ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS
MSc in COMPUTER SCIENCE
Theory and Fundamentals of Database Systems

Team Programming Assignment
Submission Deadline: January 16, 2017 (upload a zip file at eclass)
Assignment is to be carried out in groups of 2 or 3 students
In case of cheating, all related submissions shall be graded with 0

Introduction
You are asked to implement two join algorithms, SMJ (Sort Merge Join) and NLJ (Nested Loops Join). More specifically, you have to write a Java program that will take as input 2 comma-separated files and perform the user specified equi-join algorithm so as to join them on selected fields.

Description
Your program has to accept the following command line arguments:

-f1 <file1 path>: full path to file1
-a1 <file1_join_attribute>: the column to use as join attribute from file1 (counting from 0)
-f2 <file2 path>: same as above for file2
-a2 <file2_join_attribute>: same as above for file2
-j <join_algorithm_to_use>: SMJ or NLJ
-m <available_memory_size>: we use as memory metric the number of records*
-t <temporary_dir_path>: a directory to use for reading/writing temporary files
-o <output_file_path>: the file to store the result of the join

For example, in order to join two relations stored in files "R.csv" and "S.csv" on the first column of R and the second column of S, using Sort-Merge Join, having available memory = 100 records and saving the result to file "results.csv" one should execute the following command:

```
java –jar joinalgs.jar –f1 R.csv –a1 0 –f2 S.csv –a2 1 –j SMJ –m 100 –t tmp –o results.csv
```

*available_memory_size: In your program you must simulate the available memory by the use of arrays (one or more). For the sake of simplicity, we make the assumption that when we say memory=100 it means that we can load 100 records (even of different sizes, e.g. first relation may have 4 attributes while second may have 6 attributes). What is implied here is that you must read the input files in batches that can be handled by the available memory. That way your program should be capable to handle input much larger than the JVM available memory. While the program is running you can write/read any number of temporary files at your tmp directory (which should be cleaned up before the program exits).

In order to simplify the programming procedure you may also assume that all attributes of the input relations are positive integer numbers.

Attention: You are not allowed to use data management libraries or a DBMS.

Your goal is to compute the join in the shortest time possible. For instance if for the given input relations and the available memory the join can be performed is a single pass, your program should adapt accordingly.

Example

Using the folowing relations:

R

1	2	3
1	6	7
2	4	3

S

1	5	8	12
3	7	8	9
7	1	2	3
5	6	2	3

The output file of the execution

```
java -jar joinalgs.jar -f1 R.csv -a1 2 -f2 S.csv -a2 0 -j NLJ -m 100 -t tmp -o results.csv
```

shall contain (not necessarily in the shown order) the following records:

R0	R1	R2 = S0 (join column)	S1	S2	S3
1	2	3	7	8	9
2	4	3	7	8	9
1	6	7	1	2	3

There is no requirement to output a header line.

Testing

In order to test your implementation you are given 5 csv files (A.csv (150 records), B.csv (6000 records), C.csv (20000 records), D.csv (30000 records), E.csv (100000 records)) which you can download from: https://pithos.okeanos.grnet.gr/public/hu3JKuphxjwvYjjEbWTqs6

The first line of each file is the number of records it contains.

You are asked to measure execution time for the following equi-joins:

f1	a1	f2	a2	m	J
D	3	C	0	200	NLJ
D	3	B	0	200	NLJ
A	3	E	0	200	NLJ
B	1	B	2	200	NLJ
D	3	C	0	200	SMJ
D	3	B	0	200	SMJ
A	3	E	0	200	SMJ
B	1	B	2	200	SMJ

Deliverable

You must submit your work at eclass in a zip file containing:

a. **report.pdf**: a sort description of the implementation choices you made and instructions on the compilation/execution of your program. Also present the results from testing your program here. On the first page of the pdf you must mention the names and AMs of the team's members.
b. **src.zip**: the source code of your program (if you would like you can submit it as an IDE project)

Submission from one team member is enough.

Grading

Your solution will be graded based both on the correctness of your solution and on the efficiency of your code.

For any questions/clarifications about the assignment you should contact the course TA Vasilis Spyropoulos (vasspyrop@aueb.gr)