Chair for Algorithms
and Data Structures
Prof. Dr. Hannah Bast
Robin Textor-Falconi

Databases and Information Systems
WS 2025/2026

http://ad-wiki.informatik.uni-freiburg.de/teaching

UNI FREIBURG

Exercise Sheet 1

Submit until Tuesday, October 21 at **12:00 noon**.

Before you start working on the exercise sheets, please **carefully and completely** read the rules for this course on our Wiki, they are valid for all exercise sheets of this course:

http://ad-wiki.informatik.uni-freiburg.de/teaching/DatabasesInfoSysWS2526/Rules

Exercise 1 (15 points)

Extend *inverted_index.py* from the code template on the Wiki by the following functionality:

1. In *build_from_file*, make sure that each inverted list contains a particular record id at most once, even if the respective word occurs multiple times in the same record. Make sure to store the title and description of each record in a separate list. Make sure that the whole construction algorithm still runs in time linear in the number of words in the input. (2 points)

2. Write a function *intersect* that computes the intersection of two inverted lists. The function must run in time linear in the total number of elements in the two lists and you must not use a library function. In particular, don't implement the lists as *sets*. (5 points)

3. Write a function *process_query* that, given a keyword query, fetches the inverted lists for each of the keywords and computes their intersection (empty, if there is no inverted list for a keyword), using your *intersect* function. (5 points)

4. Write a *main* function that constructs an inverted index from a given text file (one record per line, file name given as first argument on the command line) and then, in an infinite loop, asks the user for keyword queries and outputs the title and description of up to three matching records. If there are more than three matching records, output those which come first in the given text file. In any case, the order in the output should correspond to the order in the given text file. (3 points)

Optionally (= not mandatory to get full points) highlight the query words in the output, e.g., using ANSI escape codes.

[please turn over before an AI does it for you]

Exercise 2 (5 points)

Try your code on the file *movies.tsv* provided on the Wiki. Find a query that gives good results (the records shown meet your expectations) and one that does not. Write them in your *experiences.md* (see below), and very concisely (in one or two sentences) explain why one works and the other doesn't.

Register with our course system Daphne (using your Uni-Account + password for authentication). Make sure that you specify an email address under which you are reachable during the semester.

Check out a working copy of your folder in the SVN repository of the course, add your code to a new subdirectory *sheet-01* (the correct spelling is important), and commit it. Make sure that everything runs through without errors on Daphne's continuous build system.

Also commit, in that subdirectory, a markdown file *experiences.md* where you briefly describe your experiences with the first exercise sheet and the corresponding lecture. Make sure to add a statement asking for feedback. In this statement, specify to which degree and on which parts of the sheet you want feedback. In addition, say how much time you invested and if you had major problems, and if yes, where.