The assignment should be done in groups of two students. You must turn in the source code of your program through Canvas. Each group must submit only one file that contains the full name, OSU email, and ONID of every member of the group.

1: Join Algorithms (10 points)

The objective of this assignment is to learn join implementation algorithms for relations and files on external memories.

Consider the following relations:

Emp (eid (integer), ename (string), age (integer), salary (double))
Dept (did (integer), dname (string), budget (double), managerid (integer))

Fields of types integer, double, and string occupy 4, 8, and 40 bytes, respectively. Each page can fit at most one tuple (record) of the input relation. Using the provided skeleton code with this assignment, implement the optimized sort-merge join algorithm for $Dept \bowtie_{Dept.managerid=Emp.eid} Emp$ in C/C++.

- **Input Files:** Each input relation is stored in a CSV file, i.e., each tuple is in a separate line and fields of each record are separated by commas. The files that store relations *Emp* and *Dept* are *Emp.csv* and *Dept.csv*, respectively. Your program must assume that the input files are in the current working directory, i.e., the one from which your program is running. We have included sample input CSV files with this assignment as sample test cases for your program. Your program must join correctly other CSV files with the same fields as the sample files.
- **Final Output:** The program must store the result in a new CSV file with the name *Join.csv* in the current working directory.
- **Main Memory Limitation:** Your program can keep up to 22 pages in main memory at any time, i.e., the size of the buffer (M) is 22. The control, local, or temporary variables, e.g., flags or counters, are excluded from this limit. The submitted solutions that use more main memory will *not* get any points.
- **Page Format:** You can use the data structure of your choosing to represent and store pages in main memory and files.
- **Sorting:** You can use/modify the code from assignment 4 to perform the sort phase of the join.
- **Types of Temporary Files (runs):** You can use the type (text or binary) of your choosing for the temporary files (runs).
- Each student has an account on hadoop-master.engr.oregonstate.edu server, which is a Linux machine. You must ensure that your program can be compiled and run on this machine. You can use the following bash command to connect to it:

  ```
  > ssh your_onid_username@hadoop-master.engr.oregonstate.edu
  ```

  Then it asks for your ONID password and probably one another question. To access this server, you must be on campus or connected to the Oregon State VPN.

- You can use the following commands to compile and run C++ code:

```
> g++ -std=c++11 main.cpp -o main.out

> main.out
```

- **Grading Criteria:** The programs that implement the correct algorithm, return correct answers, and do not use more than allowed buffers will get the perfect score. The ones that use more buffer than allowed will not get any points. The ones that implement the right algorithm but return partially correct answers will get partial credits.

2: Query Optimization (2 points)

The objective of this assignment is to practice your understanding of query optimization concepts and algorithms.

For the four relations in the following table, find the most efficient join order according to the dynamic programming algorithm used in System-R. You should give the dynamic programming table entries for evaluate the join orders. The cost of each join is the number of I/O accesses the database system performs to execute the join. Assume that the database system uses two-pass sort-merge join algorithm to perform join operations. Each page contains 4 tuples and tuples of all relations have the same size. We are interested only in left-deep join trees. Note that you should use the System-R optimizer formula to compute the size of each join output.

Relation	Number of tuples	Cardinality of A	Cardinality of B	Cardinality of C	Cardinality of D
R(A,B,C)	T(R)=4000	V(R,A) =100	V(R,B) =200	V(R,C) =100	
S(B,C)	T(S)=3000		V(S,B) =100	V(S,C) = 300	
W(B,D)	T(W)=2000		V(W,B) =100		V(W,D) =50
U(A,D)	T(U)=1000	V(U,A) =100			V(U,D) =100

Grading Criteria: Incomplete answers get partial points depending on how accurate their returned results are.