

1. Odd String Difference You are given an array of equal-length strings words. Assume that the length of each string is n. Each string words[i] can be converted into a difference integer array difference[i] of length n - 1 where difference[i][j] = words[i][j+1] - words[i][j] where $0 \leq j \leq n - 2$. Note that the difference between two letters is the difference between their positions in the alphabet i.e. the position of 'a' is 0, 'b' is 1, and 'z' is 25. For example, for the string "acb", the difference integer array is $[2 - 0, 1 - 2] = [2, -1]$. All the strings in words have the same difference integer array, except one. You should find that string. Return the string in words that has different difference integer array.

```

odd string.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/odd string.py (3.12.2)
File Edit Format Run Options Window Help
def odd(words):
    def con(s):
        return [ord(s[i + 1]) - ord(s[i]) for i in range(len(s) - 1)]
    diff_count = {}
    for word in words:
        diff = tuple(con(word))
        diff_count[diff] = diff_count.get(diff, 0) + 1
    for key, value in diff_count.items():
        if value == 1:
            odd_diff = key
            break
    for i, word in enumerate(words):
        if tuple(con(word)) != odd_diff:
            return word
words = ["adc", "wzy", "abc"]
print(odd(words))

```

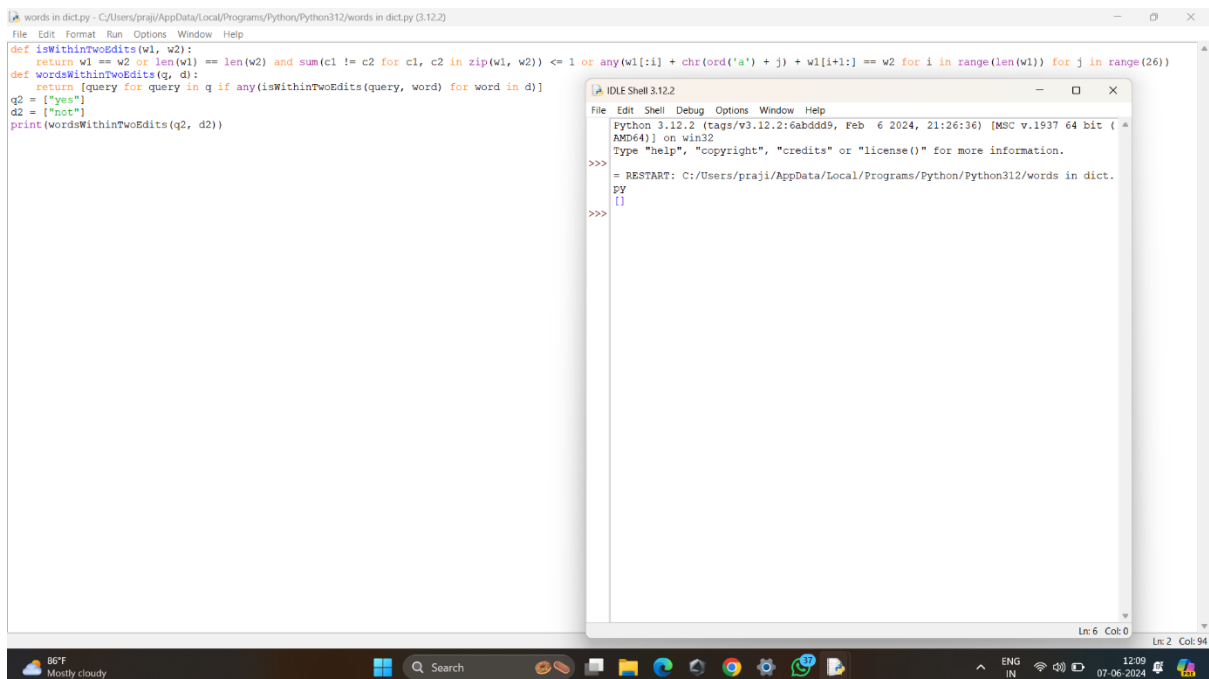
```

IDLE Shell 3.12.2
File Edit Shell Debug Options Window Help
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/odd string.py
abc
>>>

```

Time: $O(n)$

2. Words Within Two Edits of Dictionary You are given two string arrays, queries and dictionary. All words in each array comprise of lowercase English letters and have the same length. In one edit you can take a word from queries, and change any letter in it to any other letter. Find all words from queries that, after a maximum of two edits, equal some word from dictionary. Return a list of all words from queries, that match with some word from dictionary after a maximum of two edits. Return the words in the same order they appear in queries.



The screenshot shows a Python IDE with a file named 'words in dict.py'. The code defines a function 'isWithinTwoEdits' that checks if two words differ by at most two characters. It then uses this function to filter a list of queries based on a dictionary. The output is printed to the console.

```
words in dict.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/words in dict.py (3.12.2)
File Edit Format Run Options Window Help
def isWithinTwoEdits(w1, w2):
    return w1 == w2 or len(w1) == len(w2) and sum(c1 != c2 for c1, c2 in zip(w1, w2)) <= 1 or any(w1[:i] + chr(ord('a') + j) + w1[i+1:] == w2 for i in range(len(w1)) for j in range(26))
def wordsWithinTwoEdits(q, d):
    return [query for query in q if any(isWithinTwoEdits(query, word) for word in d)]
q2 = ["yes"]
d2 = ["not"]
print(wordsWithinTwoEdits(q2, d2))

IDLE Shell 3.12.2
File Edit Shell Debug Options Window Help
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb 6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/words in dict.py
>>>
```

Time: $O(m+n)$

3. Next Greater Element IV You are given a 0-indexed array of non-negative integers `nums`. For each integer in `nums`, you must find its respective second greater integer. The second greater integer of `nums[i]` is `nums[j]` such that: $j > i$, $nums[j] > nums[i]$. There exists exactly one index k such that $nums[k] > nums[i]$ and $i < k < j$. If there is no such `nums[j]`, the second greater integer is considered to be `-1`. For example, in the array `[1, 2, 4, 3]`, the second greater integer of 1 is 4, 2 is 3, and that of 3 and 4 is `-1`. Return an integer array `answer`, where `answer[i]` is the second greater integer of `nums[i]`.

```

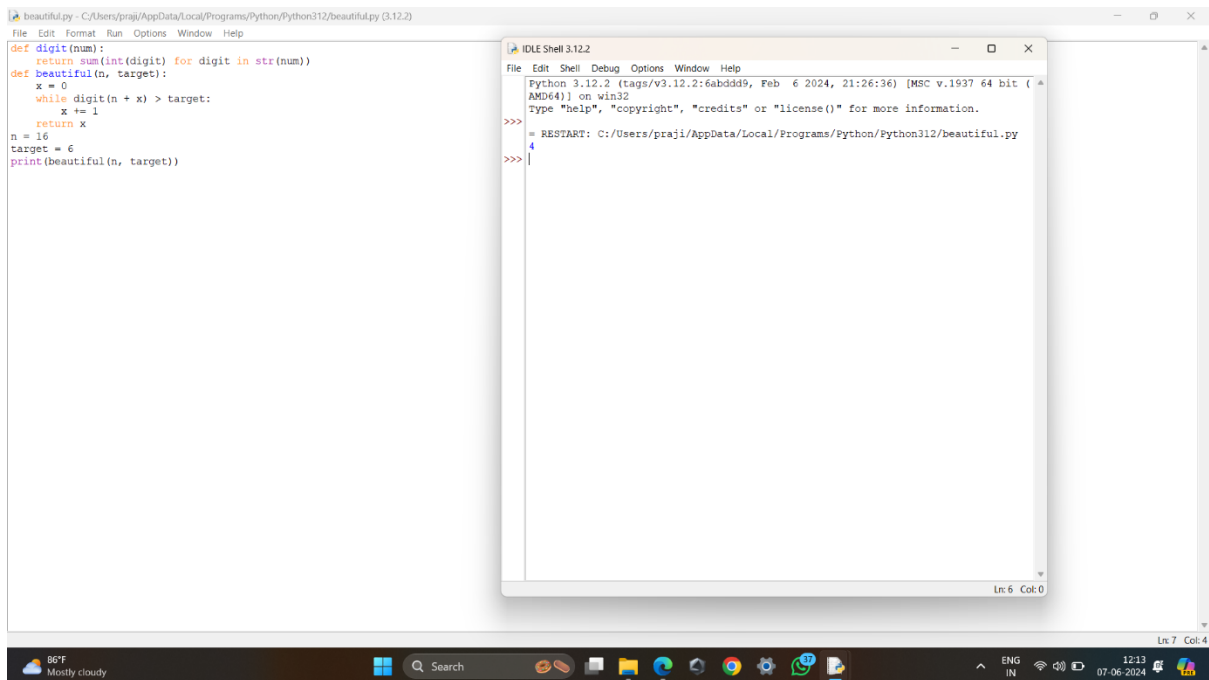
greater than number.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/greater than number.py (3.12.2)
File Edit Format Run Options Window Help
def printNGE(arr):
    arr.sort()
    for i in range(0, len(arr), 1):
        next = -1
        for j in range(i+1, len(arr), 1):
            if arr[i] < arr[j]:
                next = arr[j]
                break
        print(str(arr[i]) + " -- " + str(next))
arr = [2,4,0,9,6]
printNGE(arr)

IDLE Shell 3.12.2
Python 3.12.2 (tags/v3.12.2:6abdd49, Feb 6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/greater than number.py
0 -- -1
2 -- 4
4 -- 6
6 -- 9
9 -- -1
>>>

```

Time: $O(n^2)$

4. Minimum Addition to Make Integer Beautiful You are given two positive integers n and $target$. An integer is considered beautiful if the sum of its digits is less than or equal to $target$. Return the minimum non-negative integer x such that $n + x$ is beautiful. The input will be generated such that it is always possible to make n beautiful.



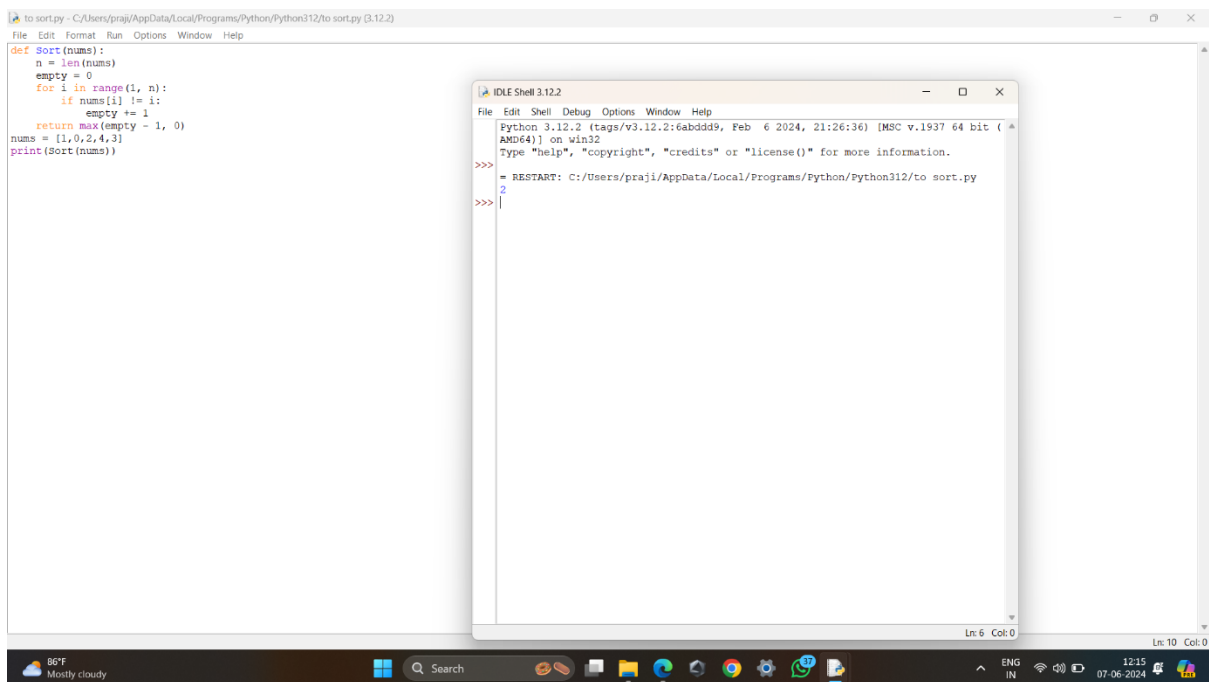
The screenshot shows a Python IDE with a script titled 'beautiful.py'. The script defines two functions: 'digit' which calculates the sum of digits of a number, and 'beautiful' which finds the minimum non-negative integer 'x' such that the sum of digits of 'n + x' is less than or equal to 'target'. The script sets 'n = 16' and 'target = 6', and prints the result of 'beautiful(n, target)'. An 'IDLE Shell 3.12.2' window is open, showing the Python interpreter's output: 'RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/beautiful.py' followed by a blank line, indicating the result of the print statement.

```
def digit(num):  
    return sum(int(digit) for digit in str(num))  
def beautiful(n, target):  
    x = 0  
    while digit(n + x) > target:  
        x += 1  
    return x  
n = 16  
target = 6  
print(beautiful(n, target))
```

```
>>>  
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/beautiful.py  
>>>
```

Time: $O(n)$

5. Sort Array by Moving Items to Empty Space You are given an integer array `nums` of size `n` containing each element from 0 to `n - 1` (inclusive). Each of the elements from 1 to `n - 1` represents an item, and the element 0 represents an empty space. In one operation, you can move any item to the empty space. `nums` is considered to be sorted if the numbers of all the items are in ascending order and the empty space is either at the beginning or at the end of the array. For example, if `n = 4`, `nums` is sorted if: • `nums = [0,1,2,3]` or • `nums = [1,2,3,0]` ...and considered to be unsorted otherwise. Return the minimum number of operations needed to sort `nums`.



The screenshot shows a Python IDE with two windows. The main window displays a Python script for sorting an array by moving items to an empty space. The script defines a function `Sort(nums)` that iterates through the array, finds the empty space (0), and moves elements to it. The initial array is `[1,0,2,4,3]`. A smaller window titled 'IDLE Shell 3.12.2' shows the execution of the script, displaying the output `2`, which represents the minimum number of operations needed to sort the array.

```
to sort.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/to sort.py (3.12.2)
File Edit Format Run Options Window Help
def Sort(nums):
    n = len(nums)
    empty = 0
    for i in range(1, n):
        if nums[i] != i:
            empty += 1
    return max(empty - 1, 0)
nums = [1,0,2,4,3]
print(Sort(nums))

IDLE Shell 3.12.2
File Edit Shell Debug Options Window Help
Python 3.12.2 (tags/v3.12.2:6abdd9, Feb 6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/to sort.py
2
>>>
```

Time: $O(N)$