Problem 1

a) Given,
Buffer 152 pages
Invoice size 200000 (outer)
Items size 500000 (inner)

Note, 1 sequential page access = Accessing 152 pages.
Thus, total sequential page access = 1315989474 / 152 = 8657826

b) Notes, all relevant attributes are uniformly distributed.

Assuming, 500000/200000 = 2.5 Item per tuple for each Invoice tuple. Thus total partitions will be 50000 /108 = 4630 (optimum k)
We know that in Hash join tuples are accessed exactly once. Assuming accessing 152 pages as 1 sequential access:

Thus, total sequential access will be (500000 + 200000) / 152 = 4606.

Problem 2

a) Java code submitted.

b) The nested loop join took more time in general because it took total lineitem.size()*orders.size() calculation.
Thought the second approach was hash based join. For this total complexity was lineitem.size()+orders.size().

c) There are three options postgre can choose: Nested loop join, Merge join and Hash join.
[i]

Postgres choose hash join: Execution Time: 4259.387 ms
Hash Join (cost=65705.00..380985.58 rows=6001263 width=12) (actual time=434.388..4142.813 rows=6001215 loops=1)

d) Forcing Merge Join with SET enable_mergejoin on executed for: Execution time: 15704.258 ms for the first time.

	QUERY PLAN text
1	Merge Join (cost=1227736.62..1340257.88 rows=6001215 width=12) (actual time=9848.726..15363.237 rows=6001215 loops=1)
2	Merge Cond: (orders.o_orderkey = lineitem.l_orderkey)
3	-> Sort (cost=215478.98..219228.98 rows=1500000 width=8) (actual time=1501.333..1836.872 rows=1500000 loops=1)
4	Sort Key: orders.o_orderkey
5	Sort Method: external merge Disk: 26488kB
6	-> Seq Scan on orders (cost=0.00..41095.00 rows=1500000 width=8) (actual time=0.081..657.003 rows=1500000 loops=1)
7	-> Materialize (cost=1012257.64..1042263.71 rows=6001215 width=8) (actual time=8347.383..11043.044 rows=6001215 loops=1)
8	-> Sort (cost=1012257.64..1027260.67 rows=6001215 width=8) (actual time=8347.373..9736.904 rows=6001215 loops=1)
9	Sort Key: lineitem.l_orderkey
10	Sort Method: external merge Disk: 105616kB
11	-> Seq Scan on lineitem (cost=0.00..172515.15 rows=6001215 width=8) (actual time=0.105..3519.987 rows=6001215 loops=1)
12	Planning Time: 0.154 ms
13	Execution Time: 15704.258 ms

Problem 3

(a)

Number of runs $N_r = N/ N_b$

$= 300000/30$

$= 10000$

Number of merge phases, $p = \lceil \log_{N_{b-1}} (N_r) \rceil$

$=2.735 = 3$

Therefore, a buffer of size 30 pages is enough for external sorting.

Minimum buffer size for 4 merge phases is as following:

$4 = \log_{N_{b-1}} (30000/ N_b)$ which yields

$N_b = 14$

So minimum buffer size required is 14

(b) Let, Total Blocks = N

Buffer Size = N_b

Output Block = b

Total runs $N_r = N/ N_b$

Possible runs per phase, $F = N_b /b + 1$

If we count each run is possible for each i/o operation then,

Total IO = N_r / (N_b /b + 1)

Total phase 1 + $\log_F N_r$

Total no of input output required IO = (Nr/ (N_b /b + 1)) (1 + $\log_F N_r$)

When N = 250000

N_b = 20

Let b = 5

Nr = 250000/20 = 12500

Possible run per phase, F = 20/5 + 1 = 6

Total i/o per phase = 12500/6 ~ 2084

Total phase = 1 + $\log_6$ 12500 ~ 6

Total IO = 2084 * 6 = 12504

[i] https://www.postgresql.org/docs/current/planner-optimizer.html