COURSE OUTCOME 1

Date: 03-10-2025

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

```
c = float(input("Enter the temperature in Celsius : "))
f = (9/5)*c+32
print(f"Temperature in Fahrenheit is : {f}")
```

OUTPUT

```
Enter the temperature in Celsius : 55.2
Temperature in Fahrenheit is : 131.36
```

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

OUTPUT

```
Enter the radius : 5.2
Area of the circle is : 84.94866535306801
Circumference of the circle is : 32.67256359733385
```

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

PROGRAM

```
from datetime import datetime, date
fy = int(input("Enter the final year : "))
cy = datetime.now().year
for i in range(cy,fy):
   if (i%4 == 0 and i%100 != 0) or(i%400 == 0):
      print(f"{i} is a leap year")
```

OUTPUT

```
Enter the final year : 2050
2028 is a leap year
2032 is a leap year
2036 is a leap year
2040 is a leap year
2044 is a leap year
2048 is a leap year
```

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

```
li = [12, 45, 7, 34, 27, 52, 20]
print(f"The List is : {li}")
print(f"Sum of numbers in the list : {sum(li)}")
print(f"Maximum number in the list : {max(li)}")
print(f"Minimum number in the list : {min(li)}")
li.sort()
print(f"Sorted list is : {li}")
```

OUTPUT

```
The List is : [12, 45, 7, 34, 27, 52, 20]
Sum of numbers in the list : 197
Maximum number in the list : 52
Minimum number in the list : 7
Sorted list is : [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

```
col1 = input("Enter colours (seperate by comma) : ").split(",")
print(f"Color 1 : {col1}")
print(f"First color in the list is : {col1[0]}")
print(f"last color in the list is : {col1[-1]}")
col2 = input("Enter colours (seperate by comma) : ").split(",")
print(f"Color 2 : {col2}")
col3 = []
for x in col1:
    if x not in col2:
        col3.append(x)
print(f"Colors only in color 1 : {col3}")
```

OUTPUT

```
Enter colours (seperate by comma) : red,green,blue,yellow,orange
Color 1 : ['red', 'green', 'blue', 'yellow', 'orange']
First color in the list is : red
last color in the list is : orange
Enter colours (seperate by comma) : blue,white,black,red,pink
Color 2 : ['blue', 'white', 'black', 'red', 'pink']
Colors only in color 1 : ['green', 'yellow', 'orange']
```

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

```
lis1 = list(map(int,input("Enter list 1 integer numbers :").split(' ')))
lis2 = list(map(int,input("Enter list 1 integer numbers :").split(' ')))
print("Length of list 1:",len(lis1))
print("Length of list 2:",len(lis2))
if len(lis1) == len(lis2):
   print(f"Both list are same length")
else:
   print(f"Both list are not same length!")

if sum(lis1) == sum(lis2):
   print(f"Both list sum are same value ({sum(lis1)})")
else:
print(f"Both list sum are not same value! list 1({sum(lis1)}),list
2({sum(lis2)})")

print("Number present in both list : ",end="")
for x in lis1:
   if x in lis2:
      print(f"{x} ",end="")
```

OUTPUT

```
Enter list 1 integer numbers :10 20 30 40 50 60
Enter list 1 integer numbers :60 50 40 30 20 10
Length of list 1: 6
Length of list 2: 6
Both list are same length
Both list sum are same value (210)
Number present in both list : 10 20 30 40 50 60
```

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

PROGRAM

```
names = ["mohammed","faseeh","johan","merin","iqbal"]
print(names)
print(f"Number of elements in the list : {len(names)}")
count = 0
for x in names:
    if 'a' in x:
        count = count+1
print(f"Number of occurrences of 'a' in the list : {count}")
```

OUTPUT

```
['mohammed', 'faseeh', 'johan', 'merin', 'iqbal']
Number of elements in the list : 5
Number of occurrences of 'a' in the list : 4
```

8. Create a dictionary of 5 student names with their marks.

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

PROGRAM

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

# a)Display the dictionary
print("Displaying the dictionary",student)

# b) Access the value of the key Sara
print("value of the key Sara",student["Sara"])

# c) Add a new student "Kiran": 80 to the dictionary and update "Minu"’s marks to 82.
student["Kiran"] = 80
student.update({"Minu" : 82})
print("Kiran added and minu updated : ",student)

# d)Delete "Tom" from the dictionary.
student.pop("Tom")

# e) Display the names of all students in the dictionary.
print("names of all students",list(student.keys()))

# f) Print all keys and values separately.
print("all keys and values separately")
```

```
print(student.keys())
print(student.values())

# g) Sort the dictionary by values in ascending order
print("values in ascending order",sorted(student.items()))

# h) Sort the dictionary by keys in ascending order
print("keys in ascending order",sorted(student.items(), key=lambda item:
item[1]))

# i) Display the student with highest marks.
print("highest marks",max(student, key=student.get))

# j) Create another dictionary “Anusha": 85, "Navin": 70, "Libin": 75. Merge
the two dictionaries
student1 = {
        "Anusha": 85,"Navin": 70,"Libin": 75
}
student.update(student1)
print("Merged dicyionary ",student))
```

OUTPUT

```
Displaying the dictionary {'Anu': 85, 'Ravi': 90, 'Minu': 75, 'Sara': 88, 'Tom':
94}
value of the key Sara 88
Kiran added and minu updated :  {'Anu': 85, 'Ravi': 90, 'Minu': 82, 'Sara': 88,
'Tom': 94, 'Kiran': 80}
names of all students ['Anu', 'Ravi', 'Minu', 'Sara', 'Kiran']
all keys and values separately
dict_keys(['Anu', 'Ravi', 'Minu', 'Sara', 'Kiran'])
dict_values([85, 90, 82, 88, 80])
values in ascending order [('Anu', 85), ('Kiran', 80), ('Minu', 82), ('Ravi', 90),
('Sara', 88)]
keys in ascending order [('Kiran', 80), ('Minu', 82), ('Anu', 85), ('Sara', 88),
('Ravi', 90)]
highest marks Ravi
Merged dicyionary  {'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

```
filename = input("Enter a filename : ")
ext = filename.split(".")

if len(ext) > 1:
        extension = ext[-1]
        print(f"The extension of the file is : {extension}")
else:
        print("The file does not have an extension ")
```

OUTPUT

```
Enter a filename : apple.pdf
The extension of the file is : pdf
```

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

PROGRAM

```
string=input("Enter string : ")
word=input("Enter word : ")
a=[]
count=0
a=string.split(" ")
for i in range(0,len(a)):
if(word==a[i]):
   count=count+1
print("Count of the word is : ")
print(count)
```

OUTPUT

```
Enter string : the is the of the
Enter word : the
Count of the word is : 3
```

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

PROGRAM

```
numbers = input("Enter the numbers : ")
num = list(map(int, numbers.split()))
num2 = []
for x in num:
        if x >100:
                num2.append("over")
        else:
                num2.append(x)
print(num2)
```

OUTPUT

```
Enter the numbers : 10 200 50 250 600
[10, 'over', 50, 'over', 'over']
```

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

PROGRAM

```
names = input("Enter first names separated by spaces: ").split()
print("List of names:", names)
print("Names in reverse order:", names[::-1])
longest_name = max(names, key=len)
print("The longest name is:", longest_name)
```

OUTPUT

```
Enter first names separated by spaces: mohammed nandana faseeh
List of names: ['mohammed', 'nandana', 'faseeh']
Names in reverse order: ['faseeh', 'nandana', 'mohammed']
The longest name is: mohammed
```

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

PROGRAM

```
text = input("Enter a word : ")
s = text[0]
text2 = text.replace(s,"$")
text2 = s + text2[1:]
print(text2)
```

OUTPUT

```
Enter a word : onion for all
oni$n f$r all
```

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

PROGRAM

```
str = input("Enter the string : ")
str2 = str[-1] + str[1:-1] + str[0]
print(str2)
```

OUTPUT

```
Enter the string : programming
grogramminp
```

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

PROGRAM

```
li = [1,20,50,61,32,33,98,96,95,67,31,51,83]
print("current list : ",li)
li2 = [x for x in li if x%2 != 0]
print("List after removing even number : ",li2)
```

OUTPUT

```
Original list: [10, 21, 4, 45, 66, 93, 1]
List after removing even numbers: [21, 45, 93, 1]
```

COURSE OUTCOME 2

Date: 03-10-2025

Implement decision making, looping constructs and functions

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

PROGRAM

```
a = int(input("Enter first number: "))
b = int(input("Enter second number: "))
c = int(input("Enter third number: "))
if a > b and a > c:
   biggest = a
elif b > c:
   biggest = b
else:
   biggest = c
if a < b and a < c:
   smallest = a
elif b < c:
   smallest = b
else:
   smallest = c
print("Biggest number:", biggest)
print("Smallest number:", smallest)
```

OUTPUT

```
Enter first number: 55
Enter second number: 26
Enter third number: 34
Biggest number: 55
Smallest number: 26
```

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

PROGRAM

```
def fibonacci(n):
    f1 = 0
    f2 = 1
    print("Fibonacci series:", f1, f2, end=" ")
    for i in range(3, n+1):
        f3 = f1 + f2
        print(f3, end=" ")
        f1 = f2
        f2 = f3

n = int(input("Enter how many Fibonacci numbers you want: "))

if n == 1:
    print("Fibonacci series: 0")
elif n == 2:
    print("Fibonacci series: 0 1")
else:
    fibonacci(n)
```

OUTPUT

```
Enter how many Fibonacci numbers you want : 10
Fibonacci series: 0 1 1 2 3 5 8 13 21 34
```

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

PROGRAM

```
def factorial(n):
    if n == 0 or n == 1:
    return 1
else:
    return n * factorial(n-1)

num = int(input("Enter a number: "))

print("Factorial of", num, "is", factorial(num))
```

OUTPUT

```
Enter a number: 6
Factorial of 6 is 720
```

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 number of steps: "))

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

OUTPUT

```
Enter number of steps: 4
1
2 4
3 6 9
4 8 12 16
```

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

PROGRAM

```
s = input("Enter a string: ")

freq = {}

for ch in s:
        if ch in freq:
                freq[ch] += 1
         else:
                 freq[ch] = 1
print("Character Frequency:")
 for ch in freq:
        print(ch, ":", freq[ch])
```

OUTPUT

```
Enter a string: programming
Character Frequency:
p : 1
r : 2
o : 1
g : 2
a : 1
m : 2
i : 1
n : 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"):
    s = s + "ly"
else:
    s = s + "ing"

print("Modified string:", s)
```

OUTPUT

```
Enter a string: read
Modified string: reading
```

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

PROGRAM

```
words = input("Enter words separated by space: ").split()
longest = 0
    for w in words:
        if len(w) > longest:
        longest = len(w)

print("Length of the longest word:", longest)
```

OUTPUT

```
Enter words separated by space: bootle book pen pencil
Length of the longest word: 6
```

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_area = lambda a: a * a
rectangle_area = lambda l, b: l * b
triangle_area = lambda b, h: 0.5 * b * h

a = float(input("Enter side of square: "))
l = float(input("Enter length of rectangle: "))
b = float(input("Enter breadth of rectangle: "))
tb = float(input("Enter base of triangle: "))
h = float(input("Enter height of triangle: "))

print("Area of Square =", square_area(a))
print("Area of Rectangle =", rectangle_area(l, b))
print("Area of Triangle =", triangle_area(tb, h))
```

OUTPUT

```
Enter side of square: 5
Enter length of rectangle: 6
Enter breadth of rectangle: 9
Enter base of triangle: 14
Enter height of triangle: 12
Area of Square = 25.0
Area of Rectangle = 54.0
Area of Triangle = 84.0
```

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

```
# Input list for positive numbers
nums = list(map(int, input("Enter integers separated by space: ").split()))
positive = [x for x in nums if x > 0]
print("Positive numbers:", positive)

# Square of N numbers
n = int(input("\nEnter N: "))
squares = [i*i for i in range(1, n+1)]
print("Squares:", squares)

# Vowels from a word
word = input("\nEnter a word: ")
vowels = [ch for ch in word if ch.lower() in 'aeiou']
print("Vowels:", vowels)

# Ordinal values of characters in word
ord_values = [ord(ch) for ch in word]
print("Ordinal values:", ord_values)
```

OUTPUT

```
Enter integers separated by space: 5 -8 6 0 9
Positive numbers: [5, 6, 9]

Enter N: 5
Squares: [1, 4, 9, 16, 25]

Enter a word: hello world
Vowels: ['e', 'o', 'o']
Ordinal values: [104, 101, 108, 108, 111, 32, 119, 111, 114, 108, 100]
```

COURSE OUTCOME 3

Date: 14-10-2025

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

graphics package

circle.py

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

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)
```

graphics/threed package

cuboid.py

```
def surfacearea(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 surfacearea(radius):
    return 4*math.pi*radius**2

def volume(radius):
    return (4/3)*math.pi*radius**3
```

main.py

```
from graphics import rectangle,circle
from graphics.threed 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 circle =",circle.area(radius))
```

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

#using cuboid module from subpackage 3D graphics
l=3
w=2
h=4
print("Surface area of cuboid=",cuboid.surfacearea(l,w,h))
print("Volume of cuboid=",cuboid.volume(l,w,h))

#using sphere module from subpackage 3D graphics
radius=2
print("Surface area of sphere=",sphere.surfacearea(radius))
print("Volume of sphere=",sphere.volume(radius))
```

OUTPUT

```
Area of a rectangle= 15
Perimeter of a rectangle= 16
Area of circle = 50.26548245743669
Perimeter of 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

operations.py

```
def sum(a,b):
    return a+b

def sub(a,b):
    return a-b

def mul(a,b):
    return a*b

def div(a,b):
    return a/b
```

main.py

```
import operations
a=int(input("Enter number 1: "))
b=int(input("Enter number 2: "))
print("Sum: ",operations.sum(a,b))
print("Sub: ",operations.sub(a,b))
print("Mul: ",operations.mul(a,b))
print("Div: ",operations.div(a,b))
```

OUTPUT

Enter number 1:20

Enter number 2:10

Sum: 30

Sub: 10

Mul: 200

Div: 2.0

COURSE OUTCOME 4

Date: 30-10-2025

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("Name: ", self.name)
        print("Rollno: ",self.rollno)
        print("Marks: ",self.marks)
    def updatemarks(self,newmarks):
        self.marks=newmarks
        name=input("Enter Student Name: ")
        rollno=int(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
newmarks=int(input("Enter new marks to update:"))

s1.updatemarks(newmarks)
#display updated details
s1.display()
```

OUTPUT

```
Enter Student Name: Anu
Enter Roll no: 1
Enter Marks: 20
Name: Anu
Rollno: 1
Marks: 20
Enter new marks to update:45
Name: Anu
Rollno: 1
Marks: 45
```

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 display(self):
        print("Name: ", self.name)
        print("Mark1: ", self.mark1)
        print("Mark2: ", self.mark2)
    def total(self):
        return self.mark1+self.mark2

name=input("Enter name: ")
mark1=int(input("Enter mark 1: "))
mark2=int(input("Enter mark 2: "))

#create student object
s1=student(name,mark1,mark2)
#display the initial details
s1.display()
#display total marks
print("Total marks: ",s1.total())
```

OUTPUT

```
Enter name: Achu
Enter mark 1: 56
Enter mark 2: 78
Name: Achu
Mark1: 56
Mark2: 78
Total marks: 134
```

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

print("Enter Marks of the s1\n")
m11=int(input("enter mark1: "))
m12=int(input("enter mark2: "))
print("Enter Marks of the s2")
m21=int(input("enter mark1: "))
m22=int(input("enter mark2: "))
s1=student(m11,m12)
s2=student(m21,m22)
```

```
print("Total Mark of s1: ",s1.total())
print("Total Mark of s2: ",s2.total())
#using overloaded + operator
ctotal=s1+s2
print("Combined total of both students: ",ctotal)
```

OUTPUT

```
Enter Marks of the s1
enter mark1: 23
enter mark2: 10
Enter Marks of the s2
enter mark1: 16
enter mark2: 10
Total Mark of s1: 33
Total Mark of s2: 26
Combined total of both students: 59
```

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

PROGRAM

```
class rectangle:
    def __init__(self,width,length):
        self.width=width
        self.length=length
    def area(self):
        return self.length*self.width
#operator overloading
    def __lt__(self,other):
        return self.area()<other.area()


print("Enter Measurement of Rectangle 1")
l1=int(input("Enter length: "))
w1=int(input("enter width: "))
print("Enter Measurement of Rectangle 2")
l2=int(input("Enter length: "))
w2=int(input("enter width: "))
r1=rectangle(l1,w1)
r2=rectangle(l2,w2)
print("Area of Rectangle 1:",r1.area())
print("Area of Rectangle 2:",r2.area())
if r1<r2:
    print("Area of Rectangle 1 is small")
```

```
else:
    print("Area of Rectangle 2 is small")
```

OUTPUT

```
Enter Measurement of Rectangle 1
Enter length: 3
enter width: 4
Enter Measurement of Rectangle 2
Enter length: 2
enter width: 2
Area of Rectangle 1: 12
Area of Rectangle 2: 4
Area of Rectangle 2 is small
```

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)
#inheritance
class mcastudent(student):
    def __init__(self,name,semester):
#call parent class constructor super
        super().__init__(name)
        self.semester=semester
    def showdetails(self):
        self.display()
        print("Semester: ",self.semester)

name=input("enter name: ")
semester=int(input("enter semester: "))
s=mcastudent(name,semester)
print("student details are: ")
s.showdetails()
```

OUTPUT

```
enter name: Fahis
enter semester: 2
student details are:
Name: ben
Semester: 2
```

COURSE OUTCOME 5

Date: 05-11-2025

Manage files and form regular expressions for effective search operations on strings and files.

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

PROGRAM

```
with open("student.txt",'w')as f:
f.write("roll number: 12 \n name: Ebin \n age: 21")
f=open("student.txt")
print(f.read())

f.close()
```

OUTPUT

```
roll number: 12
name: Ebin
age: 21
```

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

PROGRAM

```
filename = "text.txt"
lines_list = []
with open(filename, "r") as file:
    for line in file:
        lines_list.append(line.strip())
print("Contents of list:")
print(lines_list)
```

text.t

```
Apple
Banana
Cherry
```

OUTPUT

```
['Apple', 'Banana', 'Cherry']
```

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

Student.csv

```
rollno,name,age,course
101,fahis,21,btech
102,shafi,21,btech
103,iqbal,21,btech
104,faseeh,21,btech
105,johan,21,btech
```

Student.py

```
import csv
with open("student.csv") as f:
    print(f.read())
    csvr=csv.reader(f)
    for row in csvr:
    print(row)
```

OUTPUT

```
rollno,name,age,course
101,fahis,21,btech
102,shafi,21,btech
103,iqbal,21,btech
104,faseeh,21,btech
105,johan,21,btech
```

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.py

```
import csv
with open("book.csv","r") as f:
    csvr=csv.reader(f)
    print("Contents of file are \n")
    for row in csvr:
        print(row)
f=open("book.csv","r")
col=csv.reader(f)
print()
print("the spec columns are \n")
for i in col:
    print(i[0],i[1],i[3])
```

book.csv

```
book_number,title,author,price
1,Aadujeevitham,benyamin,200
2,The Goat Life,Prof.benyamin,300
3,Aadu Ka Jeevitham,benyamin ji,400
4,Aadu Dubai,Dubai.benyamin,500
5,aadujeevitham,benyamin,600
```

OUTPUT

Contents of file are

['book_number', 'title', 'author', 'price']

['1', 'Aadujeevitham', 'benyamin', '200']

['2', 'The Goat Life', 'Prof.benyamin', '300']

['3', 'Aadu Ka Jeevitham', 'benyamin ji', '400']

['4', 'Aadu Dubai', 'Dubai.benyamin', '500']

['5', 'aadujeevitham', 'benyamin', '600']

the spec columns are

book_number title price

1 Aadujeevitham 200

2 The Goat Life 300

3 Aadu Ka Jeevitham 400

4 Aadu Dubai 500

5 aadujeevitham 600

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

```
import re
word = input("Enter the word to search: ")
filename = "input.txt"
with open(filename, "r") as file:
    text = file.read()
pattern = r'\b' + re.escape(word) + r'\b'
matches = re.findall(pattern, text, flags=re.IGNORECASE)
count = len(matches)
print(f"The word '{word}' occurs {count} times in the file.")
```

input.txt

```
I love cats. Cats are looking cute.
CATS are lovely animals.
```

OUTPUT

```
Enter the word to search: cats
The word 'cats' occurs 3 times in the file.
```