MAX-DISTANCE SUB-SEQUENCE

- Given N elements (2≤ N ≤100,000) on a straight line at positions x_1,…, x_n (0≤ x ≤1,000,000,000)
- The distance of a subset of N elements is defined to be the minimum distance between two elements
- Find the subset of N given elements containing exactly C elements such that the distance is maximal.
- **Input**
  - The first line contains a positive integer T (1 <= T <= 20) which is the number of test cases.
  - Subsequent lines are T test cases with the following format:
    - Line 1: Two space-separated integers: N and C
    - Lines 2: contains x_1, x_2 , . . , x_n
- **Output**
  - For each test case output one integer: the distance of the subset found.

MAX-DISTANCE SUB-SEQUENCE

- Example

stdin	stdout
1 5 3 1 2 8 4 9	3

MAX-DISTANCE SUB-SEQUENCE- Hint

- Sắp xếp lại mảng theo thứ tự tăng dần

- Sử dụng thuật toán tìm kiếm nhị phân để thử kết quả

- Thực hiện tìm kiếm nhị phân để tìm kết quả tốt nhất d:
- *0 <= d <= (a[n] – a[1])*

- Bắt đầu thử từ giá trị d = mid:
- Kiểm tra xem có thể chọn được ít nhất c phần tử, mỗi phần tử cách nhau ít nhất d đơn vị hay không
- Nếu giá trị d = mid là 1 giá trị khả thi, ta tiếp tục tìm kiếm ở nửa bên phải.
- Ngược lại, ta sẽ tìm kiếm tiếp ở nửa bên trái

Implementation

```
int MaxDistance () {
    int l = 0, r = a[n] - a[1];
    while (l<=r) {
        int mid = (l+r)/2;
        if (check(mid)) l = mid+1;   // tiếp tục thử kết quả ở nửa bên phải
        else r = mid-1;              // tìm kết quả ở nửa bên trái
    }
    return r;
}

int main() {
    ios_base::sync_with_stdio(0); cin.tie(NULL); cout.tie(NULL);
    cin >> t;
    while (t--) {
        cin >> n >> c;
        for (int i=1; i<=n; i++)  cin >> a[i];
        sort(a+1, a+n+1);
        cout << MaxDistance() << endl;
    }
}
```

Implementation

```
#include <bits/stdc++.h>
#define MAXN 100005
using namespace std;

int t;
int n, c, a[MAXN];

bool check (int distance) {
    int sl = 1;
    int i=1, j=2;
    while (i<n) {
        while (j<=n && a[j]-a[i] < distance)  ++j;
        if (j<=n) sl++;
        if (sl>=c) return true;  // có thể lấy đủ c phần tử
        i = j;
        j++;
    }
    return false;
}
```