2110. 공유기 설치

출처	BoJ
실습/과제/자율	자율
문제유형	이분 탐색
날짜(처음 풀이)	@2024년 9월 25일
풀이 언어	Java

https://www.acmicpc.net/problem/2110

https://www.acmicpc.net/problem/2110

1. 문제 요약

도현이의 집 N개가 수직선 위에 있다. 각각의 집의 좌표는 x1, ..., xN이고, 집 여러개가 같은 좌표를 가지는 일은 없다.

도현이는 언제 어디서나 와이파이를 즐기기 위해서 집에 공유기 C개를 설치하려고 한다. 최대한 많은 곳에서 와이파이를 사용하려고 하기 때문에, 한 집에는 공유기를 하나만 설치할 수 있고, 가장 인접한 두 공유기 사이의 거리를 가능한 크게 하여 설치하려고 한다.

C개의 공유기를 N개의 집에 적당히 설치해서, 가장 인접한 두 공유기 사이의 거리를 최대로 하는 프로그램을 작성하시오.

2. 문제 입출력

입력

- 첫째 줄에 집의 개수 N (2 ≤ N ≤ 200,000)과 공유기의 개수 C (2 ≤ C ≤ N)이 하나 이상의 빈 칸을 사이에 두고 주어진다.
- 둘째 줄부터 N개의 줄에는 집의 좌표를 나타내는 xi (0 ≤ xi ≤ 1,000,000,000)가 한 줄에 하나씩 주어진다.

입력예시

```
5 3
1
2
8
4
9
```

출력

- 첫째 줄에 가장 인접한 두 공유기 사이의 최대 거리를 출력한다.

출력예시

```
3
```

📣 3. 제한사항

- **시간** : 2초
- **메모리** : 128MB

🧐 4. 접근법

▼ 초기 접근법 - 단순 비교 (이분 탐색을 몰랐다고 가정)

- 집 좌표 오름차순으로 정렬
- 각 집과 집 사이의 거리를 배열(arr[N][N])로 만들어 저장한다.

  arr[N][0]은 해당 집의 숫자

- 최대 거리이고 C는 2보다 크거나 같고, N보다 작거나 같으므로,
  - C==2이면 가장 양 끝에 설치하고, 그 거리가 result
  - C==N이면 각 집 사이의 거리 중 최대값이 result
  - else이면
    - 가장 양 끝에 먼저 2개를 설치하고, 사이에 하나씩 설치했을 때를 가정하며 숫자 비교(가장 인접한 두 공유기 사이의 거리가 큰 것이 result)
- result를 출력한다.

💻 5. 코드

▼ 이전 코드 (작성하다가 말았음)

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

public class RouterInstallationWithoutArr {
    public static void main(String[] args) throws Exception {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        StringTokenizer st = new StringTokenizer(br.readLine());

        int N = Integer.parseInt(st.nextToken());
        int C = Integer.parseInt(st.nextToken());

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

```
        // 집 좌표 오름차순 정렬
        Arrays.sort(houses);

        int result = 0;

        if (C == 2) {
            // 가장 양 끝 집에 공유기 설치
            result = houses[N - 1] - houses[0];  // 양 끝 거리
        } else if (C == N) {
            // 모든 집에 공유기 설치, 각 집 사이의 거리 중 최대값 구하기
            int maxDistance = 0;
            for (int i = 0; i < N - 1; i++) {
                maxDistance = Math.max(maxDistance, houses[i + 1] - houses[i]);
            }
            result = maxDistance;
        } else {
```

```
        }

        System.out.println(result);
```

▼ 최종 코드

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

public class Main {

    public static void main(String[] args) throws Exception {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        StringTokenizer st = new StringTokenizer(br.readLine());

        int N = Integer.parseInt(st.nextToken()); // 집의 개수
        int C = Integer.parseInt(st.nextToken()); // 공유기의 개수

        int[] houses = new int[N];

        // 집 좌표 입력
        for (int i = 0; i < N; i++) {
            houses[i] = Integer.parseInt(br.readLine());
        }

        // 집 좌표를 오름차순으로 정렬
        Arrays.sort(houses);

        // 이분 탐색 범위 설정
        int left = 1; // 최소 거리
        int right = houses[N - 1] - houses[0]; // 최대 거리
        int result = 0;

        // 이분 탐색 시작
        while (left <= right) {
            int mid = (left + right) / 2; // 현재 중간 거리
```

```
            int installed = houses[0]; // 첫 번째 집에 공유기 설치
            int count = 1; // 첫 번째 공유기 설치 완료

            // 다음 공유기를 설치할 수 있는지 확인
            for (int i = 1; i < N; i++) {
                if (houses[i] - installed >= mid) {
                    count++;
                    installed = houses[i]; // 공유기 설치
                }
            }

            if (count >= C) { // C개 이상의 공유기를 설치할 수 있다면
                result = mid; // 가능한 거리 중 가장 큰 값 저장
                left = mid + 1; // 더 큰 거리에서 탐색
            } else { // C개 미만의 공유기를 설치할 수 있다면
                right = mid - 1; // 거리를 줄여 탐색
            }
        }

        System.out.println(result); // 가장 인접한 두 공유기 사이의 최대 거리 출력
    }
}
```