Happy house 알고리즘 적용 기획서

[서울 10반 경혜안 / 권도혁]

1. 검색 기능 (KMP 알고리즘)
   1. 배경 : 동 / 아파트 이름 검색 시 효율성을 높이기 위해 KMP 알고리즘을 적용한다.
   2. 기획 방향 : KMP 알고리즘과 기존의 Brute – Force 알고리즘을 비교하여 문제 사항을 분석한다.
   3. 기획 내용

|  |  |  |
| --- | --- | --- |
| 알고리즘 | 특징 | 시간 복잡도 |
| **Brute-Force** | 본문이 되는 문자열의 맨 앞에서부터 끝까지, 찾고자 하는 문자열과 하나하나 비교하는 것 | O(MN) (M = 본문 문자열의 길이, N = 패턴의 길이 를 뜻함) |
| **KMP** | 불일치가 발생하기 직전까지 같았던 부분은 다시 비교하지 않고 패턴 매칭을 진행하는 것 | O(N+M) (M = 본문 문자열의 길이, N = 패턴의 길이 를 뜻함) |

* 1. 성능 비교
     1. KMP의 경우 반복적인 패턴을 추출하여 비교를 진행하기에 불필요한 비교 횟수가 적음
     2. Brute-Force의 경우 구현이 쉬움
     3. KMP의 경우 LPS 테이블을 만들어야 함
  2. 결론 : KMP알고리즘을 채택하여 검색 기능의 효율성을 증가시킨다.

1. 정렬 기능
   1. 배경 : 사용자가 검색한 내용을 효과적으로 전달하기 위해 정렬 알고리즘을 사용하여 나타낸다.
   2. 기획 방향 : Java 라이브러리 제공 함수 중 sort와 관련된 함수들의 장단점을 분석한다.
   3. 기획 내용

|  |  |  |
| --- | --- | --- |
| 알고리즘 | 특징 | 시간 복잡도 |
| **Arrays.sort()** | * Primitive 타입배열에서 정렬을 지원한다. * Dual pivot quick sort * 변수를 새로 선언하는 것이 아니라 저장되어있는 변수를 그대로 사용하고 내부적으로 정렬을 수행한다. | O(N^2) |
| **Collections.sort()** | * Wapper 클래스로 선언해야 한다. * Merge sort | O(N\*logN) |

* 1. 성능 비교
     1. Array sort는 타입의 제한이 있다.
     2. Array sort의 경우 내부적으로 정렬을 수행하기에 메모리가 적게 사용된다.
     3. Collection sort는 다양한 기준으로 정렬이 가능하다.
  2. 결론 : 추후 다양한 기준의 정렬을 위해 Collections.sort()을 사용한다.