

## COMP6714 (13S2) MID-TERM EXAM

TIME ALLOWED: 1 HOUR

Name: \_\_\_\_\_

Student ID: \_\_\_\_\_

**NOTE:**

- (1) Answer the questions briefly. Lengthy but irrelevant answers will be penalized.
- (2) In most of the questions, you need to show major steps.

Q1. (*30 marks*)

Answer the following questions.

- (1) What is the heuristic method to determine the execution order using the binary list merge algorithm to answer *conjunctive* keyword queries? Given a counter-example where this heuristic does not work well.

**Your Answer:**

- (2) Given at least two reasons (with simple examples) why language identification is important when indexing documents.

**Your Answer:**

- (3) Why specialized algorithms are needed to construct inverted index for large document collections?

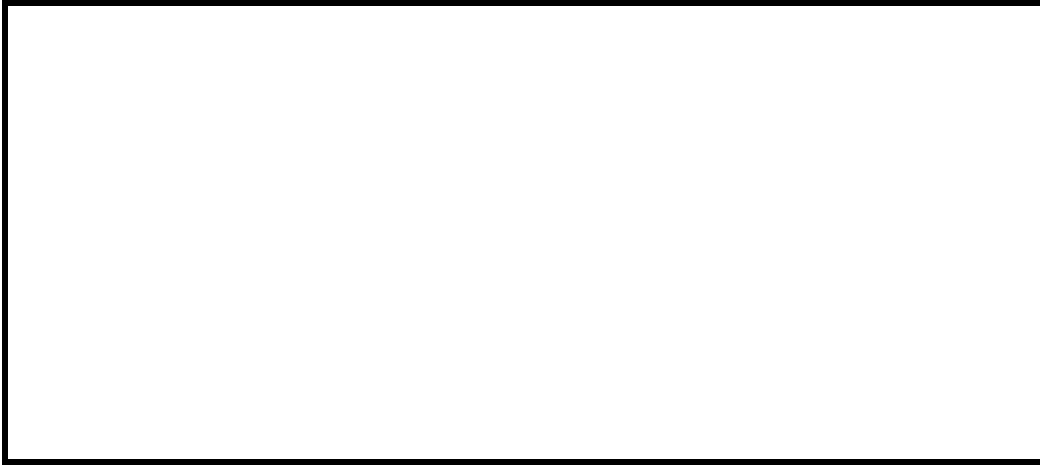
**Your Answer:**

- (4) What is the Heaps' Law and what is the Zipf's Law?

**Your Answer:**

- (5) Give an intuitive explanation of the *two* major factors we consider when choosing the best candidate for *context-sensitive* spelling correction. (For example, **taw** may be corrected to either **the** or **thaw**)

**Your Answer:**



Q2. (20 marks)

Complete the pseudocode of the function Q2 (shown below) that answers the Boolean keyword query “ $A$  AND (NOT  $B$ )”, where  $A$  and  $B$  are two different keywords. Make sure your pseudocode is easily readable (i.e., with indentation and comments if necessary).

You can assume the following functions/methods on LA or LB:

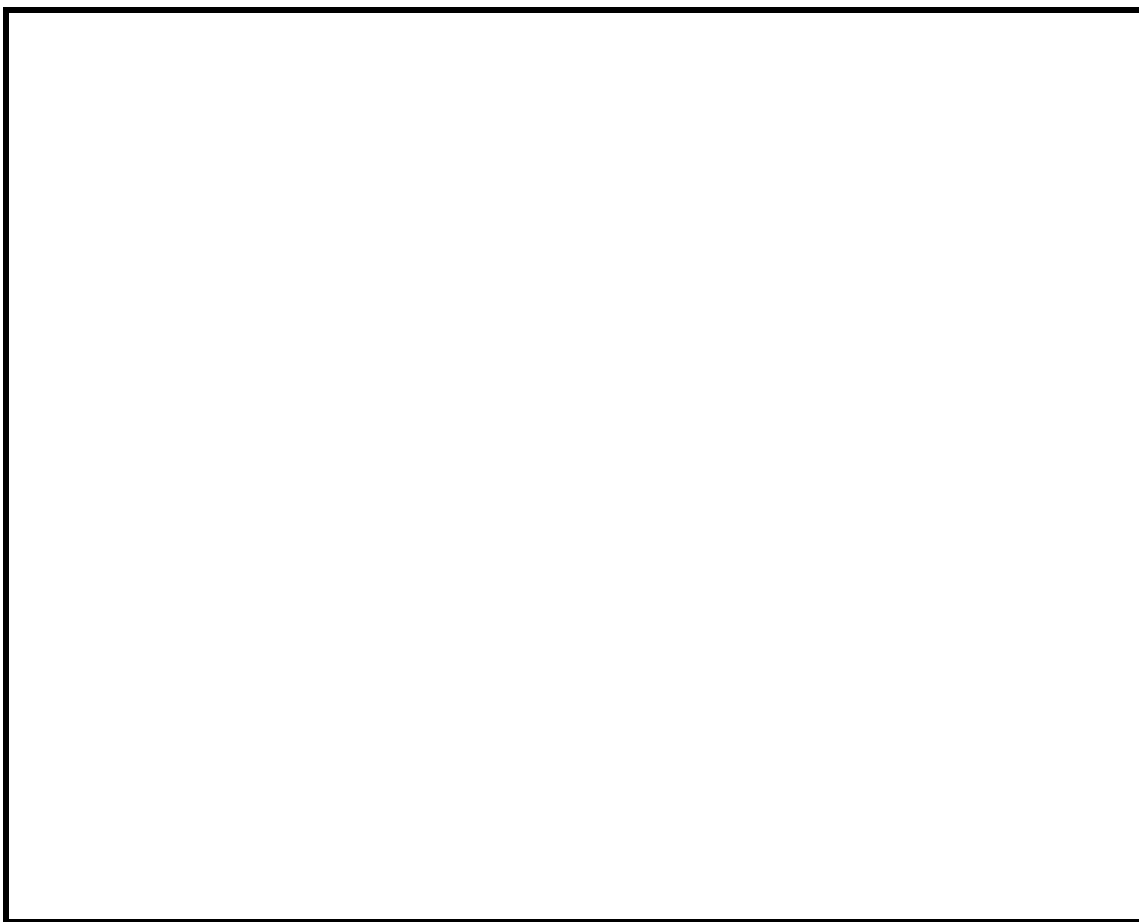
- `cur()` returns the current docID in the list;
- `eol()` returns **TRUE** if the current list is exhausted;
- `next()` moves the cursor to the next posting.

---

```
function Q2(LA, LB)
  // Let LA and LB be the corresponding inverted lists
  // of A and B, respectively
  ANSWER = []
  ...
  ...
  return ANSWER
end
```

---

**Your Answer:**



Q3. (25 marks)

Given a list of sorted strings of possibly different length, describe a method to use binary search to answer prefix query  $P^*$ , where  $P$  is a non-empty string (You may use an additional  $O(n)$  space for some auxiliary data structures)? Why  $B^+$ -tree index is still preferred over this binary-search-based solution for large collection of strings?

**Your Answer:**



Q4. (25 marks)

Consider using one of the three dynamic indexing methods, namely *immediate merge*, *no merge*, and *logarithmic merge*, to build the inverted index for a collection. Let  $|C|$  be the total number of bytes of the documents in the collection,  $M$  be the memory size in bytes, and  $B$  be the number of bytes in a disk block.

We only consider the total number of blocks read from or written to the disk as the I/O cost; and you can safely ignore the ceiling or floor functions in the analysis. For example, reading 1000 bytes from the disk costs  $\frac{1000}{b}$  I/Os.

- (1) How many sub-indexes will the *no merge* method create? What is the total I/O cost of indexing the collection?
- (2) How many sub-indexes will the *immediate merge* method create? What is the total I/O cost of indexing the collection?
- (3) How many sub-indexes will the *logarithm merge* method create (you may consider the worst case)? What is the total I/O cost of indexing the collection?

**Your Answer:**

