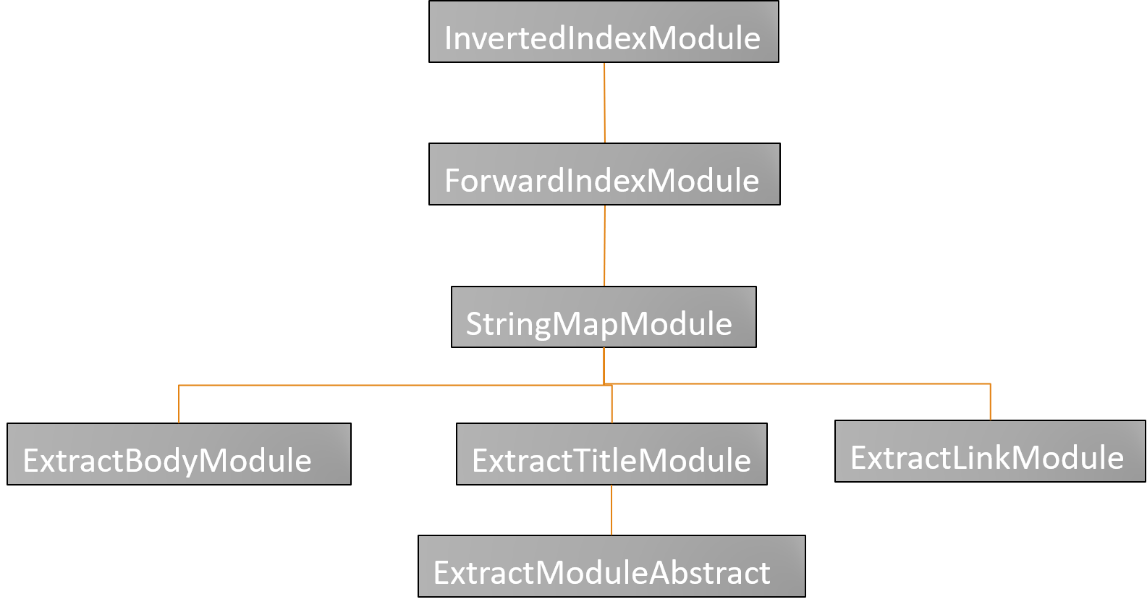
# Test Report

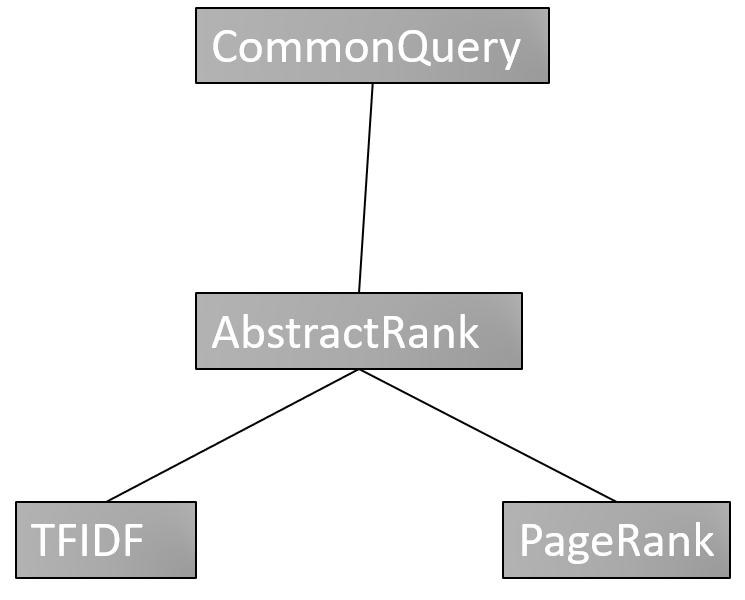
## Hierarchy Diagram

**Pre-Process**



ExtractModuleAbstract is a template to extract some content from a webpage. It is extended with three concrete classes, ExtractBodyModule, ExtractTitleModule, ExtractLinkModule to extract body, title, link from webpage respectively. StringMapModule depends on the ExtractModuleAbstract to create a map between URL and the part of content extracted. ForwardIndexModule and InvertedIndexModule will hash URL to content or hash content to URL respectively.

**Query**



## Methodology

Modified top-down method is used in our project to test the two parts of our project. The reason why we choose modified top-down method is that it focuses on the functionality testing of the system. One of the disadvantages is that there are a lot of testing stubs required. But as the structure is not that complex, the number of stub classes is acceptable.

## Testing Order

According to the hierarchy diagram, the order of modified top-down testing will be:

**Pre-process**

Unit Testing level 1:

ExtractBodyModule (Test Stub: StringMapModule)

ExtractTitleModule (Test Stub: StringMapModule)

ExtractLinkModule (Test Stub: StringMapModule)

Unit Testing level 2:

StringMapModule (Test Stub: ForwardIndexModule)

Integration Testing level 1:

ExtractBodyModule + StringMapModule (Test Stub: ForwardIndexModule)

ExtractTitleModule + StringMapModule (Test Stub: ForwardIndexModule)

ExtractLinkModule + StringMapModule (Test Stub: ForwardIndexModule)

Unit Testing level 3:

ForwardIndexModule (Test Stub: InvertedIndexModule)

Integration Testing level 2:

ExtractBodyModule + StringMapModule + ForwardIndexModule (Test Stub: InvertedIndexModule)

ExtractTitleModule + StringMapModule + ForwardIndexModule (Test Stub: InvertedIndexModule)

ExtractLinkModule + StringMapModule + ForwardIndexModule (Test Stub: InvertedIndexModule)

Unit Testing level 4:

InvertedIndexModule (Test Stub: N/A)

System Testing:

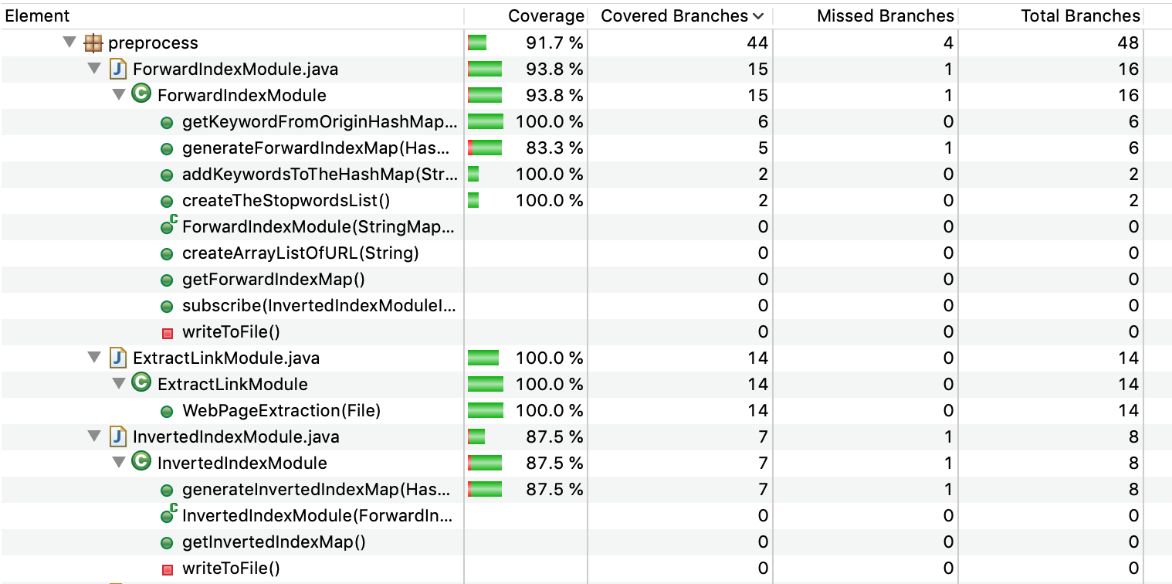
ExtractBodyModule + StringMapModule + ForwardIndexModule + InvertedIndexModule (Test Stub:N/A)

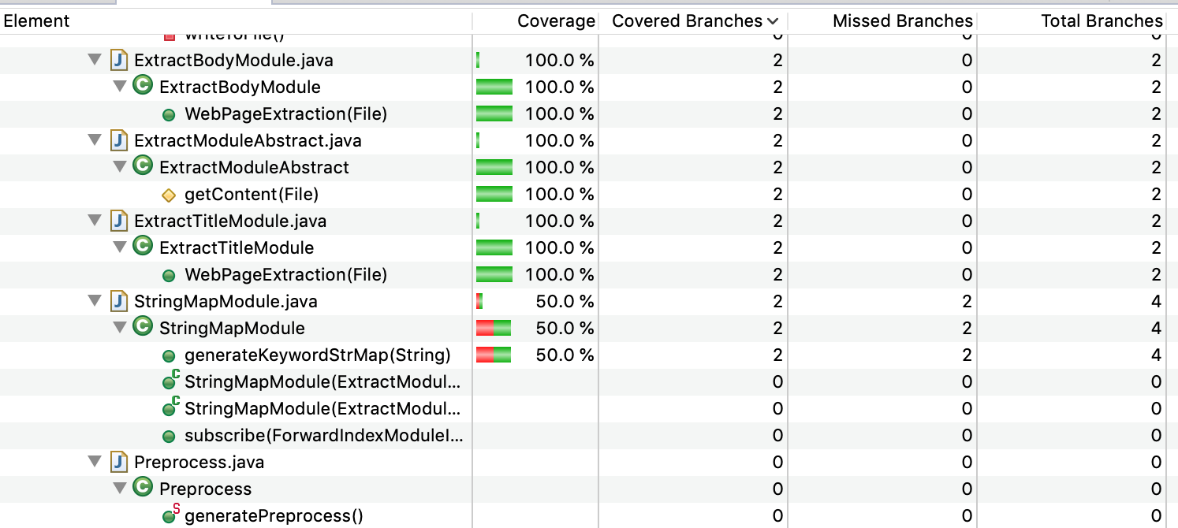
ExtractTitleModule + StringMapModule + ForwardIndexModule + InvertedIndexModule (Test Stub:N/A)

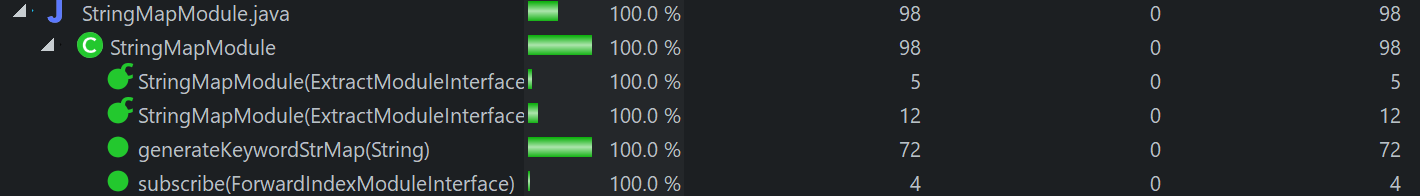
ExtractLinkModule + StringMapModule + ForwardIndexModule + InvertedIndexModule (Test Stub:N/A)

## Coverage

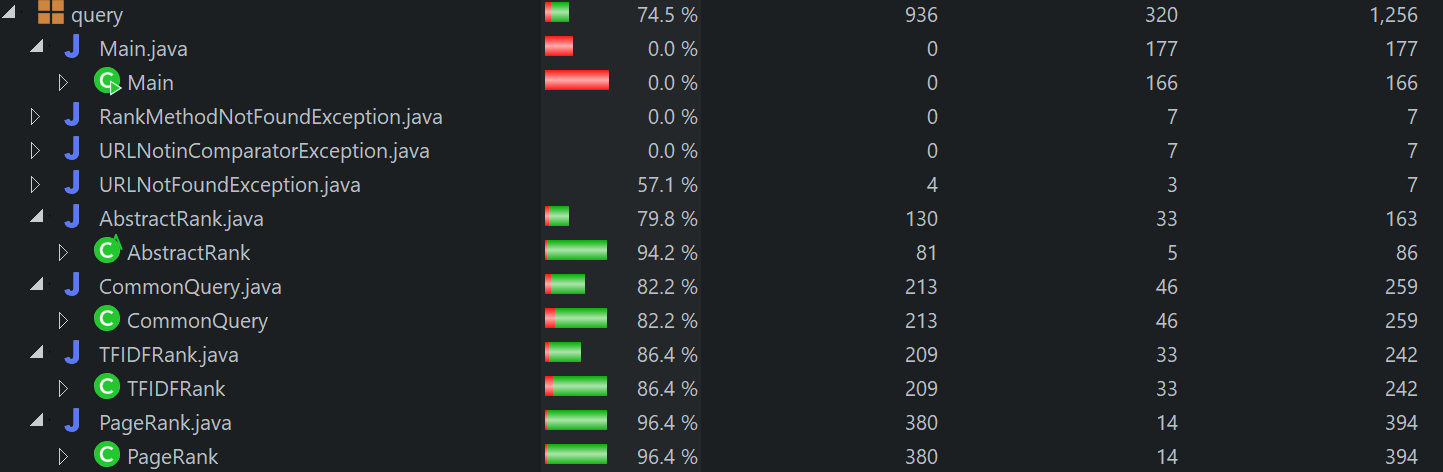
**Statement Coverage**







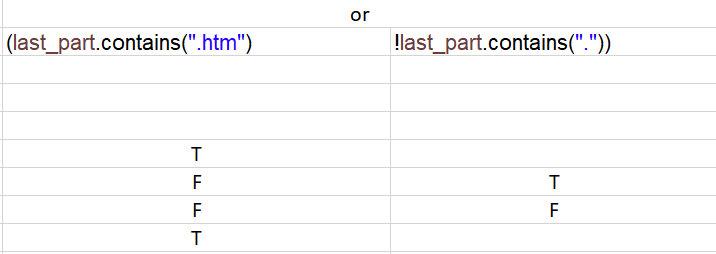
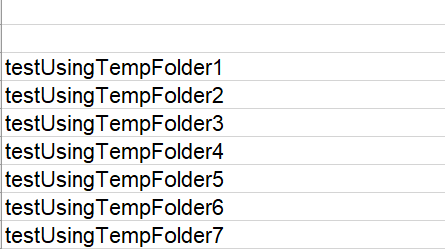
The coverage of pre-process part reaches 85% in each class.



The statement coverage of query part is pulled down partly because the Main.java class is not tested. It is hard to improve because it is ac activity interacted with users and the interaction is hard to simulate. Overall, the most important part, CommonQuery and Rank classes, have reached satisfactory statement coverage.

**MC/DC Predicate Coverage**

To reach a full MC/DC predicate coverage, test cases should cover decision coverage and condition coverage. What’s more, each condition in the decision should be able to affect the result independently. Take the following piece of code as example:



Two conditions are connected with or logic. Test cases testUsingTempFolder4 and testUsingTempFolder6 satisfies decision coverage and it also shows that last\_part.contains(“.htm”) can affect the final decision independently. Test cases testUsingTempFolder4 and testUsingTempFolder5 satisfies condition coverage and it also shows that !last\_part.contains(“”) can affect the result independently. So the testcases testUsingTempFolder4, testUsingTempFolder5 and testUsingTempFolder6 together can satisfy MC/DC predicate coverage.