MASTER OF COMPUTER APPLICATIONS

PRACTICAL RECORD WORK

ON

20MCA131 PROGRAMMING LAB

Submitted

By

…….

(Reg No : ……..)

DEPARTMENT OF COMPUTER APPLICATIONS

COLLEGE OF ENGINEERING VADAKARA

(CAPE – GOVT.OF KERALA)

DECEMBER 2021

DEPARTMENT OF COMPUTER APPLICATIONS

COLLEGE OF ENGINEERING VADAKARA

(CAPE – GOVT.OF KERALA)

CERTIFICATE

Certified that this is a bonafide record of the practical work on the course 20MCA131 PROGRAMMING LAB done by Mr/Mrs…… (Reg No:……) First semester MCA student of Department of Computer Applications at College of Engineering Vadakara in the practical fulfillment for the award of the degree of Master of Computer Applications (MCA) of APJ Abdul Kalam Technological University (KTU)

(Mrs. Bindu S Mony)

FACULTY-IN-CHARGE **HEAD OF THE DEPARTMENT**

DATE:

EXAMINERS:

INDEX			
SL.NO	**PROGRAMS**	**PAGE NO**	**REMARKS**
1	LEAP YEAR	5	
2	LIST COMPREHENSIONS	6	
3	COUNT OF WORDS	8	
4	GREATER THAN 100 STORE OVER	9	
5	OCCURRENCE OF ‘a’	10	
6	COMPARE 2 LISTS	11	
7	REPLACE WITH $	13	
8	CHARACTER EXCHANGE	14	
9	AREA OF CIRCLE	15	
10	BIGGEST OF 3 NUMBERS	16	
11	FILE EXTENSION	17	
12	FIRST AND LAST IN A LIST	18	
13	COMPUTE n+nn+nnn	19	
14	DIFFERENCE OF 2 SETS	20	
15	SWAPPING 2 STRINGS CHARACTER	21	
16	DICTIONARY SORTING	22	
17	MERGE 2 DICTIONARY	23	
18	GCD OF 2 NUMBERS	24	
19	REMOVE EVEN NUMBERS	25	
20	FACTORIAL	26	
21	FIBANOCCI SERIES	27	
22	SUM OF LIST	28	

CO1

Experiment No 1:

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

Source Code:

```
c = int(input("Enter the current year: "))
f = int(input("Enter the final year: "))
print("The future leap years are: ")
for x in range(c , f+1):
        if(x % 4 == 0 and x % 100 != 0 or x %400 == 0):
                print(x)
```

Output:

```
Enter the current year: 2021
Enter the final year: 2050
The future leap years are:
2024
2028
2032
2036
2040
2044
2048

Process finished with exit code 0
```

Experiment No 2:

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)

Source Code:

(a):

```
list = [-5, 10, 0, -14, 18, 17]
print(list)
newlist = [x for x in list if x > 0]
print("The positive numbers are: ", newlist)
```

(b):

```
n = int(input("Enter the number of numbers:"))
list = []
for i in range(n):
        a = int(input("Enter the number:"))
        list.append(a)
print(list)
list1 = [x ** 2 for x in list]
print(list1)
```

(c):

```
word = input("Enter a word: ")
vowels = [x for x in word if x in ['a', 'e', 'i', 'o', 'u']]
print("The vowels in the words are: ", vowels)
```

(d):

```
w = input("enter a word: ")
ord = [ord(x) for x in w]
print("the ordinal value is: ", ord)
```

Output:

(a):

```
[-5, 10, 0, -14, 18, 17]
The positive numbers are: [10, 18, 17]
```

(b):

```
Enter the number of numbers: 4
Enter the number: 2
Enter the number: 4
Enter the number: 6
Enter the number: 4
[2, 4, 6, 4]
[4, 16, 36, 16]
```

(c):

```
Enter a word: algorithm
The vowels in thw words are: ['a', 'o', 'i']
```

(d):

```
Enter a word: cpu
The ordinal value is: [99, 112, 117]

Process finished with exit code 0
```

Experiment No 3:

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

Source Code:

```
s = str(input("Enter line of text: "))
count = dict()
words = s.split()
for x in words:
        if x in count:
        count[x] += 1
else:
        count[x] = 1
print(count)
```

Output:

```
Enter line of text: this is an example and is easy
{'this': 1, 'is': 2, 'an': 1, 'example': 1, 'and': 1, 'easy': 1}

Process finished with exit code 0
```

Experiment No 4:

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

Source Code:

```
n = int(input("Enter the number of numbers: "))
list = []
for x in range(n):
        x = int(input("Enter the number: "))
        if x < 100:
                list.append(x)
        else:
                list.append('over')
print(list)
```

Output:

```
Enter the number of numbers: 4
Enter the number: 100
Enter the number: 150
Enter the number: 50
Enter the number: 40
['over', 'over', 50, 40]

Process finished with exit code 0
```

Experiment No 5:

Store a list of first names. Count the occurrences of ‘a’ within the list.

Source Code:

```
n = input("Enter the limit: ")
li = []
c = 0
for i in range(n):
    x = input("Enter names: ")
    li.append(x)
for i in li:
    for j in i:
        if 'a' in j:
            c = c + 1
print("The occurence of a is: ", c)
```

Output:

```
Enter the limit: 2
Enter names: abhi
Enter names: savad
The occurance of a is: 3

Process finished with exit code 0
```

Experiment No 6:

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

Source Code:

```
list1=[1, 2, 3, 4, 5]
list2=[2, 4, 6, 8]
print("Elements in list one is: ", list1)
print("Elements in list two is: ", list2)
s1 = 0
s2 = 0
if(len(list1) == len(list2)):
        print("List have same length")
else:
        print("List have different length")
for i in range(len(list1)):
        s1 = s1 + list1[i]
        print("The sum of first list is", s1)
for i in range(len(list2)):
        s2 = s2 + list2[i]
        print("The sum of list2 is", s2)
if(s1 == s2):
        print("Sum of lists is same")
else:
        print("The sum of lists is different")
print("The common element in list is: ")
for i in list1:
        for j in list2:
                if i == j:
                        print(i)
```

Output:

Elements in list one is: [1, 2, 3, 4, 5]

Elements in list two is: [2, 4, 6, 8]

List have different length

The sum of first list is 15

The sum of list2 is 20

The sum of lists is different

The common element in list is:

2

4

Process finished with exit code 0

Experiment No 7:

Get a string from an input string where all occurrences of first character replaced with '$', except first character.

[eg: onion -> oni$n]

Source Code:

```
a = input("Enter a string: ")
b = a[0]
s = a[1:len(a)]
for i in range(len(s)):
        if s[i] == a[0]:
                b = b + "$"
        else:
                b = b + s[i]
print(b)
```

Output:

```
Enter a string: onion
oni$n

Process finished with exit code 0
```

Experiment No 8:

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

Source Code:

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

Output:

```
Enter a string: python
nythop

Process finished with exit code 0
```

Experiment No 9:

Accept the radius from user and find area of circle.

Source Code:

```
r = int(input("Enter the radius: "))
area = 3.14 * r * r
print("The area of circle is: ", area)
```

Output:

```
Enter the radius: 3
The area of circle is: 28.259999999999998

Process finished with exit code 0
```

Experiment No 10:

Find biggest of 3 numbers entered.

Source Code:

```
print("Enter 3 numbers: ")
a = int(input())
b = int(input())
c = int(input())
if a > b and a > c:
        print("Largest number is : ", a)
elif b > a and b > c:
        print("Largest number is : ", b)
else:
        print("Largest number is : ", c)
```

Output:

```
Enter 3 numbers: 10
15
8
Largest number is: 15

Process finished with exit code 0
```

Experiment No 11:

Accept a file name from user and print extension of that.

Source code:

```
file = input("Enter a file name with extension: ")
print("Extension of the file is: ", file.split(".")[-1])
```

Output:

```
Enter a file name with extension: program.txt
Extension of the file is: txt

Process finished with exit code 0
```

Experiment No 12:

Create a list of colors from comma-separated color names entered by user. Display first and last colors.

Source Code:

```
n = int(input("Enter the no of colours: "))
list = []
for i in range(n):
        a = input("Enter the colour: ")
        list.append(a)
print(list)
print("The first clour is:", list[0])
print("The last colour is:", list[-1])
```

Output:

```
Enter the no of colours: 3
Enter the colour: red
Enter the colour: green
Enter the colour: yellow
['red', 'green', 'yellow']
The first clour is: red
The last colour is: yellow

Process finished with exit code 0
```

Experiment No 13:

Accept an integer n and compute n+nn+nnn.

Source Code:

```
n = int(input("enter a number: "))
nn = n * 10 + n
nnn = n * 100 + nn
print("The value of n+nn+nnn is: ", n+nn+nnn)
```

Output:

```
Enter a number: 5
The value of n+nn+nnn is: 615

Process finished with exit code 0
```

Experiment No 14:

Print out all colors from color-list1 not contained in color-list2.

Source Code:

```
col1 = ['red', 'blue', 'yellow']
col2 = ['green', 'blue']
print(col1)
print(col2)
for i in range(len(col1)):
        if col1[i] not in col2:
                print("The colour is:", col1[i])
```

Output:

```
['red', 'blue', 'yellow']
['green', 'blue']
the colour is: red
the colour is: yellow

Process finished with exit code 0
```

Experiment No 15:

Create a single string separated with space from two strings by swapping the character at position 1.

Source Code:

```
a = input("Enter first string: ")
b = input("Enter second string: ")
s1 = b[0] + a[1:]
s2 = a[0] + b[1:]
print("The swapped string is:", s1+ " +s2)
```

Output:

```
Enter first string: hello world
Enter second string: good morning
The swapped string is: gello worldhood morning

Process finished with exit code 0
```

Experiment No 16:

Sort dictionary in ascending and descending order.

Source Code:

```
dic = {'savad':5, 'abhi':20, 'nandu':17}
l = list(dic.items())
l.sort()
d = dict(l)
print("Ascending is: ", d)
l = list(dic.items())
l.sort(reverse = True)
d = dict(l)
print("Descending is: ", d)
```

Output:

```
Ascending is:  {'abhi': 20, 'nandu': 17, 'savad': 5}
Descending is:  {'savad': 5, 'nandu': 17, 'abhi': 20}

Process finished with exit code 0
```

Experiment No 17:

Merge two dictionaries.

Source Code:

```
dict1 = {"Abhi":1, "Savad":18, "Avinash":17, "Adarsh":5}

dict2 = {"Sudev":7, "Raiz":19, "Amal":2, "Niranjan":8, "Sagar":12}

dict1.update(dict2)

print("Merged : ", dict1)
```

Output:

```
Merged :  {'Abhi': 1, 'Savad': 18, 'Avinash': 17, 'Adarsh': 5, 'Sudev': 7, 'Raiz': 19, 'Amal': 2, 'Niranjan': 8, 'Sagar': 12}


Process finished with exit code 0
```

Experiment No 18:

Find gcd of 2 numbers.

Source Code:

```
num1 = int(input("Enter 1st number: "))
num2 = int(input("Enter 2nd number: "))
i = 1
while(i <= num1 and i <= num2):
        if(num1 % i == 0 and num2 % i == 0):
                gcd = i
        i = i + 1
print("GCD is: ", gcd)
```

Output:

```
Enter 1st number: 12
Enter 2nd number: 18
GCD is: 6

Process finished with exit code 0
```

Experiment No 19:

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

Source Code:

```
mylist = [12, 13, 20, 45, 43, 23, 36]
for i in mylist:
        if(i % 2 == 0):
                mylist.remove(i)
print(mylist)
```

Output:

```
[13, 45, 43, 23]

Process finished with exit code 0
```

CO2

Experiment No 20:

Program to find the factorial of a number.

Source Code:

```
fact = 1
num = int(input("Enter the number: "))
for i in range(1, num + 1):
        fact = fact * i
print("Factorial of  ", num," is: ", fact)
```

Output:

```
Enter the number: 5
Factorial of 5 is: 120

Process finished with exit code 0
```

Experiment No 21

Generate Fibonacci series of N terms.

Source Code:

```
first = 0
second = 1
limit = int(input("Enter the limit: "))
print("The fibonacci series upto ",limit," terms are:")
print(first)
print(second)
for i in range(limit - 2):
        third = first + second
        first = second
        second = third
        print(third)
```

Output:

```
Enter the limit: 6
The fibonacci series upto 6 terms are:
0
1
1
2
3
5

Process finished with exit code 0
```

Experiment No 22:

Find the sum of all items in a list.

Source Code:

```
list = []
sum = 0
limit = int(input("Enter the limit: "))
print("Enter", limit, "numbers: ")
for i in range(limit):
        list2 = int(input())
        list.append(list2)
print(list)
for i in list:
        sum = sum + i
print("Sum of list: ", sum)
```

Output:

```
Enter the limit:6
Enter 6 numbers:
3
7
4
5
12
3
[3, 7, 4, 5, 12, 3]
Sum of list: 34

Process finished with exit code 0
```

Experiment No 23:

Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square.

Source Code

```
import math
start = int(input("Enter a starting range in 4 digit:"))
end = int(input("Enter an ending range in 4 digit:"))
perfect = []
for i in range(start,end+1):
 flag = 0
 num = i
 while num > 0 :
        digit = num % 10
        if digit not in [0, 2, 4, 6, 8]:
                flag = 1
                break
        num = int(num / 10)
        if flag == 0 and math.sqrt(i) % 1 == 0:
                perfect.append(i)
print("The list of perfect square numbers are: ", perfect)
```

Output:

```
Enter a starting range in 4 digit: 1001
Enter an ending range in 4 digit: 9999
The list of perfect square numbers are: [4624, 6084, 6400, 8464]

Process finished with exit code 0
```

Experiment No 24:

Display the given pyramid with step number accepted from user.

Eg: N=4

1

2 4

3 6 9

4 8 12 16

Source Code:

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

Output:

```
Enter the step number: 4
1
2 4
3 6 9
4 8 12 16

Process finished with exit code 0
```

Experiment No 25:

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

Source Code:

```
string = input("Enter the string: ")
count = 0
for i in range(len(string)):
        if(string[i] != ' '):
                count = count + 1
print("The total number of characters in the string is ", count)
```

Output:

```
Enter the string: programming
The total number of characters in the string is 11

Process finished with exit code 0
```

Experiment No 26:

Add the 'ing' at the end of the given string, if it already ends with 'ing',then add 'ly'.

Source Code:

```
str = input("Enter a string: ")
if str.endswith("ing"):
        str = str + "ly"
else:
        str = str + "ing"
print("Modified string is:", str)
```

Output:

```
Enter a string: playing
Modified string is: playingly

Process finished with exit code 0
```

Experiment No 27:

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

Source Code:

```
n = int(input("Enter the number of words: "))
list = []
for i in range(n):
        x = (input("Enter the word: "))
        list.append(x)
print(list)
max = len(list[0])
temp = list[0]
for i in list:
        if(len(i) > max):
                max = len(i)
                temp = i
print("The word having longest length is:",temp,"and its length is:",max)
```

Output:

```
Enter the number of words: 3
Enter the word: Abhishek
Enter the word: Savad
Enter the word: Sudev
[' Abhishek ', ' Savad ', ' Sudev ']
The word having longest length is: arundhatahi and its length is: 8

Process finished with exit code 0
```

Experiment No 28:

Construct following pattern using nested loop

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

Source Code:

```
n = int(input("Enter the no of rows: "))
for i in range(1, n):
        for j in range(i):
                print(" * ",end=' ')
        print()
for i in range(n, 0, -1):
        for j in range(i):
                 print(" * ",end=' ')
        print()
```

Output:

```
Enter the no of rows: 5
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*

Process finished with exit code 0
```

Experiment No 29:

Generate all factors of a number.

Source Code:

```
n = int(input("Enter the number: "))
print("The factors of ", n, "are: ")
for i in range(1, n + 1):
        if(n % i == 0):
        print(i)
```

Output:

```
Enter the number: 30
The factors of 30 are:
1
2
3
5
6
10
15
30

Process finished with exit code 0
```

Experiment No 30:

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

Source Code:

```
a = int(input("Enter the sides of square: "))
s_area = lambda a: a * a
print("Area of square is: ", s_area(a))
l = int(input("Enter the length of rectangle: "))
b = int(input("Enter the breadth of rectangle: "))
r_area = lambda l, b: l * b
print("Area of rectangle is: ", r_area(l,b))
b = int(input("Enter the base of the triangle: "))
h = int(input("Enter the height of the triangle: "))
t_area = lambda b, h:.5 * b * h
print("Area of triangle is: ", t_area(b,h))
```

Output:

```
Enter the sides of square: 3
Area of square is: 9
Enter the length of rectangle: 4
Enter the breadth of rectangle: 5
Area of rectangle is: 20
Enter the base of the triangle: 3
Enter the height of the triangle: 4
Area of triangle is: 6.0

Process finished with exit code 0
```

CO3

Experiment No 31:

Work with built-in packages.

Source Code:

```
import math
print("3 to the power 2: ", math.pow(3, 2))
print("Square root of 64: ", math.sqrt(64))
```

Output:

```
3 to the power 2: 9.0
Square root of 64: 8.0

Process finished with exit code 0
```

Experiment No 32:

Create a package graphics with modules rectangle, circle and sub-package 3-Dgraphics with modules cuboid and sphere. Include methods to find area and perimeter of respective figures in each module. Write programs that finds area and perimeter of figures by different importing statements. (Include selective import of modules and import * statements)

Source Code:

Package Graphics:

rectangle.py

```
def area(h, w):
    return h * w
def perimeter(h, w):
    return 2 * (h + w)
```

circle.py

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

Package 3D-Graphics:

cuboid.py

```
def area(l, h, w):
     return (2 * l * h) + (2 * h * w) + (2 * l * w)
def perimeter(l, h, w):
    return 4 * (l + h + w)
```

sphere.py

```
import math
def area(r):
        return 4 * math.pi * r * r
def perimeter(r):
 return math.pi * 2 * r
```

main.py

```
import grahics.rectangle as re
import grahics.circle as c
import grahics.trd_graphics.cuboid as cb
import grahics.trd_graphics.sphere as s
print("Rectangle")
print("Area:", re.area(5, 6))
print("Perimeter:", re.perimeter(5, 6))
print("Circle")
print("Area:", c.area(4))
print("Perimeter:", c.perimeter(4))
print("Cuboid")
print("Area:", cb.area(4, 5, 6))
print("Perimeter:", cb.perimeter(4, 5, 6))
print("Shere")
print("Area:", s.area(5))
print("Perimeter:", s.perimeter(5))
```

Output:

```
Rectangle
Area: 30
Perimeter: 22
Circle
Area: 50.26548245743669
Perimeter: 25.132741228718345
Cuboid
Area: 148
Perimeter: 60
Shere
Area: 314.1592653589793
Perimeter: 31.41592653589793

Process finished with exit code 0
```

CO4

Experiment No 33:

Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two Rectangle objects by their area.

Source Code:

```
class Rectangle:
        def __init__(self, length, breadth):
                self.length = length
                self.breadth = breadth
        def area(self):
                return (self.length * self.breadth)
         def periemeter(self):
                return (2 * (self.length + self.breadth))
l1 = int(input("Enter length 1:"))
b1 = int(input("Enter breadth 1:"))
r1 = Rectangle(l1, b1)
print("Area of rectangle 1:", r1.area())
print("Periemeter of rectangle 1:", r1.periemeter())
l2 = int(input("Enter length 2:"))
b2 = int(input("Enter breadth 2:"))
r2 = Rectangle(l2, b2)
print("Area of rectangle 2:", r2.area())
print("Periemeter of rectangle 2:", r2.periemeter())
a1 = r1.area()
a2 = r2.area()
if a1 > a2:
        print("Area of rectangle 1 is high")
elif a1 == a2:
        print("Area are equal")
```

```
else:
    print("Area of rectangle 2 is high")
```

Output:

```
Enter length 1: 3
Enter breadth 1: 4
Area of rectangle 1: 12
Periemeter of rectangle 1: 14
Enter length 2: 4
Enter breadth 2: 5
Area of rectangle 2: 20
Periemeter of rectangle 2: 18
Area of rectangle 2 is high

Process finished with exit code 0
```

Experiment No 34:

Create a Bank account with members account number, name, type of account and balance. Write constructor and methods to deposit at the bank and withdraw an amount from the bank.

Source Code:

```
class Bank:
        def __init__(self, accno, name, type, bal):
                self.accno = accno
                self.name = name
                self.type = type
                 self.bal = bal
        def deposit(self, amount):
                self.bal = self.bal + amount
                print("Amount deposited successfully")
                return self.bal
         def withdraw(self, amount):
                if amount > self.bal:
                        print("Insufficient balance")
                        return self.bal
                else:
                        self.bal = self.bal - amount
                        print("Amount withdrawed successfully")
                         return self.bal
b = Bank(1001,"abhi","savings",3000)
damount = float(input("Enter the amount to be deposited: "))
print("Account balance:", b.deposit(damount))
wamount = float(input("Enter the amount to be withdrwn: "))
print("Account balance:", b.withdraw(wamount))
```

Output:

Enter the amount to be deposited: 5000

Amount deposited successfully

Account balance: 8000.0

Enter the amount to be withdrwn: 1000

Amount withdrawed successfully

Account balance: 7000.0

Process finished with exit code 0

Experiment No 35:

Create a class Rectangle with private attributes length and width. Overload '<' operator to compare the area of 2 rectangles.

Source Code:

```
class Rectangle:
        def __init__(self, length, width):
                self.__length = length
                self.__width = width
        def area(self):
                return self.__length*self.__width
        def __lt__(self, other):
                return self.area() > other.area()
l1 = int(input("Enter the length of first rectangle: "))
w1 = int(input("Enter the width of first rectangle: "))
rectangle1 = (l1, w1)
l2 = int(input("Enter the length of second rectangle: "))
w2 = int(input("Enter the width of second rectangle: "))
rectangle2 = (l2, w2)
if rectangle1 < rectangle2:
        print("Area of rectangle one is smaller")
else:
        print("Area of rectangle two is smaller")
```

Output:

Enter the length of first rectangle: 4

Enter the width of first rectangle: 5

Enter the length of second rectangle: 3

Enter the width of second rectangle: 6

Area of rectangle two is smaller

Process finished with exit code 0

Experiment No 36:

Create a class Time with private attributes hour, minute and second. Overload '+' operator to find sum of 2 time.

Source Code:

```
class Time:
        def __init__(self, hour, minute, second):
                self.__hour = hour
                self.__minute = minute
                self.__second = second
        def __add__(self, other):
                second = self.__second + other.__second
                minute = self.__minute + other.__minute
                hour = self.__hour + other.__hour
                if second > 60:
                        minute += int(minute/60)
                        second = second % 60
                 if minute > 60:
                        hour += int(minute / 60)
                        minute = (minute / 60)
                time = "{0} hours:{1} minutes:{2} seconds".format(hour, minute, second)
                return time
h1 = int(input("Enter the hour of first time: "))
m1 = int(input("Enter the minute of first time: "))
s1 = int(input("Enter the second of first time: "))
h2 = int(input("Enter the hour of second time: "))
m2 = int(input("Enter the minute of second time: "))
s2 = int(input("Enter the second of second time: "))
time1 = Time(h1, m1, s1)
time2 = Time(h2, m2, s2)
```

```
print("Sum of time:",time1 + time2)
```

Output:

```
Enter the hour of first time: 3
Enter the minute of first time: 5
Enter the second of first time: 30
Enter the hour of second time: 2
Enter the minute of second time: 8
Enter the second of second time: 20
Sum of time: 5 hours:13 minutes: 50 seconds

Process finished with exit code 0
```

Experiment No 37:

Create a class Publisher (name). Derive class Book from Publisher with attributes title and author. Derive class Python from Book with attributes price and no_of_pages. Write a program that displays information about a Python book. Use base class constructor invocation and method overriding.

Source Code:

```
class Publisher:
    def __init__(self, publisher):
        self.publisher = publisher
    def display(self):
        print("Publisher name : ", self.publisher)
class Book(Publisher):
    def __init__(self, title, author):
        self.title = title
        self.author = author
    def display(self):
        super().display()
        print("Title of the book : ", self.title)
        print("Author of the book : ", self.author)
class Python(Book):
    def __init__(self, publi, author, title, price, numb):
        self.price = price
        self.noofpages = numb
        Book.__init__(self, title, author)
        Publisher.__init__(self, publi)
    def display(self):
        super().display()
        print("Price of the book : ", self.price)
        print("Number of pages in the book", self.noofpages)
```

```
b1 = Python("SPD", "Brian Jones ", "Python Cookbook: Recipes For Mastering
Python", 1670, 1230)

b1.display()
```

Output:

```
Publisher name : SPD

Title of the book : Python Cookbook: Recipes For Mastering Python

Author of the book : Brian Jones

Price of the book : 1670

Number of pages in the book 1230


Process finished with exit code 0
```

CO5

Experiment No 38:

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

Source Code:

```
f=open("demo.txt", "r")
c=f.readlines()
li=[x.strip() for x in c]
print(li)
```

Output:

```
[' abhishek ', ' savad ', ' sudev ', ' avinash ', ' adarsh ']

Process finished with exit code 0
```

Experiment No 39:

Python program to copy odd lines of one file to other.

Source Code:

```
f1 = open("aru.txt", "r")
f2 = open("new.txt", "w")
c = f1.readlines()
for i in range(0, len(c)):
        if(i % 2 != 0):
                f2.write(c[i])
f2.close()
f2 = open("new.txt", "r")
c1 = f2.read()
print(c1)
f1.close()
f2.close()
```

Output:

```
abhishek
sudev

Process finished with exit code 0
```

Experiment No 40:

Write a Python program to read each row from a given csv file and print a list of strings.

Source Code:

```
import csv
with open("aru.csv","w",newline='') as file:
        write=csv.writer(file)
        write.writerow(["Sl No", "Movie", "Rating"])
        write.writerow(["1", "Twilight", "3"])
        write.writerow(["2", "Loard of Rings", "2"])
with open("aru.csv") as csvfile:
        data = csv.reader(csvfile)
        for r in data:
                print( r)
```

Output:

```
['Sl No', 'Movie', 'Rating']
['1', 'Twilight', '3']
['2', 'Loard of Rings', '2']

Process finished with exit code 0
```

Experiment No 41:

Write a Python program to read specific columns of a given CSV file and print the content of the columns.

Source Code:

```
import csv
with open("movie.csv", "w",newline='') as file:
        write = csv.writer(file)
        write.writerow(["Sl No", "Movie", "Rating"])
        write.writerow(["1", "Twilight", "3"])
        write.writerow(["2", "Loard of Rings", "2"])
with open("movie.csv") as csvfile:
        data = csv.DictReader(csvfile)
        for r in data:
                print(r['Sl No'], r['Movie'])
```

Output:

```
1  Twilight
2  Loard of Rings

Process finished with exit code 0
```

Experiment No 42:

Write a Python program to write a Python dictionary to a csv file. After writing the CSV file read the CSV file and display the content.

Source Code:

```
import csv
f = open("fruits.csv","w")
writer = csv.DictWriter(f,fieldnames=["Fruit", "Count"])
writer.writeheader()
writer.writerow({"Fruit":"Apple", "Count":"1"})
writer.writerow({"Fruit":"Banana", "Count":"2"})
f.close()
c = 0
f = open("fruits.csv")
reader = csv.DictReader(f)
for row in reader:
        if c == 0:
                print(f '{" ".join(row)}')
                print(f'{row["fruit"]}, {row["count"]}')
f.close()
```

Output:

```
Fruit Count
Apple, 1
Fruit Count
Banana, 2

Process finished with exit code 0
```