LAB CYCLE-1

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

CODE :

```
from datetime import date a
= date.today()
b = int(input("Enter a finishing Year : ")) print("The future
leap years : ") for yr in range(a.year, b):    if (0 == yr % 4)
and (0 != yr % 100) or (0 == yr % 400):        print(yr)
```

OUTPUT:

```
Enter a finishing Year : 2040 The
future leap years :
2024
2028
2032
2036
```

2. List comprehensions:
   a. Generate positive list of numbers from a given list of integers

CODE :

```
list1=[] n=int(input("Enter the
limit : ")) for i in range(0,n):
list1.append(int(input()))
print("List of postive integers ")
for x in list1:     if(x>0):
print(x,"\t",end="")
```

OUTPUT :

```
Enter the limit : 5
-4
-9
2
6
0

List of postive integers

2           6
```

b. Square of N numbers **CODE :** list=[] square=[] n=int(input("Enter a limit : "))

```
print("Enter the numbers") for i in range(n):
     list.append(int(input()))
  print("Square of given numbers")
  for x in list:     a=x**2
  square.append(a)
  print(square)
```

OUTPUT :

```
Enter a limit : 4
Enter the numbers
4
6
9
10

Square of given numbers
[16, 36, 81, 100]
```

c. Form a list of vowels selected from a given word **CODE :**

```
vowels=[]
vow={"a","A","e","E","i","I","o","O","u","U"}
str=input("Enter a word : ")
for i in str:     for x in vow:
if(i==x):
          vowels.append(i) print("Vowels
are")
print(vowels)
```

OUTPUT :

```
Enter a word : APPLE

Vowels are
['A', 'E']
```

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

CODE :

```
tr=input("Enter a word : ")
str1=list(str) for i in
range(len(str1)):
   str1[i] = chr(ord(str1[i])+1) print((i+1))
print(str1)
```

OUTPUT : Enter a

```
word AND
Ordinal value of A is 65
Ordinal value of N is 78
Ordinal value of D is 68
```

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

CODE :

```
str=input("Enter ascntance : ")
str1=str.split()
setstr=set(str1)
dict1={} for x
in setstr:
   i=0     for y in str1:
if(x==y):            i=i+1
dict1.update({x:i})
print(dict1)
```

OUTPUT :

```
Enter ascntance : no sentence starts with because because because is a conjunction

{'because': 3, 'a': 1, 'no': 1, 'conjunction': 1, 'with': 1, 'is': 1, 'starts': 1, 'sentence': 1}
```

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

CODE :

```
limit=int(input("enter limit"))
out_list=[] for i in range(limit):
number=int(input())     if
number<100:
out_list.append(number)
else:
      out_list.append('over')

print(out_list)
```

OUTPUT :

```
enter limit : 5
200
50
75
300
35
['over', 50, 75, 'over', 35]
```

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

CODE :

```
limit=int(input("enter limit : "))

print("enter your first name : ")
first_name_lst=[] for lim in
range(limit):
first_name=input()
    first_name_lst.append(first_name)

occur_A=sum([name.count('a')for name in first_name_lst])

print("occurrences of ‘a’ in the list is",occur_A)
```

OUTPUT :

```
enter limit : 4 enter
your first name :
Rahul
Joyal
Anu
Biyons
occurrences of ‘a’ in the list is 2
```

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

CODE :

```
lst1=[10,30,50,30,600,70]
lst2=[9,7,50,78,60,440,700]

"Check (a) Whether list are of same length "
```

```
if lst1==lst2:     print("List Have
same lentgh") else:     print("list is
not in same length")

if sum(lst1)==sum(lst2):
    print("two list have same sum") else:
    print("sum is not same")

set_lst1=set(lst1) set_lst2=set(lst2)

"& used to perform union opertion in sets"
print("{} elements are common in two lists".format(set_lst1&set_lst2))
```

OUTPUT :

```
list is not in same length sum
is not same
{50} elements are common in two lists
```

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

CODE :

```
string1=input("Enter a word : ")
fstchr=string1[0]
new_str=fstchr+string1[1:].replace(fstchr,'$')
print("{}->{}".format(string1,new_str))
```

OUTPUT

```
Enter a word : onion
onion->oni$n
```

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

CODE :

```
string=input("enter a string : ")

first_ch=string[0] last_ch=string[-1]
newstr=last_ch+string[1:-
1]+first_ch
```

```
    print("{}->{}".format(string,newstr))
```

OUTPUT

```
    enter a string : python
    python->nythop
```

9. Accept the radius from user and find area of circle.

CODE :

```
import math

radius=float(input("Enter radius of circle : "))

area_of_circle=math.pi*(radius*radius)

print("Area of circle is : ",area_of_circle)
```

OUTPUT

```
Enter radius of circle : 4
Area of circle is :  50.26548245743669
```

10. Find biggest of 3 numbers entered.

CODE :

```
num1=int(input("enter first number  : "))
num2=int(input("enter second number : "))
num3=int(input("enter third number  : "))

if num1>num2:     if num1<num3:
print("Biggest number is {}".format(num3))
else:
      print("Biggest number is {}".format(num2))
else:    if num2>num3:       print("Bigget number
is {}".format(num2))     else:
      print("Biggest number is {}".format(num3))
```

OUTPUT

```
enter first number      : 10
enter second number : 35
enter third number      : 20
```

```
Bigget number is      35
```

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

CODE :

```
filename=input("enter filename : ")

print("Extension of file name is",filename.split('.')[1])
```

OUTPUT

```
enter filename : abc.jpg
Extension of file name is jpg
```

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

CODE :

```
n=input("enter the colours : ")
splt1=n.split(",") print(splt1)
fst=splt1[0] lst=splt1[-1]
print(fst,lst)
```

OUTPUT

```
enter the colours : blue,red,orange,green
['blue', 'red', 'orange', 'green']
blue green
```

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

CODE :

```
n=(input("enter the number: "))
a=int(input("Enter the limit : "))
sum=0 for i in range(1,a+1):
sum=sum+int(n*i)
print(sum)
```

OUTPUT

```
enter the number: 5
Enter the limit : 3
```

```
615
```

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

CODE :

```
colorlst1=[]
colorlst2=[]

range1=input("enetr size of list one : ")
print("enter color list 1 : ")
for i in range(int(range1)):
colorlst1.append(input())

range2=input("enter size of list two : ")
print("enter color list 2 : ")
for i in range(int(range2)):
colorlst2.append(input())

print(" All colors from color-list1 not contained in color-list2 is
",set(colorlst1)set(colorlst2))
```

OUTPUT

```
enetr size of list one : 3
enter color list 1 :
green
blue
red

enter size of list two : 4
enter color list 2 :
green
orange
white
black

 All colors from color-list1 not contained in color-list2 is  {'red', 'blue'}
```

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

CODE :

```
string1=input("Enter String One : ") string2=input("Enter
String Two : ")
```

```
ch1_string1=string1[0] ch2_string2=string2[0]

newstring=ch2_string2+string1[1:]+" "+ch1_string1+string2[1:] print("New
string is ",newstring)
```

OUTPUT

```
Enter String One : Hey
Enter String Two : Baby

New string is  Bey Haby
```

16. Sort dictionary in ascending and descending order.

CODE :

```
size=int(input("enter size : "))
dict={} print("Key must be
uniques : ") for i in
range(size):
    k=input("enter key : ")
    dict[k]=input("enter value : ")

print("Dictionary is ::  ",dict) ascending_sorted_dict={}
descending_sorted_dict={}

for i in sorted(dict.keys()):
    ascending_sorted_dict[i]=dict[i]

for i in reversed(sorted(dict.keys())):
descending_sorted_dict[i]=dict[i]


print("Sort dictionary in ascending order    :",ascending_sorted_dict)

print("Sort dictionary in descending order    :",descending_sorted_dict)
```

OUTPUT

```
enter size : 3
Key must be uniques :
enter key : 3
enter value : a

enter key : 2
enter value : b

enter key : 7
enter value : c

Dictionary is ::   {'3': 'a', '2': 'b', '7': 'c'}

Sort dictionary in ascending order    : {'2': 'b', '3': 'a', '7': 'c'}
Sort dictionary in descending order    : {'7': 'c', '3': 'a', '2': 'b'}
```

17. Merge two dictionaries

CODE :

```
size=int(input("enter size : "))
dict1={} print("Key must be
uniques : ") for i in
range(size):     k=input("enter
key : ")
   dict1[k]=input("enter value : ")


size=int(input("enter size : "))
dict2={} print("Key must be
uniques : ") for i in
range(size):     k=input("enter
key : ")
   dict2[k]=input("enter value : ")

dict1.update(dict2)

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

OUTPUT

```
enter size : 3
Key must be uniques :
enter key : 1 enter
value : a enter
key : 2 enter
value : b enter
key : 3
enter value : c

enter size : 2 Key
must be uniques :
```

```
enter key : 4 enter
value : d enter
key : 5
enter value : e

Merged dict is  : {'1': 'a', '2': 'b', '3': 'c', '4': 'd', '5': 'e'}
```

18. Find gcd of 2 numbers.

CODE :

```
number1=int(input("Enter first number"))

number2=int(input("Enter second number"))

factors_number1= [i for i in range(1,number1+1) if number1%i==0] factors_number2=
[i for i in range(1,number2+1) if number2%i==0]

common_factors=set(factors_number1)&set(factors_number2
) if len(common_factors)!=0:     gcd=max(common_factors)
   print("Greatest Common Divisor is ",gcd)
```

OUTPUT

```
Enter first number 10
Enter second number 25
Greatest Common Divisor is  5
```

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

CODE :

```
size=int(input("Enter size of list : "))
lst_int=[] print("Enter numbers : ")
for i in range(size):
lst_int.append(int(input()))
lst_rm_int=[] lst_rm_int=[ i for i in
lst_int if i%2==1]
print("list removing even numbers : ",lst_rm_int)
```

OUTPUT

```
Enter size of list : 3 Enter
numbers :
3
4 7
list removing even numbers :  [3, 7]
```

LAB CYCLE-2

1. Program to find the factorial of a number

CODE :

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

OUTPUT

```
Enter number : 5
5 != 120
```

2. Generate Fibonacci series of N terms

CODE :

```
n=int(input("Enter the limit : "))
a=0
b=1
print("\n Fibnocci Series : ")
print("\t",a,"\t",b,"\t",end="")
for i in range(2,n):
   c=a+b
a=b
b=c
   print(c,"\t",end="")
```

OUTPUT

```
Enter the limit : 5

Fibnocci Series :
     0          1          1          2          3
```

3. Find the sum of all items in a list

CODE :

```
n=int(input("Enter range of list : "))
print(“Enter the elements\n”)

lst1=[] for i in
range(n):
   lst1.append(int(input()))

print("Sum of list is ",sum(lst1))
```

OUTPUT

```
Enter range of list : 4
Enter the elements
10
5
8
2
Sum of list is  25
```

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

CODE :

```
a=int(input("Enter the range-1 : "))
b=int(input("Enter the range-2 :
")) for i in range(a,b):     for j in
range(32,100,1):         if i==j*j:
            string=str(i)
            if int(string[0])%2==0 and int(string[1])%2==0 and int(string[2])%2==0 and
int(string[3])%2==0:
               print(i)
```

OUTPUT :

```
Enter the range 1 : 2000
Enter the range 2 : 8000
4624
6084
```

6400

5. Display the given pyramid with step number accepted from user.

Eg: N=4

```
                    1
 2                                          1
 3  6                                       1
 4   8   12
```

CODE : **Error! Bookmark not defined.**

```
n=int(input("Enter no.of steps : "))
for i in range(1,n+1):     for j in
range(1,i+1):          print(i*j,end="
")
   print("\n")
```

OUTPUT :l

```
Enter no.of steps : 5
1

2 4

3 6 9

4 8 12 16

5 10 15 20 25
```

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

CODE :

```
def char_frequency(str1):
    dict = {}
for n in str1:
        keys = dict.keys()
if n in keys:            dict[n]
+= 1         else:
dict[n] = 1     return dict
a=input("Enter the string
")
print(char_frequency(a))
```

OUTPUT :

```
Enter the string apple

{'a': 1, 'p': 2, 'l': 1, 'e': 1}
```

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

CODE :

```
string = input("Enter a string ")
if len(string) < 3:   print(string)
elif string[-3:] == 'ing':
  print(string + 'ly') else:
  print(string + 'ing')
```

OUTPUT :

```
Enter a string abc

Abcing
```

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

CODE :

```
def longestWordLength(string):
   length=0     w=''     for word
in string.split():
if(len(word)>length):
length=len(word)
w=word     return(length,w)
string=input("Enter the string")
l,w=longestWordLength(string
)
print("Longest word is",w, "and its length is",l)
```

OUTPUT :

```
Enter the string three and four is

Longest word is three and its length is 5
```

9. Construct following pattern using nested loop

```
*

* *

* * *

* * * *

* * * * *

* * * *

* * *

* *

*
```

CODE :

```
n=int(input("Enter tne number "))
for i in range(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 number 5


*

* *

* * *

* * * *

* * * * *

* * * *

* * *

* *

*
```

10. Generate all factors of a number.

CODE : def

```
print_factors(x):

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

OUTPUT :

```
Enter the number 65

The factors of 65 are:

1

5

13

65
```

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

CODE :

```
import math a=int(input("Enter the
value")) s_area=lambda a:a*a
print("Area of square is : ",s_area(a))
b=int(input("Enter the length"))
c=int(input("Enter the height"))
r_area=lambda len,ht:len*ht print("Area
of rectangle is : ",r_area(b,c))
d=int(input("Enter the base"))
e=int(input("Enter the height"))
t_area=lambda b,h:0.5*b*h
print("Area of triangle is : ",r_area(d,e))
```

OUTPUT :

```
Enter the value2

Area of square is :  4

Enter the length3

Enter the height4

Area of rectangle is :  12

Enter the base3

Enter the height2

Area of triangle is :  3
```

LAB CYCLE - 3

1. Work with built-in packages

CODE :

```
import math print(math.sqrt(25))
print(math.pi) print(math.degrees(2))
print(math.radians(60)) print(math.sin(2))
print(math.cos(0.5)) print(math.tan(0.23))
print(math.factorial(4)) import random
print(random.randint(0, 5))
print(random.random())
print(random.random() * 100) List = [1, 4,
True, 800, "python", 27, "hello"]
print(random.choice(List)) import datetime
from datetime import date import time
print(time.time())
```

OUTPUT :

```
5.0

3.141592653589793

114.59155902616465

1.0471975511965976

0.9092974268256817

0.8775825618903728

0.23414336235146527

24

1

0.03772466762864812

24.55738973412146

27

1645957476.892115
```

2. 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 programs that finds area and perimeter of figures by different importing statements. (Include selective import of modules and import * statements)

CODE :

```
area and perimeter.py from
graphics import rectangle from
graphics import circle
from graphics.threedpackage.cuboid import *

#*********rectangle******** num1=int(input("enter
length of rectangle:")) num2=int(input("enter breadth
of rectangle:"))
res1=rectangle.area(num1,num2) print("area
of rectangle:",res1)
res2=rectangle.perimeter(num1,num2)



#*********circle********

rad=int(input("enter radius of circle:"))
res3=circle.area(rad) print("area of
circle:",res3)
res4=circle.perimeter(rad)

print("perimeter of circle:",res4)

#*********cuboid********

l=int(input("enter length of cuboid:")) b=int(input("enter
breadth of cuboid:"))
h=int(input("enter hieght of cuboid:"))

res3=area(l,b,h) print("area
of cuboid:",res3)
res2=perimeter(l,b)
print("perimeter of cuboid:",res2)




#*********sphere******** from
graphics.threedpackage.sphere import *
r=int(input("enter radius of sphere"))

res4=area(r) print("area of
sphere:",res4)

res5=perimeter(r) print("perimeter
of sphere:",res5)
```

```
circle.py def
area(rad):
res3=3.14*rad*rad
return res3 def
perimeter(rad):
res4=2*3.14*rad
return res4
rectangle.py def
area(num1,num2):
res1=num1*num2
   return res1 def
perimeter(num1,num2):
res2=2*(num1+num2)
   return res2 cuboid.py
def area(l,b,h):
res5=2*(l*b+b*h+h*l)
return res5 def
perimeter(l,b):
res2=2*(l+b)     return
res2 sphere.py def
area(l,b,h):
res5=2*(l*b+b*h+h*l)
return res5 def
perimeter(l,b):
res2=2*(l+b)     return
res2
```

OUTPUT :

```
enter length of rectangle:6 enter
breadth of rectangle:4 area of
rectangle: 24 perimeter of rectangle:
20 enter radius of circle:4 area of
circle: 50.24 perimeter of circle:
25.12 enter length of cuboid:9 enter
breadth of cuboid:7 enter hieght of
```

```
cuboid:5 area of cuboid: 286
perimeter of cuboid: 32 enter radius
of sphere6 area of sphere:
452.15999999999997 perimeter of
sphere: 37.68
```

LAB CYCLE-4

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

CODE :

```
class Rectangle:     def __init__(self,
length, breadth):         self.length =
length
        self.breadth = breadth

    def area(self):         return
self.length * self.breadth


a = int(input("Enter length of rectangle: ")) b
= int(input("Enter breadth of rectangle: "))
obj = Rectangle(a, b)
print("Area of rectangle:", obj.area())
```

OUTPUT

```
Enter length of rectangle: 4

Enter breadth of rectangle: 3

Area of rectangle: 12
```

2. 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.

CODE : class bank:

```
__acc_name = ""
   __acc_no = ""
   __acc_type = ""
   __acc_balance = 0

   def __init__(self, a_name, a_no, a_type, a_balance):
self.__acc_name = a_name         self.__acc_no = a_no
self.__acc_type = a_type
      self.__acc_balance = a_balance

   def deposite(self, a_deposit):         print("Initial
balance is  : ", self.__acc_balance)
print("Deposite is  : ", a_deposit)
self.__acc_balance += a_deposit
      print("Current balance is  : ", self.__acc_balance)

   def withdraw(self):         print("Current balance is  : ",
self.__acc_balance)         self.amount = int(input("How much amount
need to withdraw : "))         if self.amount > self.__acc_balance:
print("You don't have enough balance to withdraw !!")
print("Current balance is  : ", self.__acc_balance)         else:
         print(self.amount, " is withrawed .")
self.__acc_balance -= self.amount
         print("Current balance is  : ", self.__acc_balance)

   def acc_info(self):
      print(
         "\n\n||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||\n\n")
      print("Account  holder  name   :   ",  self.__acc_name)
print("Account  number                 :    ",  self.__acc_no)
print("Account  type                   :   ",  self.__acc_type)
print("Account  Balance  is         :   ",  self.__acc_balance)
print(
         "\n\n||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||\n\n")



def main():     name = input("Enter Account holder
name : ")     no = input("Enter Account number
: ")     atype = input("Enter Account type            : ")
bal = int(input("Enter Account initial balance : "))
holder = bank(name, no, atype, bal)
```

```
    while (True):
        print("\n\n....................................................\n\n")

      opt = int(input("1)Deposite \n2)Withdraw \n3)Account info \n0)Exit\nChoose
your option :: "))
        print("\n\n........................................................\n\n")
if opt == 1:
            amount = int(input("Deposite amount : "))
holder.deposite(amount)         elif opt == 2:
            holder.withdraw()
elif opt == 3:
            holder.acc_info()
elif opt == 0:
            break
else:
            print("Invalid Option !")


if __name__ == "__main__":
while (True):         main()
```

OUTPUT :

```
Enter Account holder name : Abc Xyz Enter

Account number        : 1234567

Enter Account type            : savings

Enter Account initial balance : 10000

........................................................

1)Deposite

2)Withdraw

3)Account info

0)Exit

Choose your option :: 1
........................................................

Deposite amount : 20000

Initial balance is  :  10000

Deposite is  :  20000

Current balance is  :  30000

........................................................
```

```
1)Deposite
2)Withdraw
3)Account info
0)Exit
Choose your option :: 2
.........................................................
Current balance is  :  30000 How much
amount need to withdraw : 4000 4000  is
withrawed .
Current balance is  :  26000
.........................................................
1)Deposite
2)Withdraw
3)Account info
0)Exit
Choose your option :: 3
.........................................................
||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Account holder name  :   Abc Xyz
Account number        :   1234567
Account type              :   savings
Account Balance is      :   26000
||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
.........................................................
1)Deposite
2)Withdraw
3)Account info
0)Exit
Choose your option :: 0
```

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

CODE :

```
class Rectangle():     def
__init__(self, l, w):
self.length= l
        self.width = w

    def rectangle_area(self):
return self.length * self.width

    def __lt__(self, other):
if(self.rectangle_area()<other.rectangle_area()):
            return True
else:              return
False

l1 = int(input("enter length of the rectangle 1 : "))
w1 = int(input("enter width of the rectangle 1: "))
l2 = int(input("enter length of the rectangle 2 : "))
w2 = int(input("enter width of the rectangle 2: "))

Rectangle1= Rectangle(l1, w1) Rectangle2=
Rectangle(l2,w2)
print("area of first rectangle :",Rectangle1.rectangle_area())
print("area of second rectangle :",Rectangle2.rectangle_area())
if Rectangle1 < Rectangle2:
        print("Rectangle two is large") else:
print("Rectangle one is large or these are equal")
```

OUTPUT :

```
enter length of the rectangle 1 : 6

enter width of the rectangle 1: 4

enter length of the rectangle 2 :

8 enter width of the rectangle 2:

3 area of first rectangle : 24 area

of second rectangle : 24

Rectangle one is large or these are equal
```

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

CODE : class

```
Time:
__hour = 0
    __minute = 0
    __second = 0

    def input_time(self, hour, m, s):
self.hour = hour         self.minute
= m
        self.second = s

    def __add__(self, obj):
        day = 0
        sec1 = self.second + obj.second
        mins = sec1 // 60
sec1 = sec1 % 60
        min1 = self.minute + obj.minute + mins
        hrs = min1 // 60
min1 = min1 % 60
        hour1 = self.hour + obj.hour + hrs
day = hour1 // 24        hour1 = hour1
% 24
        return "{}:{}:{}:{}".format(day, hour1, min1, sec1)


t1 = Time() t2
= Time()

print("enter first time") hh =
int(input("enter hour")) mm =
int(input("enter minute")) ss =
int(input("enter second"))
t1.input_time(hh, mm, ss)

print("enter second time") hh =
int(input("enter hour")) mm =
int(input("enter minute")) ss =
int(input("enter second"))
t2.input_time(hh, mm, ss)

print(t1 + t2)
```

OUTPUT :

```
enter first time

enter hour3 enter

minute4 enter

second5 enter

second time

enter hour7 enter

minute8 enter

second9

0:10:12:14
```

5. 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.

CODE :

```
class Publisher:     def
__init__(self,name):
        self.name=name

    def display(self):
        print('Name:',self.name)
class Book(Publisher):     def
__init__(self,title,author):

        self.title=title
        self.author=author

class Python(Book):     def
__init__(self,name,title,author,price,noofpage):
        self.price=price
self.noofpage=noofpage

        Publisher.__init__(self,name)
        Book.__init__(self, title, author)

    def display(self):         print('Book Title:',self.title,'\n Author:',self.author,'\nPrice
:',self.price,'\n No Of Pages:',self.noofpage)


n=input("Enter book name")
```

```
t=input("title") a=input("author")
p=int(input("price"))
np=int(input("pages"))

p = Python(n,t,a,p,np) p.display()
```

OUTPUT :

```
Enter book nameuniverse

titleinto the space

authormartin price1001

pages999

Book Title: into the space

 Author: martin

Price : 1001

 No Of Pages: 999
```

LAB CYCLE - 5

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

CODE :

```
list =[] f =
open("demofile.txt" , 'r')
for x in f:   print(x)
list.append(x) print("the
list is ", list)
```

demofile.txt

```
A flower is the most beautiful part of a tree or plant.
A flower has different colors and fragrances.
Flowers are used to greeting people.
We have flower shops all over the world.
Flowers also have religious values.
In many religions, the flowers are used for prayers, funerals and other like purposes.
```

OUTPUT

```
A flower is the most beautiful part of a tree or plant.


A flower has different colors and fragrances.


Flowers are used to greeting people.


We have flower shops all over the world.


Flowers also have religious values.


In many religions, the flowers are used for prayers, funerals and other like purposes.


the list is  ['A flower is the most beautiful part of a tree or plant.\n', 'A flower has
different colors and fragrances.\n', 'Flowers are used to greeting people.\n', 'We have
flower shops all over the world.\n', 'Flowers also have religious values.\n', 'In many
religions, the flowers are used for prayers, funerals and other like purposes.\n']
```

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

CODE :

```
fn = open('demofile.txt', 'r')
fn1 = open('demo2.txt', 'w')
cont = fn.readlines()
type(cont) for i in range(0,
len(cont)):    if (i % 2 !=
0):        fn1.write(cont[i-
1])    else:        pass
fn1.close()
fn1 = open('demo2.txt', 'r')
cont1 = fn1.read()
print(cont1) fn.close()
fn1.close()
```

demofile.txt

```
A flower is the most beautiful part of a tree or plant.
A flower has different colors and fragrances.
Flowers are used to greeting people.
We have flower shops all over the world.
Flowers also have religious values.
In many religions, the flowers are used for prayers, funerals and other like purposes
```

demo2.txt

```
A flower is the most beautiful part of a tree or plant.
Flowers are used to greeting people. Flowers
also have religious values.
```

OUTPUT

```
A flower is the most beautiful part of a tree or plant.

Flowers are used to greeting people.

Flowers also have religious values.
```

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

CODE :

```
import csv with
open('people.csv', 'r') as file:
reader = csv.reader(file)     for
row in reader:         print(row)
```

people.csv

```
SN, Name, City
1, Michael, New Jersey
2, Jack, California
```

OUTPUT :

```
['SN', ' Name', ' City']

['1', ' Michael', ' New Jersey']

['2', ' Jack', ' California']
```

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

CODE :

```
import csv
colum_name=input("Enter Column name:")

with open('username.csv','r') as csvf:
   data=csv.DictReader(csvf,delimiter=",")
print("conten of "+colum_name+" is ")
for row in data:
print(row[colum_name])
```

username.csv

```
department_id,department_name,manager_id,location_id
10,Administration,200,1700
20,Marketing,201,1800
30,Purchasing,114,1700
40,Human Resources,203,2400
50,Shipping,121,1500
60,IT,103,1400
70,Public Relations,204,2700
80,Sales,145,2500
90,Executive,100,1700
```

OUTPUT :

```
Enter Column name:department_name conten

of department_name is

Administration

Marketing

Purchasing

Human Resources

Shipping

IT

Public Relations

Sales

Executive
```

5. 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

CODE :

```
import csv
csv_columns = ['id','Column1', 'Column2', 'Column3', 'Column4', 'Column5']
dict_data = {'id':['1', '2', '3'],     'Column1':[33, 25, 56],
   'Column2':[35, 30, 30],
   'Column3':[21, 40, 55],
   'Column4':[71, 25, 55],     'Column5':[10, 10, 40], } csv_file
= "temp.csv" try:    with open(csv_file, 'w') as csvfile:
writer = csv.DictWriter(csvfile, fieldnames=csv_columns)
writer.writeheader()        for data in dict_data:
writer.writerow(dict_data) except IOError:    print("I/O error")
data = csv.DictReader(open(csv_file))
print("CSV file as a dictionary:\n")
for row in data:    print(row)
```

OUTPUT :

```
CSV file as a dictionary:

{'id': "['1', '2', '3']", 'Column1': '[33, 25, 56]', 'Column2': '[35, 30, 30]', 'Column3': '[21,
40, 55]', 'Column4': '[71, 25, 55]', 'Column5': '[10, 10, 40]'}

{'id': "['1', '2', '3']", 'Column1': '[33, 25, 56]', 'Column2': '[35, 30, 30]', 'Column3': '[21,
40, 55]', 'Column4': '[71, 25, 55]', 'Column5': '[10, 10, 40]'}
```

```
{'id': "['1', '2', '3']", 'Column1': '[33, 25, 56]', 'Column2': '[35, 30, 30]', 'Column3': '[21,
40, 55]', 'Column4': '[71, 25, 55]', 'Column5': '[10, 10, 40]'}

{'id': "['1', '2', '3']", 'Column1': '[33, 25, 56]', 'Column2': '[35, 30, 30]', 'Column3': '[21,
40, 55]', 'Column4': '[71, 25, 55]', 'Column5': '[10, 10, 40]'}

{'id': "['1', '2', '3']", 'Column1': '[33, 25, 56]', 'Column2': '[35, 30, 30]', 'Column3': '[21,
40, 55]', 'Column4': '[71, 25, 55]', 'Column5': '[10, 10, 40]'}

{'id': "['1', '2', '3']", 'Column1': '[33, 25, 56]', 'Column2': '[35, 30, 30]', 'Column3': '[21,
40, 55]', 'Column4': '[71, 25, 55]', 'Column5': '[10, 10, 40]'}
```