```
package ir4;

import java.util.*;


public class PrecisionRecallCalculator {


   public static void main(String[] args) {

      // Sample documents and queries

      Map<String, String> documents = new HashMap<>();

      documents.put("Document1", "Information retrieval (IR) is the process of obtaining information from a collection of resources. It involves techniques like indexing and searching.");

      documents.put("Document2", "Text analysis involves various techniques for extracting insights from textual data. It includes tasks like sentiment analysis and named entity recognition.");


      Map<String, String> queries = new HashMap<>();

      queries.put("Query1", "information retrieval techniques obtaining information from a collection of resources");

      queries.put("Query2", "text analysis process");


      // Relevance judgments (manually annotated for simplicity)

      Map<String, List<String>> relevanceJudgments = new HashMap<>();

      relevanceJudgments.put("Query1", Arrays.asList("Document1"));

      relevanceJudgments.put("Query2", Arrays.asList("Document2"));


      // Calculate precision and recall for each query

      for (String queryId : queries.keySet()) {

         String query = queries.get(queryId);

         List<String> relevantDocuments = relevanceJudgments.getOrDefault(queryId, Collections.emptyList());


         // Tokenize the query and document

         Set<String> queryTokens = new HashSet<>(Arrays.asList(query.split(" ")));

         Set<String> documentTokens = new HashSet<>(Arrays.asList(documents.get(relevantDocuments.get(0)).split(" ")));
```

```
        // Calculate precision and recall
        double precision = calculatePrecision(queryTokens, documentTokens);
        double recall = calculateRecall(queryTokens, documentTokens);

        System.out.println("Query: " + query);
        System.out.println("Relevant Documents: " + relevantDocuments);
        System.out.println("Precision: " + precision);
        System.out.println("Recall: " + recall);
        System.out.println();
    }
}

private static double calculatePrecision(Set<String> queryTokens, Set<String> documentTokens) {
    // Calculate precision
    Set<String> intersection = new HashSet<>(queryTokens);
    intersection.retainAll(documentTokens);

    if (queryTokens.isEmpty()) {
        return 0.0;
    }

    return (double) intersection.size() / queryTokens.size();
}

private static double calculateRecall(Set<String> queryTokens, Set<String> documentTokens) {
    // Calculate recall
    Set<String> intersection = new HashSet<>(queryTokens);
    intersection.retainAll(documentTokens);

    if (documentTokens.isEmpty()) {
```

```
            return 0.0;
        }

        return (double) intersection.size() / documentTokens.size();
    }
}
```

Output :