Sample Output 0
3

Explanation 0
There are three pairs with an absolute difference of 1: (1,2), (2,3), and (3,4).

Sample Input 1
5
1 3 1 5 4
0

Sample Output 1
1

Explanation 1
There is one pair with an absolute difference of 0: (1,1) considering the position in the array, not the value.

Sample Input 2
4
1 2 2 1
1

Sample Output 2
4

Explanation 2
The pairs with an absolute difference of 1 are:
- [**1**,**2**,2,1]
- [**1**,2,**2**,1]
- [1,**2**,2,**1**]
- [1,2,**2**,**1**]

Program:

```
def countKDiff (nums, k):
    count=0
    from collections import Counter
    mydict=Counter(nums)
    for i in mydict:
```



```
        if i+k in mydict:
            count=count+ mydict[i]*mydict[i+k]
    return count
```

Ex. No. : 12.3 **Date:**

Register No.: **Name:**

Square Tiles

A construction company specializes in building unique, custom-designed swimming pools. One of their popular offerings is circular swimming pools. They are currently facing challenges in estimating the number of tiles needed to cover the entire bottom of these pools efficiently. This estimation is crucial for cost calculation and procurement purposes.

Problem Statement:

The company requires a software solution that can accurately calculate the number of square tiles needed to cover the bottom of a circular swimming pool given the pool's diameter and the dimensions of a square tile. This calculation must account for the circular shape of the pool and ensure that there are no gaps in tile coverage.

Program:

```
import math
def calculate_tiles_needed(pool_diameter_meters, tile_side_cm):
    pool_diameter_cm = pool_diameter_meters * 100
    pool_radius_cm = pool_diameter_cm / 2
    pool_area_sq_cm = math.pi * (pool_radius_cm ** 2)
    tile_area_sq_cm = tile_side_cm ** 2
    number_of_tiles = math.ceil(pool_area_sq_cm / tile_area_sq_cm)
    return number_of_tiles
pool_diameter_meters = int(input())
tile_side_cm = int(input())
tiles_needed = calculate_tiles_needed(pool_diameter_meters, tile_side_cm)
print(f"Number of tiles needed: {tiles_needed}")
```


Input Format:

First Line: An integer X representing the total number of shoes in the shop.

Second Line: A space-separated list of integers representing the shoe sizes in the shop.

Third Line: An integer N representing the number of customer requests.

Next N Lines: Each line contains a pair of space-separated values:

The first value is an integer representing the shoe size a customer desires.

The second value is an integer representing the price the customer is willing to pay for that

Register No.: **Name:**

Total Revenue

Raghu owns a shoe shop with a varying inventory of shoe sizes. The shop caters to multiple customers who have specific size requirements and are willing to pay a designated amount for their desired shoe size. Raghu needs an efficient system to manage his inventory and calculate the total revenue generated from sales based on customer demands.

Problem Statement:

Develop a Python program that manages shoe inventory and processes sales transactions to determine the total revenue generated. The program should handle inputs of shoe sizes available in the shop, track the number of each size, and match these with customer purchase requests. Each transaction should only proceed if the desired shoe size is in stock, and the inventory should update accordingly after each sale.

Program:

```
from collections import Counter
def calculate_total_revenue(shoe_inventory, customer_requests):
    inventory_counter = Counter(shoe_inventory)
    total_revenue = 0
    for size, price in customer_requests:
        if inventory_counter[size] > 0:
            total_revenue += price
            inventory_counter[size] -= 1
    return total_revenue

X = int(input())
shoe_inventory = list(map(int, input().split()))
N = int(input())
customer_requests = []
for _ in range(N):
            size, price = map(int, input().split())
            customer_requests.append((size, price))
total_revenue = calculate_total_revenue(shoe_inventory, customer_requests)

print(total_revenue)
```

Problem Statement:

Develop a Python program that reads a series of book titles and their corresponding genres from user input, categorizes the books by genre using a dictionary, and outputs the list of books under each genre in a formatted manner.

Input Format:

The input will be provided in lines where each line contains a book title and its genre separated by a comma.

Input terminates with a blank line.

Output Format:

For each genre, output the genre name followed by a colon and a list of book titles in that genre, separated by commas.

Program:

```
from collections import Counter, defaultdict
library = defaultdict(list)
while True:
    line = input().strip()
    if not line:
        break
    try:
        title, genre = map(str.strip, line.split(','))
    except ValueError:
        print("Invalid input format. Please enter as 'title, genre'")
        #print("arvijayakumar")
        continue
    library[genre].append(title)
for genre, titles in library.items():

    print(f"{genre}: {', '.join(titles)}")
```

Program:

```
from math import log
n = int(input())

print(int(log(n, 4)) == log2(n, 4))
```