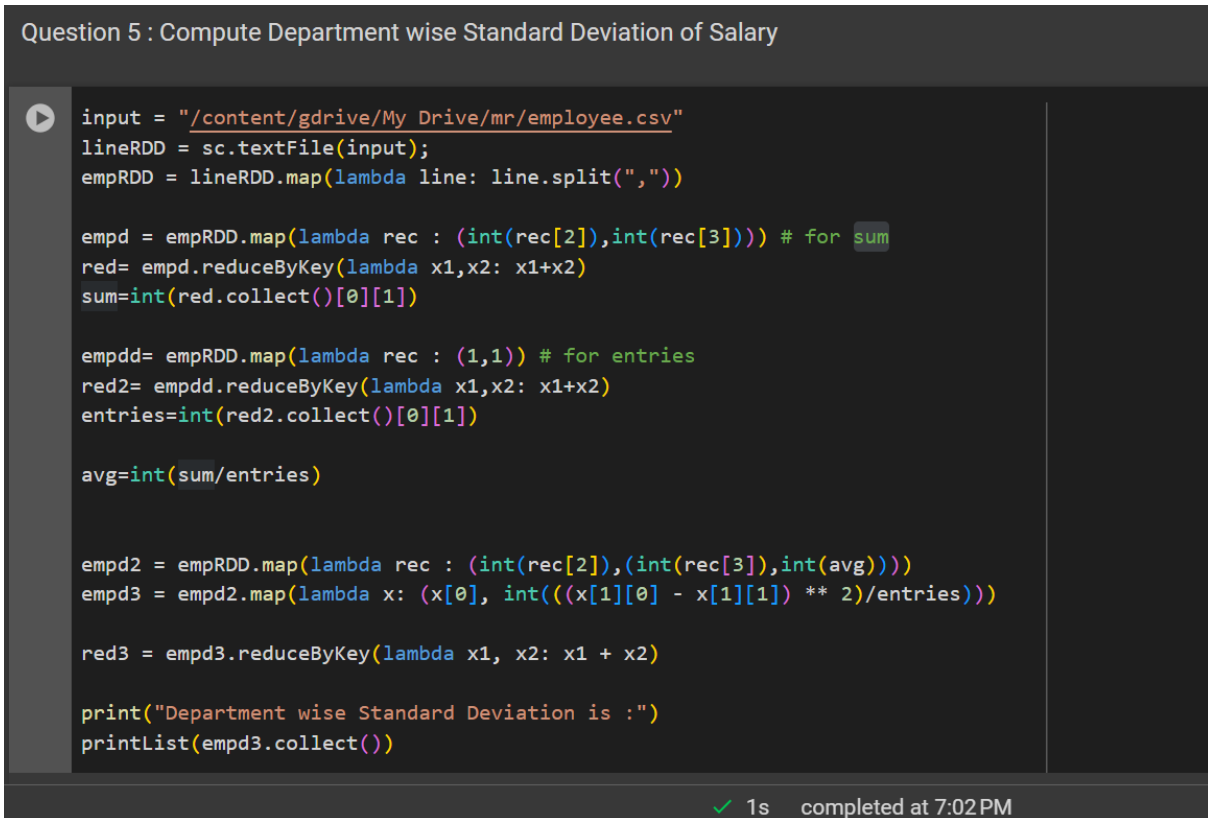


```
input = "/content/gdrive/My Drive/mr/employee.csv"
lineRDD = sc.textFile(input);
empRDD = lineRDD.map(lambda line: line.split(","))
empd = empRDD.map(lambda rec : (1,int(rec[3])))
empdd= empRDD.map(lambda rec : (1,1))
sum=0
red= empd.reduceByKey(lambda x1,x2: x1+x2)
red2= empdd.reduceByKey(lambda x1,x2: x1+x2)

sum=int(red.collect()[0][1])
entries=int(red2.collect()[0][1])
avg=int(sum/entries)
empd2 = empRDD.map(lambda rec : (1,(int(rec[3]),int(avg))))
empd3 = empd2.map(lambda x: (x[0], (x[1][0] - x[1][1]) ** 2))

red3 = empd3.reduceByKey(lambda x1, x2: x1 + x2)

print("Standard Deviation is :",int((red3.collect()[0][1])/entries))
```

```
Standard Deviation is : 630821474
```

Question 5: Compute Department wise Standard Deviation of Salary

Code:

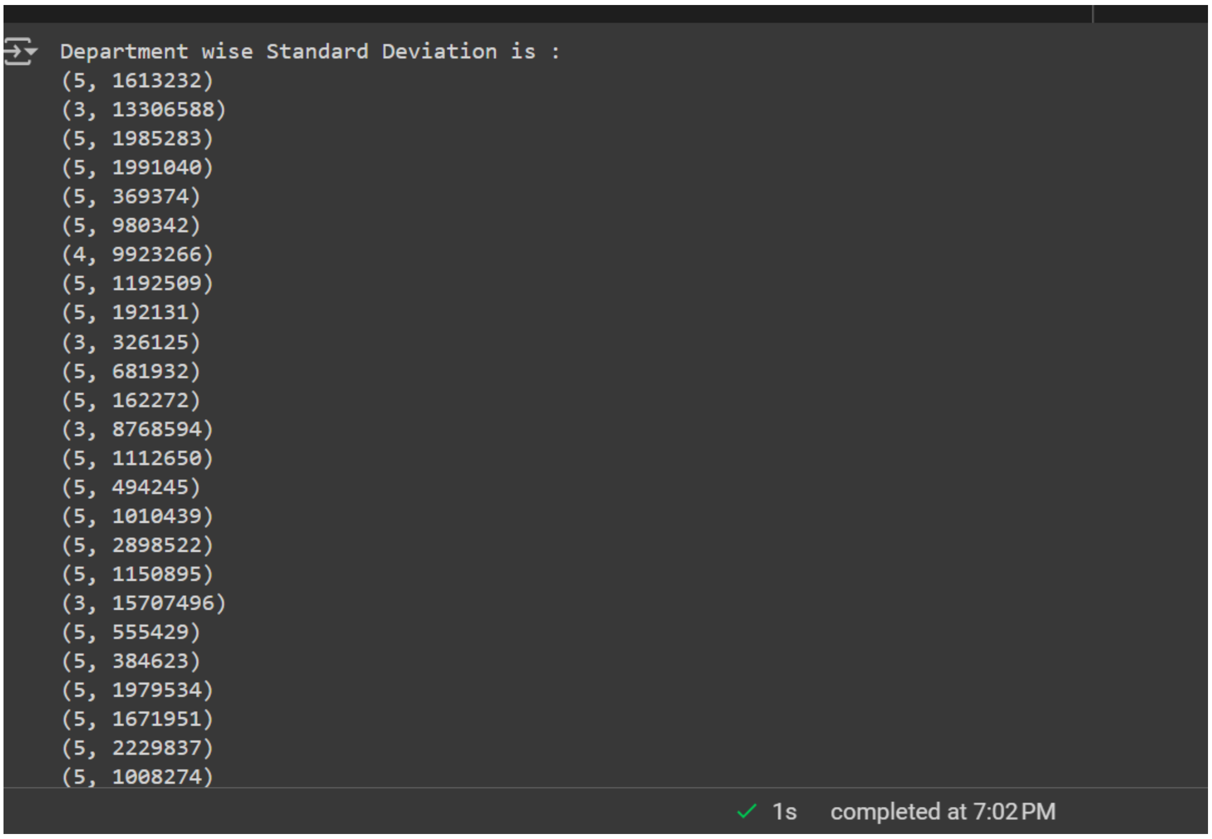


Question 5 : Compute Department wise Standard Deviation of Salary

```
input = "/content/gdrive/My Drive/mr/employee.csv"
lineRDD = sc.textFile(input);
empRDD = lineRDD.map(lambda line: line.split(","))

empd = empRDD.map(lambda rec : (int(rec[2]),int(rec[3]))) # for sum
red= empd.reduceByKey(lambda x1,x2: x1+x2)
sum=int(red.collect()[0][1])

empdd= empRDD.map(lambda rec : (1,1)) # for entries
red2= empdd.reduceByKey(lambda x1,x2: x1+x2)
entries=int(red2.collect()[0][1])

avg=int(sum/entries)


empd2 = empRDD.map(lambda rec : (int(rec[2]),(int(rec[3]),int(avg))))
empd3 = empd2.map(lambda x: (x[0], int(((x[1][0] - x[1][1]) ** 2)/entries)))

red3 = empd3.reduceByKey(lambda x1, x2: x1 + x2)

print("Department wise Standard Deviation is :")
printList(empd3.collect())
```

✓ 1s completed at 7:02 PM

Output:

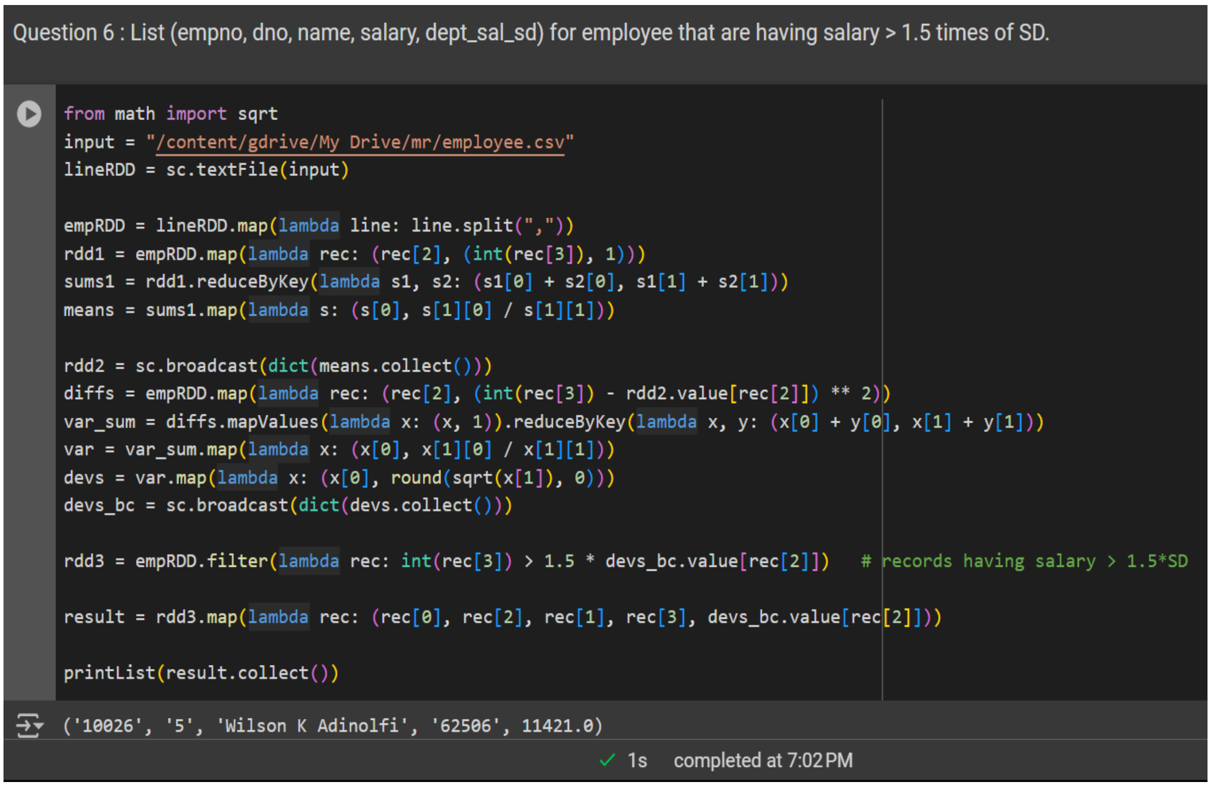


```
Department wise Standard Deviation is :
(5, 1613232)
(3, 13306588)
(5, 1985283)
(5, 1991040)
(5, 369374)
(5, 980342)
(4, 9923266)
(5, 1192509)
(5, 192131)
(3, 326125)
(5, 681932)
(5, 162272)
(3, 8768594)
(5, 1112650)
(5, 494245)
(5, 1010439)
(5, 2898522)
(5, 1150895)
(3, 15707496)
(5, 555429)
(5, 384623)
(5, 1979534)
(5, 1671951)
(5, 2229837)
(5, 1008274)
```

✓ 1s completed at 7:02 PM

Question 6: List (empno, dno, name, salary, dept_sal_sd) for employee that are having salary > 1.5 times of SD.

Code and Output:



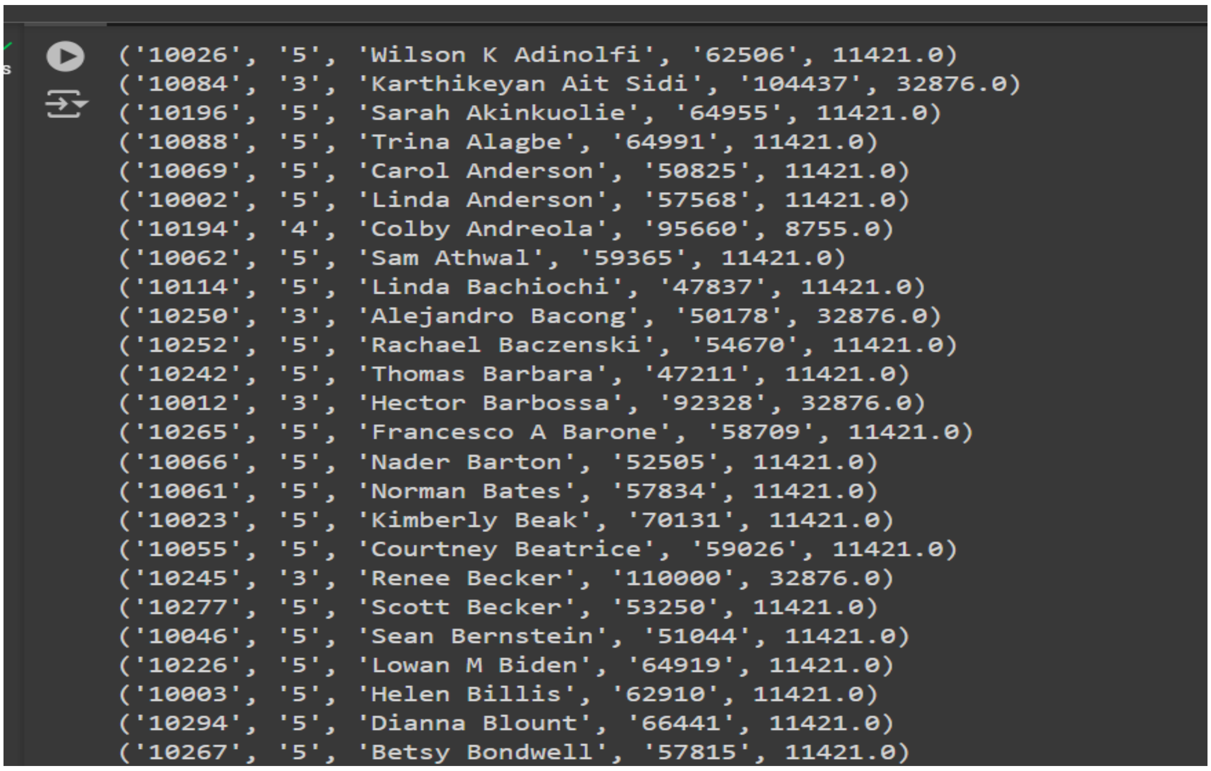
Question 6 : List (empno, dno, name, salary, dept_sal_sd) for employee that are having salary > 1.5 times of SD.

```python
from math import sqrt
input = "/content/gdrive/My Drive/mr/employee.csv"
lineRDD = sc.textFile(input)

empRDD = lineRDD.map(lambda line: line.split(","))
rdd1 = empRDD.map(lambda rec: (rec[2], (int(rec[3]), 1)))
sums1 = rdd1.reduceByKey(lambda s1, s2: (s1[0] + s2[0], s1[1] + s2[1]))
means = sums1.map(lambda s: (s[0], s[1][0] / s[1][1]))

rdd2 = sc.broadcast(dict(means.collect()))
diffs = empRDD.map(lambda rec: (rec[2], (int(rec[3]) - rdd2.value[rec[2]]) ** 2))
var_sum = diffs.mapValues(lambda x: (x, 1)).reduceByKey(lambda x, y: (x[0] + y[0], x[1] + y[1]))
var = var_sum.map(lambda x: (x[0], x[1][0] / x[1][1]))
devs = var.map(lambda x: (x[0], round(sqrt(x[1]), 0)))
devs_bc = sc.broadcast(dict(devs.collect()))

rdd3 = empRDD.filter(lambda rec: int(rec[3]) > 1.5 * devs_bc.value[rec[2]])   # records having salary > 1.5*SD

result = rdd3.map(lambda rec: (rec[0], rec[2], rec[1], rec[3], devs_bc.value[rec[2]]))

printList(result.collect())
```

```
('10026', '5', 'Wilson K Adinolfi', '62506', 11421.0)
```

✓ 1s completed at 7:02 PM

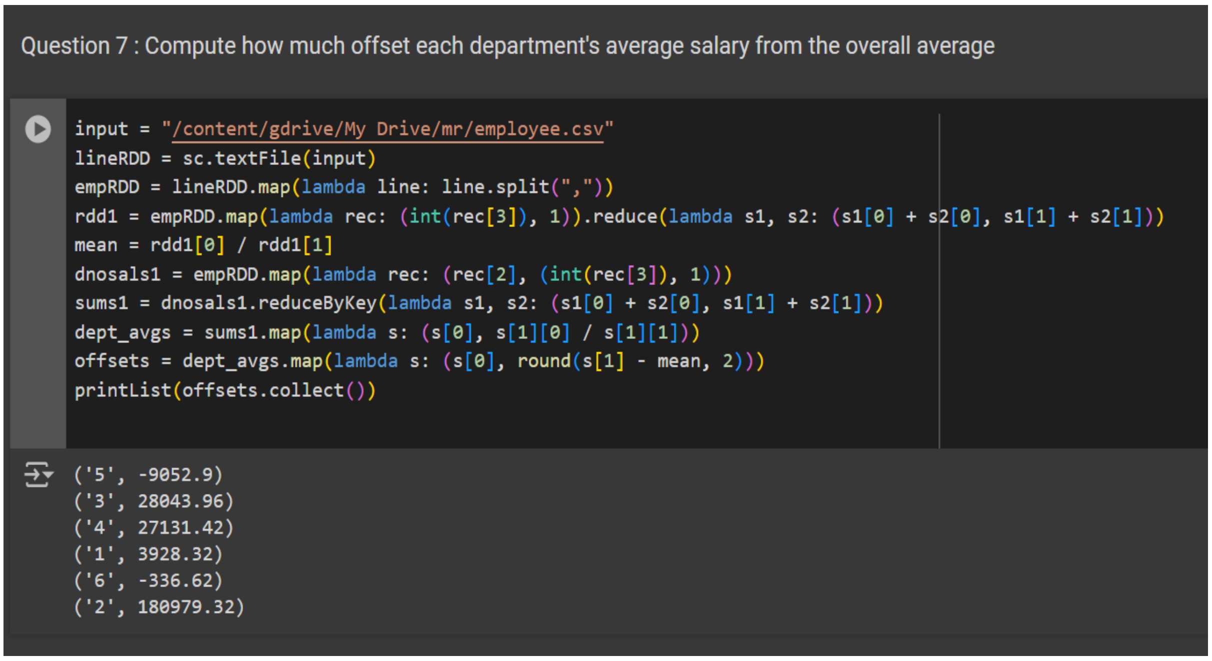
Output:



```
('10026', '5', 'Wilson K Adinolfi', '62506', 11421.0)
('10084', '3', 'Karthikeyan Ait Sidi', '104437', 32876.0)
('10196', '5', 'Sarah Akinkuolie', '64955', 11421.0)
('10088', '5', 'Trina Alagbe', '64991', 11421.0)
('10069', '5', 'Carol Anderson', '50825', 11421.0)
('10002', '5', 'Linda Anderson', '57568', 11421.0)
('10194', '4', 'Colby Andreola', '95660', 8755.0)
('10062', '5', 'Sam Athwal', '59365', 11421.0)
('10114', '5', 'Linda Bachiochi', '47837', 11421.0)
('10250', '3', 'Alejandro Bacong', '50178', 32876.0)
('10252', '5', 'Rachael Baczenski', '54670', 11421.0)
('10242', '5', 'Thomas Barbara', '47211', 11421.0)
('10012', '3', 'Hector Barbossa', '92328', 32876.0)
('10265', '5', 'Francesco A Barone', '58709', 11421.0)
('10066', '5', 'Nader Barton', '52505', 11421.0)
('10061', '5', 'Norman Bates', '57834', 11421.0)
('10023', '5', 'Kimberly Beak', '70131', 11421.0)
('10055', '5', 'Courtney Beatrice', '59026', 11421.0)
('10245', '3', 'Renee Becker', '110000', 32876.0)
('10277', '5', 'Scott Becker', '53250', 11421.0)
('10046', '5', 'Sean Bernstein', '51044', 11421.0)
('10226', '5', 'Lowan M Biden', '64919', 11421.0)
('10003', '5', 'Helen Billis', '62910', 11421.0)
('10294', '5', 'Dianna Blount', '66441', 11421.0)
('10267', '5', 'Betsy Bondwell', '57815', 11421.0)
```

Question 7: Compute how much offset each department's average salary from the overall average.

Code and Output:



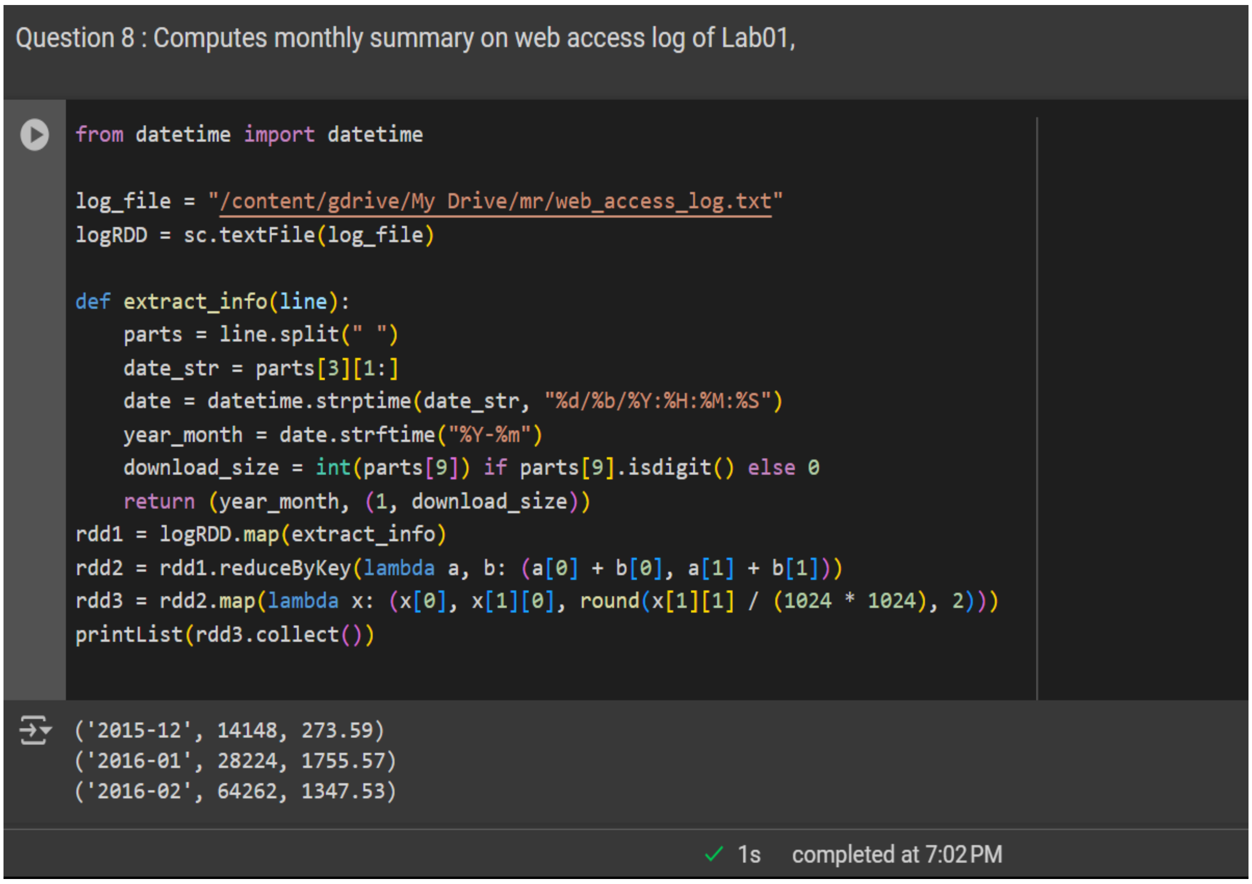
Question 7 : Compute how much offset each department's average salary from the overall average

```
input = "/content/gdrive/My Drive/mr/employee.csv"
lineRDD = sc.textFile(input)
empRDD = lineRDD.map(lambda line: line.split(","))
rdd1 = empRDD.map(lambda rec: (int(rec[3]), 1)).reduce(lambda s1, s2: (s1[0] + s2[0], s1[1] + s2[1]))
mean = rdd1[0] / rdd1[1]
dnosals1 = empRDD.map(lambda rec: (rec[2], (int(rec[3]), 1)))
sums1 = dnosals1.reduceByKey(lambda s1, s2: (s1[0] + s2[0], s1[1] + s2[1]))
dept_avgs = sums1.map(lambda s: (s[0], s[1][0] / s[1][1]))
offsets = dept_avgs.map(lambda s: (s[0], round(s[1] - mean, 2)))
printList(offsets.collect())
```

```
('5', -9052.9)
('3', 28043.96)
('4', 27131.42)
('1', 3928.32)
('6', -336.62)
('2', 180979.32)
```

Question 8: Computes monthly summary on web access log of Lab01

Code and Output:



Question 8 : Computes monthly summary on web access log of Lab01,

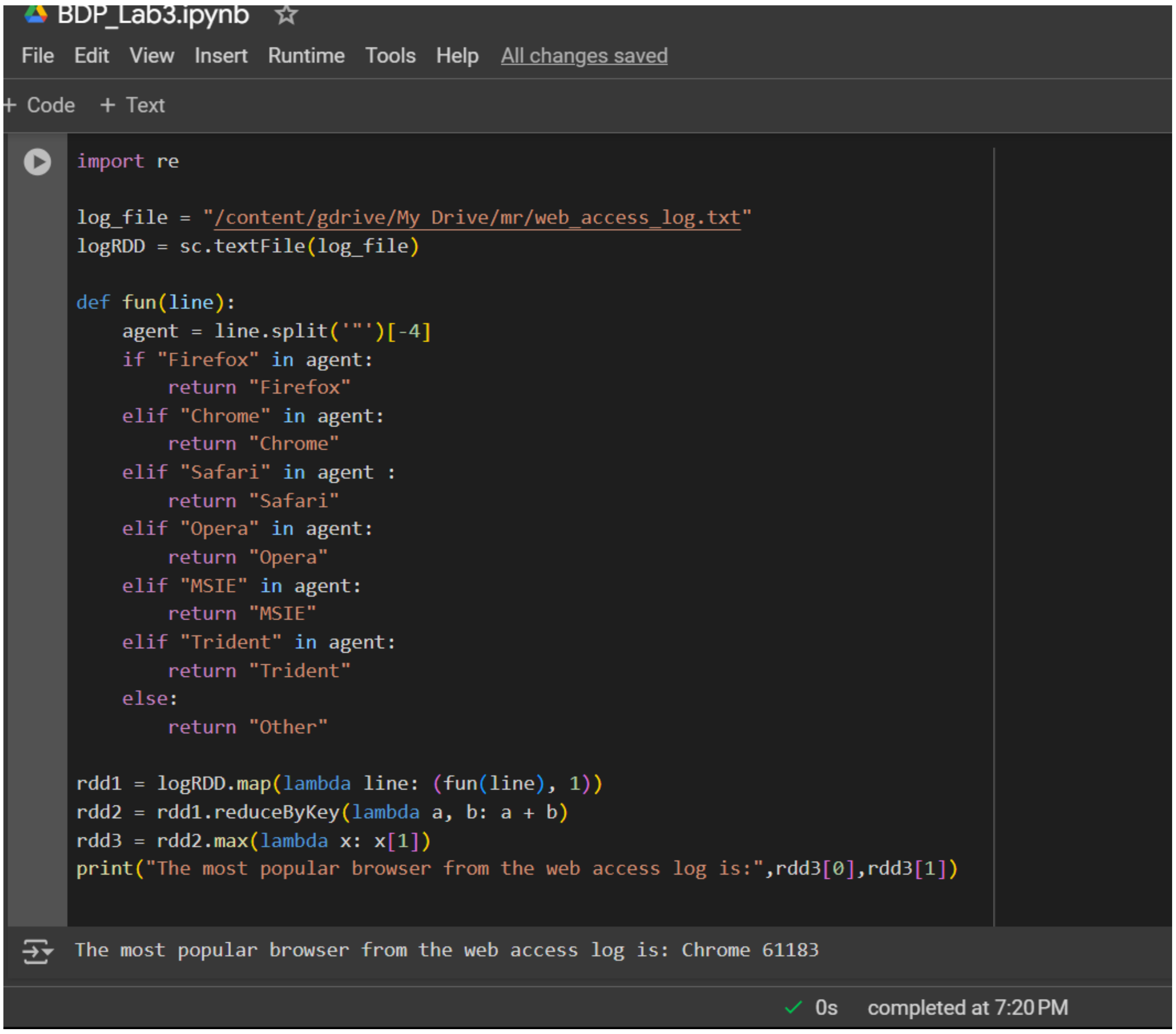
```
from datetime import datetime

log_file = "/content/gdrive/My Drive/mr/web_access_log.txt"
logRDD = sc.textFile(log_file)

def extract_info(line):
    parts = line.split(" ")
    date_str = parts[3][1:]
    date = datetime.strptime(date_str, "%d/%b/%Y:%H:%M:%S")
    year_month = date.strftime("%Y-%m")
    download_size = int(parts[9]) if parts[9].isdigit() else 0
    return (year_month, (1, download_size))
rdd1 = logRDD.map(extract_info)
rdd2 = rdd1.reduceByKey(lambda a, b: (a[0] + b[0], a[1] + b[1]))
rdd3 = rdd2.map(lambda x: (x[0], x[1][0], round(x[1][1] / (1024 * 1024), 2)))
printList(rdd3.collect())
```

```
('2015-12', 14148, 273.59)
('2016-01', 28224, 1755.57)
('2016-02', 64262, 1347.53)
```

✓ 1s completed at 7:02 PM

Question 9: Find the most popular browser (made most requests) from the “web access log” file of lab01.

Code and Output:



```python
import re

log_file = "/content/gdrive/My Drive/mr/web_access_log.txt"
logRDD = sc.textFile(log_file)

def fun(line):
    agent = line.split('"')[-4]
    if "Firefox" in agent:
        return "Firefox"
    elif "Chrome" in agent:
        return "Chrome"
    elif "Safari" in agent :
        return "Safari"
    elif "Opera" in agent:
        return "Opera"
    elif "MSIE" in agent:
        return "MSIE"
    elif "Trident" in agent:
        return "Trident"
    else:
        return "Other"

rdd1 = logRDD.map(lambda line: (fun(line), 1))
rdd2 = rdd1.reduceByKey(lambda a, b: a + b)
rdd3 = rdd2.max(lambda x: x[1])
print("The most popular browser from the web access log is:",rdd3[0],rdd3[1])
```

```
The most popular browser from the web access log is: Chrome 61183
```