

Ministère de l'Enseignement Supérieur et de la Recherche
Scientifique

Université d'Alger 1

Faculté des Sciences

Département de Mathématiques et Informatique

Module : Recherche d'information

Création d'un moteur de recherche

Elaboré par :

BENBABA Rym Amina

Groupe 1

HAMMACHE Feriel

Groupe 1

Spécialité : ISIL

Année : L3

Année universitaire : 2017-2018

Table de matière :

- I. Introduction
- II. Problématique
- III. Motivation
- IV. Indexation
 - 1. Définition
 - 2. Les analyzers
 - 3. Comment on a indexé
- V. Recherche
 - 1. Technologie utilisées
 - 2. Implémentation
- VI. Résultats
- VII. Conclusion
- Bibliographie
- Webographie

I. Introduction :

Ce projet a comme thème l'implémentation d'un moteur de recherche qui a pour but de répondre au besoin d'utilisateur qui se manifeste par la recherche de l'information souhaitée.

La recherche d'information est le domaine qui étudie la manière de retrouver des informations dans un corpus, dans notre cas c'est d'effectuer la recherche dans la collection CACM.

II. Problématique :

La collection « CACM » contient un très grand nombre de document « 3204 » Qui représentant des articles de recherche publiés dans des revues, conférences internationale, la recherche manuelle étant un processus lent ne garantissant pas de bons résultats donc ce n'est pas évident de trouver des informations facilement et dans un brève temps, donc on a pensé à implémenter un moteur de recherche pour faciliter l'accès direct à l'information et gain de temps

III. Motivation :

Parmi les caractéristiques de notre moteur de recherche on trouve :

- Facilite la tâche de recherche.
- Un accès direct à l'information.
- Répondre aux requêtes de l'utilisateur dans un temps très réduit.
- Indiquer le score relatif à chaque document.
- Indiquer le temps de réponse et le nombre de document trouvés.
- Il traite même les champs s'ils sont en majuscule.
- Il traite les requêtes booléens.
- Interface facile à utiliser et adaptable aux utilisateurs.
- L'utilisateur à le choix de cherche dans n'importe quel « field »de chaque document.
- Si le contenu de la requête n'existe pas dans la collection le moteur.
- affiche un message de l'est n'existence et lui donne des suggestions pour la correction de contenu de la requête

IV. Indexation :

1. Définition :

Un Processus qui consiste à définir un ensemble d'éléments clés permettant de caractériser le contenu d'un document afin retrouver ce document en réponse à une requête de manière efficace et rapide. [ref 1]

2. Les analyzers :

Pour l'indexation on a utilisé les analyzers suivants :

L'auteur : keywordAnalyzer.

Le titre : StandardAnalyze.

Le résumé, la référence : ShingleAnalyzerwrapper.

Et whitespaceAnalyzr pour les espaces avec 2 termes

```
HashMap<String, Analyzer> AnalyzerPerField = new HashMap<String, Analyzer>();
PerFieldAnalyzerWrapper aWrapper= new PerFieldAnalyzerWrapper
    (new ShingleAnalyzerWrapper(), AnalyzerPerField);
AnalyzerPerField.put("Title", new StandardAnalyzer());
AnalyzerPerField.put("author", new KeywordAnalyzer());
AnalyzerPerField.put("résumé", new ShingleAnalyzerWrapper(new WhitespaceAnalyzer(),2) );
AnalyzerPerField.put("Liste", new ShingleAnalyzerWrapper(new WhitespaceAnalyzer(),2));
```

3. Comment on a indexé

On a indexer les fichiers .html

Nous avons utilisé BufferedReader pour lire le document cacm

```
fis = new FileInputStream("cacm.htm");
reader = new BufferedReader(new InputStreamReader(fis));
String line = reader.readLine();
```

Et on a utilisé les boucles while pour parcourir la collection *cacm* ligne par ligne.

- Exmple :

```
        while(!line.equals(".I "+(cont))) {
            line = reader.readLine();}

    if(line.equals(".I "+(cont)) )
{
    Document samy = new Document();

    Field ID = new TextField("Identifiant", ""+cont,Field.Store.YES);
    samy.add(ID);
    System.out.println("id"+ line);
    line =reader.readLine();
    cont++;

    if(line.equals(".T"))
    {
        line = reader.readLine();
        while(!line.equals(".I "+(cont))
            && !line.equals(".T")
            && !line.equals(".A")
            && !line.equals(".C")
            && !line.equals(".W")
            && !line.equals(".X")
            && !line.equals(".B")
            && !line.equals(".N")
            && !line.equals(".K")) {

            a=a.concat(line);
            line = reader.readLine();

        }
    }
}
```

✓ Temps d'exécution : 7min.

V. Recherche :

1. Technologies utilisées

- Apache Lucene :

Lucene est une bibliothèque open source haute performance et complète écrite en Java qui permet d'indexer et de chercher du texte. [WEB1]

- Java :

Java est un langage de programmation orienté objet créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy, présenté officiellement le 23 mai 1995 au SunWorld. [WEB2]

2. Implémentation :

On a effectué le traitement sur les 5 champs demandés :

- Identifiant.
- Titre.
- Auteur.
- Résumé.
- Référence.

En 1^{er} nous avons récupéré l'élément sélectionné par l'utilisateur avec un évènement et on le convertit en minuscule pour avoir le même résultat si l'utilisateur fait une recherche en majuscule.

```
if(titre.isSelected()){ search("titre",jTextField1.getText().toLowerCase());}  
if(auteur.isSelected()){ search("author",jTextField1.getText().toLowerCase());}  
if(resume.isSelected()){ search("résumé",jTextField1.getText().toLowerCase());}  
if(ref.isSelected()){ search("Liste",jTextField1.getText().toLowerCase());}  
if(id.isSelected()){ search("Identifiant",jTextField1.getText().toLowerCase());}
```

En utilisant les booléens.

On a gardé le texte saisi par l'utilisateur dans un tableau avec la taille de texte saisi

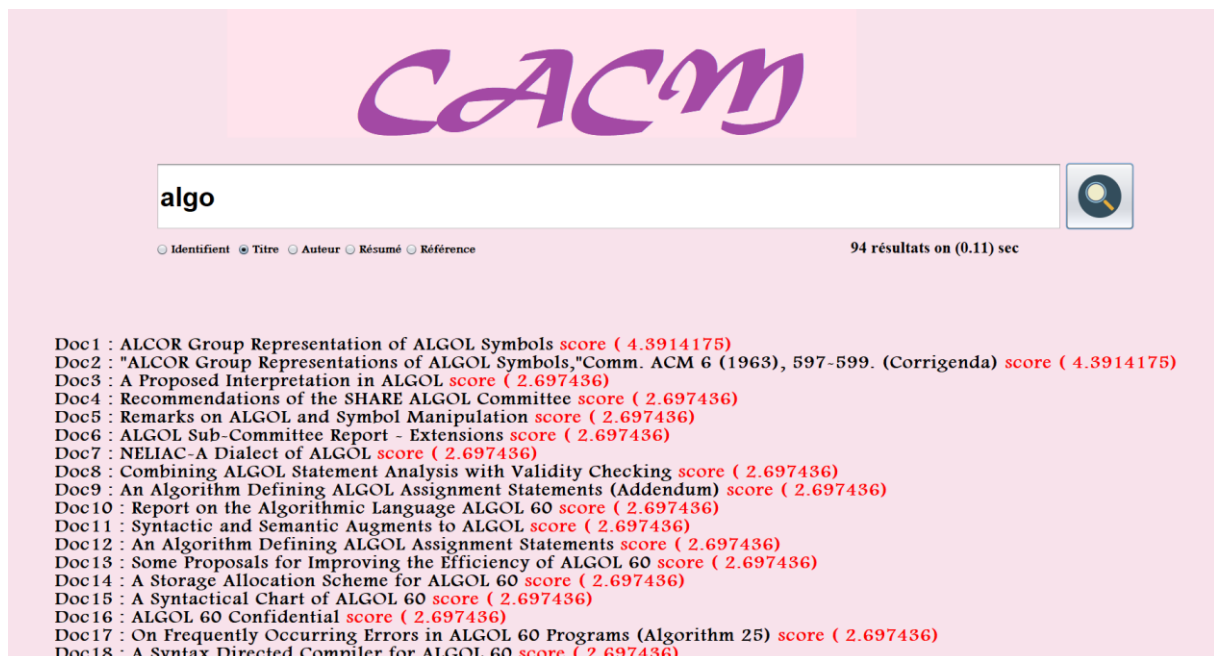
S'il trouve un symbole concaténer avec un mot il supprime le symbole et il cherche sur le mot

Voilà un petit code plus détaillé

```
BooleanQuery b = new BooleanQuery();
for(int j=0; j<tab.length;j++) {
    if (tab[j].charAt(0)=='+') {
        if (tab[j].length()!=1) {
            tab[j]=tab[j].substring(1);
            FuzzyQuery card = new FuzzyQuery(new Term(field, tab[j]),2);
            b.add(card, BooleanClause.Occur.MUST); }
        else
            {
                j=j+1;
                FuzzyQuery card = new FuzzyQuery(new Term(field, tab[j]),2);
                b.add(card, BooleanClause.Occur.MUST);
            }
    }
}
```

VI. Résultats :

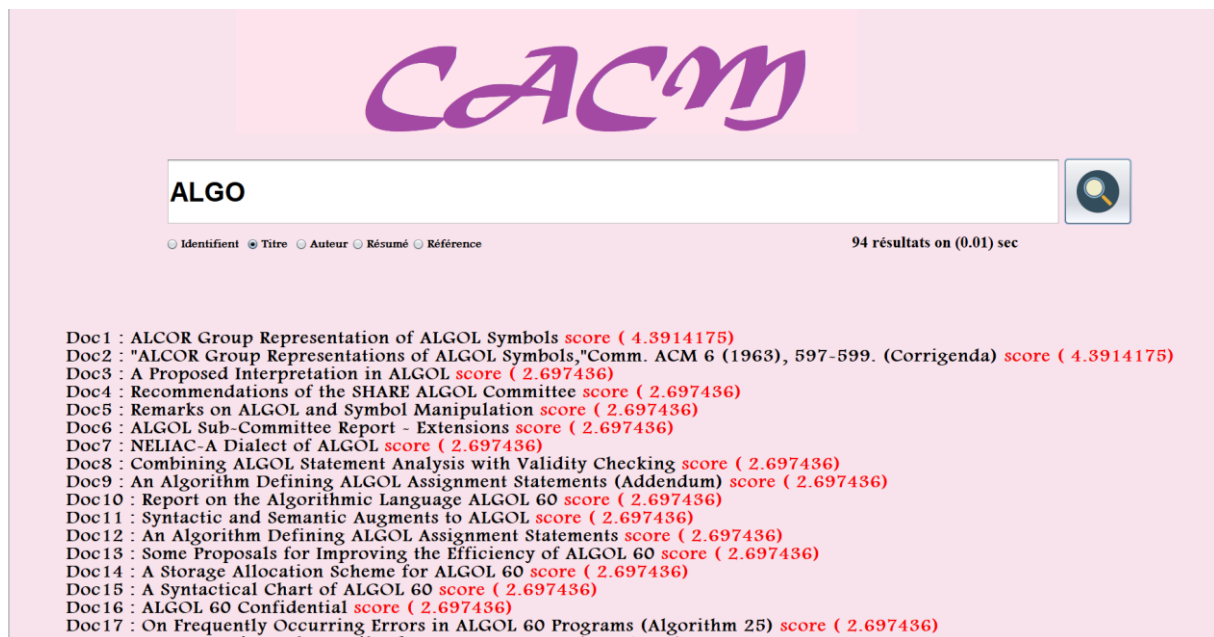
✓ Dans le cas d'une requête simple :



The screenshot displays the CACM search interface. At the top, the CACM logo is visible. Below it, a search bar contains the query "algo". To the right of the search bar is a magnifying glass icon. Below the search bar, there are radio buttons for "Identifiant", "Titre", "Auteur", "Résumé", and "Référence". To the right of these buttons, it says "94 résultats on (0.11) sec". Below this, a list of 18 documents is shown, each with a document number, title, and score. The scores are in red text.

Doc1 : ALCOR Group Representation of ALGOL Symbols **score (4.3914175)**
Doc2 : "ALCOR Group Representations of ALGOL Symbols,"Comm. ACM 6 (1963), 597-599. (Corrigenda) **score (4.3914175)**
Doc3 : A Proposed Interpretation in ALGOL **score (2.697436)**
Doc4 : Recommendations of the SHARE ALGOL Committee **score (2.697436)**
Doc5 : Remarks on ALGOL and Symbol Manipulation **score (2.697436)**
Doc6 : ALGOL Sub-Committee Report - Extensions **score (2.697436)**
Doc7 : NELIAC-A Dialect of ALGOL **score (2.697436)**
Doc8 : Combining ALGOL Statement Analysis with Validity Checking **score (2.697436)**
Doc9 : An Algorithm Defining ALGOL Assignment Statements (Addendum) **score (2.697436)**
Doc10 : Report on the Algorithmic Language ALGOL 60 **score (2.697436)**
Doc11 : Syntactic and Semantic Augments to ALGOL **score (2.697436)**
Doc12 : An Algorithm Defining ALGOL Assignment Statements **score (2.697436)**
Doc13 : Some Proposals for Improving the Efficiency of ALGOL 60 **score (2.697436)**
Doc14 : A Storage Allocation Scheme for ALGOL 60 **score (2.697436)**
Doc15 : A Syntactical Chart of ALGOL 60 **score (2.697436)**
Doc16 : ALGOL 60 Confidential **score (2.697436)**
Doc17 : On Frequently Occurring Errors in ALGOL 60 Programs (Algorithm 25) **score (2.697436)**
Doc18 : A Syntax Directed Compiler for ALGOL 60 **score (2.697436)**

- ✓ Et on voit que c'est le même cas si le contenu est écrit en majuscule



The screenshot shows the CACM search interface. At the top is the CACM logo. Below it is a search bar containing the text 'ALGO' and a magnifying glass icon. Under the search bar, there are radio buttons for 'Identifiant', 'Titre', 'Auteur', 'Résumé', and 'Référence'. To the right of these buttons, it says '94 résultats on (0.01) sec'. Below this, a list of 18 document entries is displayed, each with a document number, title, and a score in parentheses. The scores are mostly 2.697436, with Doc1 and Doc2 having higher scores of 4.3914175.

Doc1 : ALCOR Group Representation of ALGOL Symbols score (4.3914175)
Doc2 : "ALCOR Group Representations of ALGOL Symbols,"Comm. ACM 6 (1963), 597-599. (Corrigenda) score (4.3914175)
Doc3 : A Proposed Interpretation in ALGOL score (2.697436)
Doc4 : Recommendations of the SHARE ALGOL Committee score (2.697436)
Doc5 : Remarks on ALGOL and Symbol Manipulation score (2.697436)
Doc6 : ALGOL Sub-Committee Report - Extensions score (2.697436)
Doc7 : NELIAC-A Dialect of ALGOL score (2.697436)
Doc8 : Combining ALGOL Statement Analysis with Validity Checking score (2.697436)
Doc9 : An Algorithm Defining ALGOL Assignment Statements (Addendum) score (2.697436)
Doc10 : Report on the Algorithmic Language ALGOL 60 score (2.697436)
Doc11 : Syntactic and Semantic Augments to ALGOL score (2.697436)
Doc12 : An Algorithm Defining ALGOL Assignment Statements score (2.697436)
Doc13 : Some Proposals for Improving the Efficiency of ALGOL 60 score (2.697436)
Doc14 : A Storage Allocation Scheme for ALGOL 60 score (2.697436)
Doc15 : A Syntactical Chart of ALGOL 60 score (2.697436)
Doc16 : ALGOL 60 Confidential score (2.697436)
Doc17 : On Frequently Occurring Errors in ALGOL 60 Programs (Algorithm 25) score (2.697436)
Doc18 : A Syntax Directed Compiler for ALGOL 60 score (2.697436)

- ✓ Si aucun champ n'est sélectionné le moteur affiche ce message :



The screenshot shows the CACM search interface. The search bar is empty. Below the search bar, the radio buttons for 'Identifiant', 'Titre', 'Auteur', 'Résumé', and 'Référence' are all unselected. At the bottom of the interface, a message in purple text says 'veuillez sélectionner au moins un champ s'il vous plaît'.

- ✓ Si le champ de recherche est vide le moteur affiche ce message :



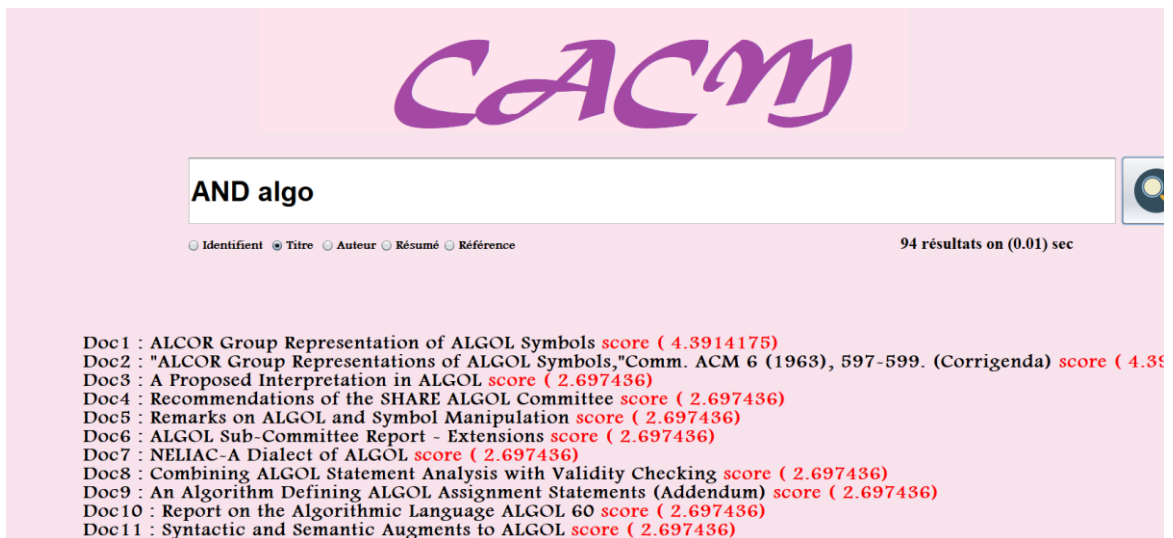
The screenshot shows the CACM search interface. The search bar is empty. Below the search bar, the radio buttons for 'Identifiant', 'Titre', 'Auteur', 'Résumé', and 'Référence' are all unselected. At the bottom of the interface, a message in red text says '😊 veuillez introduire votre recherche s'il vous plaît'.

✓ Si le texte recherché n'existe pas le moteur affiche ce message :

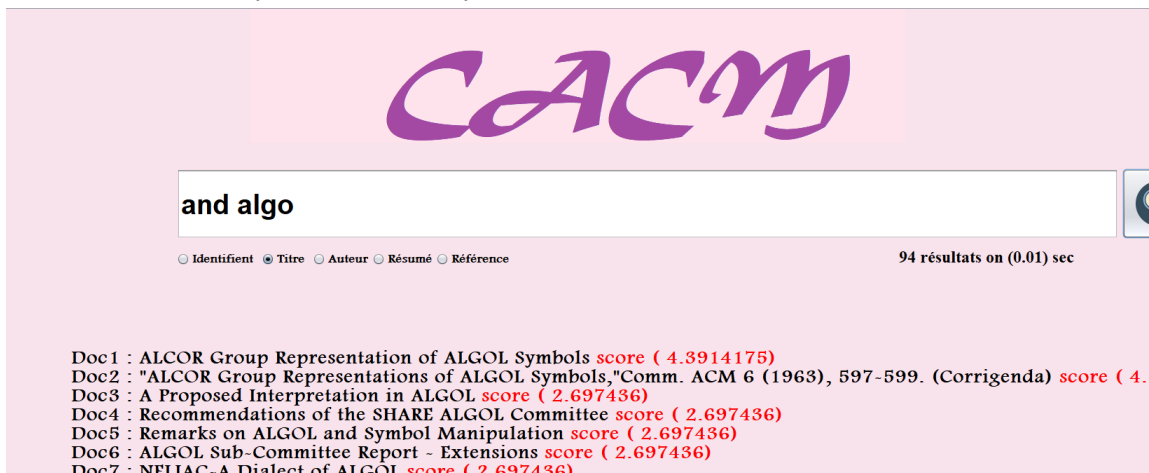


❖ Et pour les opérateurs booléens :

Nous avons traité le « AND » :



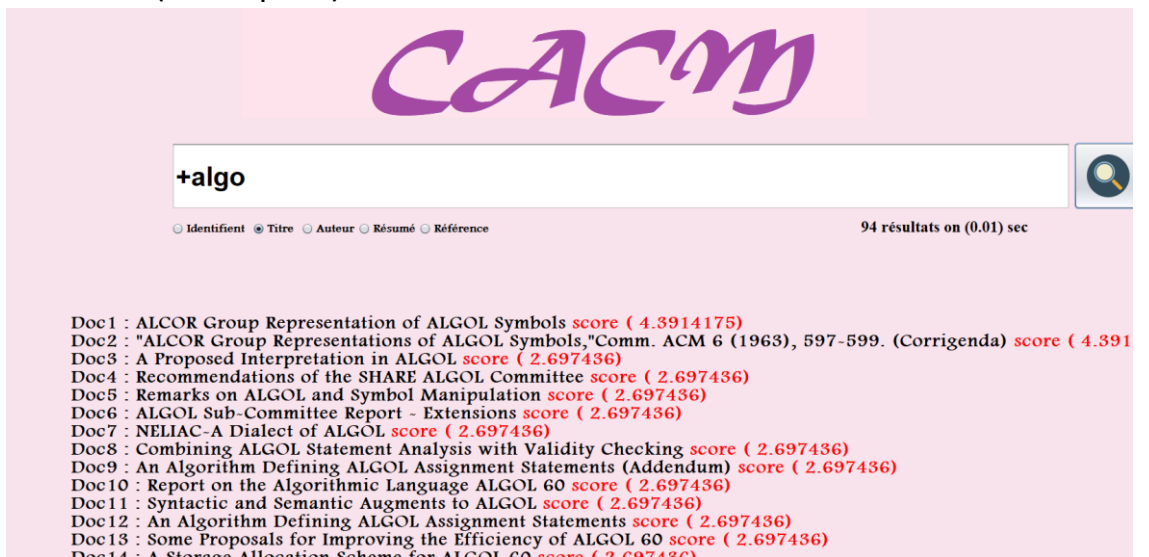
Et aussi « and » (en minuscule) :



The screenshot shows the CACM search interface. At the top is the CACM logo. Below it is a search bar containing the text "and algo". To the right of the search bar is a magnifying glass icon. Below the search bar are radio buttons for "Identifiant", "Titre", "Auteur", "Résumé", and "Référence", with "Titre" selected. To the right of these buttons is the text "94 résultats on (0.01) sec". Below this is a list of search results:

- Doc1 : ALCOR Group Representation of ALGOL Symbols **score (4.3914175)**
- Doc2 : "ALCOR Group Representations of ALGOL Symbols,"Comm. ACM 6 (1963), 597-599. (Corrigenda) **score (4.3914175)**
- Doc3 : A Proposed Interpretation in ALGOL **score (2.697436)**
- Doc4 : Recommendations of the SHARE ALGOL Committee **score (2.697436)**
- Doc5 : Remarks on ALGOL and Symbol Manipulation **score (2.697436)**
- Doc6 : ALGOL Sub-Committee Report - Extensions **score (2.697436)**
- Doc7 : NELIAC-A Dialect of ALGOL **score (2.697436)**

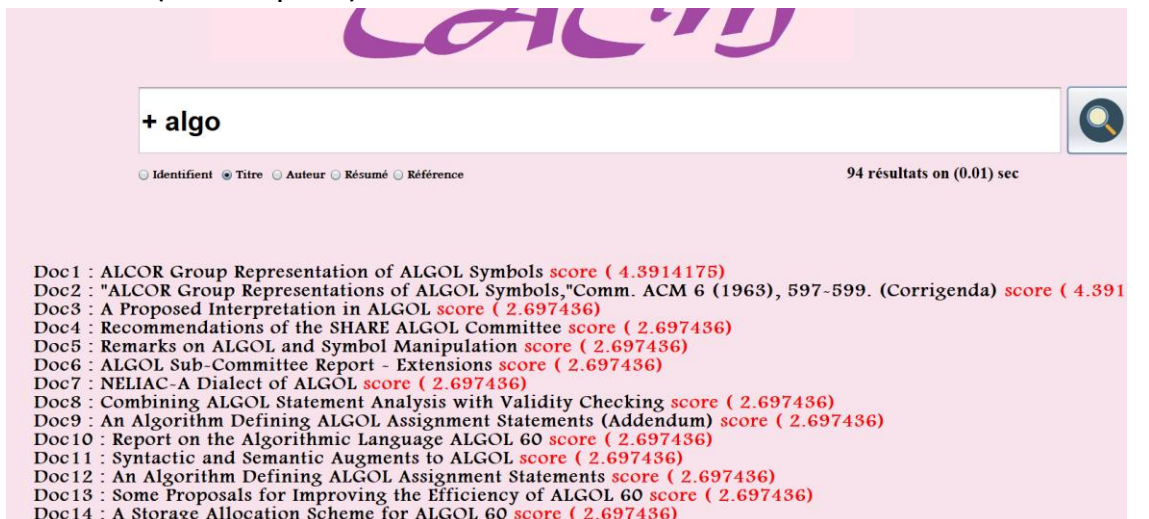
+ « and » (ans espace)



The screenshot shows the CACM search interface. At the top is the CACM logo. Below it is a search bar containing the text "+algo". To the right of the search bar is a magnifying glass icon. Below the search bar are radio buttons for "Identifiant", "Titre", "Auteur", "Résumé", and "Référence", with "Titre" selected. To the right of these buttons is the text "94 résultats on (0.01) sec". Below this is a list of search results:

- Doc1 : ALCOR Group Representation of ALGOL Symbols **score (4.3914175)**
- Doc2 : "ALCOR Group Representations of ALGOL Symbols,"Comm. ACM 6 (1963), 597-599. (Corrigenda) **score (4.3914175)**
- Doc3 : A Proposed Interpretation in ALGOL **score (2.697436)**
- Doc4 : Recommendations of the SHARE ALGOL Committee **score (2.697436)**
- Doc5 : Remarks on ALGOL and Symbol Manipulation **score (2.697436)**
- Doc6 : ALGOL Sub-Committee Report - Extensions **score (2.697436)**
- Doc7 : NELIAC-A Dialect of ALGOL **score (2.697436)**
- Doc8 : Combining ALGOL Statement Analysis with Validity Checking **score (2.697436)**
- Doc9 : An Algorithm Defining ALGOL Assignment Statements (Addendum) **score (2.697436)**
- Doc10 : Report on the Algorithmic Language ALGOL 60 **score (2.697436)**
- Doc11 : Syntactic and Semantic Augments to ALGOL **score (2.697436)**
- Doc12 : An Algorithm Defining ALGOL Assignment Statements **score (2.697436)**
- Doc13 : Some Proposals for Improving the Efficiency of ALGOL 60 **score (2.697436)**
- Doc14 : A Storage Allocation Scheme for ALGOL 60 **score (2.697436)**

+ « and » (avec espace)



The screenshot shows the CACM search interface. At the top is the CACM logo. Below it is a search bar containing the text "+ algo". To the right of the search bar is a magnifying glass icon. Below the search bar are radio buttons for "Identifiant", "Titre", "Auteur", "Résumé", and "Référence", with "Titre" selected. To the right of these buttons is the text "94 résultats on (0.01) sec". Below this is a list of search results:

- Doc1 : ALCOR Group Representation of ALGOL Symbols **score (4.3914175)**
- Doc2 : "ALCOR Group Representations of ALGOL Symbols,"Comm. ACM 6 (1963), 597-599. (Corrigenda) **score (4.3914175)**
- Doc3 : A Proposed Interpretation in ALGOL **score (2.697436)**
- Doc4 : Recommendations of the SHARE ALGOL Committee **score (2.697436)**
- Doc5 : Remarks on ALGOL and Symbol Manipulation **score (2.697436)**
- Doc6 : ALGOL Sub-Committee Report - Extensions **score (2.697436)**
- Doc7 : NELIAC-A Dialect of ALGOL **score (2.697436)**
- Doc8 : Combining ALGOL Statement Analysis with Validity Checking **score (2.697436)**
- Doc9 : An Algorithm Defining ALGOL Assignment Statements (Addendum) **score (2.697436)**
- Doc10 : Report on the Algorithmic Language ALGOL 60 **score (2.697436)**
- Doc11 : Syntactic and Semantic Augments to ALGOL **score (2.697436)**
- Doc12 : An Algorithm Defining ALGOL Assignment Statements **score (2.697436)**
- Doc13 : Some Proposals for Improving the Efficiency of ALGOL 60 **score (2.697436)**
- Doc14 : A Storage Allocation Scheme for ALGOL 60 **score (2.697436)**

Nous avons aussi traité le « et »

« Et » (en minuscule)

The screenshot shows the CAC'97 search interface. The search bar contains the text 'et algo'. Below the search bar, there are radio buttons for 'Identifiant', 'Titre', 'Auteur', 'Résumé', and 'Référence', with 'Titre' selected. To the right of the search bar, it says '94 résultats on (0.0) sec'. Below this, a list of 13 documents is displayed, each with a document number, title, and a score in red parentheses. The documents are related to the ALGOL language and its various dialects and committees.

Doc1 : ALCOR Group Representation of ALGOL Symbols score (4.3914175)
Doc2 : "ALCOR Group Representations of ALGOL Symbols,"Comm. ACM 6 (1963), 597-599. (Corrigenda) score (4.3914175)
Doc3 : A Proposed Interpretation in ALGOL score (2.697436)
Doc4 : Recommendations of the SHARE ALGOL Committee score (2.697436)
Doc5 : Remarks on ALGOL and Symbol Manipulation score (2.697436)
Doc6 : ALGOL Sub-Committee Report - Extensions score (2.697436)
Doc7 : NELIAC-A Dialect of ALGOL score (2.697436)
Doc8 : Combining ALGOL Statement Analysis with Validity Checking score (2.697436)
Doc9 : An Algorithm Defining ALGOL Assignment Statements (Addendum) score (2.697436)
Doc10 : Report on the Algorithmic Language ALGOL 60 score (2.697436)
Doc11 : Syntactic and Semantic Augments to ALGOL score (2.697436)
Doc12 : An Algorithm Defining ALGOL Assignment Statements score (2.697436)
Doc13 : Some Proposals for Improving the Efficiency of ALGOL 60 score (2.697436)

« Et » (en majuscule)

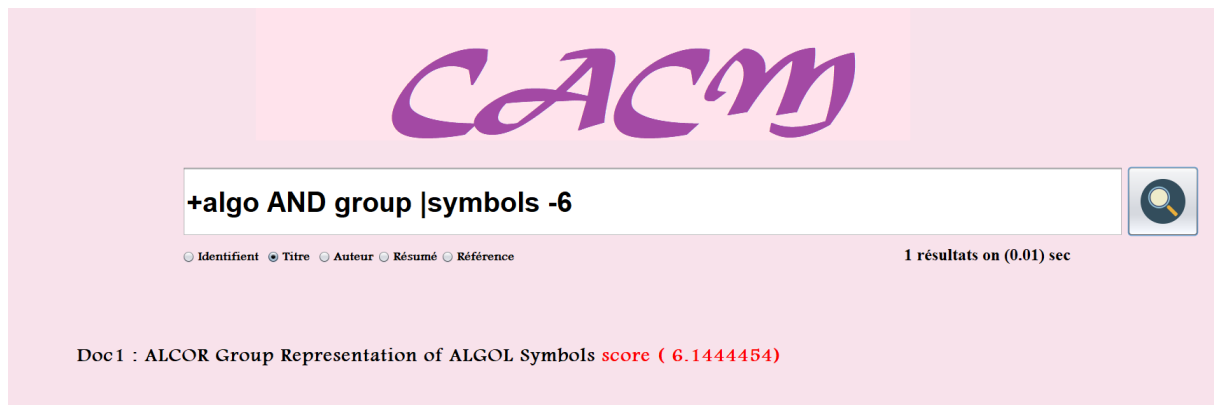
The screenshot shows the CAC'97 search interface. The search bar contains the text 'ET algo'. Below the search bar, there are radio buttons for 'Identifiant', 'Titre', 'Auteur', 'Résumé', and 'Référence', with 'Titre' selected. To the right of the search bar, it says '94 résultats on (0.0) sec'. Below this, a list of 13 documents is displayed, each with a document number, title, and a score in red parentheses. The documents are related to the ALGOL language and its various dialects and committees.

Doc1 : ALCOR Group Representation of ALGOL Symbols score (4.3914175)
Doc2 : "ALCOR Group Representations of ALGOL Symbols,"Comm. ACM 6 (1963), 597-599. (Corrigenda) score (4.3914175)
Doc3 : A Proposed Interpretation in ALGOL score (2.697436)
Doc4 : Recommendations of the SHARE ALGOL Committee score (2.697436)
Doc5 : Remarks on ALGOL and Symbol Manipulation score (2.697436)
Doc6 : ALGOL Sub-Committee Report - Extensions score (2.697436)
Doc7 : NELIAC-A Dialect of ALGOL score (2.697436)
Doc8 : Combining ALGOL Statement Analysis with Validity Checking score (2.697436)
Doc9 : An Algorithm Defining ALGOL Assignment Statements (Addendum) score (2.697436)
Doc10 : Report on the Algorithmic Language ALGOL 60 score (2.697436)
Doc11 : Syntactic and Semantic Augments to ALGOL score (2.697436)
Doc12 : An Algorithm Defining ALGOL Assignment Statements score (2.697436)
Doc13 : Some Proposals for Improving the Efficiency of ALGOL 60 score (2.697436)

Et c'est le même cas pour :

- ✓ not, NOT, non, NOT, (– sans espace) et (- avec espace)
- ✓ OR, or, OU, ou, le signe | (avec et sans espace)

Et voilà une requête avec plusieurs cas :



VII. Conclusion :

Ce projet nous a été très bénéfique car il nous a permis de bien nous familiariser à la programmation en « java » et « lucene », comme il nous a également permis de faire une grande expérience qui a amélioré nos connaissances et nos compétences dans le domaine de recherche d'information .

Nous estimons avoir atteint la plus part des objectifs tracés au départ de notre projet.

Bibliographie

[REF1] : Dr. Yassine Drias. Cour recherche information. 25Mai 2018.

Webographie

[WEB 1]: <https://fr.wikipedia.org/wiki/Lucene>

Date de consultation : 25Mai 2018.

[WEB 2]: [https://fr.wikipedia.org/wiki/Java_\(langage\)](https://fr.wikipedia.org/wiki/Java_(langage))

Date de consultation : 25Mai 2018.