[Skip to content](https://chatgpt.com/?openaicom_referred=true" \l "main)

**You said:**

what would be a better BRUTE FORCE answer to the LeetCode TwoSum problem than public class Solution

{

    public Solution()

    {

        int[] nums = {3,3};

        int t = 6;

        if(TwoSum(nums, t)!=null)

        {

            int[] returnedA = TwoSum(nums, t);

            foreach (int r in returnedA)

            {

                Console.WriteLine(r);

            }

        }

    }

    public int[] TwoSum(int[] nums, int target)

    {

            for (int x = 0; x < nums.Length; x++)

            {

                for (int y = 0; y < nums.Length; y++)

                {

                    int sum;

                    int numsx = nums[x];

                    int numsy = nums[y];

                    if (numsx != numsy || x != y)

                    {

                        sum = numsx + numsy;

                            if (sum == target)

                            {

                                int[] returnArray = { x, y };

                                return returnArray;

                            }

                    }

                }

            }

            return null;

    }

}

**ChatGPT said:**

You're on the right track with a brute force approach, but your current implementation can be improved **without changing the algorithmic complexity**, just cleaning up redundant logic and improving readability and correctness.

**Issues in Your Code:**

1. You're calling TwoSum(nums, t) **twice**. This is inefficient. You should store the result once.
2. You're checking numsx != numsy || x != y, but that condition is confusing and unnecessary. The correct condition should simply be x != y, since the problem allows using the same value twice **only if they're at different indices**.
3. The brute force approach should only check **pairs** (x, y) where y > x or x != y, to avoid redundant checks.
4. You could avoid some redundant assignments (int numsx = nums[x]; int numsy = nums[y];) unless you're doing something special with them.

**✅ Cleaned-Up Brute Force Version:**

public class Solution

{

public Solution()

{

int[] nums = { 3, 3 };

int t = 6;

int[] result = TwoSum(nums, t);

if (result != null)

{

foreach (int r in result)

{

Console.WriteLine(r);

}

}

}

public int[] TwoSum(int[] nums, int target)

{

for (int i = 0; i < nums.Length; i++)

{

for (int j = i + 1; j < nums.Length; j++)

{

if (nums[i] + nums[j] == target)

{

return new int[] { i, j };

}

}

}

return null;

}

}

**✅ Improvements Made:**

* Only checks each pair once (j = i + 1), avoiding (x,y) and (y,x) redundancy.
* Removes unnecessary checks for values (numsx != numsy is not needed).
* Avoids unnecessary re-calls and temporary variables.
* Cleaner, more readable structure.

**⏱ Time Complexity**

* Still **O(n²)** — brute force — but **cleaner and slightly more efficient in practice**.

If you eventually want a more efficient solution, consider a **hash map** for an O(n) solution. Let me know if you'd like that too!

Top of Form

Bottom of Form

ChatGPT can make mistakes. Check important info. See Cookie Preferences.

**We use cookies**

Some cookies are essential for this site to function and cannot be turned off. Others are set to help us understand how our service performs and is used, and to support our marketing efforts. Learn more in our [Cookie Policy](https://openai.com/policies/cookie-policy/). For more info or to customize your preferences, tap ‘Manage Cookies’.