Lab-1 a) Write a python program to find the best of two test average marks out of three test's marks accepted from the user.

```
first_test_m1 = int (input("Enter the marks in the first test: "))
second_test_m2 = int (input("Enter the marks in second test: "))
third_test_m3 = int (input("Enter the marks in third test: "))

if (first_test_m1 > second_test_m2):
   if (second_test_m2 > third_test_m3):
      total = first_test_m1 + second_test_m2
   else:
      total = first_test_m1 + third_test_m3
elif (first_test_m1 > third_test_m3):
   total = first_test_m1 + third_test_m2
else:
   total = second_test_m2 + third_test_m3

Avg = total / 2
print ("The average of the best two test marks is: ",Avg)
```

Lab-1 b) Develop a Python program to check whether a given number is palindrome or not and also count the number of occurrences of each digit in the input number.

```
n = int(input("Enter number:"))
num = str(n)
temp = n
rev=0
while(n>0):
   dig=n%10
   rev=rev*10+dig
   n=n//10

if(temp==rev):                # num == num[::-1]   match the string with revers string
   print("The number is a palindrome!")
else:
   print("The number isn't a palindrome!")

# number of occurrence of each digit
for i in range(10):
   if num.count(str(i)) > 0:
      print(str(i),"appears",num.count(str(i)),"times")
```

Lab – 2 a). Defined as a function F as Fn = Fn-1 + Fn-2. Write a Python program which accepts a value for N (where N >0) as input and pass this value to the function. Display suitable error message if the condition for input value is not followed.

```
def F(N):
    if N <= 0:
        print("Error:  Number of terms must be a positive  integer.")
        return
    elif N == 1:
        return [0]
    elif N == 2:
        return [0, 1]
    else:
        f_n = [0, 1]  # fn = [fn-1,fn-2]

        while len(f_n) < N:
            f_n_next = f_n[-1] + f_n[-2]  #Next term fn = fn-1+fn-2
            f_n.append(f_n_next)  #Creating a list of terms
        return f_n


# Accept input from the user
n = int(input("Enter  a positive  integer (N > 0): "))

# Call the Fibonacci function and display the result or error message
result = F(n)
if result is not None:
    print(f"The  numbers till  N are: {result}")
```

Lab- 2 b). Develop a python program to convert binary to decimal, octal to hexadecimal using functions.

```
# binary to decimal
def bin2Dec(val):
    rev=val[::-1]
    dec = 0
    i = 0
    for dig in rev:
        dec += int(dig)  * 2**i
        i += 1
    return dec

# octal to hexadecimal
def oct2Hex(val):
    rev=val[::-1]
    dec = 0
    i = 0
    for dig in rev:
        dec += int(dig)  * 8**i
        i += 1
    list=[]
    while dec != 0:
        list.append(dec%16)
        dec = dec // 16
    nl=[]
    for elem in list[::-1]:
        if elem<= 9:
            nl.append(str(elem))
        else:
            nl.append(chr(ord('A') + (elem -10)))
        hex = "".join(nl)
    return hex

num1 = input("Enter a binary number : ")
print(bin2Dec(num1))
num2 = input("Enter a octal number : ")
print(oct2Hex(num2))
```

<u>Lab-3 a).</u> Write a Python program that accepts a sentence and find the number of words, digits, uppercase letters and lowercase letters.

```
s = input("Enter a sentence: ")
w, d, u, l = 0, 0, 0, 0
l_w = s.split()
w = len(l_w)
for c in s:
    if c.isdigit():
        d = d + 1
    elif c.isupper():
        u = u + 1
    elif c.islower():
        l = l + 1

print ("No of Words: ", w)
print ("No of Digits: ", d)
print ("No of Uppercase letters: ", u)
print ("No of Lowercase letters: ", l)
```

Lab-3 b) Write a Python program to find the string similarity between two given strings

Sample Output: **Sample Output:**

```
Original String:                  Original String:
Python Exercises                  Python Exercises
Python Exercises                  Python Exercise
Similarity between two said strings:  Similarity between two said strings:
1.0                               0.967741935483871
```

```
import difflib
def string_similarity(str1, str2):
    result = difflib.SequenceMatcher(a=str1.lower(), b=str2.lower())
    return result.ratio()
str1 = 'Python Exercises'
str2 = 'Python Exercises'
print("Original string:")
print(str1)
print(str2)
print("Similarity between two said strings:")
print(string_similarity(str1,str2))
str1 = 'Python Exercises'
str2 = 'Python Exercise'
print("\nOriginal string:")
print(str1)
print(str2)
print("Similarity between two said strings:")
print(string_similarity(str1,str2))
```

# Python3 code to demonstrate
# similarity between strings
# using naive method (sum() + zip())

```
# Utility function to compute similarity
def similar(str1, str2):
        str1 = str1 + ' ' * (len(str2) - len(str1))
        str2 = str2 + ' ' * (len(str1) - len(str2))
        return sum(1 if i == j else 0
                        for i, j in zip(str1, str2)) / float(len(str1))

# Initializing strings
test_string1 = 'Python Exerises'
test_string2 = ' Python Exerises''

# using naive method (sum() + zip())
# similarity between strings
res = similar(test_string1, test_string2)

# printing the result
print ("The similarity between 2 strings is : " + str(res))
```

Lab-4 a). Write a python program to implement insertion sort and merge sort using lists

```
import random

def merge_sort(lst):
  if len(lst) > 1:
    mid = len(lst) // 2
    left_half = lst[:mid]
    right_half = lst[mid:]

    merge_sort(left_half)
    merge_sort(right_half)

    i = j = k = 0

    while i < len(left_half) and j < len(right_half):
            if left_half[i] < right_half[j]:
                lst[k] = left_half[i]
                i += 1
            else:
                lst[k] = right_half[j]
                j += 1
            k += 1

        while i < len(left_half):
            lst[k] = left_half[i]
            i += 1
            k += 1

        while j < len(right_half):
            lst[k] = right_half[j]
            j += 1
            k += 1

    return lst


def insertion_sort(arr):
    for i in range(1, len(arr)):
        key = arr[i]
        j = i - 1
        while j >= 0 and key < arr[j]:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key

my_list = []

for i in range(10):
    my_list.append(random.randint(0, 999))

print("\nUnsorted List")
print(my_list)
print("Sorting using Insertion Sort")
insertion_sort(my_list)
print(my_list)


my_list = []

for i in range(10):
    my_list.append(random.randint(0, 999))

print("\nUnsorted List")
print(my_list)
print("Sorting  using  Merge Sort")
merge_sort(my_list)
print(my_list)
```

Lab 4 b) Write a program to convert roman numbers in to integer values using dictionaries.

```
def roman2Dec(romStr):
    roman_dict ={'I': 1, 'V': 5, 'X': 10, 'L': 50, 'C': 100, 'D': 500, 'M': 1000}
    # Analyze string backwards
    romanBack = list(romStr)[::-1]
    value = 0
    # To keep track of order
    rightVal = roman_dict[romanBack[0]]
    for numeral in romanBack:
        leftVal = roman_dict[numeral]
        # Check for subtraction
        if leftVal < rightVal:
            value -= leftVal
        else:
            value += leftVal
        rightVal = leftVal
    return value


romanStr = input("Enter  a Roman Number : ")
print("Decimal number:", roman2Dec(romanStr))
```

Lab 5 a). Write a function called isphonenumber () to recognize a pattern 415-555-4242 without using regular expression and also write the code to recognize the same pattern using regular expression.

```
import re

def isphonenumber(numStr):
    if len(numStr)  != 12:
        return False
    for i in range(len(numStr)):
        if i==3 or i==7:
            if numStr[i]  != "-":
                return False
        else:
            if numStr[i].isdigit()  == False:
                return False
    return True


def reisphonenumber(numStr):
    phno_pattern  = re.compile(r'^\d{3}-\d{3}-\d{4}$')
    if phno_pattern.match(numStr):
        return True
    else:
        return False

ph_num = input("Enter a phone number : ")

print("Without using Regular Expression")
if isphonenumber(ph_num):
    print("given phone number is Valid phone number")
else:
    print("given phone number is Invalid phone number")

print("Using Regular Expression")
if reisphonenumber(ph_num):
    print("given phone number is Valid phone number")
else:
  print("given phone number is Invalid phone number")
```

b). Develop a python program that could search the text in a file for phone numbers (+919900889977) and email addresses (sample@gmail.com)

```
import re

# Define the regular expression for phone numbers
phone_regex = re.compile(r'\+\d{12}')
email_regex = re.compile(r'[A-Za-z0-9._]+@[A-Za-z0-9]+\.[A-Z|a-z]{2,}')

# Open the file for reading
with open('example.txt', 'r') as f:
    # Loop through each line in the file
    for line in f:
        # Search for phone numbers in the line
        matches = phone_regex.findall(line)
        # Print any matches found
        for match in matches:
            print(match)
        matches = email_regex.findall(line)
        # Print any matches found
        for match in matches:
            print(match)
```

Lab 6-a). Write a python program to accept a file name from the user and perform the following operations

1. Display the first N line of the file

2. Find the frequency of occurrence of the word accepted from the user in the file

```
import os.path
import sys
fname = input("Enter the filename : ")
```

```
if not os.path.isfile(fname):
    print("File", fname, "doesn't exists")
    sys.exit(0)

infile = open(fname, "r")
lineList = infile.readlines()

for i in range(len(lineList)):
    print(i+1, ":", lineList[i],end=" ")
word = input("\n Enter a word : ")
cnt = 0
for line in lineList:
    cnt += line.count(word)
print("The word", word, "appears", cnt, "times in the file")
```

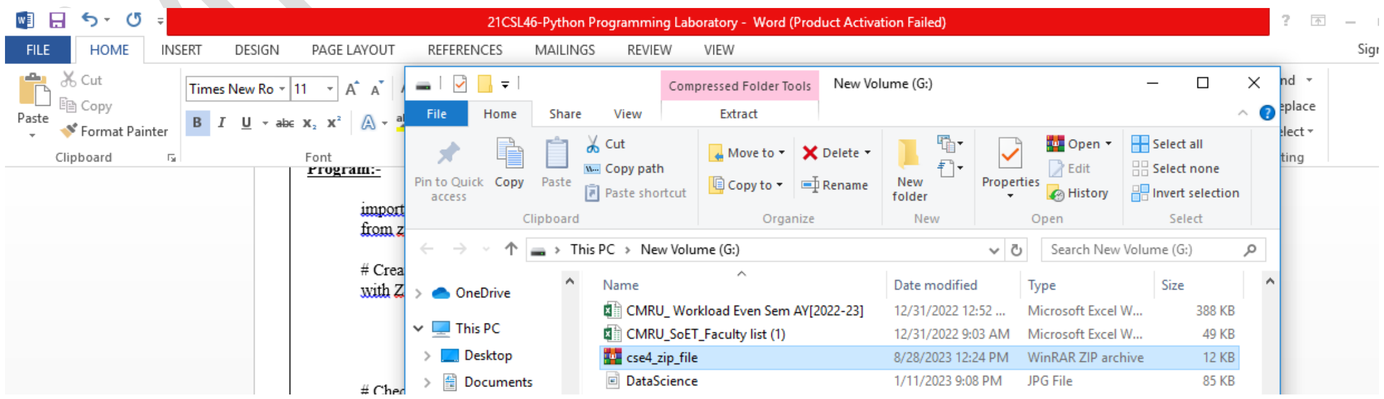
Lab 6 b). Write a python program to create a zip of a particular folder which contains several files inside it.

Program:-

```
import os
from zipfile import ZipFile

# Create a ZipFile Object
with ZipFile('G:\\cse4_zip_file.zip', 'w') as zip_object:
        # Adding files that need to be zipped
        zip_object.write('G:\\cse4_zip_file\\file1.txt')
        zip_object.write('G:\\cse4_zip_file\\wordfile.docx')

# Check to see if the zip file is created
if os.path.exists('G:\\cse4_zip_file.zip'):
        print("ZIP file created")
else:
        print("ZIP file not created")
```

Output:-

```
Zip file created
```



21CSL46-Python Programming Laboratory - Word (Product Activation Failed)

New Volume (G:)

This PC > New Volume (G:)

```
Name                                  Date modified        Type                  Size
CMRU_ Workload Even Sem AY[2022-23]   12/31/2022 12:52 ... Microsoft Excel W...  388 KB
CMRU_SoET_Faculty list (1)            12/31/2022 9:03 AM   Microsoft Excel W...   49 KB
cse4_zip_file                         8/28/2023 12:24 PM   WinRAR ZIP archive     12 KB
DataScience                           1/11/2023 9:08 PM    JPG File               85 KB
```

Lab 7. a). By using the concept of inheritance write a python program to find the area of triangle, circle and rectangle.

```
import math

class Shape:
    def __init__(self):
        self.area = 0
        self.name = ""
    def showArea(self):
        print("the  area of the",self.name,"is",self.area,"units")

class Circle(Shape):
    def __init__(self,radius):
        self.area = 0
        self.name = "Circle"
        self.radius = radius
    def calcArea(self):
        self.area = math.pi * self.radius * self.radius

class Rectangle(Shape):
    def __init__(self,length,breadth):
        self.area = 0
        self.name = "Rectangle"
        self.length = length
        self.breadth = breadth
    def calcArea(self):
        self.area = self.length * self.breadth

class Triangle(Shape):
    def __init__(self,base,height):
        self.area = 0
        self.name = "Triangle"
        self.base = base
        self.height = height
    def calcArea(self):
        self.area = self.base * self.height / 2

c1 = Circle(6)      # created c1 object for Circle class, Shape class called automatically
c1.calcArea()     # using c1 object calling the calcArea() fucntion
c1.showArea()   # using c1 object calling the showArea() function
r1 = Rectangle(15, 14)
r1.calcArea()
r1.showArea()
t1 = Triangle(13, 14)
t1.calcArea()
t1.showArea()
```

Output:-

```
the area of the Circle  is 113.09733552923255  units
the area of the Rectangle  is 210 units
the area of the Triangle  is 91.0 units
```

Lab 7. B). b) Write a python program by creating a class called Employee to store the details of Name, Employee_ID, Department and Salary, and implement a method to update salary of employees belonging to a given department.

Program:-

```
class Employee:
    def __init__(self):
        self.name = ""
        self.empId = ""
        self.dept = ""
        self.salary = 0

    def getEmpDetails(self):
        self.name = input("Enter  Employee  name :")
        self.empId = input("Enter  Employee  ID :")
        self.dept = input("Enter  Employee  Dept :")
        self.salary = int(input("Enter  Employee  Salary :"))

    def showEmpDetails(self):
        print("Employee  Details")
        print("Name  : ", self.name)
        print("ID  : ", self.empId)
        print("Dept  : ", self.dept)
        print("Salary  : ", self.salary)

    def updtSalary(self):
        self.salary = int(input("Enter  new Salary : "))
        print("Updated  Salary", self.salary)


e1 = Employee()
e1.getEmpDetails()
e1.showEmpDetails()
e1.updtSalary()
```

Output:-

```
Enter Employee  name :Ramu
Enter Employee  ID :986
Enter Employee  Dept :cse
Enter Employee  Salary :120000
Employee  Details
Name :  Ramu
ID :  986
Dept :  cse
```

```
Salary :  120000
Enter new Salary : 13000
Updated Salary 13000
```

Lab 8. a) Write a python program to find the whether the given input is palindrome or not (for both string and integer) using the concept of polymorphism and inheritance.

Program:-

```
class PaliStr:
    def __init__(self):
        self.isPali = False
    def chkPalindrome(self, myStr):
        if myStr == myStr[::-1]:
            self.isPali = True
        else:
            self.isPali = False
        return self.isPali

class PaliInt(PaliStr):
    def __init__(self):
        self.isPali = False
    def chkPalindrome(self, val):
        temp = val
        rev = 0
        while temp != 0:
            dig = temp % 10
            rev = (rev*10) + dig
            temp = temp //10
        if val == rev:
            self.isPali = True
        else:
            self.isPali = False
        return self.isPali

st = input("Enter  a string : ")
stObj = PaliStr()
if stObj.chkPalindrome(st):
    print("Given  string is a Palindrome")
else:
    print("Given  string is not a Palindrome")

val = int(input("Enter  a integer : "))
intObj = PaliInt()
if intObj.chkPalindrome(val):
    print("Given  integer is a Palindrome")
else:
    print("Given  integer is not a Palindrome")
```

Output:-

```
Enter a string : madam
Given string is a Palindrome
Enter a integer : 121
Given integer is a Palindrome

Enter a string : Ramu
Given string is not a Palindrome
Enter a integer : 123
Given integer is not a Palindrome
```

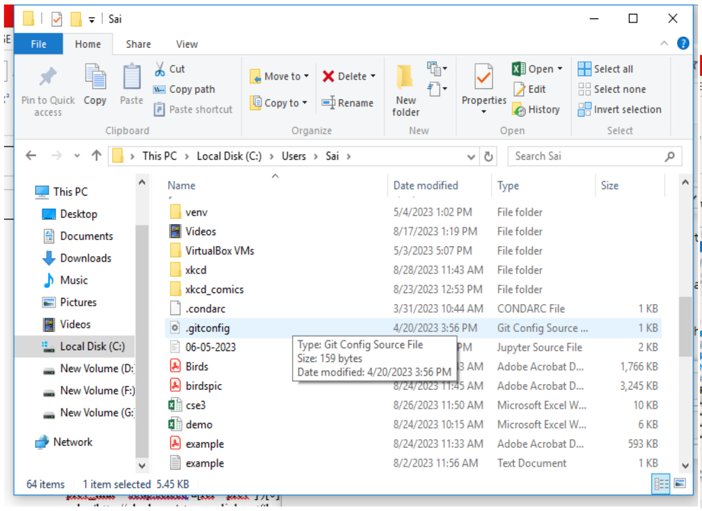
Lab 9 a). Write a python program to download the all XKCD comics

Program:-

```
import requests, os, bs4
url = 'http://xkcd.com'                # starting url
os.makedirs('xkcd',exist_ok = True)              # create a directory to store all the downloads

while not url.endswith("#"):
        # Download the page.
        print("Downloading  the page ... ")
        res = requests.get(url)
        res.raise_for_status()
        try:
                soup = bs4.BeautifulSoup(res.text,'lxml')
        except bs4.FeatureNotFound:  # lxml is not installed
                soup = bs4.BeautifulSoup(res.text,'html.parser')

        # Find the URL of the comic image.
        comic_element = soup.select('#comic img')
        if comic_element == []:
                print("No comic image found!!..")
        else:
                comic_image_url = comic_element[0].get('src')
                # download the image
                print("Downloading  the image %s .. " %(comic_image_url))
                res = requests.get('http:' + comic_image_url)
                res.raise_for_status()

                # Save the image to ./xkcd.
                file = open( os.path.join('xkcd',os.path.basename(comic_image_url))  , 'wb')
                for chunk in res.iter_content(10000):
                        file.write(chunk)
                file.close()

        # Get the Prev button's url.
        prev_link = soup.select('a[rel="prev"]')[0]
```

```
                url = 'http://xkcd.com' + prev_link.get('href')

        print("Done")
```

Output:-

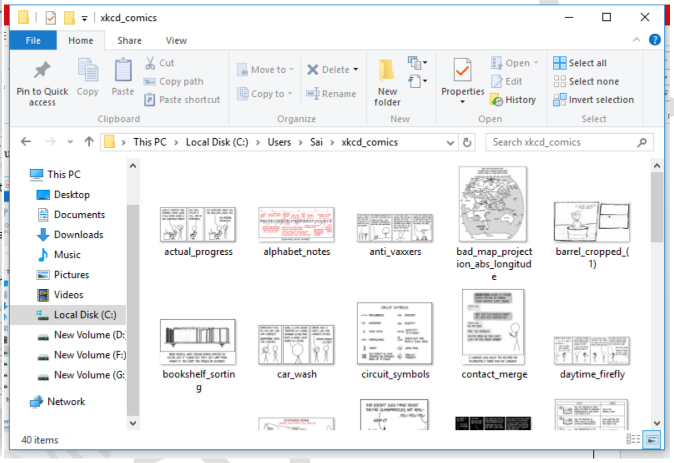


This PC > Local Disk (C:) > Users > Sai

Name	Date modified	Type	Size
venv	5/4/2023 1:02 PM	File folder	
Videos	8/17/2023 1:19 PM	File folder	
VirtualBox VMs	5/3/2023 5:07 PM	File folder	
xkcd	8/28/2023 11:43 AM	File folder	
xkcd_comics	8/23/2023 12:53 PM	File folder	
.condarc	3/31/2023 10:44 AM	CONDARC File	1 KB
.gitconfig	4/20/2023 3:56 PM	Git Config Source ...	1 KB
06-05-2023	PM	Jupyter Source File	2 KB
Birds	3 AM	Adobe Acrobat D...	1,766 KB
birdspic	8/24/2023 11:45 AM	Adobe Acrobat D...	3,245 KB
cse3	8/26/2023 11:50 AM	Microsoft Excel W...	10 KB
demo	8/24/2023 10:15 AM	Microsoft Excel W...	6 KB
example	8/24/2023 11:33 AM	Adobe Acrobat D...	593 KB
example	8/2/2023 11:56 AM	Text Document	1 KB

Type: Git Config Source File
Size: 159 bytes
Date modified: 4/20/2023 3:56 PM

64 items 1 item selected 5.45 KB



This PC > Local Disk (C:) > Users > Sai > xkcd_comics

actual_progress, alphabet_notes, anti_vaxxers, bad_map_projection_abs_longitude, barrel_cropped_(1), bookshelf_sorting, car_wash, circuit_symbols, contact_merge, daytime_firefly

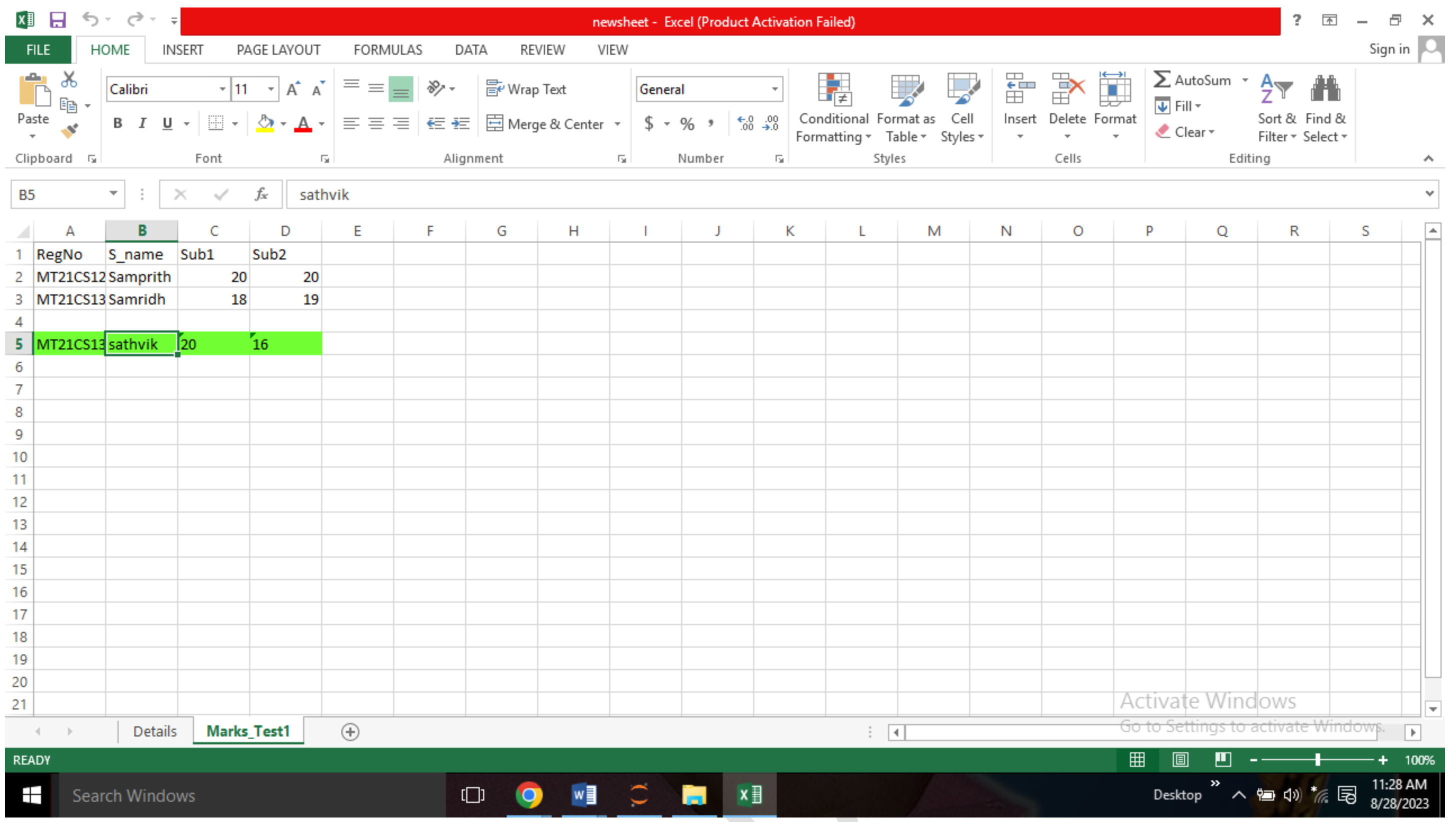
40 items

Lab 9 b). Demonstrate python program to read the data from the spreadsheet and write the data in to the spreadsheet

Program:-

```
        from openpyxl import Workbook
        from openpyxl.styles import PatternFill

        wb=openpyxl.load_workbook('student_4cse3.xlsx')        #opening the student_4cse3.xlsx file
        sh1=wb['Marks_Test1']      # assign the sheet name 'sheet name Marks_Test1' to sh1
        row =sh1.max_row
        column = sh1.max_column
        for i in range(1,row+1):
            for j in range(1,column+1):
                print(sh1.cell(i,j).value)

        sh1.cell(row=5,column=1,value='MT21CS135')
        sh1['A5'].fill=PatternFill("solid",fgColor="71FF33")
        sh1.cell(row=5,column=2,value='sathvik')
        sh1['B5'].fill=PatternFill("solid",fgColor="71FF33")
        sh1.cell(row=5,column=3,value='20')
        sh1['C5'].fill=PatternFill("solid",fgColor="71FF33")
        sh1.cell(row=5,column=4,value='16')
        sh1['D5'].fill=PatternFill("solid",fgColor="71FF33")

        wb.save('newsheet.xlsx')  # creates new xlsx file with old and new data in a new filename
```

Ouput:-



B5 sathvik

	A	B	C	D
1	RegNo	S_name	Sub1	Sub2
2	MT21CS12	Samprith	20	20
3	MT21CS13	Samridh	18	19
4				
5	MT21CS13	sathvik	20	16

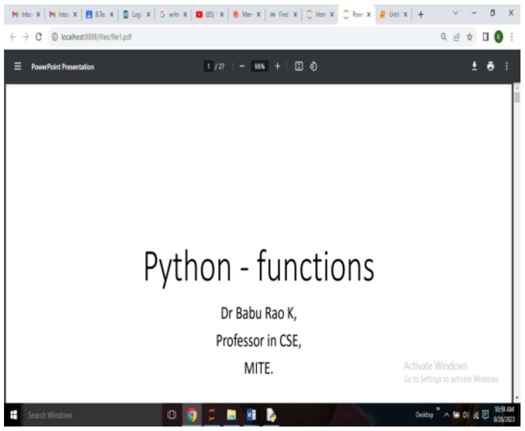
Lab 10 a). Write a python program to combine select pages from many PDFs

Program:-

```
from PyPDF2 import PdfWriter, PdfReader

num = int(input("Enter  page number you want combine from multiple  documents "))

pdf1 = open('file1.pdf', 'rb')
pdf2 = open('file2.pdf', 'rb')
pdf_writer = PdfWriter()
pdf1_reader = PdfReader(pdf1)
page = pdf1_reader.pages[num - 1]
pdf_writer.add_page(page)
pdf2_reader = PdfReader(pdf2)
page = pdf2_reader.pages[num - 1]
pdf_writer.add_page(page)

with open('outputfile.pdf', 'wb') as output:
    pdf_writer.write(output)
```

Output:-

file1.pdf



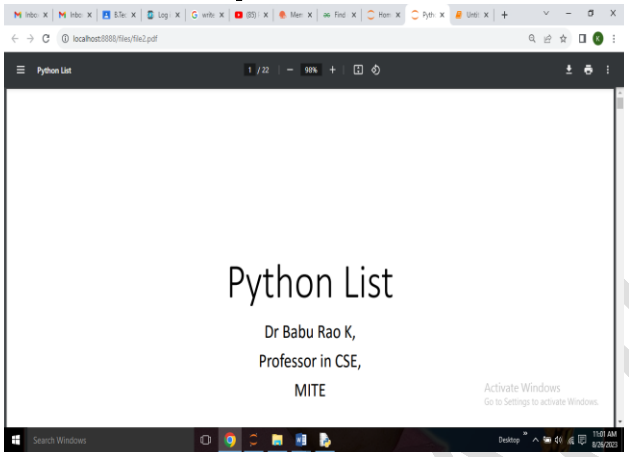
Python - functions

Dr Babu Rao K,

Professor in CSE,

MITE.

file2.pdf



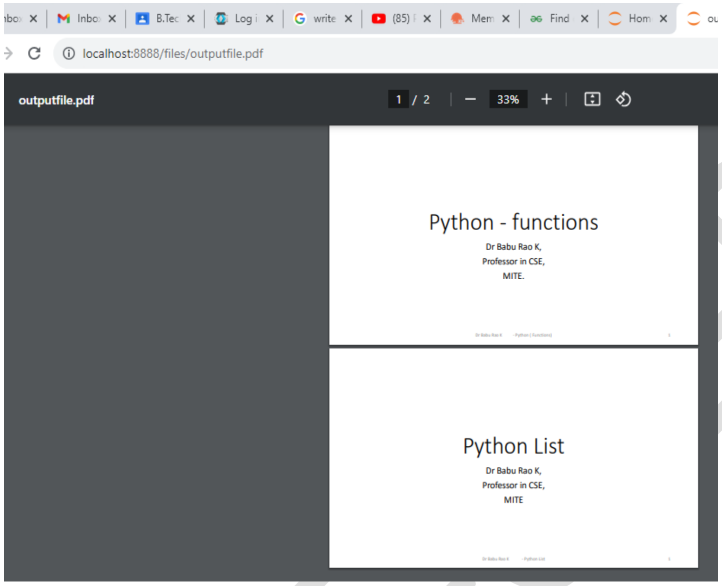
Python List

Dr Babu Rao K,

Professor in CSE,

MITE

Outputfile.pdf



Python - functions

Dr Babu Rao K,

Professor in CSE,

MITE.

Python List

Dr Babu Rao K,

Professor in CSE,

MITE

<u>Lab 10 b).</u> Write a python program to fetch current weather data from the JSON file

<u>Program:-</u>

```
import requests, json

api_key = "980fc61e5f43515a2ec709c871aa6b70"

base_url = "http://api.openweathermap.org/data/2.5/weather?"
city_name = input("Enter city name : ")

complete_url = base_url + "appid=" + api_key + "&q=" + city_name

response = requests.get(complete_url)
x = response.json()
```

```
if x["cod"] != "404":

        y = x["main"]

        current_temperature = y["temp"]

        current_pressure = y["pressure"]

        current_humidity  = y["humidity"]

        z = x["weather"]

        weather_description = z[0]["description"]

        # print following  values
        print(" Temperature (in kelvin unit)  = " + str(current_temperature) +
                "\n atmospheric pressure (in hPa unit)  = " + str(current_pressure) +
                "\n humidity  (in percentage) = " + str(current_humidity)  +
                "\n description  = " + str(weather_description))
else:
        print(" City Not Found ")
```

Output:-

```
Enter city name : Mangalore
 Temperature (in kelvin unit) = 302.63
 atmospheric pressure (in hPa unit) = 1011
 humidity (in percentage) = 74
 description = scattered clouds
```