COURSE OUTCOME 1

Date: 11-09-2025

Apply the fundamental concepts of Python programming, including input/output operations, operators, and various data types

1. Write a program to convert temperature in Celsius scale to Fahrenheit scale.

   F = (9/5) * C + 32 where F is the temperature in Fahrenheit and C is the temperature in Celsius scale.

PROGRAM

```
# Input temperature in Celsius
celsius = float(input("Enter temperature in Celsius: "))

# Conversion formula
fahrenheit = (9/5) * celsius + 32

# Display the result
print("Temperature in Fahrenheit:", fahrenheit)
```

OUTPUT

```
Enter temperature in Celsius: 25
Temperature in Fahrenheit: 77.0
```

2. Write a program to find the area and circumference of a circle. Use math module. Circumference=2*pi*r, Area=pi*r*r where r is the radius of the circle given as input

PROGRAM

```
import math
r=float(input("Enter the radius: "))
area=math.pi*(r*r)
print("Area of circle = %.2f"%area)
c=2*math.pi*r
print("Circumference of circle = %.2f"%c)
```

OUTPUT

```
Enter the radius: 4
Area of circle = 50.27
Circumference of circle = 25.13
```

3. Display future leap years from current year to a final year entered by user.

PROGRAM

```
from datetime import datetime
y1 = datetime.now().year
print("Current year= ",y1)
y2=int(input("Enter the final year: "))

print("Leap years from ",y1," to ",y2,":")
for i in range(y1,y2+1):
        if i%400==0 or i%4==0 and i%100!=0:
                print(i,end=" ")
```

OUTPUT

```
Current year=  2025
Enter the final year: 2040
Leap years from  2025  to  2040 :
2028 2032 2036 2040
```

4. Consider the list [12, 45, 7, 34, 27, 52, 20].
    a. Find the sum of elements in the list using sum().
    b. Display the largest and smallest element in the list using max() and min() .
    c. Sort the list using sorted().

PROGRAM

```
a=[12, 45, 7, 34, 27, 52, 20]
print("List= ",a)
print("Sum of the elements in the list = ",sum(a))
print("Largest element in the list = ",max(a))
print("smallest element in the list = ",min(a))
print("List after sorting = ",sorted(a))
```

OUTPUT

```
List=  [12, 45, 7, 34, 27, 52, 20]
Sum of the elements in the list =  197
Largest element in the list =  52
smallest element in the list =  7
List after sorting =  [7, 12, 20, 27, 34, 45, 52]
```

5. Create a list of colours from comma-separated colour names entered by user. Display first and last colours. Create another list of colours. Display all colours from color-list1 not contained in color-list2.

PROGRAM

```
print("Enter colour names separated by comma: ")
a=list(map(str,input().split(',')))
print("\nList = ",a)
print("\nFirst and Last colors in the list are ",a[0], " and ", a[-1])
print("\nList 1 = ",a)
b=["red","blue","green","black"]
print("List 2 = ",b)
print("\nColors from list 1 which is not contained in list 2 =")
for i in a:
    if i not in b:
        print(i,end=" ")
```

OUTPUT

```
Enter colour names separated by comma:
black,blue,white,yellow,green
List =  ['black', 'blue', 'white', 'yellow', 'green']
First and  Last colors in the list are black and green
List 1 =  ['black', 'blue', 'white', 'yellow', 'green']
List 2 =  ['red', 'blue', 'green', 'black']
Colors from list 1 which is not contained in list 2 = white yellow
```

6. Enter 2 lists of integers. Check (a) Whether lists are of same length (b) Whether list sums to same value (c) Whether any value occur in both

PROGRAM

```
a=list(map(int,input("Enter list 1: ").split()))
b=list(map(int,input("Enter list 2: ").split()))
if len(a) == len(b):
    print("Lists are of same length")
else:
    print("Lists are not of same length")
if sum(a) == sum(b):
    print("Lists sum are same")
else:
    print("Lists sum are not same")
for i in a:
    if i in b:
        print(i, " is present in both")
```

OUTPUT

```
Enter list 1: 10 20 30 40 50
Enter list 2: 50 60 70 80 90
Lists are of same length
Lists sum are not same
50 is present in both
```

7. Store a list of first names. Display the number of elements in the list. Count the occurrences of ‘a’ within the list.

PROGRAM

```
a=input(“Enter the names: ”).split()
print(“Number of elements in the list = ”,len(a))
count=0
for i in a:
    for j in i:
        if ‘a’ in j:
            count+=1
print(“Occurrences of ‘a’ is ”,count)
```

OUTPUT

```
Enter the names: Rowan Rhysand Raven
Number of elements in the list = 3
Occurrences of ‘a’ is 3
```

8. Create a dictionary of 5 student names with their marks.
   “Anu": 85, "Ravi": 90, "Minu": 75, "Sara": 88, "Tom": 94
   a. Display the dictionary
   b. Access the value of the key Sara
   c. Add a new student "Kiran": 80 to the dictionary and update "Minu"’s marks to 82.
   d. Delete "Tom" from the dictionary.
   e. Display the names of all students in the dictionary.
   f. Print all keys and values separately.
   g. Sort the dictionary by values in ascending order
   h. Sort the dictionary by keys in ascending order
   i. Display the student with highest marks.
   j. Create another dictionary “Anusha": 85, "Navin": 70, "Libin": 75. Merge the two dictionaries

PROGRAM

```
a={"Anu":85, "Ravi":90,"Minu":75,"Sara":88,"Tom":94}

print("Dictionary = ",a)

print("Value of key Sara = ",a["Sara"])

a.update({"Kiran":80})

a["Minu"]=82

print("Dictionary after adding a new student Kiran and updated Minu's Marks : \n",a)

a.pop("Tom")

print("Dictionary after deleting Tom : \n",a)

print("Name of all students: ",a.keys())

print("Keys =",a.keys())

print("Values of keys = ",a.values())

print("Values of dictionary in ascending order = ",sorted(a.values()))

print("Keys of dictionary in ascending order = \n",sorted(a.keys()))
```

```
x=max(a, key=a.get)
print("Student with highest mark = ",x)
print("First Dictionary= ",a)
b={"Anusha":85,"Navin":70,"Libin":75}
print("Second Dictionary= ",b)
a.update(b)
print("\nAfter merging first and second dictionary: \n",a)
```

OUTPUT

```
Dictionary =  {'Anu': 85, 'Ravi': 90, 'Minu': 75, 'Sara': 88, 'Tom': 94}
Value of key Sara =  88
Dictionary after adding a new student Kiran and updated Minu's Marks :
 {'Anu': 85, 'Ravi': 90, 'Minu': 82, 'Sara': 88, 'Tom': 94, 'Kiran': 80}
Dictionary after deleting Tom :
 {'Anu': 85, 'Ravi': 90, 'Minu': 82, 'Sara': 88, 'Kiran': 80}
Name of all students:  dict_keys(['Anu', 'Ravi', 'Minu', 'Sara', 'Kiran'])
Keys = dict_keys(['Anu', 'Ravi', 'Minu', 'Sara', 'Kiran'])
Values of keys =  dict_values([85, 90, 82, 88, 80])
Values of dictionary in ascending order =  [80, 82, 85, 88, 90]
Keys of dictionary in ascending order =
 ['Anu', 'Kiran', 'Minu', 'Ravi', 'Sara']
Student with highest mark =  Ravi
First Dictionary=  {'Anu': 85, 'Ravi': 90, 'Minu': 82, 'Sara': 88, 'Kiran': 80}
Second Dictionary=  {'Anusha': 85, 'Navin': 70, 'Libin': 75}
After merging first and second dictionary:
```

{'Anu': 85, 'Ravi': 90, 'Minu': 82, 'Sara': 88, 'Kiran': 80, 'Anusha': 85, 'Navin': 70, 'Libin': 75

9. Accept a filename from the user and print the extension of the filename.

PROGRAM

```
file= input('Enter Filename: ')
ext = file.split('.')[-1]
print("Extension is ", ext)
```

OUTPUT

```
Enter Filename: hello.txt
Extension is  txt
```

10. Count the occurrences of each word in a line of text.

PROGRAM

```
line = "Be who you are and say what you feel, because those who mind don't matter, and those who matter don't mind."
print(line,"\n")
words = line.lower().split()
w_counts = {}
for word in words:
    word = word.strip(".,!?;:'\"()[]{}")
    if word:
        w_counts[word] = w_counts.get(word, 0) + 1
for word, count in w_counts.items():
    print(f"'{word}': {count}")
```

OUTPUT

```
Every story I create, creates me. I write to create myself.
'every': 1
'story': 1
'i': 2
'create': 2
'creates': 1
'me': 1
'write': 1
'to': 1
'myself': 1
```

11. Prompt the user for a list of integers. For all values greater than 100, store ‘over’ instead.

PROGRAM

```
mylist = input("Enter a list of integers separated by spaces: ")
n = [int(x) for x in mylist.split()]

for i in range(len(n)):

        if n[i] > 100:

                n[i] = 'over'

print(n)
```

OUTPUT

```
Enter a list of integers separated by spaces: 50 80 150 200 30

[50, 80, 'over', 'over', 30]
```

12. Create a list of first names accepted from the user. Display the names in reverse order. Display the longest name.

PROGRAM

```
mylist = input("Enter first names separated by commas: ")
names = [name.strip() for name in mylist.split(',')]
rev = names[::-1]
lname = max(names, key=len)
print("Names in reverse order:", rev)
print("The longest name is:",lname)
```

OUTPUT

```
Enter first names separated by commas: Rhysand,Cassian,Azriel
Names in reverse order: ['Azriel', 'Cassian', 'Rhysand']
The longest name is: Rhysand
```

13. Get a string from an input string where all occurrences of first character replaced with '$', except first character. [eg: onion -> oni$n]

PROGRAM

```
s = input("Enter a string: ")
if len(s) > 0:
    first_char = s[0]
    result = first_char + s[1:].replace(first_char, '$')
else:
    result = s
print(result)
```

OUTPUT

```
Enter a string: onion
oni$n
```

14. Create a string from given string where first and last characters exchanged. [eg: python -> nythop].

PROGRAM

```
s = input("Enter a string: ")
if len(s) > 1:
        string = s[-1] + s[1:-1] + s[0]
else:
        string = s
print(string)
```

OUTPUT

```
Enter a string: python
nythop
```

15. From a list of integers, create a list removing even numbers.

PROGRAM

```
n = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
odd = [num for num in n if num % 2 != 0]
print("List= ",n)
print("Numbers after removing even numbers: ",odd)
```

OUTPUT

```
List=  [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Numbers after removing even numbers:  [1, 3, 5, 7, 9]
```

COURSE OUTCOME 2

Date: 13-10-2025

Implement decision making, looping constructs and functions

1. Find biggest and smallest of 3 numbers entered. Use if structure.

PROGRAM

```
a=int(input("Enter the first number: "))
b=int(input("Enter the second number: "))
c=int(input("Enter the third number: "))
l=a
if b>l:
        l=b
if c<l:
        l=c
print("Largest= ",l)
s=a
if b<s:
        s=b
if c<s:
        s=c
print("Smallest= ",s)
```

OUTPUT

Enter the first number: 30

Enter the second number: 10

Enter the third number: 20

Largest= 30

Smallest= 10

2. Generate first n numbers in the Fibonacci series using a user defined function.

PROGRAM

```
def fib(n):
    mylist = []
    a, b = 0, 1
    for _ in range(n):
        mylist.append(a)
        a, b = b, a + b
    return mylist
n = int(input("Enter the number of elements in Fibonacci series: "))
f = fib(n)
print("Fibonacci series:", f)
```

OUTPUT

```
Enter the number of elements in Fibonacci series: 8
Fibonacci series: [0, 1, 1, 2, 3, 5, 8, 13]
```

3. Program to find the factorial of a number using recursion.

PROGRAM

```
def fact(n):
    if n==0 or n==1:
        return 1
    else:
        while n>1:
        return n*fact(n-1)
n=int(input("Enter a number: "))
factorial=fact(n)
print("Factorial= ",factorial)
```

OUTPUT

```
Enter a number: 4
Factorial= 24
```

4. Display the given pyramid with step number accepted from user.
   Eg: N=4
   1
   2 4
   3 6 9
   4 8 12 16

PROGRAM

```
n=int(input("Enter a number: "))
for i in range(1, n + 1):
    for j in range(1, i + 1):
        print(i * j, end=" ")
    print()
```

OUTPUT

```
Enter a number: 5
1
2 4
3 6 9
4 8 12 16
5 10 15 20 25
```

5. Count the occurrence of characters (character frequency) in a string.

PROGRAM

```
s = input("Enter a string: ")
f = {}
for char in s:
    if char in f:
        f[char] += 1
    else:
        f[char] = 1
print(f)
```

OUTPUT

```
Enter a string: vengeance
{'v': 1, 'e': 3, 'n': 2, 'g': 1, 'a': 1, 'c': 1}
```

6. Add ‘ing’ at the end of a given string. If it already ends with ‘ing’, then add ‘ly’

PROGRAM

```
s = input("Enter a string: ")
if s.endswith('ing'):
        print(s + 'ly')
else:
        print(s + 'ing')
```

OUTPUT

```
Enter a string: read
reading

Enter a string: reading
readingly
```

7. Accept a list of words and return length of longest word.

PROGRAM

```
words = input("Enter words separated by spaces: ").split()
length = 0
for i in words:
    if len(i) > length:
        length = len(i)
print("Length of the longest word:",length)
```

OUTPUT

```
Enter words separated by spaces: vengeance serendipity promise
Length of the longest word: 11
```

8. Construct following pattern using nested loop

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
```

PROGRAM

```
n = 5
for i in range(1, n + 1):
        for j in range(i):
                print("*", end=" ")
        print()
for i in range(n - 1, 0, -1):
```

```
        for j in range(i):
            print("*", end=" ")
        print()
```

OUTPUT

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
```

9. Write lambda functions to find area of square, rectangle and triangle.

PROGRAM

```
square = lambda s: s ** 2
rectangle = lambda l, b: l * b
triangle = lambda b, h: 0.5 * b * h
print("Area of square with side = 4: ", square(4))
print("Area of rectangle with length=5 and breadth=3: ", rectangle(5, 3))
print("Area of triangle with base=6 and height=4: ", triangle(6,4))
```

OUTPUT

```
Area of square with side = 4: 16
Area of rectangle with length=5 and breadth=3: 15
Area of triangle with base=6 and height=4: 12.0
```

10. Use List comprehensions:

(a) Generate positive list of numbers from a given list of integers

(b) Square of N numbers

(c) Form a list of vowels selected from a given word

(d) List ordinal value of each element of a word (Hint: use ord() to get ordinal values)

PROGRAM

```
numbers = [-3, 0, 4, -1, 7, 2]
pos = [n for n in numbers if n > 0]
sqrt = [n * n for n in numbers]


word = "Promise"
vowels = "aeiouAEIOU"
v = [ch for ch in word if ch in vowels]
```

```
ord_val = [ord(ch) for ch in word]

print("List: ",numbers)
print("String: ",word)
print("\n")
print("Positive numbers from list:", pos)
print("Squares of numbers in list:", sqrt)
print("Vowels in the string:", v)
print("Ordinal value of each element of a word :",ord_val)
```

OUTPUT

```
List:  [-3, 0, 4, -1, 7, 2]
String:  Promise

Positive numbers from list: [4, 7, 2]
Squares of numbers in list: [9, 0, 16, 1, 49, 4]
Vowels in the string: ['o', 'i', 'e']
Ordinal value of each element of a word : [80, 114, 111, 109, 105, 115, 101]
```

COURSE OUTCOME 3

Date:

Apply the concepts of modules and packages to design built in and user defined packages.

1. Create a package graphics with modules rectangle, circle and sub-package 3D-graphics with modules cuboid and sphere. Include methods to find area and perimeter of respective figures in each module. Write a program that finds area and perimeter of figures by different importing statements.

PROGRAM

main.py

```
from graphics import rectangle,circle

from graphics.three_d import cuboid,sphere

length=5

width=3

print("Area of a rectangle= ",rectangle.area(length,width))

print("Perimeter of a rectangle= ",rectangle.perimeter(length,width))

radius=4

print("Area of a circle= ",circle.area(radius))

print("Perimeter of a circle= ",circle.perimeter(radius))

#using cuboid module from sub package "three_d" graphics

c_length=3

c_width=2

c_height=4
```

```
print("Surface area of cuboid=
",cuboid.surface_area(c_length,c_width,c_height))

print("Volume of cuboid= ",cuboid.volume(c_length,c_width,c_height))

#using sphere module from sub package "three_d" graphics

s_radius=2

print("Surface area of sphere= ",sphere.surface_area(s_radius))

print("Volume of sphere= ",sphere.volume(s_radius))
```

graphics

```
#graphics is a package that includes the methods circle and rectangle
```

circle.py

```
import math

def area(radius):
   return math.pi*radius*radius


def perimeter(radius):
   return 2*math.pi*radius
```

rectangle.py

```
def area(length,width):
   return length*width

def perimeter(length,width):
```

```
    return 2*(length+width)
```

three_d

#three_d is a sub package of graphics that includes the methods cuboid and sphere

cuboid.py

```
    def surface_area(l,w,h):
        return 2*(l*w+w*h+h*l)
    def volume(l,w,h):
        return l*w*h
```

sphere.py

```
    import math
    def surface_area(r):
        return 4*math.pi*r*r
    def volume(r):
        return (4/3)*math.pi*r*r*r
```

OUTPUT

```
    Area of a rectangle=  15
    Perimeter of a rectangle=  16
    Area of a circle=  50.26548245743669
    Perimeter of a circle=  25.132741228718345
    Surface area of cuboid=  52
    Volume of cuboid=  24
```

Surface area of sphere= 50.26548245743669

Volume of sphere= 33.510321638291124

2. Create a user-defined module to perform basic arithmetic operations. Import it into another program to perform arithmetic operations on two numbers given as input.

PROGRAM

main.py

```
import operations

a=10

b=20

print(a," + ",b," = ",operations.add_fn(a,b))

print(a," - ",b," = ",operations.sub_fn(a,b))

print(a," * ",b," = ",operations.mul_fn(a,b))

print(a," / ",b," = ",operations.div_fn(a,b))
```

operations.py

```
def add_fn(x,y):
   return x+y

def sub_fn(x,y):
   return x-y

def mul_fn(x,y):
   return x*y

def div_fn(x,y):
```

```
return x/y
```

OUTPUT

```
10 + 20 = 30

10 - 20 = -10

10 * 20 = 200

10 / 20 = 0.5
```

COURSE OUTCOME 4

Date: 13/10/2025

Implement object oriented programming and exception handling.

1. Design a Python class Student with attributes name, roll number, marks and methods to display and update marks. Take user input to create a student object.

PROGRAM

```
class student:
    def __init__(self,name,rollno,marks):
        self.name=name
        self.rollno=rollno
        self.marks=marks
    def display(self):
        print(self.name, self.rollno,self.marks)
    def update_marks(self,new_marks):
        self.marks=new_marks
name=input("Enter student name: ")
rollno=input("Enter roll no: ")
marks=int(input("Enter marks: "))

#create student object
s1=student(name,rollno,marks)
#display the initial details
s1.display()
#update marks
new_marks=int(input("Enter new marks to update: "))
```

```
s1.update_marks(new_marks)
#display updated details
s1.display()
```

OUTPUT

```
Enter student name: yuki
Enter roll no: 10
Enter marks: 255
yuki 10 255
Enter new marks to update: 200
yuki 10 200
```

2. Create a class Student with attributes –Name and Marks of two subjects, each out of 100. Calculate the total marks and display it.

PROGRAM

```
class student:
    def __init__(self,name,mark1,mark2):
        self.name=name
        self.mark1=mark1
        self.mark2=mark2
    def total(self):
        return self.mark1+self.mark2


name=input("Enter student name: ")
mark1=int(input("Enter mark 1: "))
mark2=int(input("Enter mark 2: "))
s1=student(name,mark1,mark2)
print("Name of student= ",s1.name)
print("Total Marks= ",s1.total())
```

OUTPUT

```
Enter student name: Yuki
Enter mark 1: 65
Enter mark 2: 40
Name of student=  Yuki
Total Marks=  105
```

3. Create a Python class named Student with two data members representing marks in two subjects (each out of 100). Define a method to display the total marks of

a student. Use operator overloading (+) to add the total marks of two Student objects and display the combined total. Accept marks as user input.

PROGRAM

```
class student:
    def __init__(self,m1,m2):
        self.m1=m1
        self.m2=m2
    def total(self):
        return self.m1+self.m2

    #operator overloading for "+"
    def __add__(self,other):
        total1=self.total()
        total2=other.total()
        return total1+total2

#main program
print("--------Marks of the first student-------- ")
m11=int(input("Enter the marks for the first subject: "))
m12=int(input("Enter the marks for the second subject: "))

print("\n--------Marks of the second student-------- ")
m21=int(input("Enter the marks for the first subject: "))
m22=int(input("Enter the marks for the second subject: "))
```

```
s1=student(m11,m12)
s2=student(m21,m22)

#calculate individual totals
print("\nTotal marks of student 1= ",s1.total())
print("Total marks of student 2= ",s2.total())

#using overloaded "+" operator
ctotal=s1+s2
print("\nCombined total of both students: ",ctotal)
```

OUTPUT

```
--------Marks of the first student--------
Enter the marks for the first subject: 80
Enter the marks for the second subject: 60

--------Marks of the second student--------
Enter the marks for the first subject: 100
Enter the marks for the second subject: 50

Total marks of student 1=  140
Total marks of student 2=  150

Combined total of both students:  290
```

4. Create a class Rectangle with attributes length and width. Overload < operator to compare area of two rectangles.

PROGRAM

```
class rectangle:
    def __init__(self,l,b):
        self.l=l
        self.b=b
    def area(self):
        return self.l*self.b

    #operator overloading for "<"
    def __lt__(self,other):
        return self.area()<other.area()

print("----------Rectangle 1----------")
l1=int(input("Enter the length: "))
w1=int(input("Enter the breadth: "))

print("\n----------Rectangle 2----------")
l2=int(input("Enter the length: "))
w2=int(input("Enter the breadth: "))

r1=rectangle(l1,w1)
r2=rectangle(l2,w2)
```

```
#using overloaded "<" operator
if r1<r2:
   print("\nArea of rectangle 2 is greater")
else:
   print("\nArea of rectangle 1 is greater")
```

OUTPUT

```
----------Rectangle 1----------
Enter the length: 5
Enter the breadth: 3

----------Rectangle 2----------
Enter the length: 10
Enter the breadth: 3

Area of rectangle 2 is greater
```

5. Create a Python program to demonstrate inheritance. Define a parent class Student that stores the name of a student and display it using a method. Define a child class MCAStudent that inherits from Student and add an additional data member for the semester. Use the super() function to call the parent class constructor. Display the student's name and semester using appropriate methods. Accept the values through user input.

PROGRAM

```
class student:
    def __init__(self,name):
        self.name=name
    def display(self):
        print("Name= ",self.name)


#child class inheriting from parent class
class MCAStudent(student):
    def __init__(self,name,sem):
        #call parent class constructor using super()
        super().__init__(name)
        self.sem=sem
    def show_details(self):
        self.display()
        print("Semester= ",self.sem)


#Main program
name=input("Enter the student name: ")
sem=input("Enter the semester: ")
```

```
#create an object of child class
s=MCAStudent(name,sem)
print("Student details are: ")
s.show_details()
```

OUTPUT

```
Enter the student name: Yuki
Enter the semester: 5
Student details are:
Name=  Yuki
Semester=  5
```

COURSE OUTCOME 5

DATE: 05/11/2025

1. Write a program to create a file named student.txt and store student details such as roll number, name, and age.

PROGRAM

2. Write a program to read a text file line by line and store it into a list.

3. Write a program to read each row from a CSV file with fields roll number, name, age, course and display a list of strings.

PROGRAM

name.csv

```
1,Rhysand,22,MCA
2,Azriel,21,MCA
3,Cassian,20,MCA
4,Rowan,22,MCA
5,Aelin,20,MCA
```

program3.py

```
import csv
with open("name.csv",'r') as f:
    csvr=csv.reader(f)
    for row in csvr:
        print(row)
```

OUTPUT

```
['1', 'Rhysand', '22', 'MCA']
['2', 'Azriel', '21', 'MCA ']
['3', 'Cassian', '20', 'MCA']
['4', 'Rowan', '22', 'MCA']
['5', 'Aelin', '20', 'MCA']
```

4. Write a program to read specific columns of a CSV file book with fields book number, title, author, price. Display specific columns, book number, title and price.

PROGRAM

book.csv

```
01,Seven Year Slip,Ashley Poston,650
02,Throne of Glass,Sarah J.Maas,850
03,Empire of Storms,Sarah J.Maas,900
04,All Rhodes Leads Here,Marrianna Zapta,600
05,The Right Move,Liz Tomforde,450
```

book.py

```
import csv
with open("book.csv","r") as f:
    csvr=csv.reader(f)
    print("The contents of files are: ")
    for row in csvr:
        print(row)
f.close()
with open("program4.csv","r") as f:
    col=csv.reader(f)
    print("\nThe specific columns are: ")
    for i in col:
        print(i[0],i[1],i[3])
    f.close()
```

OUTPUT

The contents of files are:

['01', 'Seven Year Slip', 'Ashley Poston', '650']

['02', 'Throne of Glass', 'Sarah J.Maas', '850']

['03', 'Empire of Storms', 'Sarah J.Maas', '900']

['04', 'All Rhodes Leads Here', 'Marrianna Zapta', '600']

['05', 'The Right Move', 'Liz Tomforde', '450']

The specific columns are:

01 Seven Year Slip 650

02 Throne of Glass 850

03 Empire of Storms 900

04 All Rhodes Leads Here 600

05 The Right Move 450

5. Using Regular Expression, search for a particular word (given as input) in a text file. Count the number of occurrences of the word in the file.

PROGRAM

demo.txt

Twinkle, twinkle, little star,

How I wonder what you are!

Up above the world so high,

Like a diamond in the sky.

Twinkle, twinkle, little star,

How I wonder what you are!

main.py

```
import re
word = input("Enter the word to search: ")
with open("demo.txt", "r") as f:
text = f.read()
pattern = r"\b" + re.escape(word) + r"\b"
matches = re.findall(pattern, text, re.IGNORECASE)
print(f"Occurrences of '{word}':", len(matches))
```

OUTPUT

```
Enter the word to search: twinkle
Occurrences of 'twinkle': 4
```