Bash File for Spark and Pig

```
export JAVA_HOME=$(readlink -f $(which javac) | awk 'BEGIN {FS="/bin"} {print $1}')
if ! command -v pig/(spark-shell --version) &> /dev/null
then
export PATH=$(echo $PATH):$(pwd)/bin
fi
```

Bash File for Hadoop

```
export JAVA_HOME=$(readlink -f $(which javac) | awk 'BEGIN {FS="/bin"} {print $1}')
export PATH=$(echo $PATH):$(pwd)/bin
export CLASSPATH=$(hadoop classpath)
```

QB solution Part B

6)

```
customer=LOAD ‘customer.txt’ USING PigStorage(‘,’) as
(id:int,age:int,name:chararray);
order=LOAD ‘order.txt’ USING PigStorage(‘,’) as
(oid:int,cust_id:int,name:chararray);
join_result=JOIN customer BY id, orders BY cust_id;
cust_order =ORDER student BY age DESC/ASC;
Dump join_result;
Dump cust_order;
```

5)

```
student =LOAD ‘student.txt’ USING PigStorage(‘,’) as
(id:int,age:int,name:chararray,city:chararray)
filter_data = FILTER student BY city == 'Bangalore’
group_data = GROUP student by age;
Dump filter_data;
Dump group_data;
```

4)

```
import sys
if(len(sys.argv)!=3):
        print("Provide Input File and Output Directory")
        sys.exit(0)
from pyspark import SparkContext
sc =SparkContext()
f = sc.textFile(sys.argv[1])
temp=f.map(lambda x: (x.split(',')[7],1))
data=temp.countByKey()
dd=sc.parallelize(data.items())
dd.saveAsTextFile(sys.argv[2])
```

```
temp=f.map(lambda x: (x.split(',')[3],1))
cout=temp.countByKey()
cc=sc.parallelize(cout.items())
cc.saveAsTextFile(sys.argv[3])
```

3)

```
    import sys
    if(len(sys.argv)!=3):
            print("Provide Input File and Output Directory")
            sys.exit(0)
    from pyspark import SparkContext
    sc =SparkContext()
    f = sc.textFile(sys.argv[1])
    temp=f.map(lambda x: (x.split(',')[16],1))
    data=temp.countByKey()
    dd=sc.parallelize(data.items())
    dd.saveAsTextFile(sys.argv[2])
    temp=f.map(lambda x: (x.split(',')[2],1))
    c=temp.countByKey()
    cout=sc.parallelize(c.items())
    cout.saveAsTextFile(sys.argv[3])
```

2)

```
    Import sys
    if(len(sys.argv)!=6):
                print("please provide correct input and output file")
                sys.exit(0)
    From pyspark import SparkContext
    sc=SparkContext()
    f=sc.textFile(sys.argv[1])
    data=f.map(lambda x:((x.split(',')[11],float(x.split(',')[8])))
    magni=data.reduceByKey(lambda a,b:a if a>b else b)
    magni.saveAsTextFile(sys.argv[2])
    data=f.map(lambda x:(x.split(',')[11],float(x.split(',')[9])))
    dep=data.reduceByKey(lambda a,b:a if a>b else b)
    dep.saveAsTextFile(sys.argv[3])
    data=f.map(lambda x:(x.split(',')[11],float(x.split(',')[6])))
    lat=data.reduceByKey(lambda a,b:a if a>b else b)
    lat.saveAsTextFile(sys.argv[4])
    data=f.map(lambda x:(x.split(',')[11],float(x.split(',')[7])))
    lon=data.reduceByKey(lambda a,b : a if a>b else b)
    lon.saveAsTextFile(sys.argv[5])
```

0067011990999991950051507004+68750+023550FM-12+038299999V0203301N00671220001CN
9999999N9+00001+99999999999

1)

```
import sys
if(len(sys.argv)!=4):
   print("eroor")
   sys.exit(0)
from pyspark import SparkContext
sc=SparkContext()
f=sc.textFile(sys.argv[1])
temp=f.map(lambda x:(int(x[15:18]),int(x[87:92])))
maxi min=temp.reduceByKey(lambda a,b:a if a>b else b)
maxi min.saveAsTextFile(sys.argv[2])
min=temp.reduceByKey(lambda a,b:a if a<b else b)
min.saveAsTextFile(sys.argv[3])
```

Part A

W Hadoop Programs.docx