Search Engine Coding Challenge

Implement a simple search engine.

A search engine contains a set of documents. Each document consists of a unique ID and a list of tokens. The search engine responds to queries by finding documents which contain certain tokens and returning their IDs.

The program should accept a sequence of commands from standard input and respond on standard output. Commands are terminated by newlines. There are two types of commands:

1. The index command
2. The query command

The index Command

index *doc-id token1 … tokenN*

The index command adds a document to the index. The doc-id is an integer. Tokens are arbitrary alphanumeric strings. A document can contain an arbitrary number of tokens greater than zero. The same token may occur more than once in a document. If the doc-id in an index command is the same as in a previously seen index command, the previously stored document should be completely replaced (i.e., only the tokens from the latest command should be associated with the doc-id).

Examples:

index 1 soup tomato cream salt

index 2 cake sugar eggs flour sugar cocoa cream butter

index 1 bread butter salt

index 3 soup fish potato salt pepper

When the program successfully processes an index command, it should output

index ok *doc-id*

If the program sees an invalid index command (e.g, a token contains a non-alphanumeric character, or a doc-id is not an integer) it should report an error to standard output and continue processing commands. The error output should have the following form

index error *error message*

The query Command

query *expression*

The query command returns a list of document ids that have the matching tokens provided in the *expression*

For Senior Engineers:

Where expression is an arbitrary expression composed of alphanumeric tokens and the special symbols &, |, (, and). The most simple expression is a single token, and the result of executing this query is a list of the IDs of the documents that contain the token. More complex expressions can be built built using the operations of set conjunction (denoted by &) and disjunction (denoted by |). The & and | operation have equal precedence and are commutative and associative. Parentheses have the standard meaning. Parentheses are mandatory: *a* | *b* | *c* is not valid, *(a* | *b)* | *c* must be used (this is to make parsing queries simpler).

Logically, to execute the query the program looks at every document previously specified by the index command, checks if the document matches the query, and outputs the doc-id if it does. However this is suboptimal and much more efficient implementations exist.

Upon reading the query command the program should execute the query and produce the following output

query results *doc-id1 doc-id2 …*

The doc-ids of the matching documents can be output in arbitrary order. If there is an error, the output should be

query error *error message*

Examples, given the index commands shown in the example above:

in: query butter

out: query results 2 1

in: query sugar

out: query results 2

in: query soup

out: query results 3

in: query (butter | potato) & salt

out: query results 1 3

Please implement a search engine in any language you like. You may use standard and open-source libraries and you may consult online materials or books. Please try to make the implementation reasonably fast, for a fairly large number of documents, say in the tens of thousands (you don't need to test it on large sets of documents, just think about what this would be for your design). Query speed is more important than indexing speed.