GRIFFITH COLLEGE DUBLIN

BIG DATA MANAGEMENT AND ANALYTICS

BIG DATA MANAGEMENT ASSIGNMENT 1

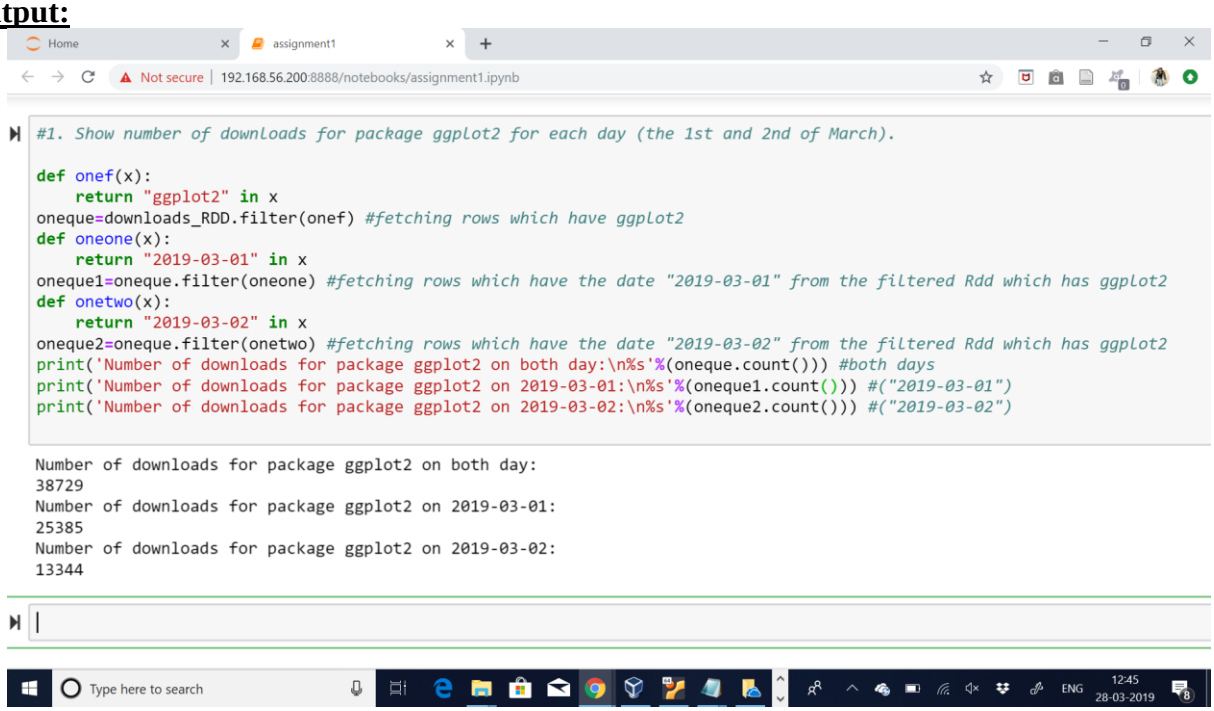
ARAVINDAN SRINIVASAN(2981707) 28/03/2019
aravindan.srinivasan@student.griffith.ie

1)Show number of downloads for package ggplot2 for each day (the 1st and 2nd of March).

Coding:

```
def onef(x):
    return "ggplot2" in x
oneque=downloads_RDD.filter(onef) #fetching rows which have ggplot2
def oneone(x):
    return "2019-03-01" in x
oneque1=oneque.filter(oneone) #fetching rows which have the date "2019-03-01" from the filtered Rdd which has ggplot2
def onetwo(x):
    return "2019-03-02" in x
oneque2=oneque.filter(onetwo) #fetching rows which have the date "2019-03-02" from the filtered Rdd which has ggplot2
print('Number of downloads for package ggplot2 on both day:\n%s'%(oneque.count())) #both days
print('Number of downloads for package ggplot2 on 2019-03-01:\n%s'%(oneque1.count())) #("2019-03-01")
print('Number of downloads for package ggplot2 on 2019-03-02:\n%s'%(oneque2.count())) #("2019-03-02")
```

Output:

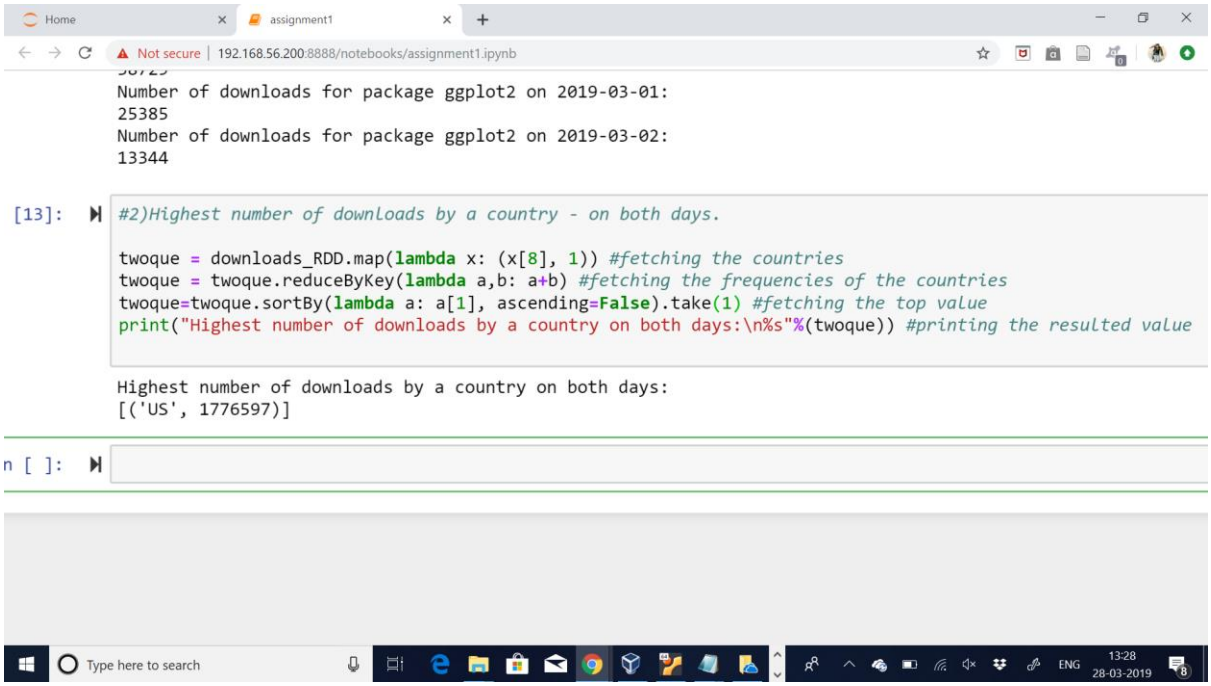


2) Highest number of downloads by a country - on both days.

Coding:

```
twoque = downloads_RDD.map(lambda x: (x[8], 1)) #fetching the countries
twoque = twoque.reduceByKey(lambda a,b: a+b) #fetching the frequencies of the countries
twoque=twoque.sortBy(lambda a: a[1], ascending=False).take(1) #fetching the top value
print("Highest number of downloads by a country on both days:\n%s"%(twoque)) #printing the resulted value
```

Output:

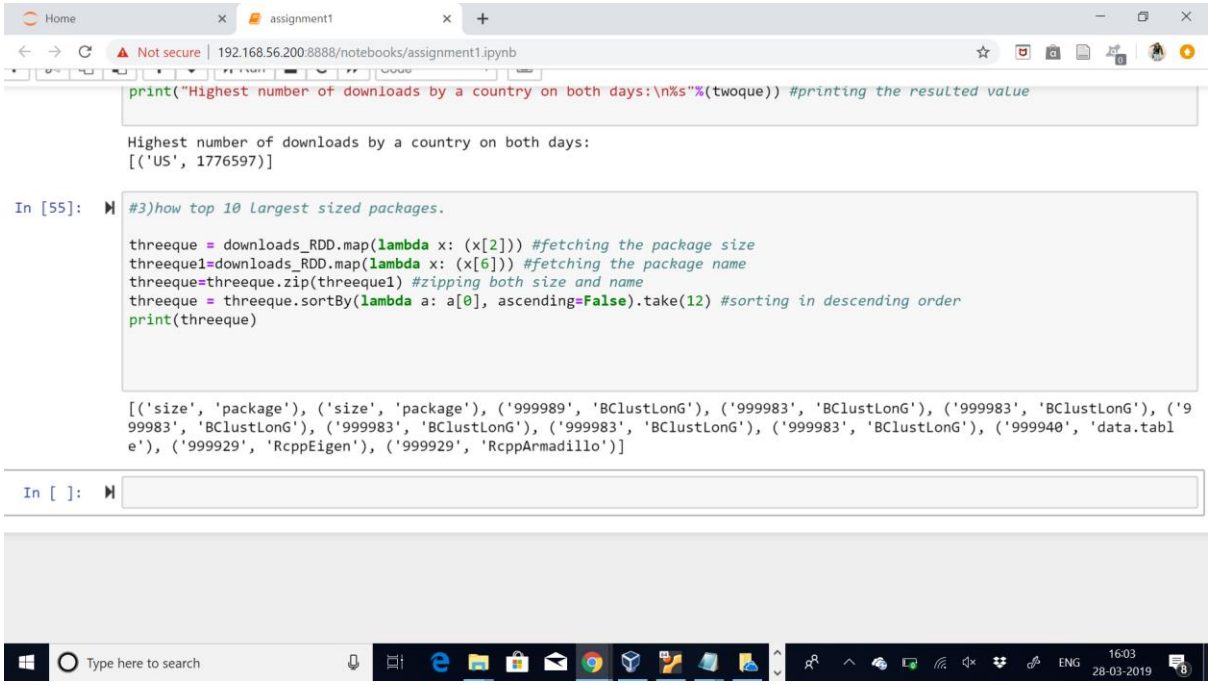


3)how top 10 largest sized packages.

Coding:

threeque = downloads_RDD.map(lambda x: (x[2])) #fetching the package size
threeque1=downloads_RDD.map(lambda x: (x[6])) #fetching the package name
threeque=threeque.zip(threeque1) #zipping both size and name
threeque = threeque.sortBy(lambda a: a[0], ascending=False).take(12) #sorting in descending order
print(threeque)

Output:

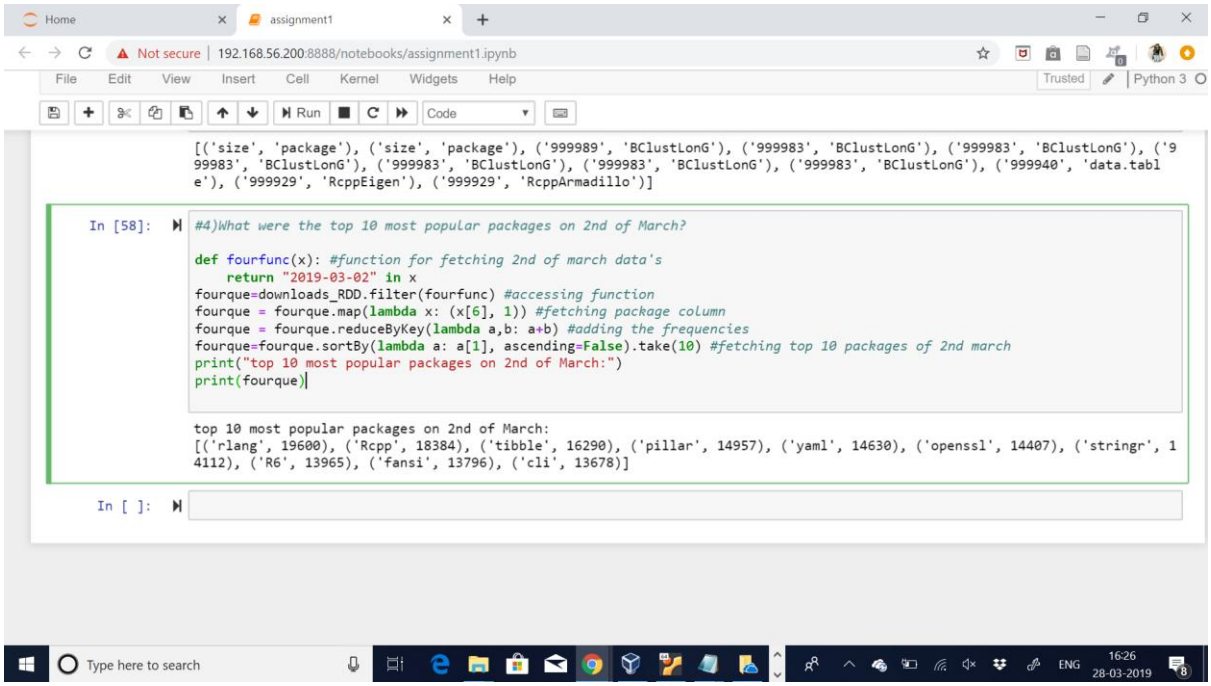


4)What were the top 10 most popular packages on 2nd of March?

Coding:

```
def fourfunc(x): #function for fetching 2nd of march data's
    return "2019-03-02" in x
fourque=downloads_RDD.filter(fourfunc) #accessing function
fourque = fourque.map(lambda x: (x[6], 1)) #fetching package column
fourque = fourque.reduceByKey(lambda a,b: a+b) #adding the frequencies
fourque=fourque.sortBy(lambda a: a[1], ascending=False).take(10) #fetching top 10
packages of 2nd march
print("top 10 most popular packages on 2nd of March:")
print(fourque)
```

Output:

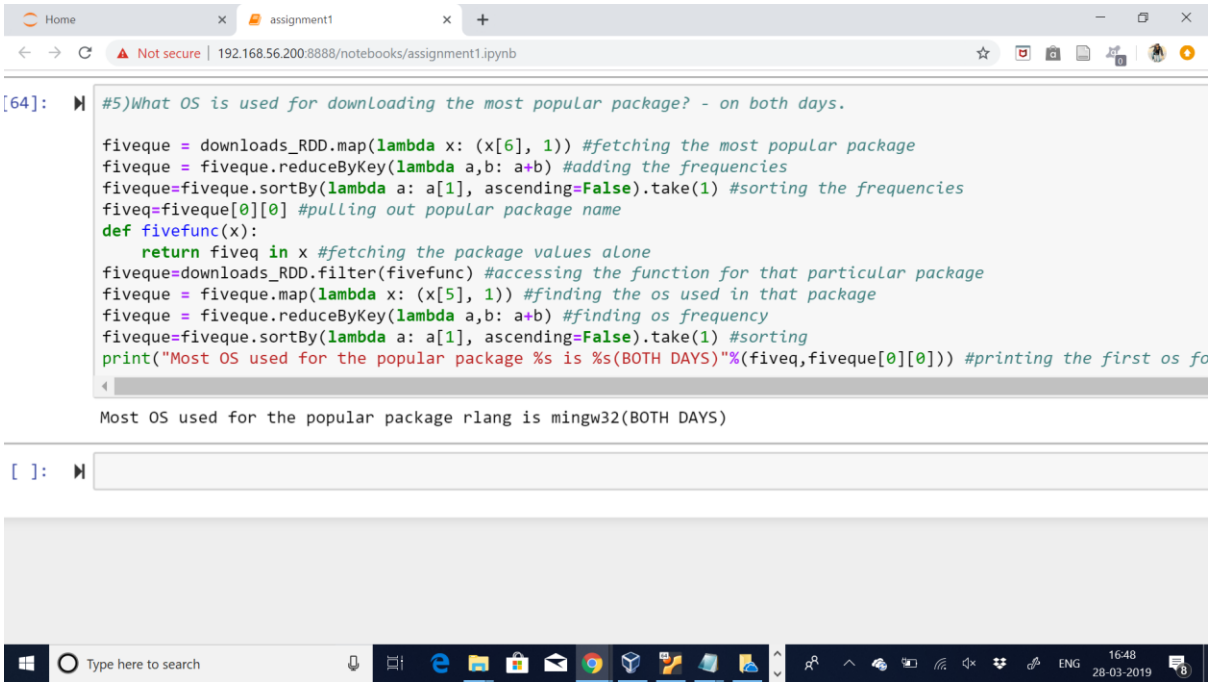


5)What OS is used for downloading the most popular package? - on both days.

Coding:

```
fiveque = downloads_RDD.map(lambda x: (x[6], 1)) #fetching the most popular package
fiveque = fiveque.reduceByKey(lambda a,b: a+b) #adding the frequencies
fiveque=fiveque.sortBy(lambda a: a[1], ascending=False).take(1) #sorting the frequencies
fiveq=fiveque[0][0] #pulling out popular package name
def fivefunc(x):
    return fiveq in x #fetching the package values alone
fiveque=downloads_RDD.filter(fivefunc) #accessing the function for that particular
package
fiveque = fiveque.map(lambda x: (x[5], 1)) #finding the os used in that package
fiveque = fiveque.reduceByKey(lambda a,b: a+b) #finding os frequency
fiveque=fiveque.sortBy(lambda a: a[1], ascending=False).take(1) #sorting
print("Most OS used for the popular package %s is %s(BOTH
DAYS)"%(fiveq,fiveque[0][0])) #printing the first os for poplular package
```

Output:

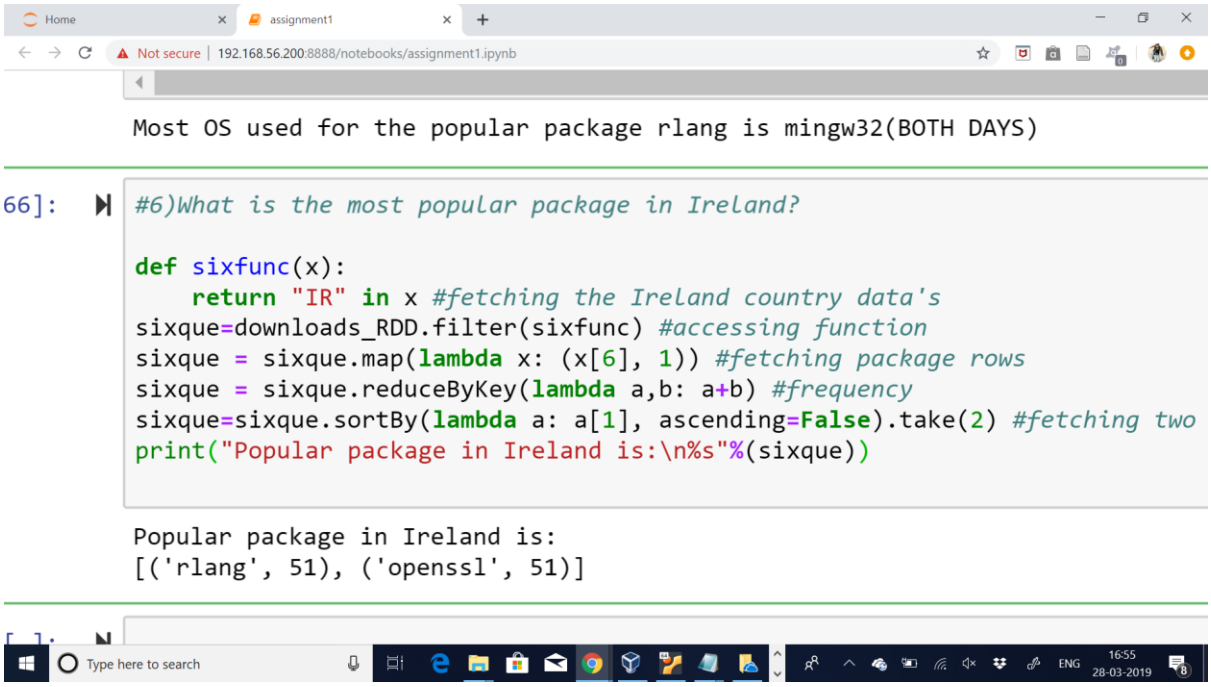


6. What is the most popular package in Ireland?

Coding:

```
def sixfunc(x):
    return "IR" in x #fetching the Ireland country data's
sixque=downloads_RDD.filter(sixfunc) #accessing function
sixque = sixque.map(lambda x: (x[6], 1)) #fetching package rows
sixque = sixque.reduceByKey(lambda a,b: a+b) #frequency
sixque=sixque.sortBy(lambda a: a[1], ascending=False).take(2) #fetching two values
since both have same frequencies
print("Popular package in Ireland is:\n%s"%(sixque))
```

Output:

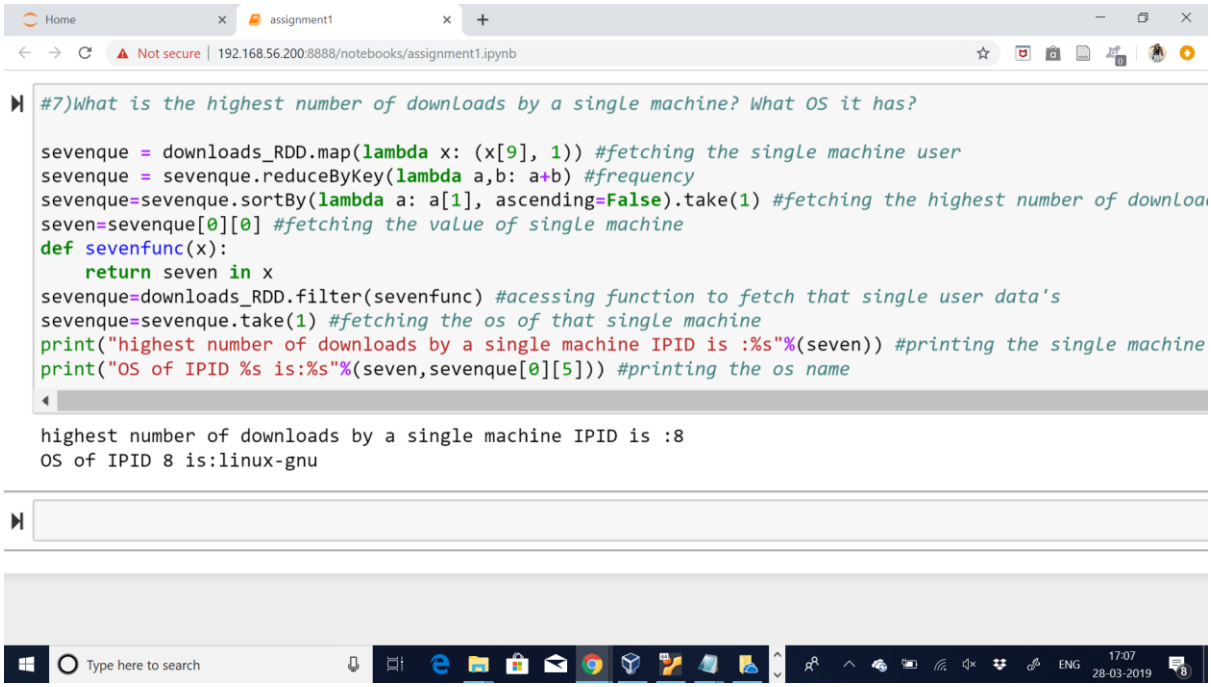


7)What is the highest number of downloads by a single machine? What OS it has?

Coding:

```
sevenque = downloads_RDD.map(lambda x: (x[9], 1)) #fetching the single machine user
sevenque = sevenque.reduceByKey(lambda a,b: a+b) #frequency
sevenque=sevenque.sortBy(lambda a: a[1], ascending=False).take(1) #fetching the
highest number of downloaded by a single machine
seven=sevenque[0][0] #fetching the value of single machine
def sevenfunc(x):
    return seven in x
sevenque=downloads_RDD.filter(sevenfunc) #acessing function to fetch that single user
data's
sevenque=sevenque.take(1) #fetching the os of that single machine
print("highest number of downloads by a single machine IPID is :%s"%(seven)) #printing
the single machine
print("OS of IPID %s is:%s"%(seven,sevenque[0][5])) #printing the os name
```

Output:

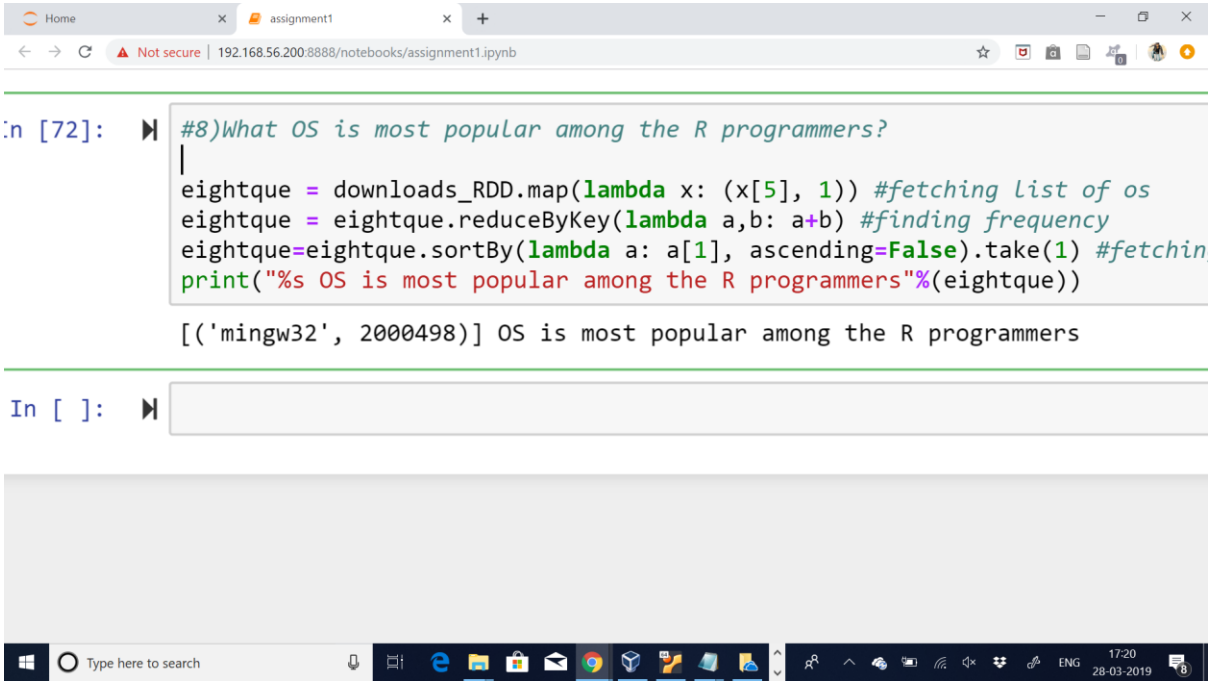


8) What OS is most popular among the R programmers?

Coding:

```
eightque = downloads_RDD.map(lambda x: (x[5], 1)) #fetching list of os
eightque = eightque.reduceByKey(lambda a,b: a+b) #finding frequency
eightque=eightque.sortBy(lambda a: a[1], ascending=False).take(1) #fetching first values
print("%s OS is most popular among the R programmers"%(eightque))
```

Output:

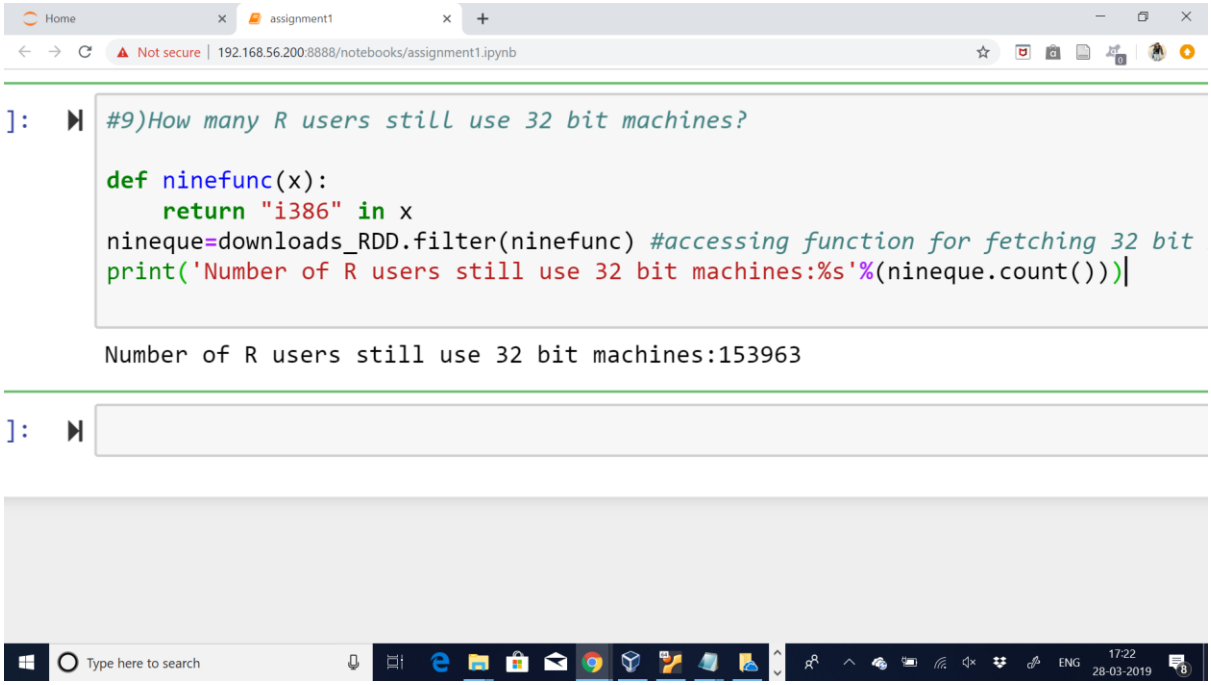


9) How many R users still use 32 bit machines?

Coding:

```
def ninefunc(x):
    return "i386" in x
nineque=downloads_RDD.filter(ninefunc) #accessing function for fetching 32 bit
machines
print('Number of R users still use 32 bit machines:%s'%(nineque.count()))
```

Output:



10) List total number of incomplete records - lines which have missing values.

Coding:

```
def tenfunc(x):
    return "NA" in x
tenque=downloads_RDD.filter(tenfunc) #accessing function for NA values
print('Total number of incomplete records:%s'%(tenque.count()))
```

Output: