**Assignment Set 1 – EEE A**

**Roll No. : CB.EN.U4EEE18014**

|  |  |  |  |
| --- | --- | --- | --- |
| **Problem** | **Number of Test Cases passed** | **Number of Test Cases failed** | **Total Number of Test Cases** |
| Strong Password | 90 | 0 | 90 |
| Super Reduced String | 16 | 0 | 16 |
| Caesar Cipher | 12 | 0 | 12 |
| Making Anagrams | 16 | 0 | 16 |
| Beautiful Binary String | 12 | 0 | 12 |
| Remove Duplicates from Sorted Array | 161 | 0 | 161 |
| Two Sum | 53 | 0 | 53 |
| Move Zeroes | 21 | 0 | 21 |
| Max Consecutive Ones | 42 | 0 | 42 |
| Squares of a Sorted Array | 137 | 0 | 137 |
| Create Target Array in the given order | 44 | 0 | 44 |
| Find Numbers with Even Number of Digits | 104 | 0 | 104 |

**Problem No 1 => Strong Password**

#!/bin/python3

import math

import os

import random

import re

import sys

# Complete the minimumNumber function below.

def minimumNumber(n, password):

    numbers="0123456789"

    lower\_case="abcdefghijklmnopqrstuvwxyz"

    upper\_case="ABCDEFGHIJKLMNOPQRSTUVWXYZ"

    special\_characters="!@#$%^&\*()-+"

    num=l=u=s=c=t=lens=0

    if n>=6:

        for i in password:

            if(i in numbers):

                num+=1

            if(i in lower\_case):

                l+=1

            if(i in upper\_case):

                u+=1

            if(i in special\_characters):

                s+=1

    elif(n<6):

        lens=6-n

        for i in password:

            if(i in numbers):

                num+=1

            if(i in lower\_case):

                l+=1

            if(i in upper\_case):

                u+=1

            if(i in special\_characters):

                s+=1

    if(num==0):

        c+=1

    if(l==0):

        c+=1

    if(u==0):

        c+=1

    if(s==0):

        c+=1

    if(lens>=c):

        c=lens

    return c

if \_\_name\_\_ == '\_\_main\_\_':

    fptr = open(os.environ['OUTPUT\_PATH'], 'w')

    n = int(input())

    password = input()

    answer = minimumNumber(n, password)

    fptr.write(str(answer) + '\n')

    fptr.close()

**Problem Number 2 => Super Reduced String**

#!/bin/python3

import math

import os

import random

import re

import sys

# Complete the superReducedString function below.

def superReducedString(s):

    a = []

    for i in s:

        if(not a):

            a.append(i)

        else:

            if(a[-1]==i):

                a.pop()

            else:

                a.append(i)

    if(not a):

        return "Empty String"

    else:

        return ''.join(a)

if \_\_name\_\_ == '\_\_main\_\_':

    fptr = open(os.environ['OUTPUT\_PATH'], 'w')

    s = input()

    result = superReducedString(s)

    fptr.write(result + '\n')

    fptr.close()

**Problem Number 3 => Caesar Cipher**

#!/bin/python3

import math

import os

import random

import re

import sys

# Complete the caesarCipher function below.

def caesarCipher(s, k):

    b=""

    if(k>26):

            while(k>=26):

                k=k-26

    if(k<=26):

        for i in s:

            m=ord(i)

            if(m>96 and m<123):

                e=m+k

                if(e>=123):

                    e=e-26

                    b+=chr(e)

                else:

                    b+=chr(e)

            elif(m>64 and m<91):

                e=m+k

                if(e>90):

                    e=e-26

                    b+=chr(e)

                else:

                    b+=chr(e)

            else:

                e=m

                b+=chr(e)

    return b

if \_\_name\_\_ == '\_\_main\_\_':

    fptr = open(os.environ['OUTPUT\_PATH'], 'w')

    n = int(input())

    s = input()

    k = int(input())

    result = caesarCipher(s, k)

    fptr.write(result + '\n')

    fptr.close()

**Problem Number 4 => Making Anagrams**

#!/bin/python3

import math

import os

import random

import re

import sys

# Complete the makingAnagrams function below.

def makingAnagrams(s1, s2):

    a=[i for i in s1]

    b=[i for i in s2]

    a.sort()

    b.sort()

    c=0

    l=[]

    for i in a:

        if i in a and i in b and i not in l:

            c+=abs(a.count(i)-b.count(i))

        elif i in a and i not in b and i not in l:

            c+=a.count(i)

        l.append(i)

    for j in b:

        if j not in a and j in b and j not in l:

            c+=b.count(j)

        l.append(j)

    return c

if \_\_name\_\_ == '\_\_main\_\_':

    fptr = open(os.environ['OUTPUT\_PATH'], 'w')

    s1 = input()

    s2 = input()

    result = makingAnagrams(s1, s2)

    fptr.write(str(result) + '\n')

    fptr.close()

**Problem Number 5 => Beautiful Binary String**

#!/bin/python3

import math

import os

import random

import re

import sys

# Complete the beautifulBinaryString function below.

def beautifulBinaryString(b):

    x=b.count("010")

    return x

if \_\_name\_\_ == '\_\_main\_\_':

    fptr = open(os.environ['OUTPUT\_PATH'], 'w')

    n = int(input())

    b = input()

    result = beautifulBinaryString(b)

    fptr.write(str(result) + '\n')

    fptr.close()

**Problem Number 6 => Remove Duplicates from Sorted Array**

class Solution:

def removeDuplicates(self, nums: List[int]) -> int:

nums.sort()

for i in range(len(nums)-1,0,-1):

if(nums[i]==nums[i-1]):

nums.pop(i)

**Problem Number 7 => Two Sum**

class Solution:

def twoSum(self, nums: List[int], target: int) -> List[int]:

n=len(nums)

for i in range(0,n):

for j in range(0,n):

if(i!=j):

sum=nums[i]+nums[j]

if (target==sum):

return i,j

**Problem Number 8 => Move Zeroes**

class Solution:

def moveZeroes(self, nums: List[int]) -> None:

l=len(nums)

for i in range(l-1,-1,-1):

if nums[i]==0:

nums.pop(i)

nums.append(0)

**Problem Number 9 => Max Consecutive Ones**

class Solution:

def findMaxConsecutiveOnes(self, nums: List[int]) -> int:

c=0

a=[]

for i in nums:

if(i!=0):

c+=1

else:

c=0

a.append(c)

return max(a)

**Problem Number 10 => Squares of a Sorted Array**

class Solution:

def sortedSquares(self, nums: List[int]) -> List[int]:

a=[]

for i in nums:

p = i \* i

a.append(p)

a.sort()

return a

**Problem Number 11 => Create Target Array in the given order**

class Solution:

def createTargetArray(self, nums: List[int], index: List[int]) -> List[int]:

target=[]

for i in range(len(nums)):

target.insert(index[i],nums[i])

return target

**Problem Number 12 =>** Find Numbers with Even Number of Digits

class Solution:

def findNumbers(self, nums: List[int]) -> int:

cn=0

for i in nums:

c=0

while(i!=0):

i=int(i/10)

c+=1

if((c%2)==0):

cn+=1

return cn