

October University for Modern Sciences & Arts

**Faculty of Computer Science**

CS313

Data Storage

**Course Project**

**Search Engine for Arabic Wikipedia**

**NAME: Yahia Mahmoud Abdel Rahman**

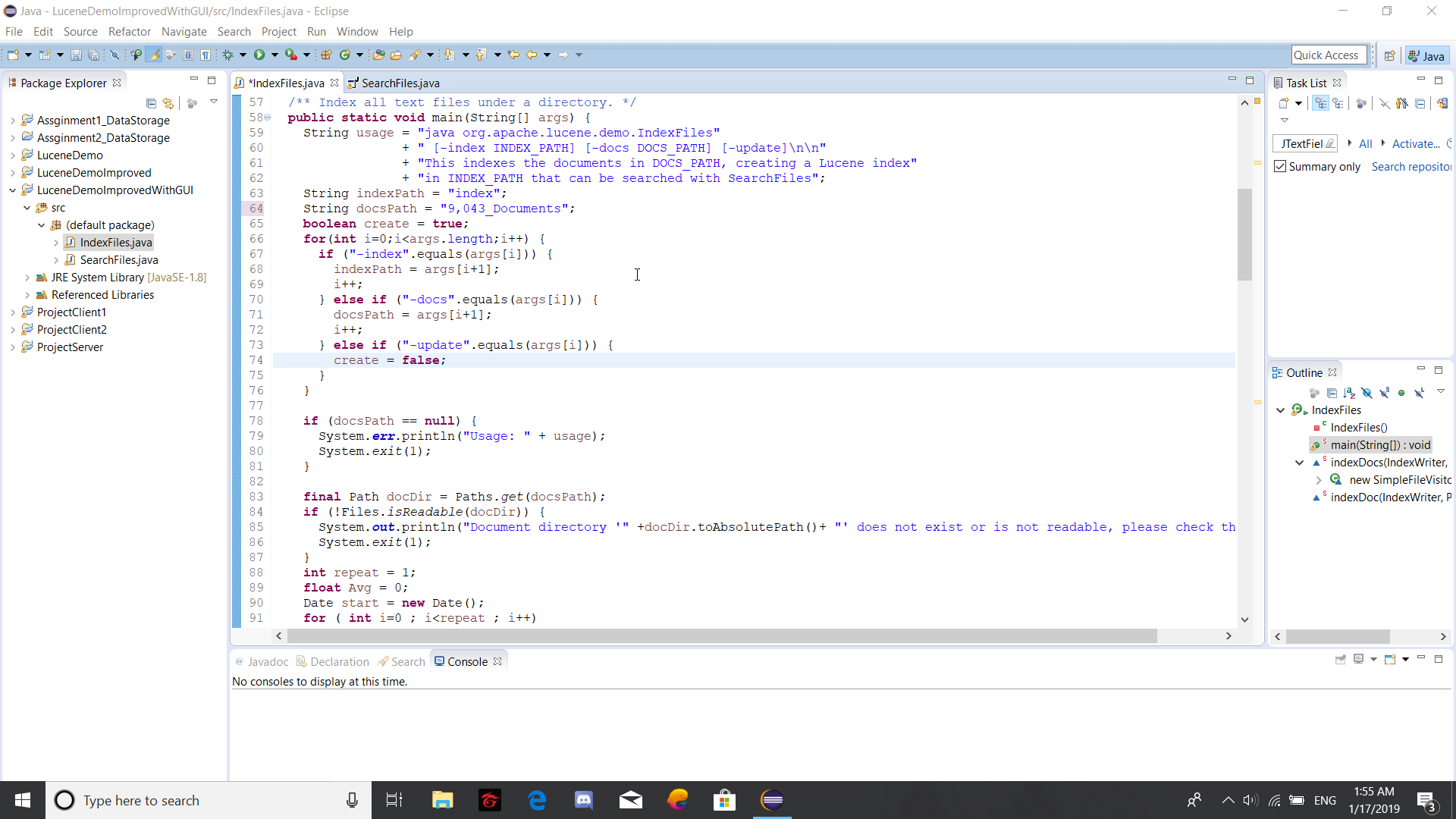
**ID: 173269**

**Search Engine For Arabic Wikipedia**

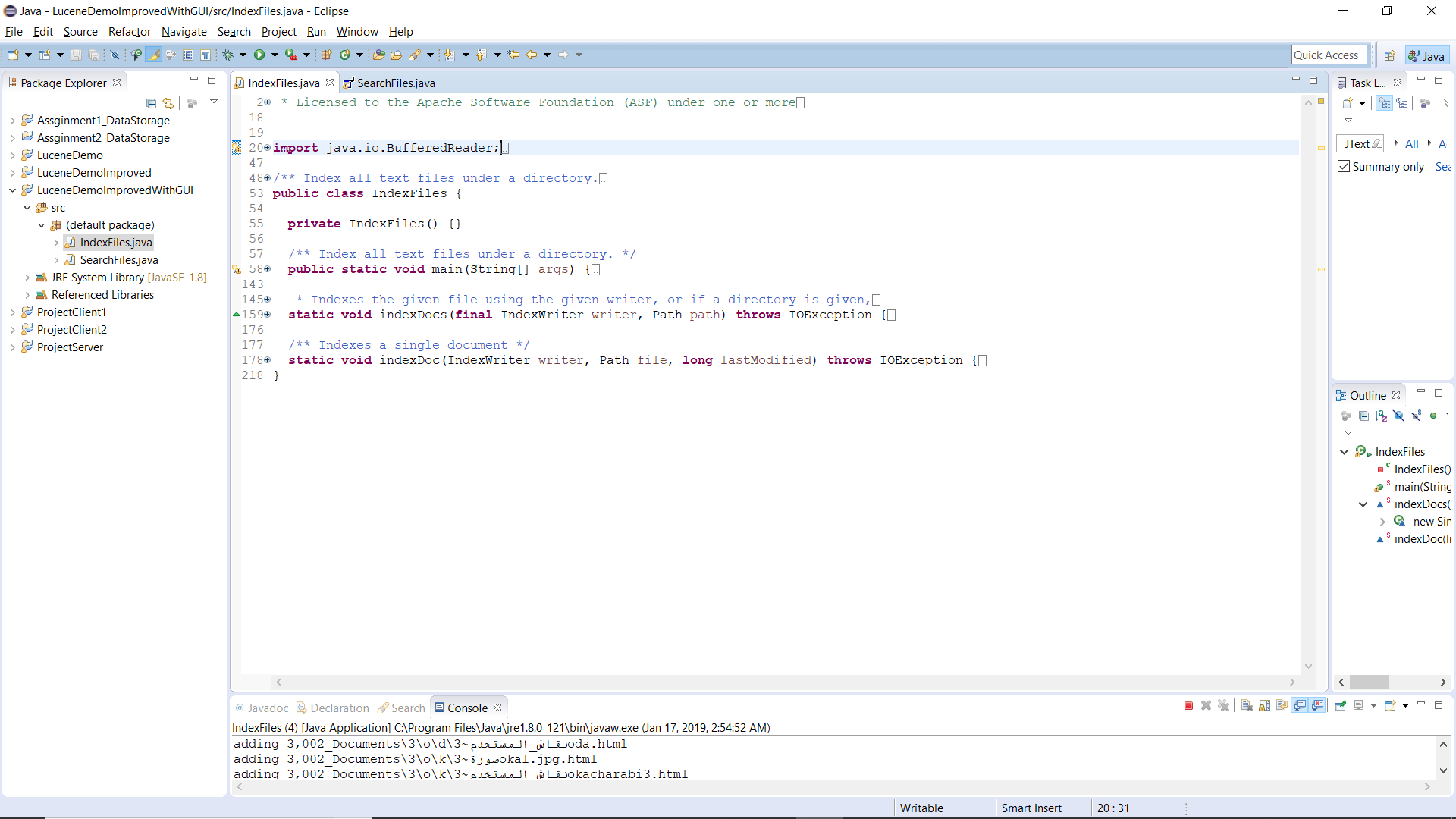
1. **Detailed Report:**
2. **Description of the problem**: I implemented a search engine for Arabic Wikipedia using lucene demo. Since it is a demo application for indexing Arabic documents from Wikipedia and then searching for queries that user need from the documents that have been indexed by the application. The application is implemented with an easy UI (User Interface) that users can handle it easy without any complexity.
3. **System architecture:** The architecture of the system was based on two parts. The first part, a front-end using java programming code with the help of eclipse which is the most widely used Java IDE. And the second part, a back-end using lucene demo libraries and two basic source codes for indexing the documents (IndexFile.java: code to create a Lucene index) and searching for queries (SearchFile.java: code to search a Lucene index).
4. **Implementation details:** The implementation of the application is consist of two main classes (IndexFiles.java and SearchFiles.java). The **first class** is used for indexing Arabic documents from Wikipedia, so it takes the location of the documents and then index them in a directory called index and it saves the documents location to use them later. When it start indexing the documents, it use some readymade methods made by Lucene. So it start analyze the doucments using Arabic analyzer method then it use IndexWriterConfig method to create a new index in the directory, removing any previous indexed documents or add new documents to an existing index. Then use IndexWriter to write them in directories and index documents. The **second class** is used to search for queries entered by the users which is tokenized and make every term on its own, then it use IndexReader method to read indexed document from the directories. After that it search about the tokenized query in the indexed documents and find the most relevant documents then return them to the user.
5. **Discussion of the results:** The improvements that is made in the application are in both IndexFiles and SearchFiles classes. First at the **IndexFiles class**, the application has been improved by using as much RAM as we can afford, so we used this method (iwc.setRAMBuffersizeMB(1024.0)) to increase the usage of the RAM up to 1 GB to increase indexing speed. And then we used this method (writer.forceMerge(1)) to maximize search performance, It has proven beneficial for search performance to always foreMerge down to one segment. However, this takes significant time. Second at the SearchFiles class, the application has been improved by an easy user interface(JFrame by java) for the user to enter queries in a text field and then click on search button then results returns in a text area. The second improvement in the SearchFiles class is using one instance of IndexSearcher method, since by sharing a single IndexSearcher across queries and across threads in the application.  
   The results of the indexing and searching for a query (ex. أنا أعرف) after and before the improvements are:
6. Before improvements:
7. Indexing 102 documents took an average time of repeating indexing 100 time equal 1633 ms and searching average time of repeating searching 100 time equal 57 ms.
8. Indexing 3,002 documents took an average time of repeating indexing 100 time equal 53448 ms and searching average time of repeating searching 100 time equal 35 ms.
9. Indexing 61,742 documents took an average time of repeating indexing 100 time equal 706,396 ms and searching average time of repeating searching 100 time equal 132 ms.
10. After Improvements:
11. Indexing 102 documents took an average time of repeating indexing 100 time equal 1,010 ms and searching average time of repeating searching 100 time equal 25 ms.
12. Indexing 3,002 documents took an average time of repeating indexing 100 time equal 43,979 ms and searching average time of repeating searching 100 time equal 33 ms.
13. Indexing 61,742 documents took an average time of repeating indexing 100 time equal 422,722 ms and searching average time of repeating searching 100 time equal 43 ms.

So there is a huge improvement, after using all these improvement methods.

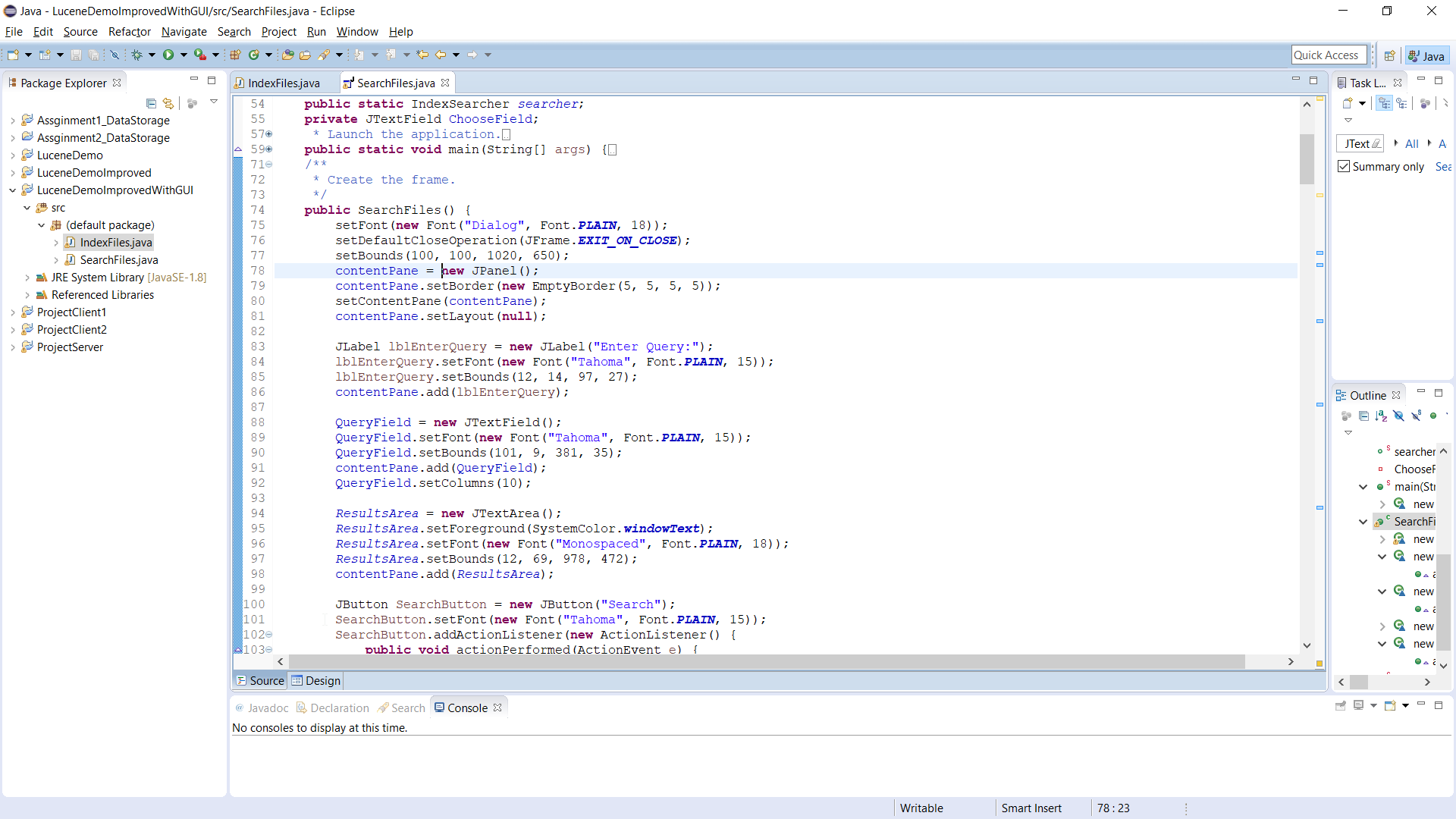
1. Screen Shots:
2. IndexFiles.Java



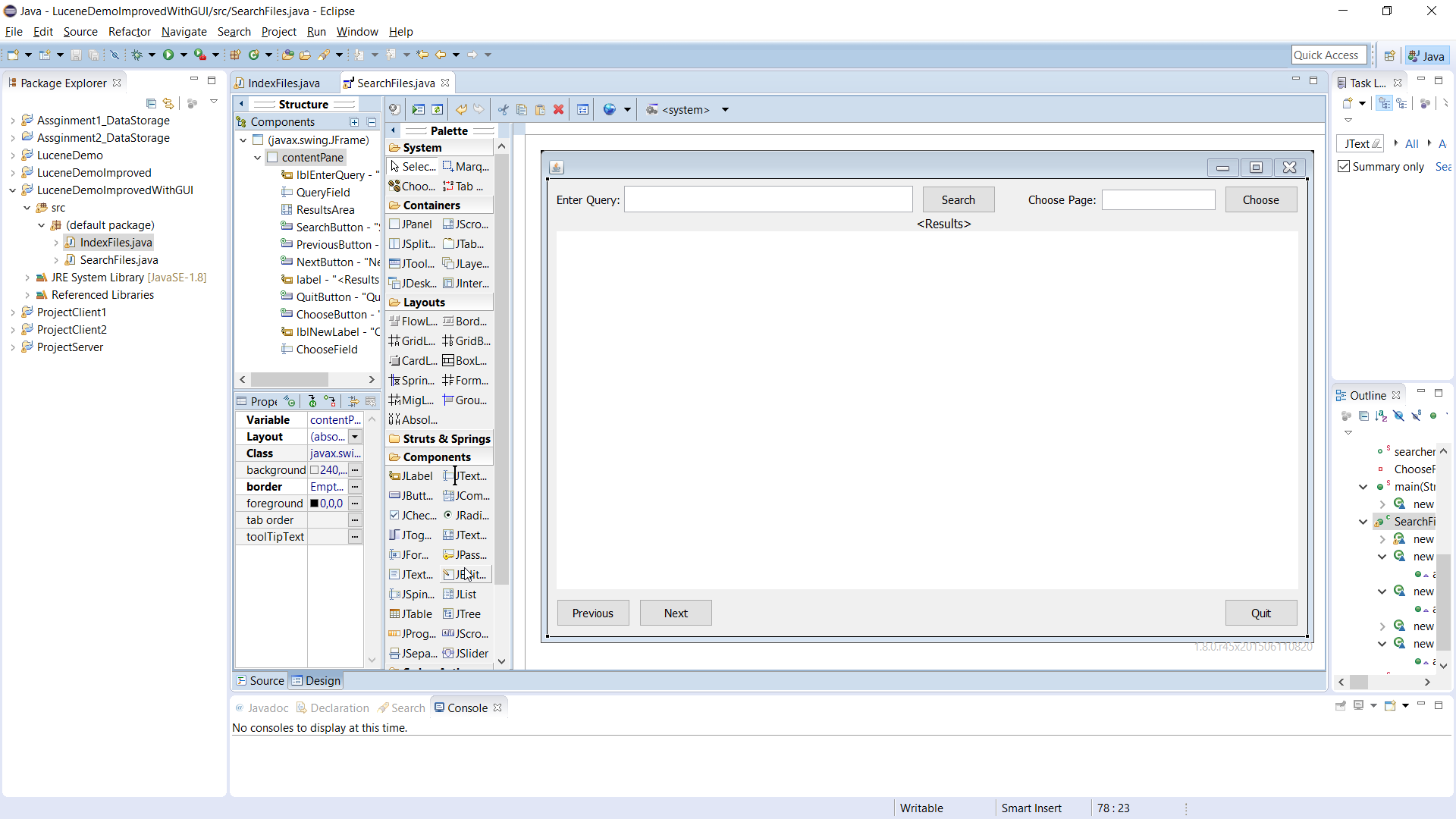
1. IndexFiles.Java Running



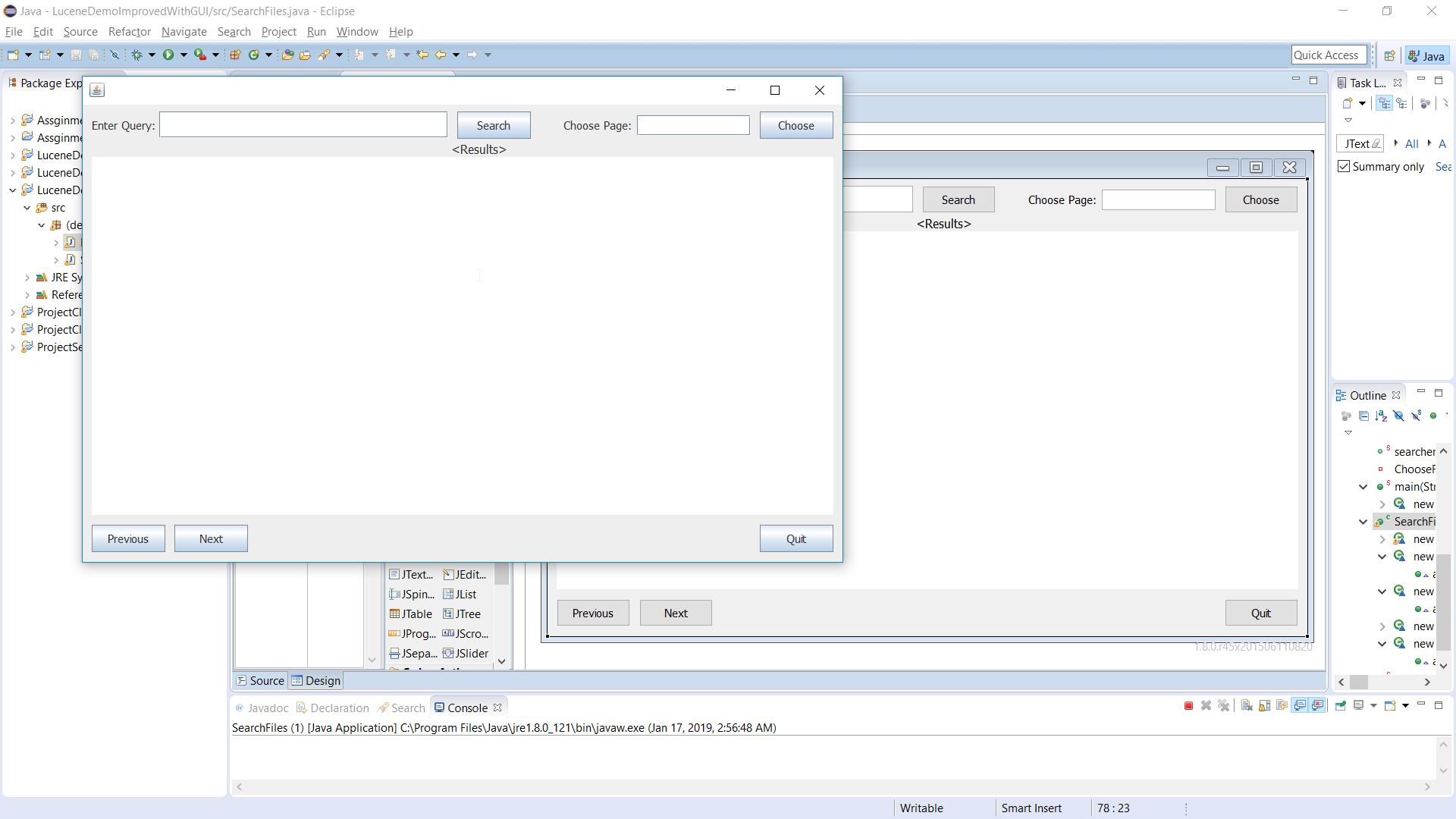
1. SearchFiles.java



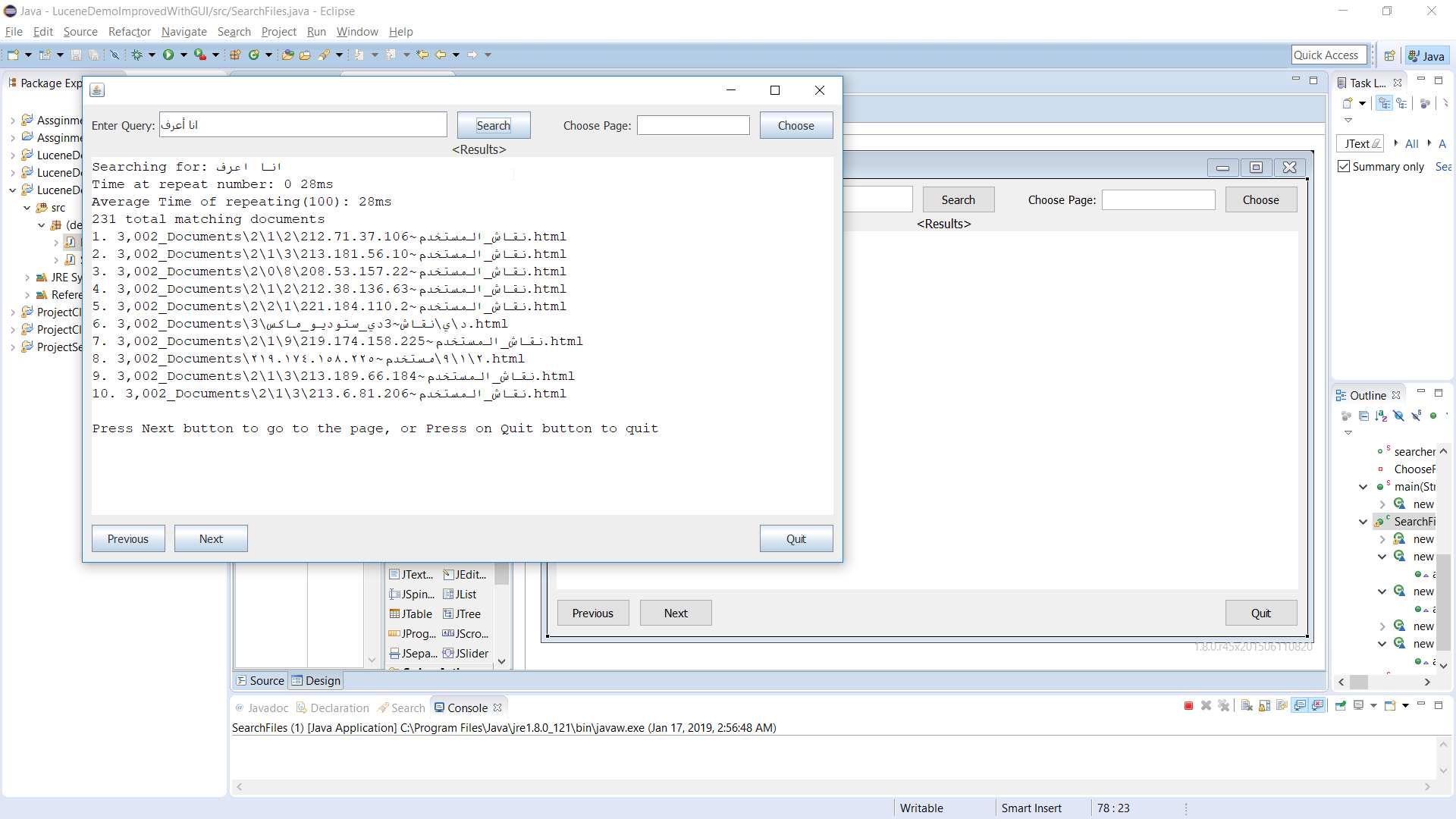
1. SearchFile.java GUI



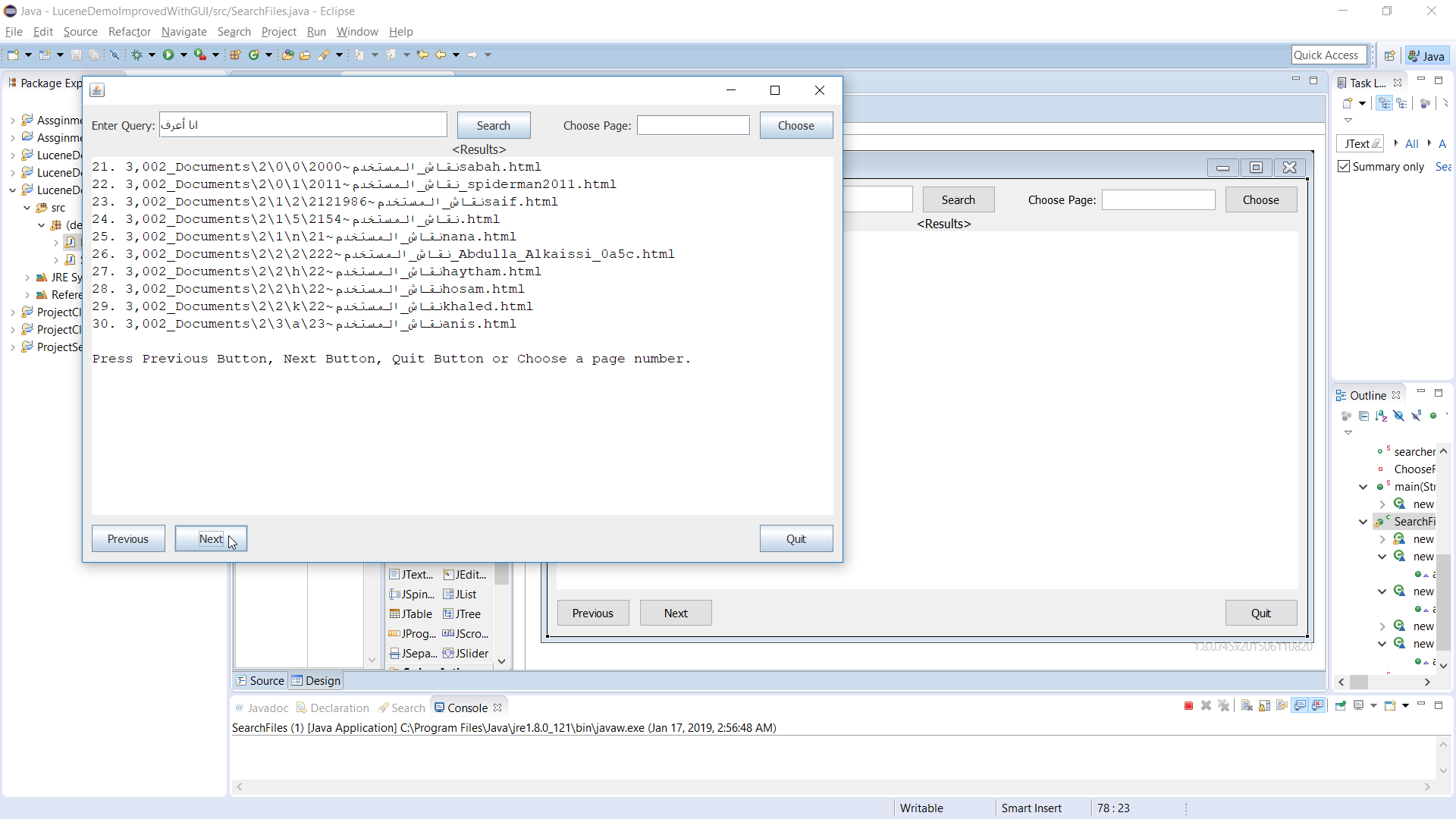
1. SearchFile.java GUI Running.



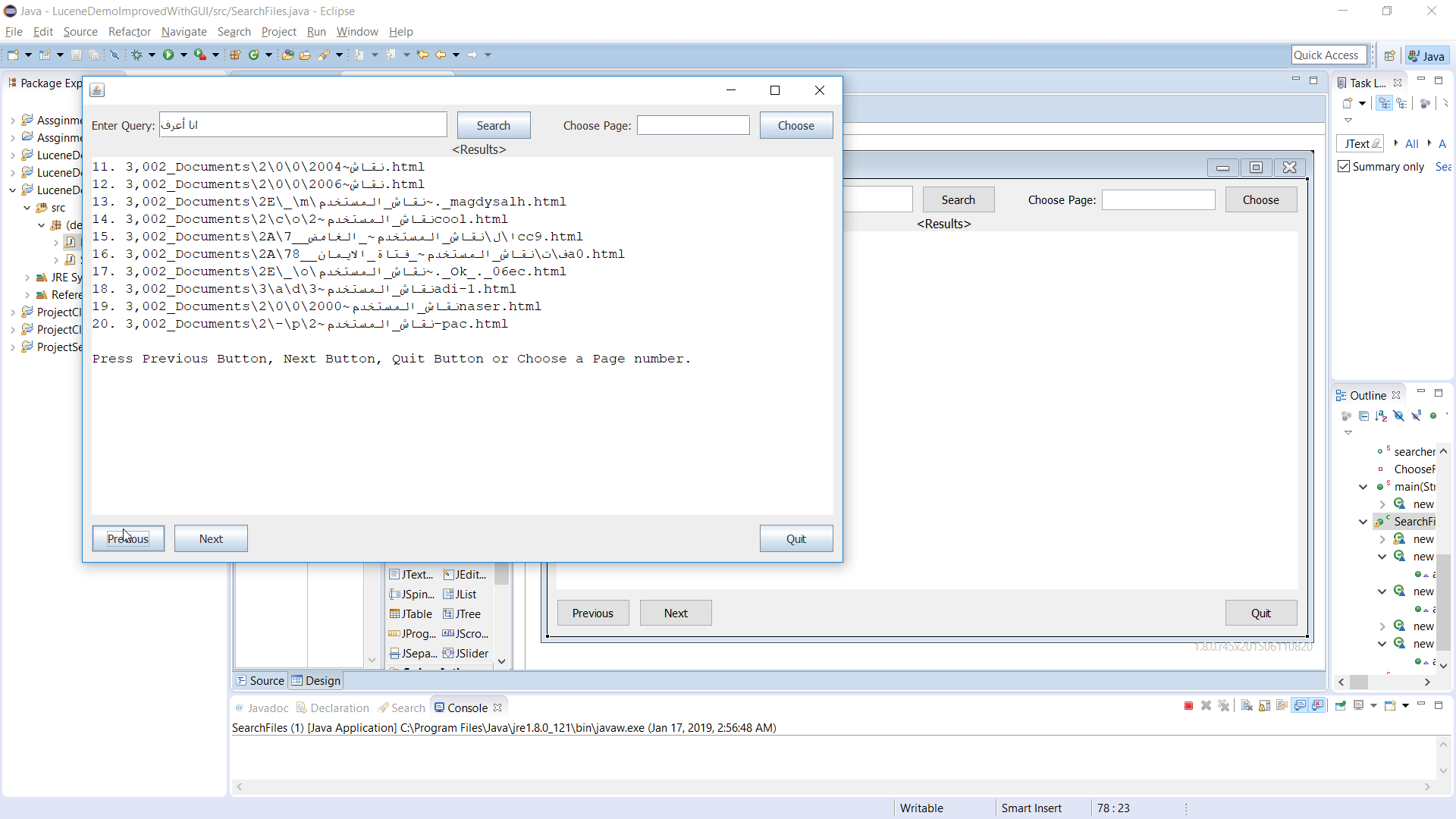
1. SearchFile.java GUI Running and entering query (أنا أعرف) above on the left corner of the form and then click on search button and Results in the TextArea below.



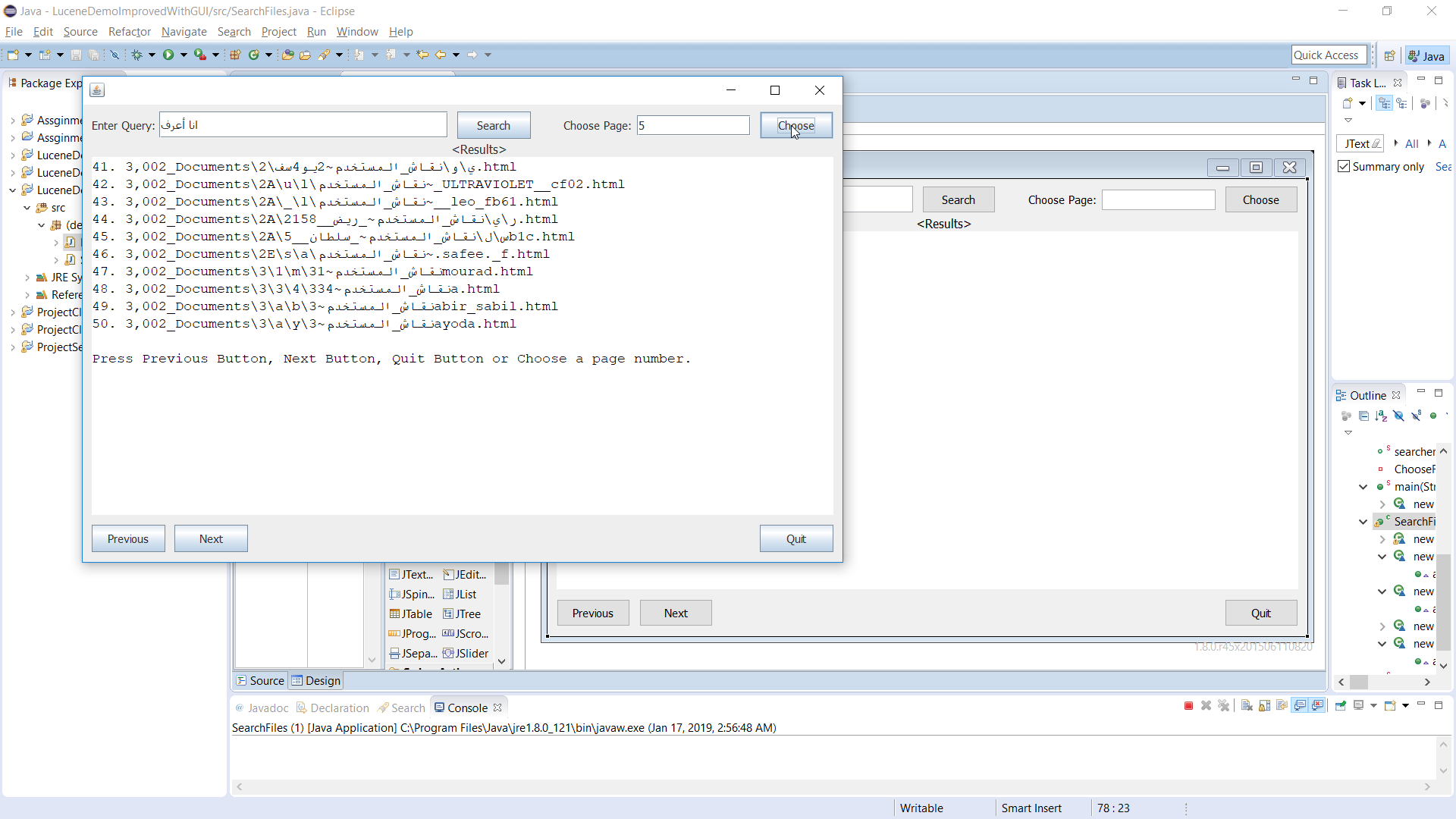
1. Clicking on Next button to show the next page in TextArea.



1. Clicking on Previous button to show the previous page in TextArea.



1. Choosing a page number at the TextField above on the right corner of the form (Page Number 5) then click on Choose button then Results will be in the TextArea.



1. Choosing a page number at the TextField above on the right (Page Number 2) then click on Choose button then Results will be in the TextArea.

