Time Complexity and Sorting
Time Complexity
Crash course in Big O notation
Examples

This is O(1)

```java
int a = 5;
int b = 7;
int c = 4;
int d = a + b + c + 153;
```

This is O(n)

```java
for (int i = 1; i <= n; i++) {
    // constant time code here
}
```

Still O(n)

```java
for (int i = 1; i <= 5 * n + 17; i++) {
    // constant time code here
}
```

```java
for (int i = 1; i <= n + 457737; i++) {
    // constant time code here
}
```
More examples

```
1 for (int i = 1; i <= n; i++) {
2     for (int j = 1; j <= m; j++) {
3         // constant time code here
4     }
5 }
```

$O(nm)$

```
1 for (int i = 1; i <= n; i++) {
2     for (int j = i; j <= n; j++) {
3         // constant time code here
4     }
5 }
```

$O(n^2)$

```
1 for (int i = 1; i <= n; i++) {
2     for (int j = 1; j <= n; j++) {
3         // constant time code here
4     }
5 }
6     for (int i = 1; i <= m; i++) {
7         // more constant time code here
8 }
```

$O(n^2 + m)$
## Time/Big O table for competitive programming

<table>
<thead>
<tr>
<th>$n$</th>
<th>Possible complexities</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n \leq 10$</td>
<td>$O(n!), O(n^7), O(n^6)$</td>
</tr>
<tr>
<td>$n \leq 20$</td>
<td>$O(2^n \cdot n), O(n^5)$</td>
</tr>
<tr>
<td>$n \leq 80$</td>
<td>$O(n^4)$</td>
</tr>
<tr>
<td>$n \leq 400$</td>
<td>$O(n^3)$</td>
</tr>
<tr>
<td>$n \leq 7500$</td>
<td>$O(n^2)$</td>
</tr>
<tr>
<td>$n \leq 7 \cdot 10^4$</td>
<td>$O(n \sqrt{n})$</td>
</tr>
<tr>
<td>$n \leq 5 \cdot 10^5$</td>
<td>$O(n \log n)$</td>
</tr>
<tr>
<td>$n \leq 5 \cdot 10^6$</td>
<td>$O(n)$</td>
</tr>
<tr>
<td>$n \leq 10^{18}$</td>
<td>$O(\log^2 n), O(\log n), O(1)$</td>
</tr>
</tbody>
</table>
Example problem: odd sum

B. Odd sums

Time limit per test: 1 second
Memory limit per test: 256 megabytes
Input: standard input
Output: standard output

You are given a list with \( n \) integers. Find how many pairs of distinct elements sum to an odd number.

Input
The input starts with a line with a single integer \( n \), between 1 and 100,000. Then follow \( n \) lines, each with a single integer between 1 and 1,000,000,000 representing the list.

Output
The output should be a single integer.

Examples

<table>
<thead>
<tr>
<th>input</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>input</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

Note
In the first example, the pairs are (1, 2), (3, 2), (5, 2), (1, 4), (3, 4), (5, 4).
In the second example, the pairs are (1, 2), (1, 2), (1, 2).
Naive solution

```java
import java.util.Scanner;

public class OddsNaive {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[] ls = new int[n];

        for (int i = 0; i < n; i++)
            ls[i] = sc.nextInt();

        long ans = 0;
        for (int i = 0; i < n; i++)
            for (int j = i + 1; j < n; j++)
                if ((ls[i] + ls[j]) % 2 == 1)
                    ans++;

        System.out.println(ans);
    }
}
```

This is $O(n^2)$
Better solution

```java
import java.util.Scanner;

public class OddsBetter {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        long countOdds = 0;
        long countEvens = 0;

        for (int i = 0; i < n; i++) {
            if (sc.nextInt() % 2 == 0) {
                countEvens++;
            } else {
                countOdds++;
            }
        }
        System.out.println(countEvens * countOdds);
    }
}

This is O(n)
Sorting
Know how to fast sort in your favorite language!!

**C++**

Use: 

```cpp
sort(arr, arr + N)
```

where \( N \) is the number of elements to be sorted

You will need to import `<algorithm>` and this is part of the stl namespace, so you need to:

```cpp
using namespace std;
```
Two examples in C++

```cpp
#include <bits/stdc++.h>
using namespace std;

int main() {
    int arr[] = {5, 1, 3, 2, 4}; int N = 5;
    sort(arr, arr + N);
    for (int i = 0; i < N; i++)
        cout << arr[i] << " "; //1 2 3 4 5
    cout << endl;

    int arr2[] = {5, 1, 3, 2, 4};
    sort(arr2 + 1, arr2 + 4);
    for (int i = 0; i < N; i++)
        cout << arr2[i] << " "; //5 1 2 3 4
}
```
Two examples in Java

```java
import java.util.*;

class Main {
    public static void main (String[] args) {
        int arr[] = {5, 1, 3, 2, 4};
        Arrays.sort(arr);
        for (int i = 0; i < arr.length; i++)
            System.out.print(arr[i] + " "); //1 2 3 4 5
    }
}
```
Sources:

Most of the code shown and the time table were taken from: https://usaco.guide/