

[백준]2467_용액

▼ 상태	풀었지만 깃허브에 안올린 것
📅 날짜	@2022년 4월 21일
≡ 공부유형	스터디
≡ 알고리즘	이진탐색
▼ 사이트	백준
☑ 깃허브	<input type="checkbox"/>

2467번: 용액

KOI 부설 과학연구소에서는 많은 종류의 산성 용액과 알칼리성 용액을 보유하고 있다. 각 용액에는 그 용액의 특성을 나타내는 하나의 정수가 주어져있다. 산성 용액의 특성값은 1부터 1,000,000,000까지의

<https://www.acmicpc.net/problem/2467>

BAE/KJOON
ONLINE JUDGE

개념

투포인터

풀이

-99 -2 -1 4 98

start = 인덱스 0

end = 인덱스 size -1 (맨 끝)

// -10억~10억 사이의 정수가 (2값이 최대로 크게 더해도 20억이라 int 안에서 해결 가능)

// 2~10만개 사이의 개수만큼 : 리스트 길이 = 이만큼 이진 탐색 진행

// 투 포인터?

코드

```
package BinarySearch;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.Arrays;
import java.util.StringTokenizer;

public class BinarySearch_BOJ_2467_LJE { // 용액

    static void BinarySearch() {
        int start = 0; // 시작
        int end = N-1; // 끝
        int min = Integer.MAX_VALUE;

        while(start<end) {
            int sum = list[start] + list[end];

            if(sum==0) { // 0이면 바로 끝내주기
                liquid[0]=list[start];
                liquid[1]=list[end];
                break;
            }
            if(Math.abs(sum)<min) { //새로운 합이 더 작다면 값 저장
                liquid[0]=list[start];
                liquid[1]=list[end];
                min = Math.abs(sum);
            }

            if(sum>=0){ // 합이 0보다 크다면 0기준 밸런스가 오른쪽에 치우친거니까
                end -= 1; //오른쪽을 줄여주기
            }else { // 0보다 작으면 왼쪽에 치우친거니까
                start += 1; // 왼쪽을 줄여주기
            }
        }
    }

    static int N,list[],liquid[];
    public static void main(String[] args) throws NumberFormatException, IOException {
        /* 입력 */
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        N = Integer.parseInt(br.readLine());

        list = new int[N]; // list : 용액들 저장
        liquid = new int[2]; // liquid : 출력할 용액

        StringTokenizer st = new StringTokenizer(br.readLine());
        for (int i = 0; i < N; i++) {
            list[i] = Integer.parseInt(st.nextToken());
        }
    }
}
```

```

/* 탐색 */
Arrays.sort(list); // 이진 탐색은 항상 정렬 후
//   for (int i = 0; i < N; i++) {
//       System.out.println(list[i]+" ");
//   }

BinarySearch(); // 탐색

Arrays.sort(liquid); // 오름차순으로 출력 위해 정렬
System.out.println(liquid[0] +" "+liquid[1]);
}
}

```

참고

▼ 참고 코드

투 포인터

입력 데이터는 오름차순으로 주어진다고 했으므로 따로 정렬을 해주지 않아도 된다. 첫 원소는 가장 작은 값이고, 끝 원소는 가장 큰 값이므로 투 포인터를 사용하여 서로 가운데로 향하여 움직이게 한 다음 가장 0에 가까운 수를 구해주면 된다.

1. 0과 n-1을 각 투포인터로 지정한다.
2. 투포인터를 사용하여 합이 0에 가까운 구간을 탐색한다.
 - a. $\text{min} > \text{Math.abs}(\text{arr}[\text{left}] + \text{arr}[\text{right}])$ 합이 0에 가장 가까운 값이 갱신되면 저장해준다.
 - b. $\text{sum} \geq 0$ 합이 0보다 크거나 같다면 더 작아져야 하므로 오른쪽 포인터를 이동한다. $\text{right}--$;
 - c. $\text{sum} < 0$ 합이 0보다 작다면 더 커져야 하므로 왼쪽 포인터를 이동한다. $\text{left}--$;

이진탐색

투 포인터와 아이디어는 같지만 탐색방식이 다르다. 하나의 원소($\text{arr}[i]$)를 찍한 다음에 나머지 원소 중 현재 원소*-1와 가장 가까운 원소를 이진탐색을 통해 찾아준다. 그래야 0에 가장 가까운 값을 찾을 수 있기 때문이다.

1. $\text{arr}[i]$ 기준점을 뺀다.
2. $[i+1 \sim n-1]$ 구간에 있는 원소 중 $\text{arr}[i]*-1$ 와 가장 가까운 값을 이진탐색을 통해 탐색한다.

- a. $\text{min} > \text{Math.abs}(\text{arr}[i] + \text{arr}[\text{mid}])$ 0에 가장 가까운 값이 갱신되면 저장해준다.
- b. $\text{arr}[\text{mid}] \geq -\text{arr}[i]$ 기준점*-1보다 크다면 위치를 $\text{right} = \text{mid} - 1$;로 옮겨준다.
- c. $\text{arr}[\text{mid}] < -\text{arr}[i]$ 기준점*-1보다 작다면 위치를 $\text{left} = \text{mid} + 1$;로 옮겨준다.

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.StringTokenizer;

public class Main {

    static int n;
    static long[] arr;
    public static void main(String[] args) throws IOException{
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        n = Integer.parseInt(br.readLine());

        arr = new long[n];
        StringTokenizer st = new StringTokenizer(br.readLine());
        for(int i=0; i<n; i++) {
            arr[i] = Long.parseLong(st.nextToken());
        }

        long min = Long.MAX_VALUE;
        int ml = 0, mr = 0;
        for(int i=0; i<n-1; i++) {
            int left = i+1;
            int right = n-1;
            while(left<=right) {
                int mid = (left+right)/2;
                long sum = Math.abs(arr[i]+arr[mid]);

                if(min > sum) {
                    min = sum;
                    ml = i; mr = mid;
                }
                if(arr[mid]>= -arr[i]) {
                    right = mid-1;
                }else{
                    left = mid+1;
                }
            }
        }
        System.out.println(arr[ml]+" "+arr[mr]);
    }
}
```

[BOJ] 백준 2467번 용액 (Java)

KOI 부설 과학연구소에서는 많은 종류의 산성 용액과 알칼리성 용액을 보유하고 있다. 각 용액에는 그 용액의 특성을 나타내는 하나의 정수가 주어져있다. 산성 용액의 특성값은 1부터

 <https://loosie.tistory.com/559>

