Optimisation. Exercice d'application du cours.

Soit le schéma : R(A,B), S(B,C). On désire évaluer la requête : $\sigma_{A='\text{toto}'}(R \bowtie S)$.

On s'intéresse à comparer l'efficacité de différents algorithmes de jointure. On dispose des fonctions suivantes. Le parcours séquentiel d'une table se fait à l'aide d'une fonction Scan(.) qui prend en entrée une table et renvoie le tuple suivant de cette table. La première fois que Scan(R) est appelée elle renvoie le premier tuple de R, la seconde fois le second tuple de R, etc. Lorsque tous les tuples de R ont été renvoyés, Scan(R) renvoie la valeur *null*.

La fonction Ajout-Result(.) rajoute son argument au résultat de l'algorithme dans lequel elle est appelée.

Enfin, si p est un tuple, $p.A$ est son champ A.

1) On considère l'algorithme A_0 suivant (avec une syntaxe à la Java). Quelle est sa complexité en fonction du nombre de tuples dans R et dans S dans le cas le pire?

```
do {
     p = Scan(R);
     if (p.A == 'toto')
         do {
             q = Scan(S);
             if (q.B == p.B)
                 Ajout-Result(p.A, p.B, q.C);
         }
         while (q != null);
}
while (p != null);
```

2) On introduit maintenant un index sur l'attribut A de R, un autre sur l'attibut B de R et un dernier sur l'attribut B de S. Chacun de ces index est un arbre B+.

On a maintenant une fonction Scan-Index() qui prend en entrée une table et une valeur pour un attribut de cette table et retourne, en utilisant l'index approprié, successivement les tuples de la table ayant la bonne valeur sur l'attribut. Ainsi le premier appel à Scan-Index(R,A=x) retourne le premier tuple de R ayant x pour attribut A, le second appel à Scan-Index(R,A=x) retourne le second tuple de R ayant x pour attribut A, etc. Lorsque tous les tuples ont été renvoyés, Scan-Index(R,A=x) retourne la valeur *null*.

On considère l'algorithme A_1 suivant.

```
do {
    p = Scan(R);
    if (p.A == 'toto')
        do {
            q = Scan-Index(S, B=p.B);
            Ajout-Result(p.A, p.B, q.C);
        }
        while (q != null);
}
while (p != null);
```

On considère les situations suivantes (et pour chacune il n'y a aucune autre contrainte).
Pour chacune, donnez la complexité de A_1.

a) A='toto' est un critère très sélectif (c'est-à-dire satisfait par très peu de lignes de R).

b) Dans R il ne peut y avoir deux lignes avec la même valeur de B, et on ne sait rien sur la sélectivité de A='toto'.

c) Dans S il ne peut y avoir deux lignes avec la même valeur de B, et on ne sait rien sur la sélectivité de A='toto'.

d) Dans S il ne peut y avoir deux lignes avec la même valeur de B, et A='toto' n'est pas sélectif (satisfait par 50% des lignes de R).