SOLUTION SKETCH, INF3800/INF4800 V13

QUESTION 1: NAÏVE BAYES

a) See Equations 13.2 and 13.3 (p. 239.) Conditional independence assumption (p. 246), positional independence for the multinomial model (p. 247).

b) See Sections 13.2-13.4. Table 13.3 (p. 248) summarizes. Multinomial model typically used in practice.

c) This is really Table 13.1 (p. 241) in disguise. There are 3 documents in *c=canada*. In these 3 documents there are 8 tokens in total, whereof 5 are *bieber*. All 3 *c=canada* documents contain *bieber*. See also Examples 13.1 (p. 241) and 13.2 (p. 244).
1. Multinomial: P = (5+1)/(8+6)=6/14=3/7 = 0.43.
2. Bernoulli: P = (3+1)/(3+2) = 4/5 = 0.80.

QUESTION 2: QUERY EVALUATION

a) See Section 2.3 (p. 33.) Only effective for conjunctive queries, not disjunctive queries. Even for conjunctive queries, skip lists could slow down search in some settings (p. 44.)

b) See Section 7.1.5 (p. 129.) Having impact-ordered posting lists implies using TAAT evaluation, because the ordering isn't global.

c) See Section 7.1.4 (p. 127.) Allows premature termination of the posting list scanning, for better performance in ranked retrieval. Static quality scores could represent, e.g., PageRank score (Section 21.2.2), or a measure derived from user reviews (p. 127), or a popularity/buzz score derived from an analytics service.

QUESTION 3: XML SEARCH

a) See Section 10.2 (p. 184.)

b) See Section 10.2 and lecture slides. E.g., choice of indexing unit (and its impact on document statistics), nested elements, schemas (both lack thereof and schema heterogeneity), queries that involve structure and content (and UI challenges for users), evaluation metrics.

QUESTION 4: EVALUATION METRICS

a) F_1 = 1/(0.5(1/R+1/P)) = 2PR/(P + R). See Section 8.3 (p. 144.)

b) See Section 8.4 (p. 148.) With P = R, this is where F_1 = P = R.

c) P = 8/(8+10) = 8/18 = 0.44. R = 8/20 = 2/5 = 0.40.

QUESTION 5: SUFFIX ARRAYS

a) Let *bieber* be entry 0 and *belieber* be entry 1. See table below. Then sort the table rows lexicographically by the second column. The suffix array is the first column in the sorted table. Here (x, y) means position index y of entry x.

(0, 0)	*bieber*
(0, 1)	*ieber*
…	…
(0, 4)	*er*
(0, 5)	*r*
(1, 0)	*belieber*
(1, 1)	*elieber*
…	…
(1, 6)	*er*
(1, 7)	*r*

b) See paper. Vanilla explanation of binary search and prefix searching is fine. LCP fanciness to be considered a bonus, not required to explain.

QUESTION 6: PERMUTERM INDEXES

a) See Section 3.2.1 (p. 49.). Would expand *bieber* to all rotations *bieber$*, *ieber$b*, *eber$bi*, etc.

b) Convert *bi*er* to the prefix search *er$bi**.