1. Write a program (WAP) that calculates the total price of an item after adding the

taxes on the given cost. The item has 5-percent state GST and 12-percent Central

GST. The program should display all the amounts.

**Program:**

price=int(input('Enter the price:'))

sgst=price\*0.05

cgst=price\*0.12

print('cgst is',cgst,'sgst',sgst,sep=" ")

print('total price:',price+sgst+cgst)

2. Given the following four element {Bread, 36.50, Cake, 25} WAP that will

display the following formatted string.

The price of the Cake is 25 and the price of Bread is 36.50

**Program:**

str='the price of {} is {} and '\*2

str=str[:-5]

print(str.format('cake',25,'Bread',36.50))

3. WAP in python that will print the following string using one print function and

format function.

|bat<<<<||\*\*cat\*\*||====dat|

**Program:**

str='|{}'+'<'\*4+'||'+'\*'\*2+'{}'+'\*'\*2+'||'+'='\*4+'{}|'

print(str.format('bat','cat','dat'),sep="")

4. WAP that will display all the ASCII characters from A to z.

for i in range(65,91):

print(i,chr(i))

5. WAP that reverse a string. Don’t use reverse method or any inbuilt methods.

str=input()

print(str[::-1])

6. WAP to find the sum of all the odd-valued terms less than 2 million in a Fibonacci series.

Program:

a,b,count\_odd=0,1,0

while b<2000000:

if b%2 is 1:

count\_odd+=b

a,b=b,a+b

print(b)

7. WAP that counts the number's factors of the given number and print this value

using format() method.

**Code:**

n,count\_fact=int(input()),0

for i in range(2,(n//2)+1):

if n%i is 0:

count\_fact+=1

print('The number of factors for {} is {}'.format(n,count\_fact+2))

8. WAP that creates three integer variable. If one of the variable is greater than equal to sum of other two variable then print False otherwise print True.

**Code:**

a,b,c=int(input()),int(input()),int(input())

#another method to take input [a,b,c]=list(map(int,input().split()))

if a >=b+c or b>=c+a or c>=a+b:

print('False')

else:

print('True')

9. WAP in python to find the square root of a given number (Newton’s method).

**Code:**

'''formula for newtons square root method:

for k iterations square root of a is x(k+1) where x(k+1) is

x(k+1) = a^0.5+(1/2x(k))(x(k)-a^0.5)^2

'''

a=int(input('Enter a number'))

ans=1

for k in range(0,120):

ans=a\*\*0.5+(1/2\*ans)\*(ans-a\*\*0.5)\*\*2

print(ans)

10.Luhn checksum algorithm: Consider each digit of the credit card to have a zero-

based index: the first is at index 0, and the last is at index 15. Start from the

rightmost digit and process each digit one at a time. For digits at even-numbered

indexes (the 14th digit, 12th digit, etc.), simply add that digit to the cumulative

sum. For digits at odd-numbered indexes (the 15th, 13th, etc), double the digit's

value, then if that doubled value is less than 10, add it to the sum. If the doubled

number is 10 or greater, add each of its digits separately into the sum. If the sum

value is a multiple of 10 then the Credit card is valid else the card is invalid. WAP

to implement Luhn checksum algorithm for a given credit card number.

**Code:**

#taking all digits into a list

n=str(input())

n=list(map(int,n))

ans\_sum=0

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

if i%2 is 0:

ans\_sum+=n[i]

else:

val=n[i]\*2

if val<10:

ans\_sum+=n[i]

else:

while val>0:

ans\_sum+=val%10

val//=10

if ans\_sum%10 is 0:

print('Valid Credit card number')

else:

print('Invalid Credit card number')

11. Write a small program that assigns an angle in degrees to a variable called

degrees. The program converts this angle to radians and assigns it to a variable

called radians. To convert from degrees to radians, use the formula radians =

degrees ∗ 3.14/180 (where we are using 3.14 to approximate π). Print the angle

in both degrees and radians using format () function.

**Code:**

degrees=int(input())

radians=degrees\*3.14/180

print(‘Degree: {} Radians: {}’.format(degrees,radians))

12. Given the temperature T (in degrees Fahrenheit) and the wind speed v (in miles per hour), the National Weather Service defines the effective temperature (the wind chill) to be: w = 35.74 + 0.6215T + (0.4275T − 35.75)v ^0.16

Compose a program that calculates w and display it for given T and v. The value

of T lies in between -50 to 50 and the value of v is less than 3 or greater than 120.

**Code:**

T=int(input())

v=int(input())

if T<-50 or 50<T or v<3 or 120<v:

print('Not possible to calculate')

else:

w = 35.74+0.6215\*T + (abs(0.4275\*T-35.75)\*v)\*\*0.16

print(w)

4. Compose a program that writes three given integer in ascending order and

descending order. Don’t use any build in functions.

**Code:**

a=int(input())

b=int(input())

c=int(input())

if a<b:

a,b=b,a

if a<c:

a,c=c,a

if b<c:

b,c=c,b

print('Descending order is')

print(a,b,c)

a,c=c,a

print('Ascending order is')

print(a,b,c)

5. Develop a program that calculates the 7th root of 345675453435. Use Newton

Rapson,s method.

6. Compose a program that compute the value of sin(35) using the standard sine formula.

**Code:**

#convert to radians

x=(35\*3.14)/180

val=x-((x\*\*3)/6)+((x\*\*5)/120)-((x\*\*7)/(7\*6\*120))+((x\*\*9)/(9\*8\*7\*6\*120))

print('sin({}) is {}'.format(35,val))

7. Each new term in the Fibonacci sequence is generated by adding the previous two terms. By starting with 0 and 1, the first few terms will be: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ... By considering the terms in the Fibonacci sequence whose values do not exceed four million, find the sum of the terms those are divisible by 3 and 7.

Code:

a,b=0,1

sum1=0

while b<4000000:

if b%3 is 0 or b%7 is 0:

sum1+=b

a,b=b,a+b

print(sum1)

1. Compose a program that takes one command line argument n and writes an n-by-n table such that there is a \* in row i and column j if the gcd of I and j is 1, otherwise there is a space in that position.

**Code:**

def find\_gcd(i,j):

li=[]

for k in range(1,min(i,j)+1):

if i%k is 0 and j%k is 0:

li.append(k)

return max(li)

n=int(input())

for i in range(1,n+1):

for j in range(1,n+1):

if find\_gcd(i,j) is 1:

print('\*',end=' ')

else:

print(' ',end=' ')

print()

2. Euler’s totient function is an important function in number theory: φ(n) is defined as the number of positive integers less than or equal to n that are relative prime with n. Compose a function that takes an integer argument n and returns φ(n). Write a program that takes an integer from command

line, calls the function, and writes the result.

**Code:**

def find\_gcd(i,j):

li=[]

for k in range(1,min(i,j)+1):

if i%k is 0 and j%k is 0:

li.append(k)

return max(li)

print('the required values are ',end=' ')

n=int(input())

for i in range(1,n+1):

if find\_gcd(i,n) is 1:

print(i,end=' ')

3. Compose a function foo() that takes four bool arguments and returns True

if an odd number of arguments are True, and False otherwise.

**Code:**

def foo(a,b,c,d):

if a is True and c is True:

return True

else:

return False

print(foo(True,False,False,True))

4. Compose a function that will compute ex = 1 + x + x2/2! + x3/3! + …… Compose a program that will take x of type float as input and calculate e x using the defined function.

**Code:**

def fact(n):

if n is 1:

return 1

else:

return n\*fact(n-1)

x=float(input('enter a number'))

sum=0

for i in range(1,100):

val=(x\*\*i)/fact(i)

sum+=val

print(sum)

5. Compose a program that takes an integer n, n > 10 as input and reads any n − 10 distinct integers between 10 and n, and determine the missing integer. Don’t use list in your program.

**Code:**

n=int(input())

temp=n-10

sum=0

while temp>0:

val=int(input())

if val<10 and val>n:

continue

sum+=val

temp-=1

sum1=0

for i in range(10,n+1):

sum1+=i

print(sum1-sum)

6(check). Consider a two-player game in which there are n initial pebbles on a table. The players take turns, removing either one or two pebbles from the table, and the player who removes the final pebble wins. Suppose that Alice and Bob play this game, each using a simple strategy:

o Alice always removes a single pebble

o Bob removes two pebbles if an even number of pebbles is on the table, and one otherwise

Given n initial pebbles and Alice starting, who wins the game? Write a recursive program to solve this game.

#Sir, in this question I am getting bob as winner if n is greator than 2 for any value and manually the output coming is also bob

#check this question once

def guess\_winner(n):

n-=1

if n==0:

return 'alice'

if n%2==0:

n-=2

else:

n-=1

if n==0:

return 'bob'

return guess\_winner(n)

for i in range(1,100):

print(guess\_winner(i))

1. Write a function called most\_frequent that takes a string and prints the letters in decreasing order of frequency.

#Write a function called most\_frequent that takes a string and prints the letters in decreasing order of frequency.

def most\_frequent(c):

d={}

for char in c:

if char not in d:

d[char]=1

else:

d[char]+=1

d=rev\_dict(d)

d=list(d.items())

d.sort(reverse=True)

d=dict(d)

for values in d.values():

for item in values:

print(item,end='')

def rev\_dict(d):

rev={}

for key,value in d.items():

if value not in rev:

rev[value]=[key]

else:

rev[value].append(key)

return rev

most\_frequent('sahoo')

1. Implement a recursive function for computing the greatest common divisor of two integer values.

#gcd of two numbers

def gcd(a,b,n,val):

if a%n is 0 or b%n is 0:

val=n

if n is min(a,b):

return val

return gcd(a,b,n+1,val)

gcd(10,20,1,0)

1. Given a list of numbers, return a list where all adjacent duplicate elements have been reduced to a single element.  Ex: [1, 2, 2, 3, 2] returns [1, 2, 3, 2].  You may create a new list or modify the passed in list.

l=list(map(int,input().split()))

l=list(set(l))

print(l)

1. Given an array of integers, compose a program that finds the length of the longest contiguous sequence of equal values where the values of the elements just before and just after this sequence are smaller.

l=list(map(int,input().split()))

li=[]

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

if l[i]<l[i+1]:

j=i+1

count=0

while l[j]==l[j+1]:

count+=1

j+=1

if l[j+1]<l[j]:

li.append(3+count)

print(max(li))

1. Compose a program that reads in text from standard input and writes the number of words in the text.

l=input()

count\_word=1

print(l.count(' ')+1)

1. Write a function called find\_dups that takes a list of integers as its input argument and returns a set of those integers that occur two or more times in the list.

Code:

def find\_dups(s):

dup=[]

for item in set(s):

if s.count(item)>=2:

dup.append(item)

return dup

find\_dups(list(map(int,input().split())))

2.     A DNA sequence is a string made up of the letters A, T, G, and C. To find the complement of a DNA sequence, As are replaced by Ts,Tsby As, Gs by Cs, and Cs by Gs. For example, the complement of AATTGCCGT is TTAACGGCA. Write a program that will compute the complement of a DNA sequence using used defined function.

def complement(str):

for i in range(0,len(str)):

if str[i]=='A':

str=str[:i]+'T'+str[i+1:]

elif str[i]=='T':

str=str[:i]+'A'+str[i+1:]

elif str[i]=='G':

str=str[:i]+'C'+str[i+1:]

elif str[i]=='C':

str=str[:i]+'G'+str[i+1:]

return str

str=input()

print(complement(str))

1. Enter two numbers x and y. Write a program that will print those numbers between x and y whose sum of the digits is an odd number and a member of the Fibonacci series starting with 0 and 1.

def check\_fibonacci(n):

a,b=0,1

if n==0 or n==1:

return True

while b<=n:

if b==n:

return True

a,b=b,a+b

return False

def check\_sum(n):

sum=0

while n!=0:

sum+=n%10

n//=10

return sum%2==1

x,y=int(input()),int(input())

for i in range(x,y+1):

if check\_fibonacci(i) and check\_sum(i):

print(i)

Read the data of the file 4L.txt. This data represents an unrecognized pattern with “abc”. Extract that pattern and display. Also, write that pattern in another text file named as “4p.txt”

import os

os.chdir("C:/Users/IIITK3/Downloads")

input\_file=open('4L.txt','r')

output\_file=open('4A.txt','a')

for line in input\_file:

x=[]

for i in range(0,len(line)-3):

val=line.find('abc',i)

x.append(val)

x.append(val+1)

x.append(val+2)

y=set(x)

x=list(y)

for i in range(0,len(line)):

if i not in x:

line=line[0:i]+' '+line[i+1:]

print(line)

output\_file.write(line)

The file “5L.txt” contains a list of few students and the marks that they have got in four lab tests out of total mark 20. Write a program that should calculate the average marks for each student. If it is over 16 it should write their Name, Roll\_No and their average marks to a new file, if not it should do nothing. The program should also count how many students have averaged above 16.

import os

os.chdir("C:/Users/IIITK3/Downloads")

count1=0

input\_file=open('5L.txt','r')

file2=open('5R.txt','w')

for count,line in enumerate(input\_file):

result=line.split(',')

if count==0:

file2.writelines(['Name','Roll No','Avg Marks'])

file2.writelines('\n')

continue

else:

result.append((int(result[3])+int(result[4])+int(result[5])+int(result[6]))/4)

if result[-1]<16:

continue

else:

count1+=1

file2.writelines([result[0],' ',result[1],' ',result[2],' ',str(result[-1])])

file2.writelines('\n')

file2.write(f'The total students is {count1}')

input\_file.close()

file2.close()

You are given a file “6L.txt” that contains a bunch of names. Some of the names are a first name and a last name separated by spaces, like Akash Agarwal, while others have a middle name, like Akhilesh Kumar Chaurisiya. There are no names consisting of just one word or more than three words. Write a program that asks the user to enter initials, like AA or JKA, and prints all the names that match those initials. Note that initials like AA should match both Akash Agarwal and Atul Kumar Ayer.

import os

os.chdir("C:/Users/IIITK3/Downloads")

with open('6L.txt','r') as input\_file:

y=[]

names\_list=[]

for line in input\_file:

x=line.split()

k=len(x)

if k==2:

u=x[0][0]+x[1][0]

elif k==3:

u=x[0][0]+x[1][0]+x[2][0]

names\_list.append([line,u])

check\_initial=input()

for item in names\_list:

if item[1]==check\_initial or item[1][0]+item[1][-1] == check\_initial:

print(f'{item[0]}')

Write python codes to solve the following:

A) Read the CSV file (Lab-6.csv) and display its content  
    This file contains the data of CS branch of the Batch 2019. Is their any mistake in the dataset provided, Check this through program.  
     If not how many rows have the wrong entries and display the rows with data  
     Correct those rows with the corrected value (CS, 2019)  
     Find the average, maximum, and minimum CGPA of the students  
     Write the corrected data with the computed values to another new csv file.

import os

os.chdir('C:/Users/IIITK3/Downloads')

count\_wrong,avg,sum,min,max=0,0,0,0,0

output\_file=open('Lab-6as.csv','w')

with open('Lab-6 (1).csv','r') as input\_file:

count=0

for count,line in enumerate(input\_file):

l=0

if count>0:

a=line.split(',')

if a[1]!='CS':

count\_wrong+=1

l=1

if a[2]!='2019':

if l==0:

count\_wrong+=1

a[2]='2019'

sum+=eval(a[-1])

if count==1:

min=max=(a[-1])

else:

if (a[-1])<min:

min=a[-1]

if (a[-1])>max:

max=a[-1]

x=','.join(a)

output\_file.writelines(x)

output\_file.write(f'Min CGPA is {min}')

output\_file.write(f'Max CGPA is {max}')

output\_file.write(f'Max CGPA is {sum/(count+1)}')

B) Read the file (Lab-6\_2.csv)  
    Display the number of rows and columns of this dataset.  
    Display the time at which the information regarding each row is collected.  
    Display how many rows have missing values  
    Replace the missing values with average value of the remaining values of that column.

    Write the computed data to the new file.

import os

os.chdir('C:/Users/IIITK3/Downloads')

count\_unknown,aud\_jpy\_open,aud\_jpy\_high,aud\_jpy\_low,aud\_jpy\_close=0,0,0,0,0

with open('Lab-6\_2.csv','r') as input\_file:

count=0

for count,line in enumerate(input\_file):

if count is 0:

print(f'No of columns is {len(line)}')

if count>0:

a=line.split(',')

for i in range(0,len(a)):

if a[i] is '':

count\_unknown+=1

if a[1]!='':

aud\_jpy\_open+=eval(a[1])

if a[2]!='':

aud\_jpy\_high+=eval(a[2])

if a[3]!='':

aud\_jpy\_low+=eval(a[3])

if a[4]!='':

aud\_jpy\_close+=eval(a[4])

aud\_jpy\_open,aud\_jpy\_high,aud\_jpy\_low,aud\_jpy\_close=aud\_jpy\_open/count,aud\_jpy\_high/count,aud\_jpy\_low/count,aud\_jpy\_close/count

print(f'Count of missing data is {count\_unknown}')

output\_file=open('Lab-6\_2(output).csv','w')

with open('Lab-6\_2.csv','r') as input\_file:

for count,line in enumerate(input\_file):

if count>0:

a=line.split(',')

if a[1] is '':

a[1]=f'{aud\_jpy\_open}'

if a[2] is '':

a[2]=f'{aud\_jpy\_high}'

if a[3] is '':

a[3]=f'{aud\_jpy\_low}'

if a[4] is '':

a[4]=f'{aud\_jpy\_close}'

output\_file.write(','.join(a))

) Read the csv file (Lab\_6\_3.csv) that contains the information regarding the availability of alcohol for consumption of the month December, 2018 and answer the following questions through python code and write your answer to a new csv file(named as Lab\_6\_3\_out.csv).

a) How many attributes are in the CSV file and what are those?  
b) Display the distinct values that each attribute has.  
c) Is there any attribute having missing values?  
d) Fill the missing values by the mean of the remaining data.  
e) Find the maximum data value and print the corresponding Series\_title  
f) Display total volume of wine available in the month December,2108.  
g) Which Series\_title has maximum data entries in the csv file?

h) Find the year in which the Fortified wine brand has discontinued.

import csv

li=[] #holds all the rows used in all operations

with open('Lab\_6\_3.csv','r') as file:

di={} #holds the values of different wines and their count

series=[] #holds the last series

l=0

dis=[] #distinct vales

for row in csv.reader(file):

li.append(row)

for item in row:

dis.append(item)

total=len(li)-1

for i in range(0,len(li)):

if li[i][6] not in series:

series.append(li[i][6])

series.append(li[i][6])

for j in range(0,len(li[0])):

if li[i][j]=='':

mis\_attr=li[0][j]

if i>1 and li[i][1]!='':

l=l+float(li[i][1])

if li[i][6] not in di:

di[li[i][6]]=0

series=set(series)

series=list(series)

l=l/total

maxi=float(li[i][1])

Vol\_wine=0

series\_count=0

max\_series\_count=0

for i in range(0,len(li)):

if i>=1:

if li[i][1]=='':

li[i][1]=str(l)

k=float(li[i][1])

if li[i][6]=='Wine':

Vol\_wine=Vol\_wine+k

if float(maxi)<(k):

maxi=li[i][1]

ser\_title=li[i][6]

max\_series\_count=1

max\_series\_name='NULL'

for item in di.keys():

series\_count=0

for i in range(0,len(li)):

if li[i][6]==item:

series\_count+=1

di[item]=series\_count

if max\_series\_count<series\_count:

max\_series\_count=series\_count

max\_series\_name=item

get=''

for item in series:

if 'Fortified wine' in item:

get=item

print('The no. of attributes in csv file are',len(li[0]),'and they are',li[0])

print('The distinct values in series are')

print(list(set(dis)))

print('The missing attr is',mis\_attr)

print('Data after filling missing values are ',dis)

print('The maximum data value is ',maxi)

print('Volume of wine available is',Vol\_wine)

print('The maximum data value and their corresponding series name is:',end='')

print(max\_series\_count,max\_series\_name)

print('The Fortified wine is discontinued in the year',get[-5:-1])

Regular Expressions:

Write a regular expression that will match any string that starts with 'iiitk' and ends with '2018' with any number of characters, including none, in between. Implement your regular expression.

#Regular Expressions

import re

pat=re.compile('^iiitk(\w)\*(\W)\*(\d)\*2018$')

string=input('Enter a string')

if pat.match(string):

print('yes')

else:

print('no')

Write a program using regular expression that will match any Python (.py) file from your working directory and returns as a list. Hint: os.walk() returns a list of three items. It contains the name of the root directory, a list of the names of the subdirectories, and a list of the filenames in the current directory.

**Code:**

import os

import re

os.chdir('C:/Users/IIITK2/Desktop/check files')

pat=re.compile('[a-z0-9]\*\.py')

for directory\_path,directory\_names,file\_names in os.walk(".",topdown=False):

for item in file\_names:

if pat.search(item):

with open(item,'r') as req:

li=req.readlines()

pat1=re.compile('def [a-z]\*')

for item2 in li:

if pat1.search(item2):

xr=item2.split('(')

print(xr[0][4:])

The file emails.txt (see the attachment) contains email addresses of a set of newly admitted anonymous students. From this email addresses, Write Python code using regular expression that will generate email addresses for the domain [iiitkottayam.ac.in](http://iiitkottayam.ac.in/).  Step 1: read in the file. Step 2: create a Python regular expression to identity the usename Step 3: extract the associated username Step 4: Make new email address with [iiitkottayam.ac.in](http://iiitkottayam.ac.in/) domain name, e.g. [mrk@iiitkottayam.ac.in](mailto:mrk@iiitkottayam.ac.in)

import os,re

os.chdir('C:/Users/IIITK3/Downloads')

with open('emails.txt','r') as input\_file:

for count,line in enumerate(input\_file):

pat=re.compile('(\w)\*@(\w)\*.(\w)\*')

if pat.search(line):

use=line.split('@')

print(use[0]+'@iiitkottayam.ac.in')

Write a python code that takes the name of a Python source file as input and outputs a list of the names of all the user-defined functions defined in that file.

import os,re

os.chdir('C:/Users/IIITK3/Downloads')

with open('aditya.py','r') as input\_file:

for count,line in enumerate(input\_file):

pat=re.compile('def [a-zA-z0-9]\*')

if pat.match(line):

val=line.split('(')

print(val[0][4:])

Using the previous code, write a function that takes a list of filenames for Python programs and outputs a list of user-defined functions and their source file as tuples.

import os,re

os.chdir('C:/Users/IIITK3/Downloads')

def print\_function\_names(filename):

li=[]

with open(filename,'r') as input\_file:

for count,line in enumerate(input\_file):

pat=re.compile('def [a-zA-z0-9]\*')

if pat.match(line):

val=line.split('(')

li.append(val[0][4:])

return (filename,li)

li=input('Enter the file names').split()

for item in li:

print(print\_function\_names(item))

Write a python code using regular expression to validate an IP address of a computer.

import re

pat=re.compile("(25[0-5]|2[0-4][0-9]|[0-1]?[0-9][0-9]?\.){3}(25[0-5]|2[0-4][0-9]|[0-1]?[0-9][0-9]?)$")

if re.match(pat,"122.168.136.212"):

print('Valid IP Address')

else:

print('Invalid IP Address')

Write a user defined python function that takes a postfix and a file name as argument and then creates a list of words from that file that precedes the postfix. Define your pattern to ignore the case of the word. Implement your function.

#Question Doubt