Day 77 coding Statement :

You are given an array A of N elements. For any ordered triplet (i,j,k) such that i, j, and k are pairwise distinct and $1 \le i,j,k \le N$, the value of this triplet is $(A_i - A_j) \cdot A_k$. You need to find the **maximum** value among all possible ordered triplets.

Note: Two ordered triplets (a,b,c) and (d,e,f) are only equal when $a=d$ **and** $b=e$ **and** $c=f$. As an example, (1,2,3) and (2,3,1) are two different ordered triplets.

Input Format

- The first line of the input contains a single integer T - the number of test cases. The test cases then follow.
- The first line of each test case contains an integer N.
- The second line of each test case contains N space-separated integers $A_1, A_2, \ldots, A_N$.

Output Format

For each test case, output the maximum value among all different ordered triplets.

Sample Input

3

3

1 1 3

5

3 4 4 1 2

5

23 17 21 18 19

Sample Output

2

12

```
import java.util.*;
import java.lang.*;
import java.io.*;

public class Program {
	public static void main(String[] args) throws java.lang.Exception {
		BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
		int t = Integer.parseInt(br.readLine());
		while (t-- > 0) {
			int n = Integer.parseInt(br.readLine());
			String s = br.readLine();
			String sr[] = s.split(" ");
			int ar[] = new int[n];
			for (int i = 0; i < n; i++)
				ar[i] = Integer.parseInt(sr[i]);
			Arrays.sort(ar);
			long cout = Integer.MIN_VALUE;
			cout = (long) (ar[n - 1] - ar[0]) * ar[n - 2];
			System.out.println(cout);
		}
	}
}
```