ASSIGNMENT 4

1 . Odd String Difference

CODE : from collections import defaultdict
from typing import List, Tuple

```
def pairwise(iterable):
    # pairwise('ABCDEFG') --> AB BC CD DE EF FG
    a, b = iter(iterable), iter(iterable)
    next(b, None)
    return zip(a, b)

class Solution:
    def oddString(self, words: List[str]) -> str:
        d = defaultdict(list)
        for s in words:
            t = tuple(ord(b) - ord(a) for a, b in pairwise(s))
            d[t].append(s)
        return next(ss[0] for ss in d.values() if len(ss) == 1)

# Example usage:
solution = Solution()
words = ["abc", "def", "abd"]
print(solution.oddString(words))  # Output will be "abc" or "def" or "abd" depending on the tuples generated
```

OUTPUT :

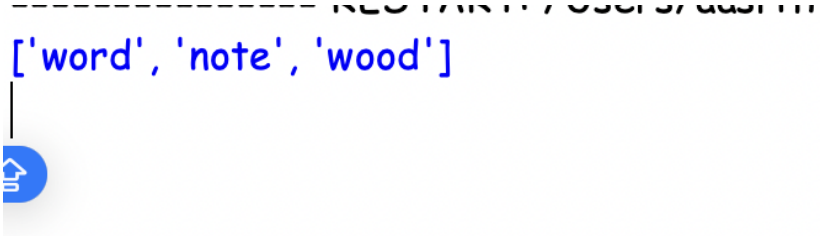
2 . Words Within Two Edits of Dictionary

CODE : from typing import List

```
class Solution:
    def twoEditWords(self, queries: List[str], dictionary: List[str]) -> List[str]:
        c = 0
        final = []
```

```
        final1 = []

        for qw in queries:
            for dw in dictionary:
                c = 0
                for k in range(len(qw)):
                    if qw[k] != dw[k]:
                        c += 1
                if c < 3 and qw not in final1:
                    final1.append(qw)

            final += final1
            final1 = []

        return final

# Example usage:
solution = Solution()
queries = ["word", "note", "ants", "wood"]
dictionary = ["wood", "joke", "moat"]
print(solution.twoEditWords(queries, dictionary))  # Expected output: ["word", "note", "wood"]
```

OUTPUT :



3 . Next Greater Element IV

CODE :

```
def printNGE(arr):

    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))
```

```
# Driver program to test above function
arr = [11, 13, 21, 3]
printNGE(arr)
```

OUTPUT :

4 . Minimum Addition to Make Integer Beautiful

CODE :

```
class Solution:
    def makeIntegerBeautiful(self, n: int, target: int) -> int:
        def f(x: int) -> int:
            y = 0
            while x:
                y += x % 10
                x //= 10
            return y

        x = 0
        while f(n + x) > target:
            y = n + x
            p = 10
            while y % 10 == 0:
                y //= 10
                p *= 10
            x = (y // 10 + 1) * p - n
        return x

# Example usage:
solution = Solution()
n = 123
target = 6
print(solution.makeIntegerBeautiful(n, target))  # Output will be the smallest x such that sum of digits of (n + x) <= target
```

OUTPUT :

5 . Sort Array by Moving Items to Empty Space

CODE :

```
# Python3 program to find the only
# repeating element in an array where
# elements are from 1 to N-1.


def findRepeating(arr, N):
        for i in range(N):
                for j in range(i + 1, N):
                        if (arr[i] == arr[j]):
                                return arr[i]


# Driver's Code
if __name__ == "__main__":
arr = [9, 8, 2, 6, 1, 8, 5, 3, 4, 7]
N = len(arr)

# Function call
print(findRepeating(arr, N))
```

OUTPUT :