信息检索》第二次作业

在认真阅读学习《An Introduction to Information Retrieval》Chapter 1 “Boolean Retrieval” 的基础上，完成如下作业：

题目1：Write out a postings merge algorithm,
in the style of Figure 1.6 (page 11) for handling the Boolean query

x AND NOT y

```
# algorithm for AND
INTERSECT(p1, p2)
    answer <- ()
    while p1 != NIL and p2 != NIL
    do if docID(p1) == docID(p2)
        then ADD(answer, docID(p1))
            p1 <- next(p1)
            p2 <- next(p2)
        else if docID(p1) < docID(p2)
            then p1 <- next(p1)
            else p2 <- next(p2)
    return answer
```

```
# algorithm for AND NOT
AND_NOT(p1, p2)
    answer <- ()
    while p1 != NIL
    do if p2 == NIL
            ADD(answer, docID(p1))
            p1 <- next(p1)
        else if docID(p1) < docID(p2)
            then ADD(answer, docID(p1))
                p1 <- next(p1)
        else if docID(p1) == docID(p2)
            p1 <- next(p1)
            p2 <- next(p2)
            else p2 <- next(p2)
```

题目2：For the queries below, can we still run through the intersection in time $O(x+y)$, where x and y are the lengths of the postings lists for `Brutus` and `Ceaser`? If not, what can we achieve

- Brutus AND NOT Caesar
- Brutus OR NOT Caesar

Looking at the algorithm above, we can see that `Brutus AND NOT Caesar` can be done in $O(x+y)$. Instead of collecting documents that occur in both postings list like the `AND` algorithm, collect those that occur in the first one and not the second list. For the second one, the time complexity is $O(n)$, where n is the total number of documents in the collection. This is because the length of the results is only bounded by N, not by the length of the postings list.