Government Engineering College, Aurangabad

FYMCA

Roll no: MC22F14F060

Q. 1 Write a Python function to check whether a number is perfect or not from 1 to 10000

```
def perfect_n0(number):
    div_sum = sum(i for i in range(1, number) if number % i == 0)
    return divisors_sum == number
# Checking numbers from 1 to 10,000
for num in range(1, 10001):
    if perfect_no(num):
        print(f"{num} is a perfect number")
```

```
============ RESTART: C:/Users/USER/OneDrive/Desktop/Assignment 2.py ===========
6 is a perfect number
28 is a perfect number
496 is a perfect number
8128 is a perfect number
>>>
```

Q.2 Write a recursive python program to calculate the sum of a list of numbers

```
def calculate_sum(numbers):
    if len(numbers) == 0:
        return 0
    else:
        return numbers[0] + calculate_sum(numbers[1:])

my_list = [1, 2, 3, 4, 5]
result = calculate_sum(my_list)
print(f"The sum of {my_list} is: {result}")
```

```
============ RESTART: C:/Users/USER/OneDrive/Desktop/Assignment 2.py ======
The sum of [1, 2, 3, 4, 5] is: 15
>>>
```

Q.3 Write a recursive python program to calculate the sum of the positive integers of n+(n-2)+(n-4)... (until n-x=0)

```
def calculate_sum(n):
    if n <= 0:
        return 0
    else:
        return n + calculate_sum(n – 2)
num = int(input(“Enter a two digit number: “))
result = calculate_sum(num)
print(f”The sum of the series for n={num} is: {result}”)
```

```
============ RESTART: C:/Users/USER/OneDrive/Desktop/Assignment 2.py ======
Enter a two digit number: 55
The sum of the series for n=55 is: 784
>>>
```

Q. 4 Python program using function to check whether the string is symmetrical or palindrom or otherwise.

```
def check_string(string):
    # Remove whitespace and convert to lowercase
    string = string.replace(" ", "").lower()
    if string == string[::-1]:
        return "Palindrome"
    elif string == string[::-1]:
        return "Symmetrical"
    else:
        return "Neither"
my_string = input("Enter a word: ")
result = check_string(my_string)
print(f"The string '{my_string}' is {result}")
```

```
============ RESTART: C:/Users/USER/OneDrive/Desktop/Assignment 2.py ==
Enter a word: Mrunal
The string 'Mrunal' is Neither
>>>
============ RESTART: C:/Users/USER/OneDrive/Desktop/Assignment 2.py ==
Enter a word: NAYAN
The string 'NAYAN' is Palindrome
>>>
```

Q.5 Write a function called singleLetterCount . This function takes in two parameters (two strings). The first parameter should be a word and second should be a letter. Use try except statement to ensure that the parameters are correctly passed. The function should be case sensitive. If the letter is not found in the word, the function should return 0.

```
def singleLetterCount(word, letter):
    try:
        count = word.count(letter)
        return count
    except AttributeError:
        return 0


word = input("Enter a word: ")
letter =input("Enter a letter: ")
result = singleLetterCount(word, letter)
print(f"The letter '{letter}' appears in the word '{word}' {result} time(s)")
```

```
============ RESTART: C:/Users/USER/OneDrive/Desktop/Assignment 2.py ====
Enter a word: Applications
Enter a letter: p
The letter 'p' appears in the word 'Applications' 2 time(s)
>>>
```

Q.6 Write a function which accepts a sequence of comma separated 4 digit binary numbers as its input and then check whether they are divisible by 5 or not. The numbers that are divisible by 5 are to be printed in a comma separated sequence. (n= int('0100',2)) . User assertion to ensure that every number is of 4 digits only.

```
def check_divisible_by_five(numbers):
    numbers_list = numbers.split(',')
    divisible_by_five = []


    for num in numbers_list:
        assert len(num) == 4, "Each number should be 4 digits long."


        decimal_num = int(num, 2)
        print("Given number : ", decimal_num,"that is :",num)
        if decimal_num % 5 == 0:
            divisible_by_five.append(num)
```

```
    result = ",".join(divisible_by_five)
    return result


binary_numbers = "0100,1010,0011,1111,0110"
result = check_divisible_by_five(binary_numbers)
print(f"The numbers divisible by 5 are: {result}")
```

```
============ RESTART: C:/Users/USER/OneDri
Given number :  4 that is : 0100
Given number :  10 that is : 1010
Given number :  3 that is : 0011
Given number :  15 that is : 1111
Given number :  6 that is : 0110
The numbers divisible by 5 are: 1010,1111
>>>
```

Q.7 Write a python class BankAccount with attributes like account number, balance, date of opening and customer name and methods like deposit, withdraw and check balance.

```
class BankAccount:
    def __init__(self, account_number, balance, date_of_opening, customer_name):
        self.account_number = account_number
        self.balance = balance
        self.date_of_opening = date_of_opening
        self.customer_name = customer_name

    def deposit(self, amount):
        if amount > 0:
            self.balance += amount
            print(f"Deposit of {amount} successful.")
        else:
            print("Invalid deposit amount.")

    def withdraw(self, amount):
        if amount > 0:
            if self.balance >= amount:
```

```
            self.balance -= amount
            print(f"Withdrawal of {amount} successful.")
        else:
            print("Insufficient balance.")
    else:
        print("Invalid withdrawal amount.")

def check_balance(self):
    print(f"Account Balance: {self.balance}")



account = BankAccount("1234567890", 5000, "2023-06-15", "John Doe")


account.check_balance()
account.deposit(2000)
account.check_balance()
account.withdraw(3000)
account.check_balance()
```

```
============ RESTART: C:/Users/USER/OneDrive/Desktop/Assignment 2.py =
Account Balance: 5000
Deposit of 2000 successful.
Account Balance: 7000
Withdrawal of 3000 successful.
Account Balance: 4000
>>>
```

Q.8 Write a program to find the most repeated word in a text file

```
count = 0;
word = "";
maxCount = 0;
words = [];
#Opens a file in read mode
file = open("data.txt", "r")
#Gets each line till end of file is reached
for line in file:
    #Splits each line into words
```

```
    string = line.lower().replace(',',"").replace('.',"").split(" ");
    #Adding all words generated in previous step into words
    for s in string:
        words.append(s);
#Determine the most repeated word in a file
for i in range(0, len(words)):
    count = 1;
    #Count each word in the file and store it in variable count
    for j in range(i+1, len(words)):
        if(words[i] == words[j]):
            count = count + 1;
    #If maxCount is less than count then store value of count in maxCount
    #and corresponding word to variable word
    if(count > maxCount):
        maxCount = count;
        word = words[i];


print("Most repeated word: " + word);
file.close();
```

```
============ RESTART: C:/Users/USER/OneDrive/Desktop/Assignment 2.py ===
Most repeated word: computer
>>>
```

Q.9 Write a python program to overload unary minus operator which takes a list and returns a new list with unique elements of the first list

```
class UniqueList:
    def __init__(self, lst):
        self.lst = lst


    def __neg__(self):
        unique_elements = list(set(self.lst))
        return UniqueList(unique_elements)



original_list = [11,1, 1,2, 3, 2, 4, 3, 5, 1]
unique_list = UniqueList(original_list)
```

```
result = -unique_list
print("Original list:", original_list)
print("Unique list:", result.lst)
```

```
''' 
============ RESTART: C:/Users/USER/OneDrive/Desktop/Assignment 2.py ===
Original list: [11, 1, 1, 2, 3, 2, 4, 3, 5, 1]
Unique list: [1, 2, 3, 4, 5, 11]
>>> 
```

Q.10 Write a python class to find the three elements that sum to zero from a list of 10 numbers. (input list: [-25,-10,-7,-3,2,4,8,10] output list: [[-10,2,8],[-7,-3,10]])

```
class SumToZeroFinder:
    def find_three_elements(self, nums):
        result = []
        nums.sort()
        for i in range(len(nums)-2):
            left = i + 1
            right = len(nums) - 1
            while left < right:
                current_sum = nums[i] + nums[left] + nums[right]
                if current_sum == 0:
                    result.append([nums[i], nums[left], nums[right]])
                    left += 1
                    right -= 1
                elif current_sum < 0:
                    left += 1
                else:
                    right -= 1
        return result
numbers = [-25, -10, -7, -3, 2, 4, 8, 10]
finder = SumToZeroFinder()
result = finder.find_three_elements(numbers)
print("Output list:", result)
```

```
============ RESTART: C:/Users/USER/OneDrive/Desktop/Assignment 2.py =======
Output list: [[-10, 2, 8], [-7, -3, 10]]
>>> 
```