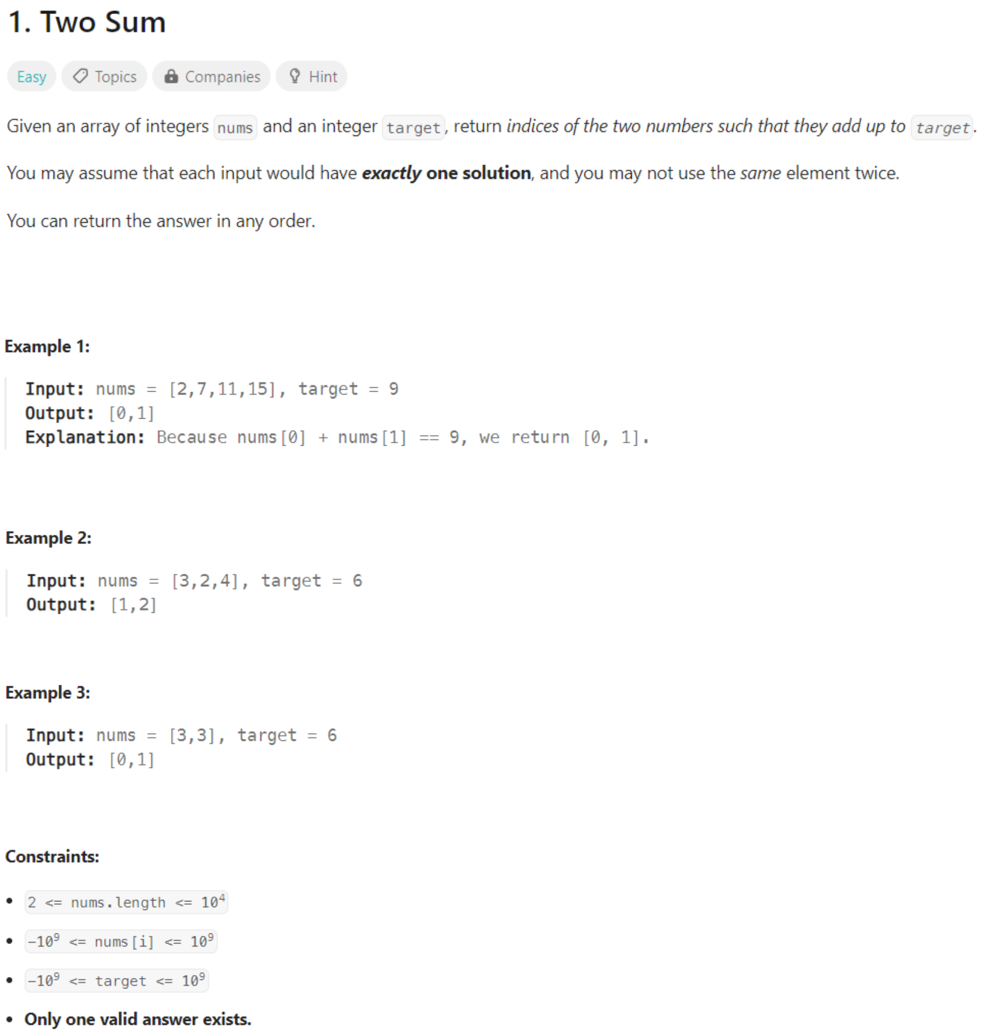
Problema de Two Sum



1. Two Sum

Given an array of integers `nums` and an integer `target`, return *indices of the two numbers such that they add up to `target`*.

You may assume that each input would have ***exactly* one solution**, and you may not use the *same* element twice.

You can return the answer in any order.

Example 1:

```
Input: nums = [2,7,11,15], target = 9
Output: [0,1]
Explanation: Because nums[0] + nums[1] == 9, we return [0, 1].
```

Example 2:

```
Input: nums = [3,2,4], target = 6
Output: [1,2]
```

Example 3:

```
Input: nums = [3,3], target = 6
Output: [0,1]
```

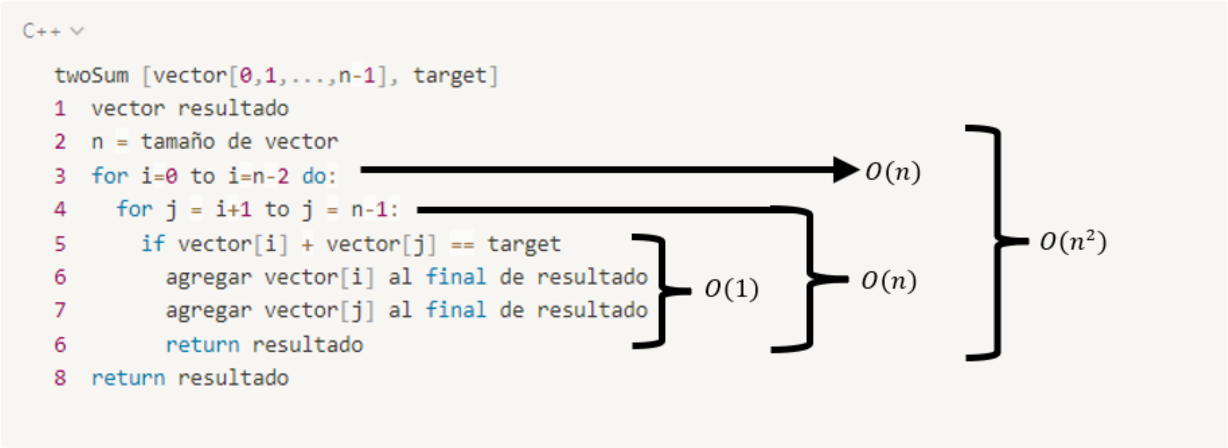
Constraints:

- `2 <= nums.length <=` 10^4
- -10^9 `<= nums[i] <=` 10^9
- -10^9 `<= target <=` 10^9
- **Only one valid answer exists.**

Pseudocódigo.

```
twoSum [vector[0,1,...,n-1], target]
vector resultado
n = tamaño de vector
for i=0 to i=n-2 do:
  for j = i+1 to j = n-1:
    if vector[i] + vector[j] == target
      agregar vector[i] al final de resultado
      agregar vector[j] al final de resultado
      return resultado
return resultado
```

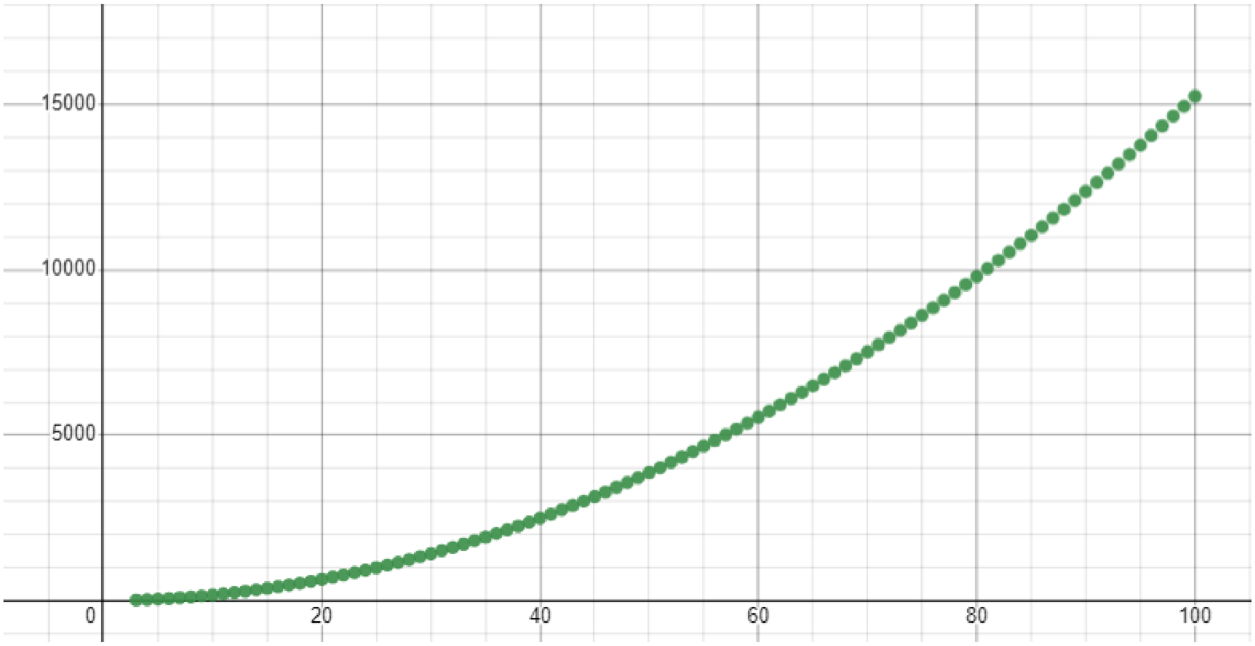
Análisis a priori:



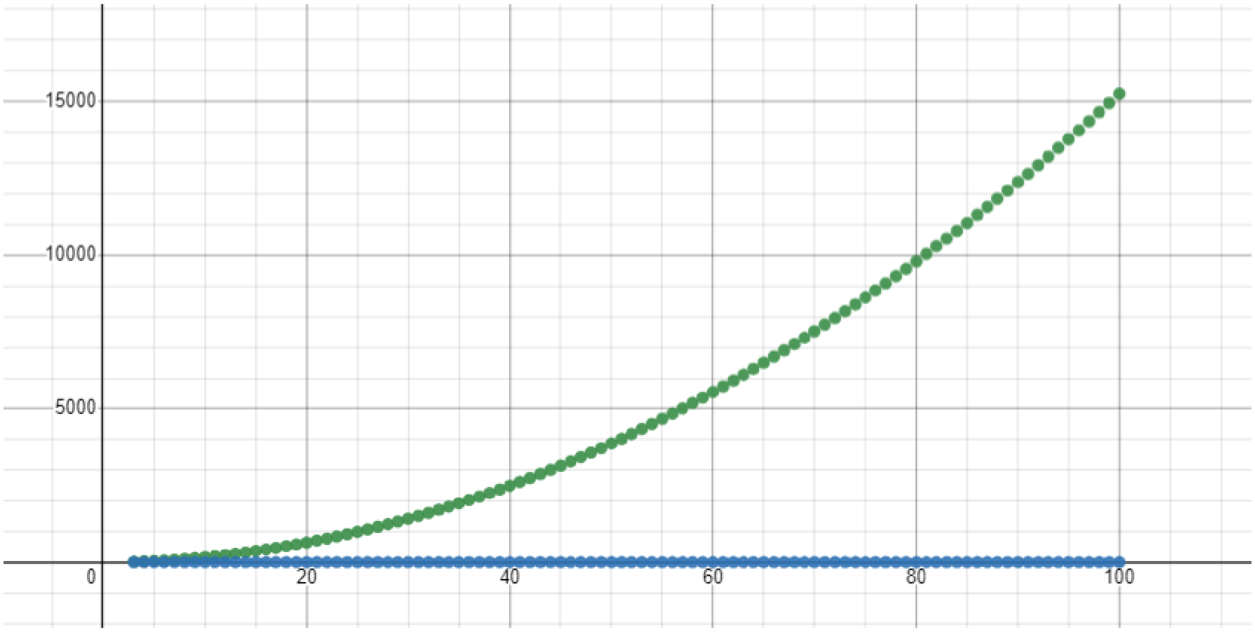
```
twoSum [vector[0,1,...,n-1], target]
vector resultado
n = tamaño de vector
for i=0 to i=n-2 do:
  for j = i+1 to j = n-1:
    if vector[i] + vector[j] == target
      agregar vector[i] al final de resultado
      agregar vector[j] al final de resultado
      return resultado
return resultado
```

$$\therefore twoSum \in O(n^2)$$

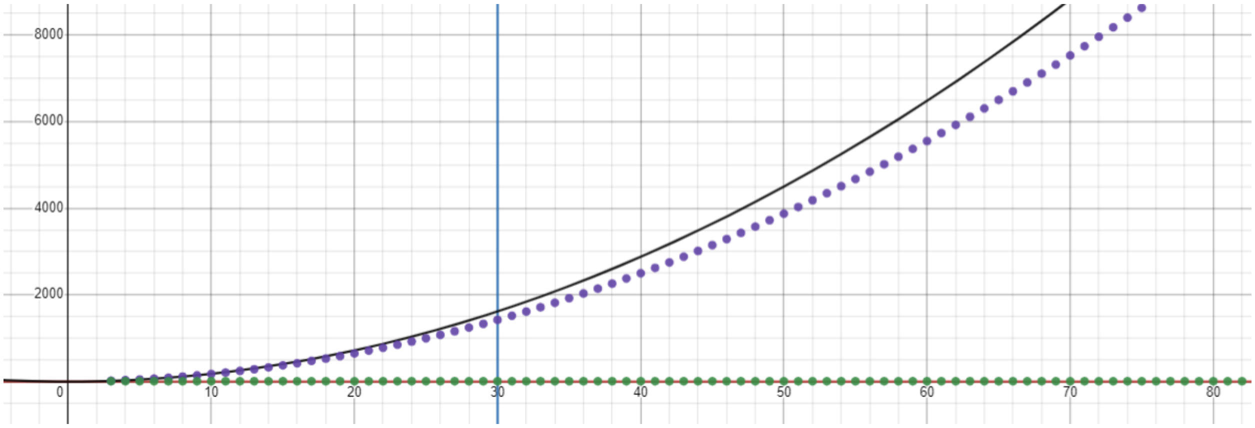
Análisis a posteriori



$$\therefore twoSum \in O(n^2)$$



$$\therefore twoSum \in \Omega(1)$$



$g(n) = 1.8n^2$ $\quad z(n) = 0$ $\quad \forall n \geq 30$

Código

```
vector<int> twoSum(vector<int>& nums, int target) {
    vector<int> result ={};
    int n = nums.size();
    for(int i=0;i<n-1;i++){
        for(int j=i+1;j<n;j++){
            if((nums[i]+nums[j])==target){
                result.push_back(i);
                result.push_back(j);

                return result;
            }
        }
    }
    return result;
}
```

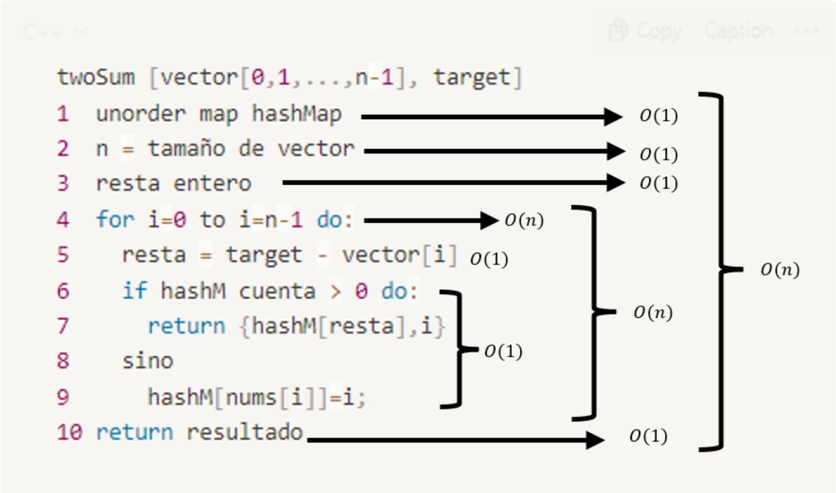
Solución Lineal.

Pseudocódigo.

```
twoSum [vector[0,1,...,n-1], target]
1  unorder map hashMap
```

```
n = tamaño de vector
resta entero
for i=0 to i=n-1 do:
  resta = target - vector[i]
  if hashM cuenta > 0 do:
    return {hashM[resta],i}
  sino
    hashM[nums[i]]=i;
return resultado
```

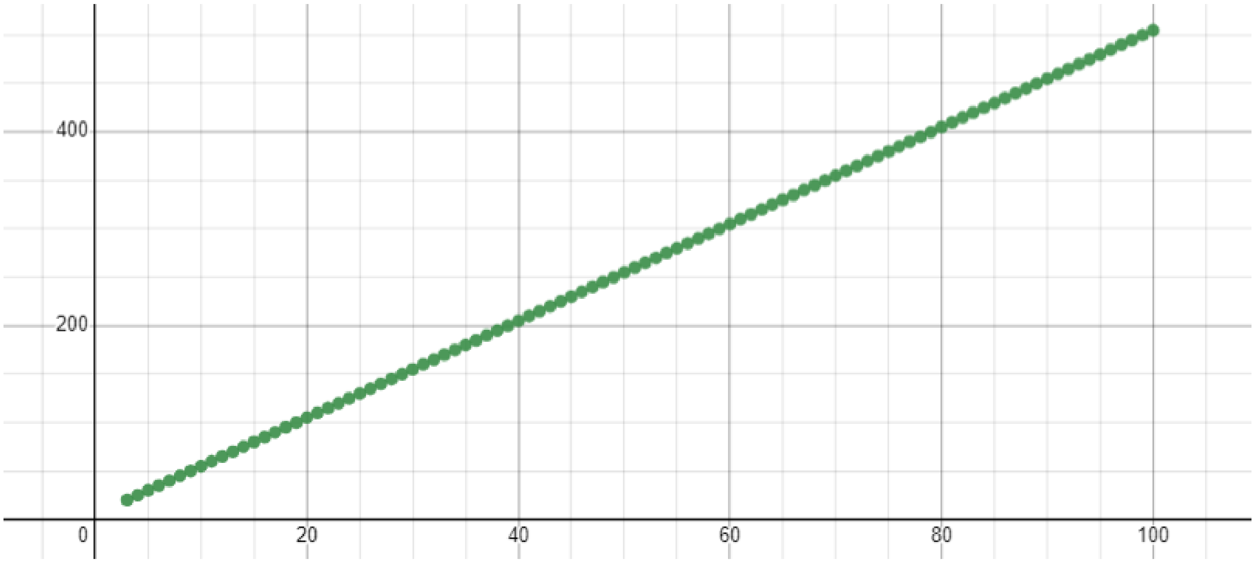
Análisis a priori:



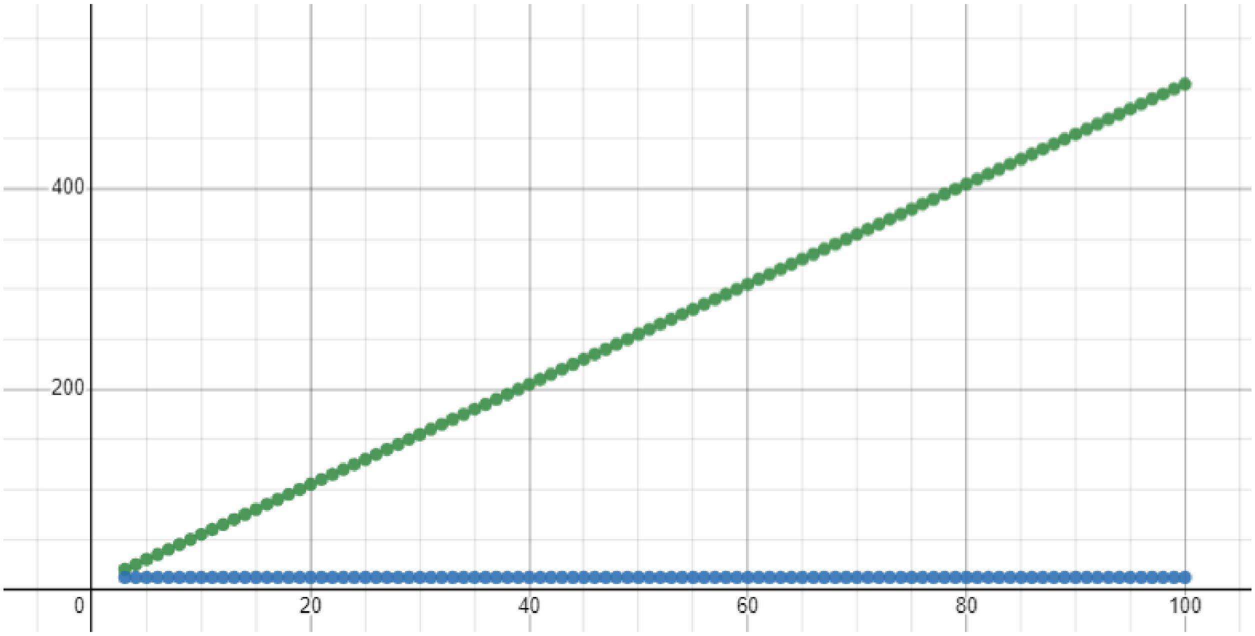
```
twoSum [vector[0,1,...,n-1], target]
unorder map hashMap
n = tamaño de vector
resta entero
for i=0 to i=n-1 do:
  resta = target - vector[i]
  if hashM cuenta > 0 do:
    return {hashM[resta],i}
  sino
    hashM[nums[i]]=i;
return resultado
```

$$\therefore twoSum \in O(n)$$

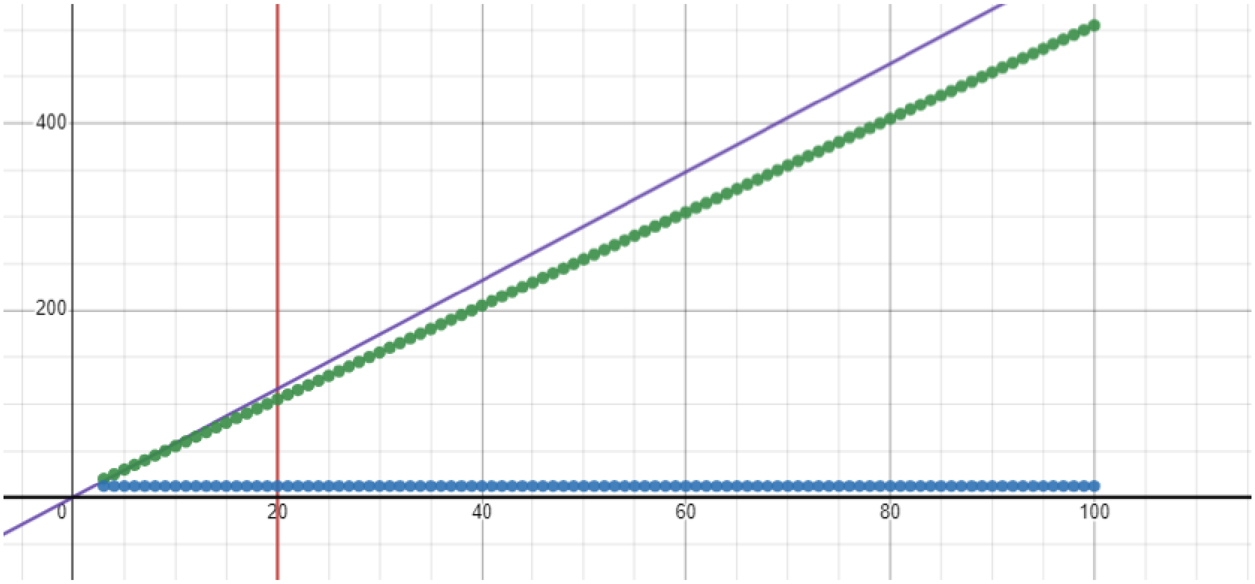
Análisis a posteriori.



$$\therefore twoSum \in O(n)$$



$$\therefore twoSum \in \Omega(1)$$



$g(n) = 5.8n$ $z(n) = 0$ $\forall n \geq 20$

Código

```
vector<int> twoSum(vector<int>& nums, int target) {
        unordered_map<int,int> hashM;
        int n = nums.size(),resta;
        for(int i=0;i<n;i++){
            resta = target-nums[i];
            if(hashM.count(resta)>0)
                return {hashM[resta],i};
            else
                hashM[nums[i]]=i;
        }
        return {};
}
```

Gracias por su atención.