(1) Over the last few weeks, we saw several papers that use the following technique. In order to index an object O, they first transform O into an alternate representation O’. This alternate representation is then indexed. To answer a query Q, the query is transformed into a query Q’ such that if Q matches O, we are guaranteed that Q’ matches O’. Which of the following papers does not use this technique? Note that more than one may be a correct answer; circle all that apply.

(a) An Efficient Index Structure for String Databases
(b) Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases
(c) Fast Subsequence Matching in Time-Series Databases
(d) The String B-Tree: A New Data Structure...

(2) When indexing subsequences, one problem that must be dealt with is that the subsequence starting at each position in each sequence must be indexed. How does the paper “Fast Subsequence Matching in Time-Series Databases” recommend that we deal with this issue?

(a) Use a discrete FFT rather than the continuous version of the transform
(b) Index the first few Fourier coefficients for each subsequence starting at each point in each sequence
(c) Index MBRs that enclose the first few Fourier coefficients for each subsequence starting at each point in each sequence
(d) Ask a separate query for each starting point in the query sequence

(3) In the paper “A Database Index for Biological Sequences” the authors basically describe how to implement a suffix tree. However, a big problem is that suffix trees don’t work well on disk because they are too slow to build since they require too many random disk I/Os. Which of the following is not a way that the authors overcome this?

(a) They forgo a guaranteed O(n) running time for a possible O(n^2) running time that works well on disk and tends to have O(nlogn) performance on biological data
(b) They make multiple passes through the data, each time only building a portion of the tree that is small enough to hold in memory
(c) They forgo the use of suffix links, which can require random disk I/Os
(d) They do not store the suffix number of each node

(4) In the OASIS paper, what is the priority queue for?

(a) This ensures that we only expend energy to find the best matches, suffix tree nodes are maintained in order of the best possible score for each node
(b) This is to allow fast concatenation of all of the strings traversed in the suffix tree
(c) This allows for O(nm/log2n) disk I/Os in order to build the S-W matrix
(d) This allows for fast traversal of the suffix tree by ensuring that we only traverse to nodes that are currently buffered in memory.

(5) In the paper “An Efficient Index Structure For String Databases”, what is the difference between the entries of level i and level i + 1 of the MRS data structure?

(a) The entries of level i + 1 correspond to representations of sliding windows twice as big as the entries on level i.
(b) The entries of level i are representations of the ith string in the database, as opposed to the i + 1th string.
(c) The entries of level i + 1 keep more wavelet coefficients than entries on level i.
(d) The entries of level i + 1 have larger MBRs than the entries on level i.

(6) What is a “twig join”?

(a) This is an operation that computes all similar substructures within two XML documents.
(b) This is an operation that tries to find all occurrences of a small fragment of a tree in an XML document.
(c) This is an operation that tries to find all repeated tree fragments in an XML document.
(d) This is an operation that tries to find all occurrences of a given regular expression in an XML document.

(7) In the X-Rank paper, we are given two options for locating query keywords: DIL and RDIL. Which of the following best describes the difference between them?

(a) The RDIL list is useful when the document collection is such that space overhead might be a problem.
(b) The RDIL lists are not indexed, but the DIL lists are.
(c) The DIL lists are organized by node ID, whereas the RDIL lists are organized by rank.
(d) The DIL list is most useful when multiple query keywords are used.

(8) In the XISS paper, how do the author make their XML numbering scheme suitable for dynamic documents?

(a) They describe how deletions and insertions can be performed by only updating local preorder and postorder numbers.
(b) It won’t work with dynamic documents.
(c) After an insertion, every node below the parent of the inserted node needs to be re-traversed and re-numbered.
(d) When traversing the tree to perform the numbering, additional unused slots are reserved to accommodate future insertions.