# Problem 2903: Find Indices With Index and Value Difference I

## Problem Information

**Difficulty:** Easy
**Acceptance Rate:** 0.00%
**Paid Only:** No

## Problem Description

You are given a

0-indexed

integer array

nums

having length

n

, an integer

indexDifference

, and an integer

valueDifference

.

Your task is to find

two

indices

i

and

j

, both in the range

[0, n - 1]

, that satisfy the following conditions:

abs(i - j) >= indexDifference

, and

abs(nums[i] - nums[j]) >= valueDifference

Return

an integer array

answer

,

where

answer = [i, j]

if there are two such indices

,

and

answer = [-1, -1]

otherwise

. If there are multiple choices for the two indices, return

any of them

.

Note:

i

and

j

may be

equal

.

Example 1:

Input:

nums = [5,1,4,1], indexDifference = 2, valueDifference = 4

Output:

[0,3]

Explanation:

In this example, i = 0 and j = 3 can be selected. abs(0 - 3) >= 2 and abs(nums[0] - nums[3]) >= 4. Hence, a valid answer is [0,3]. [3,0] is also a valid answer.

Example 2:

Input:

nums = [2,1], indexDifference = 0, valueDifference = 0

Output:

[0,0]

Explanation:

In this example, i = 0 and j = 0 can be selected. abs(0 - 0) >= 0 and abs(nums[0] - nums[0]) >= 0. Hence, a valid answer is [0,0]. Other valid answers are [0,1], [1,0], and [1,1].

Example 3:

Input:

nums = [1,2,3], indexDifference = 2, valueDifference = 4

Output:

[-1,-1]

Explanation:

In this example, it can be shown that it is impossible to find two indices that satisfy both conditions. Hence, [-1,-1] is returned.

Constraints:

1 <= n == nums.length <= 100

0 <= nums[i] <= 50

0 <= indexDifference <= 100

0 <= valueDifference <= 50

## Code Snippets

### C++:

```
class Solution {
public:
vector<int> findIndices(vector<int>& nums, int indexDifference, int
valueDifference) {

}
};
```

### Java:

```
class Solution {
public int[] findIndices(int[] nums, int indexDifference, int
valueDifference) {

}
}
```

### Python3:

```
class Solution:
def findIndices(self, nums: List[int], indexDifference: int, valueDifference:
int) -> List[int]:
```

### Python:

```
class Solution(object):
def findIndices(self, nums, indexDifference, valueDifference):
"""
:type nums: List[int]
:type indexDifference: int
:type valueDifference: int
:rtype: List[int]
"""
```

### JavaScript:

```
/**
* @param {number[]} nums
* @param {number} indexDifference
* @param {number} valueDifference
* @return {number[]}
*/
var findIndices = function(nums, indexDifference, valueDifference) {

};
```

**TypeScript:**

```
function findIndices(nums: number[], indexDifference: number,
valueDifference: number): number[] {

};
```

**C#:**

```
public class Solution {
public int[] FindIndices(int[] nums, int indexDifference, int
valueDifference) {

}
}
```

**C:**

```
/**
* Note: The returned array must be malloced, assume caller calls free().
*/
int* findIndices(int* nums, int numsSize, int indexDifference, int
valueDifference, int* returnSize) {

}
```

**Go:**

```
func findIndices(nums []int, indexDifference int, valueDifference int) []int
{

}
```

**Kotlin:**

```
class Solution {
fun findIndices(nums: IntArray, indexDifference: Int, valueDifference: Int):
IntArray {

}
}
```

**Swift:**

```
class Solution {
func findIndices(_ nums: [Int], _ indexDifference: Int, _ valueDifference:
Int) -> [Int] {

}
}
```

**Rust:**

```
impl Solution {
pub fn find_indices(nums: Vec<i32>, index_difference: i32, value_difference:
i32) -> Vec<i32> {

}
}
```

**Ruby:**

```
# @param {Integer[]} nums
# @param {Integer} index_difference
# @param {Integer} value_difference
# @return {Integer[]}
def find_indices(nums, index_difference, value_difference)

end
```

**PHP:**

```
class Solution {

/**
* @param Integer[] $nums
* @param Integer $indexDifference
* @param Integer $valueDifference
* @return Integer[]
*/
function findIndices($nums, $indexDifference, $valueDifference) {

}
}
```

**Dart:**

```
class Solution {
List<int> findIndices(List<int> nums, int indexDifference, int
valueDifference) {

}
}
```

**Scala:**

```
object Solution {
def findIndices(nums: Array[Int], indexDifference: Int, valueDifference:
Int): Array[Int] = {

}
}
```

**Elixir:**

```
defmodule Solution do
@spec find_indices(nums :: [integer], index_difference :: integer,
value_difference :: integer) :: [integer]
def find_indices(nums, index_difference, value_difference) do

end
end
```

**Erlang:**

```
-spec find_indices(Nums :: [integer()], IndexDifference :: integer(),
ValueDifference :: integer()) -> [integer()].
find_indices(Nums, IndexDifference, ValueDifference) ->
```

```
.
```

**Racket:**

```
(define/contract (find-indices nums indexDifference valueDifference)
(-> (listof exact-integer?) exact-integer? exact-integer? (listof
exact-integer?))
)
```

## Solutions

### C++ Solution:

```
/*
* Problem: Find Indices With Index and Value Difference I
* Difficulty: Easy
* Tags: array
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public:
vector<int> findIndices(vector<int>& nums, int indexDifference, int
valueDifference) {

}
};
```

### Java Solution:

```
/**
* Problem: Find Indices With Index and Value Difference I
* Difficulty: Easy
* Tags: array
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public int[] findIndices(int[] nums, int indexDifference, int
valueDifference) {

}
}
```

**Python3 Solution:**

```
"""
Problem: Find Indices With Index and Value Difference I
Difficulty: Easy
Tags: array

Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(1) to O(n) depending on approach
"""

class Solution:
def findIndices(self, nums: List[int], indexDifference: int, valueDifference:
int) -> List[int]:
# TODO: Implement optimized solution
pass
```

**Python Solution:**

```
class Solution(object):
def findIndices(self, nums, indexDifference, valueDifference):
"""
:type nums: List[int]
:type indexDifference: int
:type valueDifference: int
:rtype: List[int]
"""
```

**JavaScript Solution:**

```
/**
* Problem: Find Indices With Index and Value Difference I
* Difficulty: Easy
* Tags: array
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* @param {number[]} nums
* @param {number} indexDifference
* @param {number} valueDifference
* @return {number[]}
*/
var findIndices = function(nums, indexDifference, valueDifference) {

};
```

### TypeScript Solution:

```
/**
* Problem: Find Indices With Index and Value Difference I
* Difficulty: Easy
* Tags: array
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

function findIndices(nums: number[], indexDifference: number,
valueDifference: number): number[] {

};
```

### C# Solution:

```
/*
* Problem: Find Indices With Index and Value Difference I
* Difficulty: Easy
* Tags: array
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

public class Solution {
public int[] FindIndices(int[] nums, int indexDifference, int
valueDifference) {

}
}
```

### C Solution:

```
/*
* Problem: Find Indices With Index and Value Difference I
* Difficulty: Easy
* Tags: array
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* Note: The returned array must be malloced, assume caller calls free().
*/
int* findIndices(int* nums, int numsSize, int indexDifference, int
valueDifference, int* returnSize) {

}
```

### Go Solution:

```
// Problem: Find Indices With Index and Value Difference I
// Difficulty: Easy
// Tags: array
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

func findIndices(nums []int, indexDifference int, valueDifference int) []int
{

}
```

**Kotlin Solution:**

```
class Solution {
fun findIndices(nums: IntArray, indexDifference: Int, valueDifference: Int):
IntArray {

}
}
```

**Swift Solution:**

```
class Solution {
func findIndices(_ nums: [Int], _ indexDifference: Int, _ valueDifference:
Int) -> [Int] {

}
}
```

**Rust Solution:**

```
// Problem: Find Indices With Index and Value Difference I
// Difficulty: Easy
// Tags: array
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

impl Solution {
pub fn find_indices(nums: Vec<i32>, index_difference: i32, value_difference:
i32) -> Vec<i32> {

}
}
```

**Ruby Solution:**

```
# @param {Integer[]} nums
# @param {Integer} index_difference
# @param {Integer} value_difference
# @return {Integer[]}
def find_indices(nums, index_difference, value_difference)

end
```

**PHP Solution:**

```
class Solution {

/**
* @param Integer[] $nums
* @param Integer $indexDifference
* @param Integer $valueDifference
* @return Integer[]
*/
function findIndices($nums, $indexDifference, $valueDifference) {

}
}
```

**Dart Solution:**

```
class Solution {
List<int> findIndices(List<int> nums, int indexDifference, int
valueDifference) {

}
}
```

**Scala Solution:**

```
object Solution {
def findIndices(nums: Array[Int], indexDifference: Int, valueDifference:
Int): Array[Int] = {
}
}
```

**Elixir Solution:**

```
defmodule Solution do
@spec find_indices(nums :: [integer], index_difference :: integer,
value_difference :: integer) :: [integer]
def find_indices(nums, index_difference, value_difference) do

end
end
```

**Erlang Solution:**

```
-spec find_indices(Nums :: [integer()], IndexDifference :: integer(),
ValueDifference :: integer()) -> [integer()].
find_indices(Nums, IndexDifference, ValueDifference) ->
.
```

**Racket Solution:**

```
(define/contract (find-indices nums indexDifference valueDifference)
(-> (listof exact-integer?) exact-integer? exact-integer? (listof
exact-integer?))
)
```