Binary Search

Date	
작성일시	@2022년 8월 3일 오전 10:29
강의 번호	
유형	
강사명	
강의자료	☐
노션 복습	☐
코딩 복습	☐
주말숙제(교제)	☐
정리	☐

이진 탐색법_Binary Search

- 원하는 데이터를 찾는 알고리즘
- (전제조건) 반드시 찾는 데이터 전체가 정렬되어야만 사용할 수 있다.
- 절반씩 대상 데이터를 줄여가면서 탐색한다.

▼ 이진 탐색법 알고리즘

0. 데이터 정렬 후(전제조건)

1. 가운데 요소를 찾는 처리 - 두 숫자의 가운데는 평균으로 구할 수 있음

> ex) 0 + 6 = 6 / 2 = 3
> 0 + 2 = 2 / 2 = 1

2. 가운데 요소와 원하는 데이처를 비교하는 처리 - head 와 tail을 지정

> center 중간값
> head 가장 앞에 값

> tail 제일 뒤의 값
> (head + tail) / 2 = center
>
> > ****요소들의 개수가 짝수 일때, 예를 들면 요소가 6라고 생각해보면 center 후보가 2개가 된다. 하지만 2.5라는 첨자는 있을수 없다. 이럴 경우는 소수점 이하 부분을 제거한 정수 부분을 취하여 인덱스를 사용하면 전혀 문제가 없다.

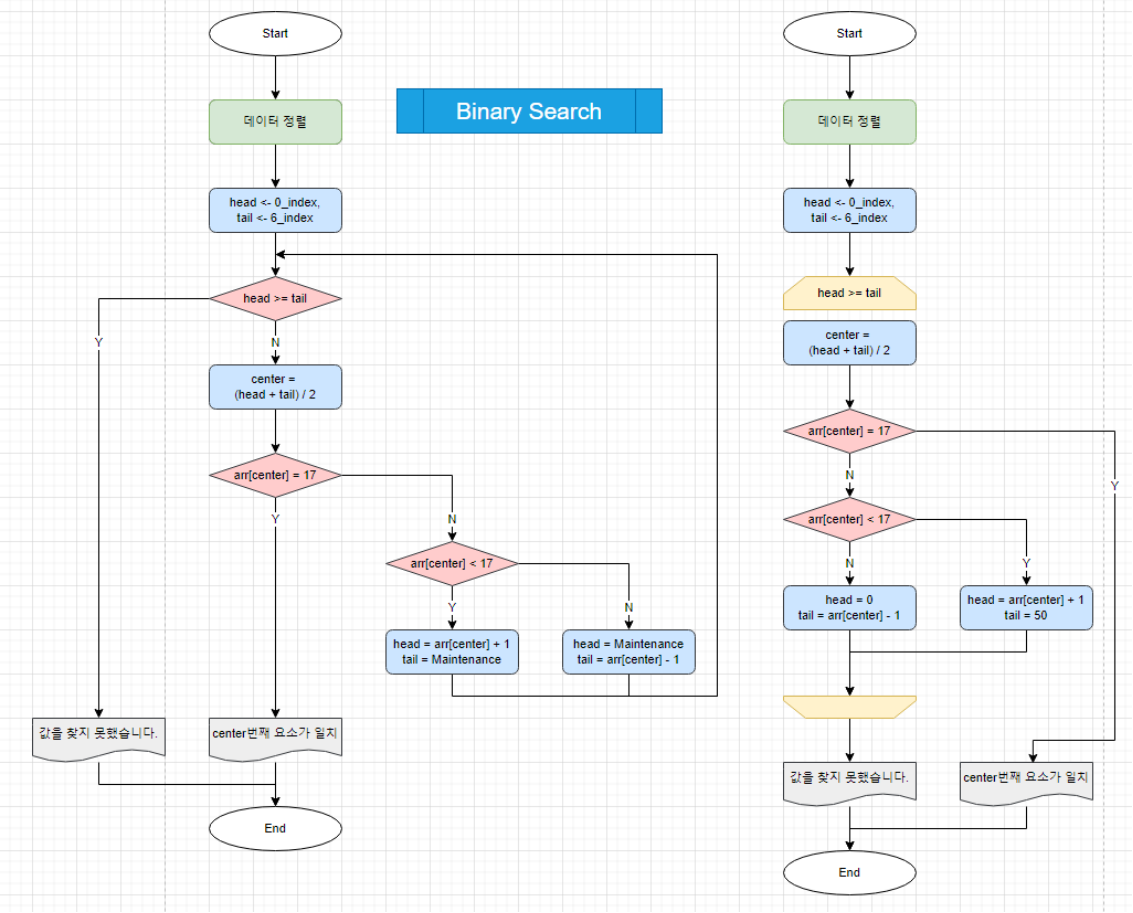
a. 평균 계산을 통한 가운데 요소의 값과 찾는 값을 비교하여 만약 첫 번째에 일치한다면 프로그램을 종료한다.
b. 하지만 No일 경우 즉, 원하는 데이터가 아닐 경우 두 가지의 경우의 수가 나타난다. (찾는 값보다 작은 경우와 큰 경우)
c. 이 두가지의 경우 모두 탐색 범위를 반으로 줄이는 처리로 넘어간다.

3. 탐색 범위 를 절반으로 줄이는 처리
   a. 원하는 데이터(17)가 가운데 데이터(9)보다 큰 경우 * arr[center] <17

      이 경우 전체 검색 범위의 뒷부분으로 대상을 절반으로 좁힌다. 따라서 탐색 범위으 맨 앞 요소는 arr[center] 보다 하나 큰 첨자를 갖는 요소가 된다.
      head = center +1, tail은 그대로 사용
   b. 원하는 데이터(17)가 가운데 데이터(9)보다 작은 경우 * arr[center] > 17

      이 경우 전체 검색 범위의 앞으로 대상을 절반으로 좁힌다. 따라서 탐색 범위의 맨 뒷 요소는 arr[center] 보다 하나 작은 첨자를 갖는 요소가 된다.
      head는 그대로 사용, tail = center -1

▼ Flow chart



▼ Java

```
*방법 1
public class Binary {
  public static void main(String[] args) {
    int[] arr = {11,13,17,18,23,29,31};
    int head = 0;
    int tail = 6;
    int ans = 17;

    while(head <= tail) {
      int center = (head + tail) /2 ;
      if(arr[center] == ans) {
        System.out.println( center+1 +"번째 요소 일치");
        break;
      } else if (arr[center] < ans) {
        head = center+1;
      } else {
        tail = center-1;
      }
    }
    if(head < tail) { //반복을 방지
```

```
    System.out.println("찾는 값이 없습니다.");
    }
  }
} Outputs : 3번째 요소 일치
```

```
*방법 2
public class Binary {
  public static void main(String[] args) {
    int[] arr = {11,13,17,18,23,29,31};
    int ans = 17;
    int result =bsearch(arr, ans);
    if(result != -1) {
    System.out.println( result+1 +"번째 요소 일치"); //index는 0부터 시작
    }else {
      System.out.println("찾는 값이 없습니다.");
    }
  }

  public static int bsearch(int[] arr, int ans) {
    int head = 0;
    int tail = 6;
    while(head <= tail) {
      int center = (head + tail) /2 ;
      if(arr[center] == ans) {
        return center;
      } else if (arr[center] < ans) {
        head = center+1;
      } else {
        tail = center-1;
      }
    }
    return -1;
    //index에는 -1이 없다. 따라서 index없는 값은 던져서 없는 경우를 표시
  }
} Outputs : 3번째 요소 일치
```