```
/*Day 78 coding Statement :

For a given array B1?,B2?,…,BM? of length at least 3, let's define its weight as the largest
value of (Bi?−Bj?)⋅(Bj?−Bk?)
over all possible triples (i,j,k) with 1≤i,j,k≤M and i!=j, j!=k, k!=i.
You are given a sorted array A1?,A2?,…,AN? (that is, A1?≤A2?≤…≤AN?).
Calculate the sum of weights of all contiguous subarrays of A of
length at least 3. That is, count the sum of weights of arrays [Ai?,Ai+1?,…,Aj?] over all
1≤i<j≤N with j−i≥2.*/
import java.util.*;
import java.lang.*;
import java.io.*;
class Main
{
 public static void main (String[] args) throws java.lang.Exception
 {
 MyScanner sc = new MyScanner();
 PrintWriter out = new PrintWriter(new BufferedOutputStream(System.out));
 int tt = sc.nextInt();
 while (tt-- > 0) {
 int n = sc.nextInt();
 int [] a = new int[n];
 TreeSet<Integer> set = new TreeSet<>();
 for (int i = 0; i < n; i++) {
 a[i] = sc.nextInt();
 set.add(a[i]);
 }
 long ans = 0;
 for (int i = 0; i < n; i++) {
 for (int j = i + 2; j < n; j++) {
 int s = a[i];
 int e = a[j];
 int mean = (s + e) / 2;
 long res = 0;
 Integer lo = set.lower(mean);

 if (lo != null) {
 res = Math.max(res, multiply(e - lo, lo -s));
 }
 Integer hi = set.higher(mean);
 if (hi != null) {
 res = Math.max(res, multiply(e - hi, hi -s));
 }
 if (set.contains(mean)) {
 res = Math.max(res, multiply(e - mean, mean -s));
 }
 ans += res;
 }
 }
 out.println(ans);
 }
 out.close();
 }
 static long multiply(int x, int y) {
 return (long) x * (long) y;
 }
 static void sort(long[] a) {
 ArrayList<Long> q = new ArrayList<>();
 for (long i : a) q.add(i);
 Collections.sort(q);
 for (int i = 0; i < a.length; i++) a[i] = q.get(i);
 }
```

```
public static class MyScanner {
BufferedReader br;
StringTokenizer st;
public MyScanner() {
br = new BufferedReader(new InputStreamReader(System.in));
}
String next() {
while (st == null || !st.hasMoreElements()) {
try {
st = new StringTokenizer(br.readLine());
} catch (IOException e) {
e.printStackTrace();
}
}
return st.nextToken();
}
int nextInt() {
return Integer.parseInt(next());
}
long nextLong() {
return Long.parseLong(next());
}
double nextDouble() {
return Double.parseDouble(next());

}
String nextLine(){
String str = "";
try {
str = br.readLine();
} catch (IOException e) {
e.printStackTrace();
}
```

```
 return str;
 }
 }
}
```