

컴퓨터알고리즘 11주차

2016112158 김희수



[날짜]

[회사 이름]

[회사 주소]

11주차 과제는 Burrow-Wheeler Transform을 구현하는 과제다. 구현테스트를 위해 original String을 입력받고 oringal을 BWT를 통해 BWTString을 만든다. 그 후, BWTString만 가지고 original String을 복원하면 끝이다.

Original String을 BWTString으로 변환하는 과정에는 table을 생성하고, 생성된 table을 정렬하는 과정이 포함된다. table은 int, string의 튜플을 원소로 가지는 벡터이다. 여기서 int는 componentIds라 하여 각 원소를 나타내는 id라 할 수 있고, string은 그 원소의 내용(스트링)이다. fill\_table함수는 original string을 시작으로 문자하나씩 밀면서 table을 채워나간다.

SortTable은 table안의 스트링들을 비교해서 정렬한다. 비교는 스트링을 기준으로 사전식 비교를 하며 스트링뿐만 아니라 componentId도 스왑되어야한다. 그렇게 정렬되었으면 테이블의 마지막 column이 BWTString이 된다.

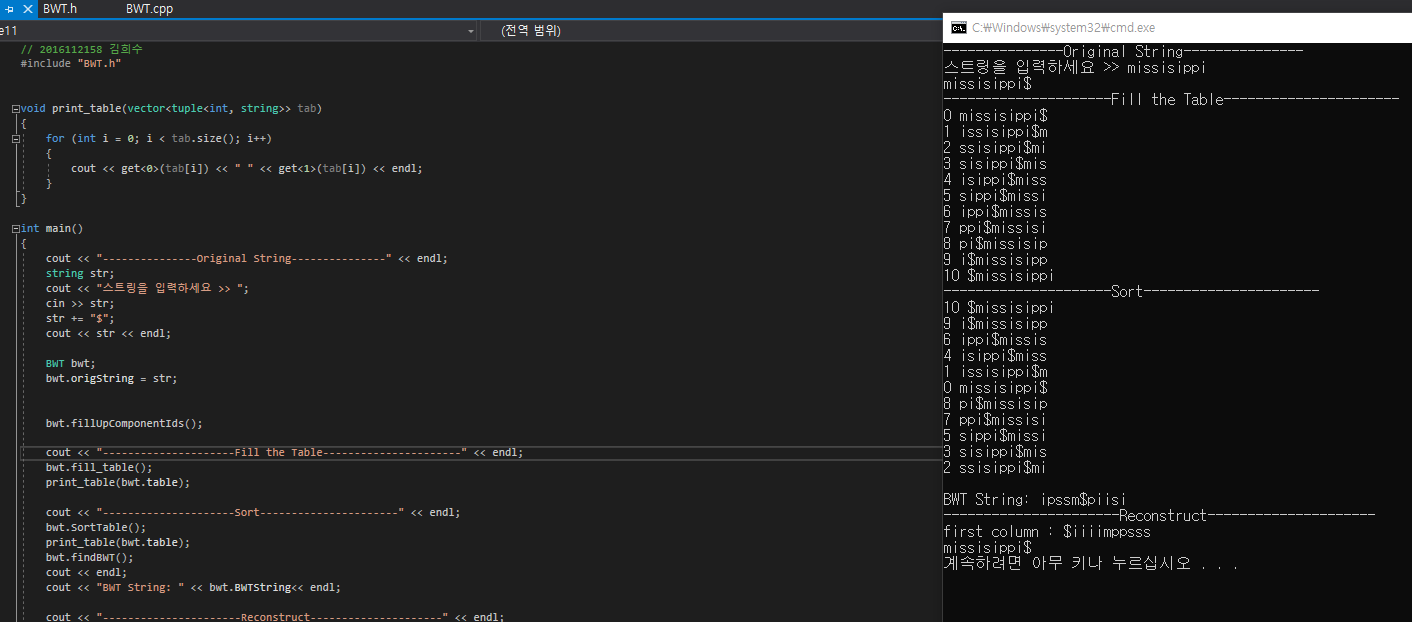
BWTString이 완성되었으면 이것을 가지고 본래의 스트링을 복원해보자. 복원은 연결리스트를 이용하였다. first column의 문자가 가리키는 BWTString 문자의 인덱스를 저장하기 위해 l\_shift배열을 선언하였다. acaacg$를 예로 들면 first column은 $aaaccg가 된다. l\_shift의 인덱스는 first column의 원소를 가리킨다. l\_shift[0]에는 first column의 0번째 문자에 해당하는 $를 BWTString에서 찾아서 그 인덱스를 저장한다. 따라서 l\_shift[0] = 2. l\_shift[1]에는 first column의 1번째 문자에 해당하는 a를 BWTString에서 찾아서 그 인덱스를 저장한다. 따라서 l\_shift[1] = 3

연결리스트 list는 크기를 128로 주어서 문자들의 아스키 코드를 연결리스트의 인덱스로 할 수 있게 하였다. list에는 BWTString의 문자를 인덱스로 하여 각 문자의 스트링 내에서의 상대적위치를 data로 하여 저장한다. 그 후 l\_shift 초기화를 위해 computeLShift함수를 호출한다.

computeLShift함수는 연결리스트에서 BWTString의 문자의 상대적 위치를 l\_shift에 저장한다.

이제 BWTString의 길이만큼 l\_shift의 원소에 해당하는 BWTString을 출력해주면 original String이 복원된다. 단, 이때 시작은 BWTString에서 $의 위치의 l\_shift여야 한다.

(missisippi 실행화면)



( 영문이름 kimheesu 실행화면)

