```
# p01.py
"""
The finance department of a company wants to calculate the monthly pay of one of its employee.
Monthly pay should be calculated as mentioned in the formula below and display all the employee
details. Monthly Pay= No. of hours worked in a week * Pay rate per hour * No .of weeks in a month
Write a Python Program to implement the problem.
"""

week_hours = float(input("Enter weekly working hours : "))
hourly_pay = float(input("Enter pay rate per hour : "))
working_weeks = float(input("Enter working weeks in a month : "))

monthly_pay = week_hours * hourly_pay * working_weeks

print("Calculated monthly pay : Rs", monthly_pay)

##############################################
# p02_a.py
"""
The finance department of a company wants to calculate the monthly net pay of one of its employee
by finding the income tax to be paid (in Indian Rupees) and the net salary after the income tax
deduction. The employee should pay the income tax based on the following table:
Gross Salary(In Rs) | Tax Percentage
--------------------|---------------
Below 5,000         | Nil
5,001 to 10,000     | 10%
10,001 to 20,000    | 20%
More than 20,000    | 30%

Display the employee id, basic salary, allowances, gross pay, income tax and net pay using Python
Programming.
"""

emp_id = int(input("Enter Employee ID : "))
basic_pay = float(input("Enter Basic Pay : "))
allowances = float(input("Enter Allowances : "))

gross_pay = basic_pay + allowances

if gross_pay > 20000:
    income_tax = gross_pay * 0.3
elif gross_pay > 10000:
    income_tax = gross_pay * 0.2
elif gross_pay > 5000:
    income_tax = gross_pay * 0.1
else:
    income_tax = 0

net_pay = gross_pay - income_tax

print("Employee ID : ", emp_id)
print("Basic Pay   : Rs", basic_pay)
print("Allowances  : Rs", allowances)
print("Gross Pay   : Rs", gross_pay)
print("Income Tax  : Rs", income_tax)
print("Net Pay     : Rs", net_pay)

##############################################
# p02_b.py
"""
In the retail application, display the details of the customer like bill id, customer id, bill
amount and customer name. But the retail shop wants the customer name to be between 3 to 20
characters. Write a Python Program to implement the case study.
"""
bill_id = input("Enter the Bill ID : ")
cust_id = input("Enter the Customer ID : ")
bill_amt = float(input("Enter the bill amount : "))
cust_name = input("Enter the Customer name :")
flag = True
while flag:
    if(len(cust_name)<3 or len(cust_name)>20):
        print("The name is invalid!")
        cust_name=input("Enter the Customer Name : ")
else:
    flag = False  print("BILL ID : ",bill_id)
print("CUSTOMER ID : ",cust_id) print("BILL AMOUNT : ",bill_amt)
print("CUSTOMER NAME : ",cust_name)
```

```
##########################################
# p03.py
#Write a Python program to check whether the given string is palindrome or not.

def easy_pal(string):
    if string == string[::-1]:
        print(string + " is a palindrome")
    else:
        print(string + " is not a palindrome")

def harder_pal(string):
    LENGTH = len(string)

    for i in range(LENGTH//2):
        if string[i] != string[LENGTH - i - 1]:
            print(string + " is not a palindrome")
            return

    print(string + " is a palindrome")

inp = input("Enter a string: ")

easy_pal(inp)

harder_pal(inp)

##########################################
# p04.py
#Write a Python program to generate first ‘n’ Fibonacci numbers.

def fibonaci(n):
    if n==0:
        return 0
    if n==1:
        return 1
    return fibonaci(n-1)+fibonaci(n-2)

n=int(input("Enter n:"))
print("The fibonacci series:")
for i in range(n):
    print(fibonaci(i))

##########################################
# p05.py
"""
Consider the scenario of processing marks of students for a course in student management system.
Given below is the list of marks scored by students. Find top three scorers for the course and also
display the average marks scored by all students. Implement the solution using Python Programming.
Student Name Marks Scored
John    |86.5
Jack    |91.2
Jill    |84.5
Harry   |72.1
Joe     |80.5
"""

student={"John":86.5,"Jack":91.2,"Jill":84.5,"Harry":72.1,"Joe":80.5}
marks=list(student.values())
marks.sort()#The top three scorers will be the last three marks holders
for s in student:#To get the name for the top scorer marks
    if student[s]==marks[len(marks)-1]:
        top1=s
    elif student[s]==marks[len(marks)-2]:
        top2=s
    elif student[s]==marks[len(marks)-3]:
        top3=s
#To compute the average
avg=0
for i in marks:
    avg+=i
avg/=len(marks)

print("The top three scorers are:",top1,",",top2,",",top3)
print("The average score:",avg)

##########################################
```

```
# p06.py
#Write a program to count and display the number of capital letters in a given string.

s=input("Enter the string")
cnt=0
for ch in s:
    if ch.isupper():
        cnt+=1
print("The number of capital letters are:",cnt)

##############################################
# p07.py
#Write a program to count the number of each vowel in a given string.

vowels={"a","A","e","E","i","I","o","O","u","U"}
s=input("Enter the string:")
cnt=0
for ch in s:
    if ch in vowels:
        cnt+=1
print("The number of vowels:",cnt)

##############################################
# p08.py
#Write a program to remove all punctuations like “’!()-[]{};:’’’,\,<>,/,?,@,#,$, %^&*_~” from the
string provided by the user.

s=input("Enter the string:")
for ch in s:
    if ch in "'!()-[];:,\\<>/?@#$%^&*_~\"": #To include " give it as /"
        s=s.replace(ch,"")
print(s)

##############################################
# p09.py
'''
Consider two strings, String1 and String2 and display the merged_string as output. The
merged_string should be the capital letters from both the strings in the order they appear. Sample
Input: String1: I Like C String2: Mary Likes Python Merged_string should be ILCMLP
'''

String1=input("Enter string 1:")
String2=input("Enter string 2:")
merged_string=""
for ch in String1:
    if ch.isupper():
        merged_string=merged_string+ch
for ch in String2:
    if ch.isupper():
        merged_string=merged_string+ch
print(merged_string)

##############################################
# p10.py
'''
With a given integral number n, write a program to generate a dictionary that contains (i, i*i)
such that is an integral number between 1 and n (both included) and then the program should print
the dictionary. Suppose the following input is supplied to the program: 8 Then, the output should
be:
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64}
'''
n=int(input("Enter n:"))
d=dict()
for i in range(1,n+1):
    d[i]=i*i

print(d)

##############################################
# p11.py
'''
Write a binary search function which searches an item in a sorted list. The function should return
the index of element to be searched in the list.
'''

def binary_search(sorted_list, search_item):
    lower, upper = 0, len(sorted_list)
```

```
    while lower <= upper:
        mid_index = (lower + upper) // 2 # // will only retuen int after division
        mid_item = sorted_list[mid_index]
        if search_item == mid_item:
            return mid_index + 1
        if mid_item > search_item:
            upper = mid_index - 1
        else:
            lower = mid_index + 1
    return -1

sorted_list = input("Enter list items [in sorted order] : ")
search_item = int(input("Enter the element to search : "))
sorted_list = sorted_list.split(' ')
sorted_list = list(map(int, sorted_list))   # converts elements from string to integers
result = binary_search(sorted_list, search_item)
if result is -1:
    print("Element not found")
else:
    print(search_item, "found at position ", result)

##############################################
# p12.py
'''
Design a class named Rectangle to represent a rectangle. class contains:
* Two data fields named width and height .
* A constructor that creates a rectangle with the specified width and height .
* The default values are 1 and 2 for the width and height, respectively.
* A method named getArea() that returns the area of this rectangle.
* A method named getPerimeter() that returns the perimeter.
Write a test program that creates two Rectangle objects—one with width 4 and height 40 and the
other with width 3.5 and height 35.7. Display the width, height, area, and perimeter of each
rectangle in this order.
'''

class Rectangle:
    def __init__(self,width=1,height=2):
        self.width=width
        self.height=height
    def getArea(self):
        return self.width*self.height
    def getPerimeter(self):
        return 2*(self.width+self.height)

def main():
    r1=Rectangle(4,40)
    r2=Rectangle(3.5,35.7)
    print("For Rectangle 1:")
    print("Width:",r1.width," Height:",r1.height," Area:",r1.getArea()," 
Perimeter:",r1.getPerimeter())
    print("For Rectangle 2:")
    print("Width:",r2.width," Height:",r2.height," Area:",r2.getArea()," 
Perimeter:",r2.getPerimeter())

main()

##############################################
# p13.py
"""
Design a class named Account that contains:
* A private int data field named id for the account.
* A private float data field named balance for the account.
* A private float data field named annualInterestRate that stores the current interest rate.
* A constructor that creates an account with the specified id (default 0), initial balance (default
100), and annual interest rate (default 0).
* The accessor and mutator methods for id , balance , and annualInterestRate .
* A method named getMonthlyInterestRate() that returns the monthly interest rate.
* A method named getMonthlyInterest() that returns the monthly interest.
* A method named withdraws that withdraws a specified amount from the account.
* A method named deposit that deposits a specified amount to the account.
(Hint: The method getMonthlyInterest() is to return the monthly interest amount, not the interest
rate. Use this formula to calculate the monthly interest: balance*monthlyInterestRate.
monthlyInterestRate is annualInterestRate / 12 . Note that annualInterestRate is a percent (like
4.5%). You need to divide it by 100 .)
Write a test program that creates an Account object with an account id of 1122, a balance of
$20,000, and an annual interest rate of 4.5%. Use the withdraw method to withdraw $2,500, use the
deposit method to deposit $3,000, and print the id, balance, monthly interest rate, and monthly
```

```
interest.
"""

class Account:
    def __init__(self,id=0,balance=100.0,annualInterestRate=0.0):
        self.__id=id
        self.__balance=balance
        self.__annualInterestRate=annualInterestRate
    def getId(self): #Accessor Functions
        return self.__id
    def getBalance(self):
        return self.__balance
    def getannualInterestRate(self):
        return self.__annualInterestRate
    def setId(self,id): #Mutator Fucntions
        self.__id=id
    def setBalance(self,balance):
        self.__balance=balance
    def setannualInterestRate(self,annualInterestRate):
        self.__annualInterestRate=annualInterestRate
    def getMonthlyInterestRate(self):
        return self.__annualInterestRate/(12*100)
    def getMonthlyInterest(self):
        return self.__balance*self.getMonthlyInterestRate()
    def withdraw(self,amt):
        self.__balance-=amt
    def deposit(self,amt):
        self.__balance+=amt

a=Account(1122,20000,4.5)
a.withdraw(2500)
a.deposit(3000)
print("Id:",a.getId()," Balance:",a.getBalance()," Monthly Interest
Rate:",a.getMonthlyInterestRate()," Monthly Interest:",a.getMonthlyInterest())

##############################################
# p14.py
'''
Write a function that returns the number of days in a year using the following header:
def numberOfDaysInAYear(year):
Write a test program that displays the number of days in the years from 2010 to 2020.
'''

def numberOfDaysInAYear(year):
    if year%4==0 and year%100!=0 or year%400==0:
        return 366
    else:
        return 365

days=0
yr=2010
while yr<2020:
    days+=numberOfDaysInAYear(yr)
    yr+=1
print("The number of days between 2010 and 2020 are:",days)

##############################################
# p15.py
'''
(Display matrix of 0s and 1s) Write a function that displays an n-by-n matrix using the following
header:
def printMatrix(n):
Each element is 0 or 1, which is generated randomly. Write a test program that prompts the user to
enter n and displays an n-by-n matrix.
Sample run:
Enter n: 3
0 1 0
0 0 0
1 1 1
'''
import random

def gen_random_matrix(order):
    for i in range(order):
        for j in range(order):
            print(random.randint(0, 1), end=' ')
        print('')
```

```
#test
order = int(input("Enter order of matrix : "))
gen_random_matrix(order)

##########################################
# p16.py
#To grade MCQ of students based on the answer key

student=[["A","B","A","C","C","D","E","E","A","D"],
         ["D","B","A","B","A","D","E","E","A","D"],
         ["E","D","D","A","B","D","E","E","A","D"],
         ["C","B","A","E","D","C","E","E","A","D"],
         ["A","B","D","C","C","D","E","E","A","D"],
         ["B","B","E","C","C","D","E","E","A","D"],
         ["B","B","A","C","C","D","E","E","A","D"],
         ["E","B","E","C","C","D","E","E","A","D"]]

answer=["D","B","D","C","C","D","A","E","A","D"]
cnt=0 # For student number
for s in student: #s refers to individual student
    marks=0
    i=0
    while i<len(answer): # Compare every key and the answer
        if s[i]==answer[i]:
            marks+=1
        i+=1
    print("The marks of student ",cnt,":",marks)
    cnt+=1

##########################################
# p17.py
'''
(Find the index of the smallest element) Write a function that returns the index of the smallest
element in a list of integers. If the number of such elements is greater than 1, return the
smallest index.
Use the following header: def indexOfSmallestElement(lst):
Write a test program that prompts the user to enter a list of numbers, invokes this function to
return the index of the smallest element, and displays the index.
'''
def indexOfSmallestElement(lst):
    LENGTH = len(lst)
    small_element, small_index = lst[0], 0
    for i in range(LENGTH):
        if lst[i] < small_element:
            small_element = lst[i]
            small_index = i
    return small_index + 1

#test
element_list = input("Enter list items : ")
# Convert list elements to int
element_list = element_list.split(' ')
print(element_list)
element_list = list(map(int, element_list))
print("Index of smallest element : ", indexOfSmallestElement(element_list))

##########################################

# p18.py
'''
Write the following function that tests whether the list has four consecutive numbers with the same
value:
def isConsecutiveFour(values):
Write a test program that prompts the user to enter a series of integers and reports whether the
series contains four consecutive numbers with the same value.
'''

def isConsecutiveFour(element_list):
    LENGTH = len(element_list)

    for i in range(LENGTH - 3):
        cur_ele = element_list[i]

        if element_list[i : i + 4] == [cur_ele, cur_ele, cur_ele, cur_ele]:
            return True
```

```
    return False


element_list = input("Enter list elements : ")
element_list = element_list.split(' ')
print("occurance of consecutive four in list : ", isConsecutiveFour(element_list))

#valid input : 1 2 2 2 2 3 4 (four consecutive 2s)
#invalid input : 1 2 3 4 5

##########################################
# p19.py
'''
Write a program that will count the number of characters, words, and lines in a file. Words are
separated by a white-space character. Your program should prompt the user to enter a filename.
'''

file_name = input("Enter file name : ").strip()

try:
    file = open(file_name) # Or use "if not(os.path.isfile(fname)):" import os.path
    word_count = 0
    line_count = 0
    char_count = 0
    for line in file:
        line_count += 1
        char_count += len(line) # len(line.strip()) To ignore the newline chars
        words=line.split(" ")
        word_count += len(words) # OR word_count += line.count(' ') + 1   ## number of spaces + 1

    print("Number of Lines : ", line_count)
    print("Number of Words : ", word_count)
    print("Number of Chars : ", char_count)

except FileNotFoundError:
    print("file not found")

##########################################
# p20.py
'''
Suppose that a text file contains an unspecified number of scores. Write a program that reads the
scores from the file and displays their total and average. Scores are separated by blanks. Your
program should prompt the user to enter a filename.
'''

file_name = input("Enter file name : ")

try:
    file = open(file_name)
    scores = []

    total,n = 0,0
    for line in file:
        scores = line.split(' ')
        scores = list(map(int, scores))
        n+=len(scores)
        print(scores)
        for score in scores:
            total += score

    print("Total score : ", total)
    print("Average score : ", total / n)

except FileNotFoundError:
    print("file not found")

##########################################
# p21.py
'''
Design a class named Triangle that extends the GeometricObject class. The Triangle class contains:
* Three float data fields named side1 , side2 , and side3 to denote the three sides of the triangle.
* A constructor that creates a triangle with the specified side1 , side2 , and side3 with default
values 1.0 .
* The accessor methods for all three data fields.
* A method named getArea() that returns the area of this triangle.
* A method named getPerimeter() that returns the perimeter of this triangle.
* A method named __str__() that returns a string description for the triangle.
```

```
Write a test program that prompts the user to enter the three sides of the triangle, a color, and 1
or 0 to indicate whether the triangle is filled. The program should create a Triangle object with
these sides and set the color and filled properties using the input. The program should display the
triangle's area, perimeter, color, and True or False to indicate whether the triangle is filled or
not.
'''

import math

class GeometricObject:
    def __init__(self, color, filled):
        self.color = color
        self.filled = filled

    def getPerimeter(self):
        pass
    def getArea(self):
        pass
    def getColor(self):
        return self.color
    def getFilled(self):
        return self.filled

class Triangle(GeometricObject):
    '''
    Represents a triangle having sides s1,,s2 and s3.Computes Areas and also the Perimeter
    '''
    def __init__(self, color, filled, side1 = 1.0, side2 = 1.0, side3 = 1.0):
        GeometricObject.__init__(self, color, filled)
        self.side1 = side1
        self.side2 = side2
        self.side3 = side3

    # Accessor methods
    def getSide1(self):
        return self.side1
    def getSide2(self):
        return self.side2
    def getSide3(self):
        return self.side3
    def getPerimeter(self):
        return self.side1 + self.side2 + self.side3
    def getArea(self):
        # Heron's formula
        p = self.getPerimeter() / 2;
        area = math.sqrt(p * (p - self.side1) * (p - self.side2) * (p - self.side3))
        return area

    def __str__(self):
        return Triangle.__doc__ #returns the comment inder the class
    # def __str__(self):
    #     return  "Side 1    : " + str(self.side1) + "\n" + \
    #             "Side 2    : " + str(self.side2) + "\n" + \
    #             "Side 3    : " + str(self.side3) + "\n" + \
    #             "Perimeter : " + str(self.getPerimeter()) + "\n" + \
    #             "Area      : " + str(self.getArea()) + "\n" + \
    #             "Filled    : " + str(self.filled) + "\n" + \
    #             "Color     : " + str(self.color)

triangle1 = Triangle("RED", True)
print(triangle1)

###########################################
# p22.py
'''
Write a program that draws a rectangle or an oval, as shown in Figure below: The user selects a
figure from a radio button and specifies whether it is filled by selecting a check button.
'''
from tkinter import *


def process():
    canvas.delete("r1", "r2", "o1", "o2")
    if var1.get() == 1 and var2.get() == 1:
        canvas.create_rectangle(100, 100, 1000, 500, fill="black", tags="r1")
    elif var1.get() == 1 and var2.get() == 0:
        canvas.create_rectangle(100, 100, 1000, 500, tags="r2")
```

```
    elif var1.get() == 0 and var2.get() == 0:
        canvas.create_oval(10, 10, 190, 90, tags="o1")
    elif var1.get() == 0 and var2.get() == 1:
        canvas.create_oval(10, 10, 190, 90, fill="black", tags="o2")


window = Tk()
frame = Frame(window)
var1 = IntVar()
var2 = IntVar()
canvas = Canvas(window, width=1280, height=720, bg="white")
r1 = Radiobutton(frame, text="Rectangle", variable=var1, value=1, command=process)
r2 = Radiobutton(frame, text="Oval", variable=var1, value=0, command=process)
c1 = Checkbutton(frame, text="Filled", variable=var2, onvalue=1, offvalue=0, command=process)
canvas.pack()
frame.pack()
r1.grid(row="1", column="1")
r2.grid(row="1", column="2")
c1.grid(row="1", column="3")
window.mainloop()

##############################################
# p23.py
'''
Write a program that calculates the future value of an investment at a given interest rate for a
specified number of years. The formula for the calculation is as follows:
futureValue = investmentAmount * (1 + monthlyInterestRate) ^ (years * 12)
Use text fields for users to enter the investment amount, years, and interest rate. Display the
future amount in a label field when the user clicks the Calculate button, as shown.
'''
from tkinter import *


def processButton():
    futureValue = float(investmt_amt.get()) * pow(1 + (float(annual_intrst_rate.get()/100) / 12),
float(years.get()) * 12)
    l5["text"] = str(futureValue)


window = Tk()
window.title("Investment Calculator")
investmt_amt = DoubleVar()
l1 = Label(window, text="Investment Amount")
t1 = Entry(window, textvariable=investmt_amt)
years = DoubleVar()
l2 = Label(window, text="Years")
t2 = Entry(window, textvariable=years)
annual_intrst_rate = DoubleVar()
l3 = Label(window, text="Annual Interest Rate (%)")
t3 = Entry(window, textvariable=annual_intrst_rate)

l4 = Label(window, text="Future Value")
l5 = Label(window)
button = Button(window, text="Calculate", command=processButton)
l1.grid(row="1", column="1")
t1.grid(row="1", column="2")
l2.grid(row="2", column="1")
t2.grid(row="2", column="2")
l3.grid(row="3", column="1")
t3.grid(row="3", column="2")
l4.grid(row="4", column="1")
l5.grid(row="4", column="2")
button.grid(row="5", column="2")
window.mainloop()

##############################################
# p24.py
#Write a program that displays a still fan.
from tkinter import *

window=Tk();
canvas=Canvas(window,width=500,height=500,bg="white")
canvas.create_arc(0,0,500,500,start=0,extent=45,fill="black")
canvas.create_arc(0,0,500,500,start=90,extent=45,fill="black")
canvas.create_arc(0,0,500,500,start=180,extent=45,fill="black")
canvas.create_arc(0,0,500,500,start=270,extent=45,fill="black")
canvas.pack()
```

```
window.mainloop()

##########################################
# p25.py
# Custom exception TriangleError
import math
import sys
class TriangleError(RuntimeError):
    def __init__(self,s1,s2,s3):
        self.__side1=s1
        self.__side2=s2
        self.__side3=s3
    def getS1(self):
        return self.__side1
    def getS2(self):
        return self.__side2
    def getS3(self):
        return self.__side3
    def printError(self):
        print("The sum of any two sides of a triangle must always be greater than the third side")
        sys.exit()

class GeometricObject:
    def __init__(self, color, filled):
        self.color = color
        self.filled = filled

    def getPerimeter(self):
        pass
    def getArea(self):
        pass
    def getColor(self):
        return self.color
    def getFilled(self):
        return self.filled

class Triangle(GeometricObject):
    '''
Represents a triangle having sides s1,s2 and s3.Computes Areas and also the Perimeter
    '''
    def __init__(self, color, filled, side1 = 1.0, side2 = 1.0, side3 = 1.0):
        GeometricObject.__init__(self, color, filled)
        self.side1 = side1
        self.side2 = side2
        self.side3 = side3
        try:
            if self.side1+self.side2<=self.side3 or self.side1+self.side3<=self.side2 or
self.side3+self.side2<=self.side1:
                raise TriangleError(self.side1,self.side2,self.side3)
            else:
                pass
        except TriangleError as e:
            e.printError()

    # Accessor methods
    def getSide1(self):
        return self.side1
    def getSide2(self):
        return self.side2
    def getSide3(self):
        return self.side3
    def getPerimeter(self):
        return self.side1 + self.side2 + self.side3
    def getArea(self):
        # Heron's formula
        p = self.getPerimeter() / 2;
        area = math.sqrt(p * (p - self.side1) * (p - self.side2) * (p - self.side3))
        return area

    def __str__(self):
        return Triangle.__doc__ #returns the comment inder the class
    # def __str__(self):
    #     return  "Side 1    : " + str(self.side1) + "\n" + \
    #             "Side 2    : " + str(self.side2) + "\n" + \
    #             "Side 3    : " + str(self.side3) + "\n" + \
    #             "Perimeter : " + str(self.getPerimeter()) + "\n" + \
    #             "Area      : " + str(self.getArea()) + "\n" + \
```

```
    #             "Filled    : " + str(self.filled) + "\n" + \
    #             "Color     : " + str(self.color)

triangle1 = Triangle("RED", True, 5,8,4)
print(triangle1)
triangle2 = Triangle("RED", True, 2,2,8)

##############################################
# p26.py
#MultiThreading - one thread to print number of vowels and other to check palindrome

from threading import Thread
import time

class Mythread(Thread):
    def __init__(self,threadID,name,string):
        Thread.__init__(self)
        self.name=name
        self.string=string
        self.threadID=threadID
    def run(self):
        print("Starting ",self.name)
        if self.threadID==1:
            self.vowelcount()
        else:
            self.palindrome()
        print("Exiting ",self.name)
    def vowelcount(self):
        vowels={"a","A","e","E","i","I","o","O","u","U"}
        cnt=0
        for ch in self.string:
            if ch in vowels:
                cnt+=1
        print("The number of vowels:",cnt)
    def palindrome(self):
        srev=self.string[::-1]
        if self.string==srev:
            print("String is a palindrome")
        else:
            print("String is not a palindrome")


thread1=Mythread(1,"Find Vowel","I love Python Programming")
thread2=Mythread(2,"Palindrome","malayalam")
thread1.start()
thread2.start()
thread1.join()
thread2.join()
print("Exiting main thread")

##############################################
# p27.py
'''Design a Tkinter interface to perform the following operations on a database by considering the
table
Student (USN: String, Name: String, Age: Int, Branch: String).
Display the success and failure message using MessageBox
a. Insert student details
b. Search the student details with USN=”4NM06CS001"'''
import pymysql
from tkinter import *
from tkinter import messagebox

mydb=pymysql.connect(host="localhost",user="root",passwd="root",database="stud")
mycursor=mydb.cursor()

def processInsert():
    sql="INSERT INTO Student VALUES ('"+var1.get()+"','"+var2.get()+"','"+str(var3.get())
+"','"+var4.get()+"')"
    try:
        mycursor.execute(sql)
        mydb.commit()
        messagebox.showinfo("Insertion","Insertion successful")
    except:
        messagebox.showerror("Insertion","Insertion Unsuccessful")
        mydb.rollback()

def processSearch():
```

```
    sql="SELECT * FROM Student WHERE USN='"+var1.get()+"'"
    mycursor.execute(sql)
    for row in mycursor:
        if(var1.get()==row[0]):
            messagebox.showinfo("Search","Search successful")
            var1.set(row[0])
            var2.set(row[1])
            var3.set(row[2])
            var4.set(row[3])
            return
    messagebox.showerror("Search","Search Unsuccessful")

window=Tk()
window.title("Student")
l1=Label(window,text="USN")
l2=Label(window,text="Name")
l3=Label(window,text="Age")
l4=Label(window,text="Branch")
var1=StringVar()
var2=StringVar()
var3=IntVar()
var4=StringVar()
t1=Entry(window,textvariable=var1)
t2=Entry(window,textvariable=var2)
t3=Entry(window,textvariable=var3)
t4=Entry(window,textvariable=var4)
b1=Button(window,text="Insert",command=processInsert)
b2=Button(window,text="Search",command=processSearch)
l1.grid(row="1",column="1")
t1.grid(row="1",column="2")
l2.grid(row="2",column="1")
t2.grid(row="2",column="2")
l3.grid(row="3",column="1")
t3.grid(row="3",column="2")
l4.grid(row="4",column="1")
t4.grid(row="4",column="2")
b1.grid(row="5",column="1")
b2.grid(row="5",column="2")
window.mainloop()

##########################################
# p28.py
'''Design a Tkinter interface to perform the following operations on a database by considering the
table
Employee (SSN:Int, Fname: String, LName: String, Age: Int,Place: String, Salary: Int).
Display the success and failure message using MessageBox
a. Insert employee details
b. Delete the details of employee whose id = 1001 and place = “XYZ”
c. Update the employee details'''
import pymysql
from tkinter import *
from tkinter import messagebox
mydb=pymysql.connect(host="localhost",user="root",passwd="root",database="Emp")
mycursor=mydb.cursor()

def processInsert():
    sql="INSERT INTO Employee VALUES ('"+var1.get()+"','"+var2.get()+"','"+var3.get()
+"','"+str(var4.get())+"','"+var5.get()+"','"+str(var6.get())+"')"
    try:
        mycursor.execute(sql)
        mydb.commit()
        messagebox.showinfo("Insertion","Insertion successful")
    except:
        messagebox.showerror("Insertion","Insertion Unsuccessful")
        mydb.rollback()

def processDelete():
    sql="DELETE FROM Employee WHERE SSN='"+var1.get()+"'"
    try:
        mycursor.execute(sql)
        mydb.commit()
        messagebox.showinfo("Deletion","Deletion successful")
    except:
        messagebox.showerror("Deletion","Deletion Unsuccessful")
        mydb.rollback()

def processUpdate():
```

```
    sql="UPDATE Employee SET Fname='"+var2.get()+"', Lname='"+var3.get()+"', Age='"+str(var4.get())
+"', Place='"+var5.get()+"', Salary='"+str(var6.get())+"' WHERE SSN='"+var1.get()+"'"
    try:
        print(sql)
        mycursor.execute(sql)
        mydb.commit()
        messagebox.showinfo("Updation","Updation successful")
    except:
        messagebox.showerror("Updation","Updation Unsuccessful")
        mydb.rollback()

window=Tk()
window.title("Employee")
l1=Label(window,text="SSN")
l2=Label(window,text="First Name")
l3=Label(window,text="Last Name")
l4=Label(window,text="Age")
l5=Label(window,text="Place")
l6=Label(window,text="Salary")
var1=StringVar()
var2=StringVar()
var3=StringVar()
var4=IntVar()
var5=StringVar()
var6=IntVar()
t1=Entry(window,textvariable=var1)
t2=Entry(window,textvariable=var2)
t3=Entry(window,textvariable=var3)
t4=Entry(window,textvariable=var4)
t5=Entry(window,textvariable=var5)
t6=Entry(window,textvariable=var6)
b1=Button(window,text="Insert",command=processInsert)
b2=Button(window,text="Delete",command=processDelete)
b3=Button(window,text="Update",command=processUpdate)
l1.grid(row="1",column="1")
t1.grid(row="1",column="2")
l2.grid(row="2",column="1")
t2.grid(row="2",column="2")
l3.grid(row="3",column="1")
t3.grid(row="3",column="2")
l4.grid(row="4",column="1")
t4.grid(row="4",column="2")
l5.grid(row="5",column="1")
t5.grid(row="5",column="2")
l6.grid(row="6",column="1")
t6.grid(row="6",column="2")
b1.grid(row="7",column="1")
b2.grid(row="7",column="2")
b3.grid(row="8",column="1",columnspan="2")
window.mainloop()

##############################################
# p28_client.py
"""
Write a Client/Server Socket program to demonstrate the file transfer operation using Python
Programming.
"""
# client

import socket
import sys


HOST = socket.gethostname()
PORT = 9999

soc = socket.socket(socket.AF_INET,   socket.SOCK_STREAM)
soc.connect((HOST, PORT))
print("[+] Connected with Server")

# get file name to send
file_name = input("Enter file name to send : ")
# open file
try:
    file = open(file_name, "rb")
    # send file
    print("[+] Sending file...")
```

```
    data = file.read()
    soc.sendall(data)

    # close connection
    soc.close()
    file.close()
    print("[-] Disconnected")

except FileNotFoundError:
    print("Error : file not found")

##############################################
# p28_server.py
import socket
import sys


HOST = socket.gethostname()
PORT = 9999

soc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
soc.bind((HOST, PORT))
soc.listen(5)

print("Listening ...")

while True:
    conn, addr = soc.accept()
    print("[+] Client connected: ", addr)

    # content will be put in recv.txt
    file = open("recv.txt", "wb")
    while True:
        # get file bytes
        data = conn.recv(4096)
        if not data:
            break
        # write bytes on file
        file.write(data)
    file.close()
    print("[+] Download complete!")

    # close connection
    conn.close()
    print("[-] Client disconnected")

##############################################
# p28_client.py
"""
Write a client/server program where the client program takes the expression (n1 op n2 where n1 and
n2 are operands and op can be +,-,*,/ ) from the user and sends the expression to the server
program. The server program performs the specified operation and sends the result to the client
program and displays it on the user's console.
"""
# client

import socket

s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
host=socket.gethostname()
port=9999
s.connect((host,port))
msg="112/22"
s.send(msg.encode('ascii'))
ans=s.recv(1024)
print(ans.decode('ascii'))

##############################################
# p28_server.py
import socket

s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
host=socket.gethostname()
port=9999
s.bind((host,port))
s.listen(1)
```

```
while True:
    c,addr=s.accept()
    exp=str(c.recv(1024))
    print("Recieved: ",exp)
    n1=""
    n2=""
    for i in range(len(exp)):
        if exp[i].isdigit():
            n1=n1+exp[i]
        elif exp[i]=='+' or exp[i]=='-' or exp[i]=='*' or exp[i]=='/':
            break;
    n1=int(n1)
    op=exp[i]
    for j in range(i+1,len(exp)):
        if exp[j].isdigit():
            n2=n2+exp[j]
        else:
            break;
    n2=int(n2)
    if op=="+":
        res=n1+n2
    elif op=="-":
        res=n1-n2
    elif op=="*":
        res=n1*n2
    elif op=="/":
        res=n1//n2

    c.send(str(res).encode('ascii'))
    c.close()

##########################################
<html>
        <head>
        </head>
        <body>
                <form action="cgi-bin/p31.py" method="post" target="_blank">
                        <input type="checkbox" name="maths"> MATHS
                        <input type="checkbox" name="physics"> PHYSICS
                        <input type="submit" value="SUBMIT">
                </form>
        </body>
</html>
#!/usr/bin/python3
# p31.py
# Write a CGI script to demonstrate the concept of check button.

import cgi, cgitb
form = cgi.FieldStorage()
if form.getvalue('maths'):
        math_flag = "ON"
else:
        math_flag = "OFF"
if form.getvalue('physics'):
        physics_flag = "ON"
else:
        physics_flag = "OFF"

print ("Content-type:text/html")
print()
print ("<html>")
print ("<head>")
print ("<title>Checkbox -CGI Program</title>")
print ("</head>")
print ("<body>")
print ("<h2> CheckBox Maths is : %s</h2>" % math_flag)
print ("<h2> CheckBox Physics is : %s</h2>" % physics_flag)
print ("</body>")
print ("</html>")

##########################################
<html>
        <head>
        </head>
        <body>
                <form action="cgi-bin/p32.py" method="post" target="_blank">
                        <input type="radio" name="subject" value="MATHS"> MATHS
```

```
                        <input type="radio" name="subject" value="PHYSICS"> PHYSICS
                        <input type="submit" value="SUBMIT">
                </form>
        </body>
</html>
#!/usr/bin/python3
# p31.py
# Write a CGI script to demonstrate the concept of radio button.

import cgi, cgitb
form = cgi.FieldStorage()
if form.getvalue('subject'):
        subject = form.getvalue('subject')
else:
        subject = "Not set"

print ("Content-type:text/html")
print()
print ("<html>")
print ("<head>")
print ("<title>Radio-CGI Program</title>")
print ("</head>")
print ("<body>")
print ("<h2> Selected Subject is %s</h2>" % subject)
print ("</body>")
print ("</html>")

##############################################
```