1. Consider the query $R(A,B) \ join S(C,D) \ join T(E,F)$ (the join condition is $B=C$ and $D=E$). Suppose $M = 100$, and $B(R) = 30$, $B(S) = 200$, $B(T) = 60$, $B(R \ join S) = 80$, $B(S \ join T) = 50$. Design an optimal query plan that uses only main-memory hash join algorithms. Your plan may store intermediate results to disk if necessary.

Load $R$ & $T$ into memory and create hash tables of them. Then read blocks of $S$ one at a time, performing the joins in the following graph. All intermediate results are pipelined.

```
  HashJoin B=C
   /           \
  /               \
HashJoin D=E
   /     \
R     T     S
```
2. Consider the algebra plan below. Each of the joint operators is a main memory hash join algorithm, using the Open( ), GetNext( ), Close( ) interface. Assuming that all joins are pipelining, show the execution steps for computing the entire join.

Where R, S, T, U have the following content:

<table>
<thead>
<tr>
<th></th>
<th>R</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A1 B</td>
<td>B D1</td>
</tr>
<tr>
<td></td>
<td>A2 B</td>
<td>B D2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>U</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>D1 F</td>
<td>F H1</td>
</tr>
<tr>
<td></td>
<td>D2 F</td>
<td>F H2</td>
</tr>
</tbody>
</table>
T0.open
T2.open
   U.open
   U.getNext
   U.getNext
   U.getNext // got NULL
   U.close
   T.open
T2.getNext
   T.getNext
T2.getNext
T2.getNext
   T.getNext
T2.getNext
   T2.getNext // got NULL
T2.close
   T.close
T1.open
   S.open
   S.getNext
   S.getNext
   S.getNext // got NULL
   S.close
   R.open
T0.getNext
   T1.getNext
   R.getNext
T0.getNext
T0.getNext
   T1.getNext
T0.getNext
T0.getNext
   T1.getNext
   R.getNext
T0.getNext
T0.getNext
   T1.getNext
T0.getNext
T0.getNext
   T1.getNext
   R.getNeXt
   R.getNext // got NULL
T0.close
   T1.close
   R.close
(b) [10 points] Consider the following query, where \( \Join \) denotes the natural join:

\[
R(A, B) \Join S(B, C) \Join T(C, D) \Join U(D, E)
\]

Here we only consider left linear plans

i. How many different left linear plans exist for this query?

\[ n! \]

ii. Show two different left linear plans without cartesian products.

\[
(((R \text{ join } S) \text{ join } T) \text{ join } U) \\
(((T \text{ join } S) \text{ join } U) \text{ join } R)
\]

iii. How many different plans without cartesian product exists for this query?

\[ 2^{(n-1)} \]