

**M.Sc. (Five Year Integrated) in Computer Science
(Artificial Intelligence & Data Science)**

First Semester

Laboratory Record

21-805-0106: PYTHON PROGRAMMING LAB

*Submitted in partial fulfillment
of the requirements for the award of degree in
Master of Science (Five Year Integrated)
in Computer Science (Artificial Intelligence & Data Science) of
Cochin University of Science and Technology (CUSAT)
Kochi*



Submitted by

**HIRA MOHAMMED K
(80521011)**

**DEPARTMENT OF COMPUTER SCIENCE
COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY (CUSAT)
KOCHI-682022**

MARCH 2022

DEPARTMENT OF COMPUTER SCIENCE
COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY (CUSAT)
KOCHI, KERALA-682022



*This is to certify that the practical laboratory record for **21-805-0106: Python Lab** is a record of work carried out by **HIRA MOHAMMED K(80521011)**, in partial fulfillment of the requirements for the award of degree in **Master of Science (Five Year Integrated)** in **Computer Science (Artificial Intelligence & Data Science)** of Cochin University of Science and Technology (CUSAT), Kochi. The lab record has been approved as it satisfies the academic requirements in respect of the first semester laboratory prescribed for the Master of Science (Five Year Integrated) in Computer Science degree.*

Faculty Member in-charge

Dr. Shailesh S.
Assistant Professor
Department of Computer Science
CUSAT

Dr. Philip Samuel
Professor and Head
Department of Computer Science
CUSAT

Table of Contents

Sl.No.	Program	Pg.No.
1	Operations on numbers	pg 1
2	Area of Triangle	pg 3
3	Employee payslip	pg 5
4	Happy sad numbers	pg 8
5	Operations on strings	pg 11
6	Pair of Rabbits	pg 16
7	Operations on list of integers	pg 18
8	File Handling	pg 22
9	List of N Boxes	pg 27
10	Calculation of area of two shapes	pg 30

Operations on numbers

AIM

Develop a program to read a four-digit number and find its sum of digits,reverse,difference between the product of digits at the odd position and the product of digits at the even position.

THEORY

- Input()-used to take user input.By default it return the user input in form of a string
- Strings-String is a sequence of character
- Arithmetic operators-Used to perform mathematical operations like addition,subtraction,multiplication and division

PROGRAM

```
#function to find digits
def digits_number(n):
    d1=(n//1000)%10 #finding digit1
    d2=(n//100)%10  #finding digit2
    d3=(n//10)%10   #finding digit3
    d4=n%10         #finding digit4
    return d1,d2,d3,d4

#function to find sum
def find_sum(d1,d2,d3,d4):
    sum=d1+d2+d3+d4
    print("Sum: ",sum)

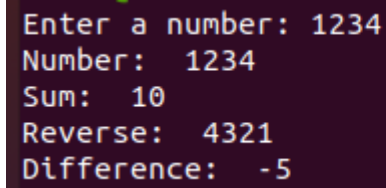
#function to find reverse
def reverse_of_number(d1,d2,d3,d4):
    reverse = (d4*1000) + (d3*100) + (d2*10) + d1
    print("Reverse: ",reverse)

#function to find reverse
def find_difference(d1,d2,d3,d4):
    difference=(d1*d3)-(d2*d4) #finding difference of products
    print("Difference: ",difference)

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

```
print("Number: ",num)
#calling functions
digit1,digit2,digit3,digit4 = digits_number(num)
find_sum(digit1,digit2,digit3,digit4)
reverse_of_number(digit1,digit2,digit3,digit4)
find_difference(digit1,digit2,digit3,digit4)
```

SAMPLE INPUT-OUTPUT



```
Enter a number: 1234
Number: 1234
Sum: 10
Reverse: 4321
Difference: -5
```

TEST CASES

Test case no.	Test description	Input	Expected Outcome	Actual Outcome	Result
1	Check for a four digit number	num=1234	Sum:10 Reverse:4321 Difference:-5	Sum:10 Reverse:4321 Difference:-5	Pass
2	Check for another number	num=5961	Sum:21 Reverse:1695 Difference:21	Sum:21 Reverse:1695 Difference:21	Pass

GITHUB LINK

[Click Here for the Code](#)

RESULT

Program executed successfully. Result obtained .

Area of Triangle

AIM

Develop a program to read the three sides of two triangles and calculate the area of both. Define a function to read the three sides and call it. Also, define a function to calculate the area. Print the total area enclosed by both triangles and each triangle's contribution (

THEORY

- Datatype-Data types are the classification or categorization of data items. It represents the kind of value that tells what operations can be performed on a particular data.
- Functions- Function is a group of related statements that performs a specific task.
- Expressions-An expression is a combination of operators and operands that is interpreted to produce some other value
- Built in function-Python has several functions that are readily available for use. These functions are called built-in functions.

PROGRAM

```
def read_sides(): #function to read the sides of triangle
    a = int(input("Enter length of side1: "))
    b = int(input("Enter length of side2: "))
    c = int(input("Enter length of side3: "))
    return(a,b,c)

def area_triangle(a,b,c): #function to find area of the two triangles
    s=(a+b+c)/2
    area= (s*(s-a)*(s-b)*(s-c))** 0.5
    return(area)

print("Triangle 1 : ") #printing area of both the triangles
A,B,C=read_sides() #calling function
print("Area of triangle 1 is ",area_triangle(A,B,C))
print("\n")
print("Triangle 2: ")
X,Y,Z=read_sides() #calling function
print("Area of triangle 2 is ",area_triangle(X,Y,Z))
print("\n")
totalarea= area_triangle(A,B,C)+area_triangle(X,Y,Z)
print("Total area= ",totalarea)
print("Contribution of triangle1 = ",(area_triangle(A,B,C)/totalarea)*100,"%")
print("Contribution of triangle2 = ",(area_triangle(X,Y,Z)/totalarea)*100,"%")
```

SAMPLE INPUT-OUTPUT

```
Triangle 1 :
Enter length of side1: 3
Enter length of side2: 4
Enter length of side3: 5
Area of triangle 1 is  6.0

Triangle 2:
Enter length of side1: 6
Enter length of side2: 7
Enter length of side3: 8
Area of triangle 2 is  20.33316256758894

Total area=  26.33316256758894
Contribution of triangle1 =  22.784957881909886 %
Contribution of triangle2  =  77.21504211809011 %
```

TEST CASES

Test Case no.	Test Description	Input	Expected Outcome	Actual Output	Result
1.	Check for different sides for both triangles	Triangle 1 side 1 = 3 side 2 = 4 side 3 = 5 Triangle 2 side 1 = 6 side 2 = 7 side 3 = 8	Area of triangle 1 is 6.0 Area of triangle 2 is 20.33316256758894 Total area = 26.3331625678894 Contribution of triangle1 = 22.784957881909886% Contribution of triangle2 = 77.21504211809011%	Area of triangle 1 is 6.0 Area of triangle 2 is 20.33316256758894 Total area = 26.3331625678894 Contribution of triangle1 = 22.784957881909886% Contribution of triangle2 = 77.21504211809011%	Pass
2	Check for two triangles having same sides	Triangle 1 side 1 = 3 side 2 = 4 side 3 = 5 Triangle 2 side 1 = 3 side 2 = 4 side 3 = 5	Area of triangle 1 is 6.0 Area of triangle 2 is 6.0 Total area = 12.0 Contribution of triangle1 = 50.0% Contribution of triangle2 = 50.0%	Area of triangle 1 is 6.0 Area of triangle 2 is 6.0 Total area = 12.0 Contribution of triangle1 = 50.0% Contribution of triangle2 = 50.0%	Pass

GITHUB LINK

[Click Here for the Code](#)

RESULT

Program executed successfully. Result obtained .

Employee payslip

AIM

Develop a program to read the employee's name, code, and basic pay and calculate the gross salary, deduction, and net salary according to the following conditions. Define a function to find each of the components. Finally, generate a payslip.

THEORY

- Conditional branching- A programming instruction that directs the computer to another part of the program based on the results of a comparison

PROGRAM

```
def gross_salary(BP,DA,HRA,MA): #function to find the gross salary
    GS=BP+DA+HRA+MA
    return(GS)
def pay_deduction(PT,PF,IT): #function to find the deduction
    D=PT+PF+IT
    return(D)
def net_salary(GS,D): #function to find the net salary
    netsalary=GS-D
    return(netsalary)
def employee_payslip():

    if(BP<10000):    #finding the components
        DA=(5/100)*BP
        HRA=(2.5/100)*BP
        MA=500
        PT=20
        PF=(8/100)*BP
        IT=0
    elif(BP<30000):
        DA=(7.5/100)*BP
        HRA=(5/100)*BP
        MA=2500
        PT=60
        PF=(8/100)*BP
        IT=0
    elif(BP<50000):
        DA=(11/100)*BP
        HRA=(7.5/100)*BP
```



```
    MA=5000
    PT=60
    PF=(11/100)*BP
    IT=(11/100)*BP
else:
    DA=(2.5/100)*BP
    HRA=(11/100)*BP
    MA=7000
    PT=80
    PF=(12/100)*BP
    IT=(20/100)*BP

GS = gross_salary(BP,DA,HRA,MA)
D = pay_deduction(PT,PF,IT)
netSalary = net_salary(GS,D)

print("\tPAYSALIP\t")

print("Name of the employee: ",name)
print("Basic pay:",BP)
print("Gross salary: ",GS)
print("Deduction: ",D)
print("Net Salary: ",netSalary)

name=input("Enter name of the employee: ") #input details
code=int(input("Enter the code: "))
BP=int(input("Enter basic pay: "))
print("\n")

employee_payslip()
```

SAMPLE INPUT-OUTPUT

```
Enter name of the employee: Hira Mohammed K
Enter the code: 11
Enter basic pay: 9000

      PAYSLIP
Name of the employee:  Hira Mohammed K
Basic pay: 9000
Gross salary:  10175.0
Deduction:   740.0
Net Salary:  9435.0
```

TEST CASES

Test Case no.	Test Description	Input	Expected Outcome	Actual Output	Result
1.	Check for Basic Pay below 10,000	Basic pay = 9,000	Gross salary:10175.0 Deduction:740.0 Net Salary:9435.0	Gross salary:10175.0 Deduction:740.0 Net Salary:9435.0	Pass
2.	Check for Basic Pay below30,000 and above10,000	Basic pay = 20,000	Gross salary:25000.0 Deduction:1660.0 Net Salary:23340.0	Gross salary:25000.0 Deduction:1660.0 Net Salary:23340.0	Pass
3.	Check for Basic Pay below50,000 and above30,000	Basic pay = 40,000	Gross salary:52400.0 Deduction:8860.0 Net Salary:43540.0	Gross salary:52400.0 Deduction:8860.0 Net Salary:43540.0	Pass
4.	Check for Basic Pay above50,000	Basic pay = 60,000	Gross salary:75100.0 Deduction:19280.0 Net Salary:55820.0	Gross salary:75100.0 Deduction:19280.0 Net Salary:55820.0	Pass

GITHUB LINK

[Click Here for the Code](#)

RESULT

Program executed successfully. Result obtained .

Happy sad numbers

AIM

Develop a program to perform the following task:

- a. Define a function to check whether a number is happy or not.
- b. Define a function to print all happy numbers within a range.
- c. Define a function to print first N happy numbers

THEORY

- While Loops- while loop is used to execute a block of statements repeatedly until a given condition is satisfied. And when the condition becomes false, the line immediately after the loop in the program is executed.
- for in Loop- For loops are used for sequential traversal. For example: traversing a list or string or array etc
- Nested loop- Its allows to use one loop inside another loop.

PROGRAM

```
def check_happy(n): #function to check whether a number is happy or sad
```

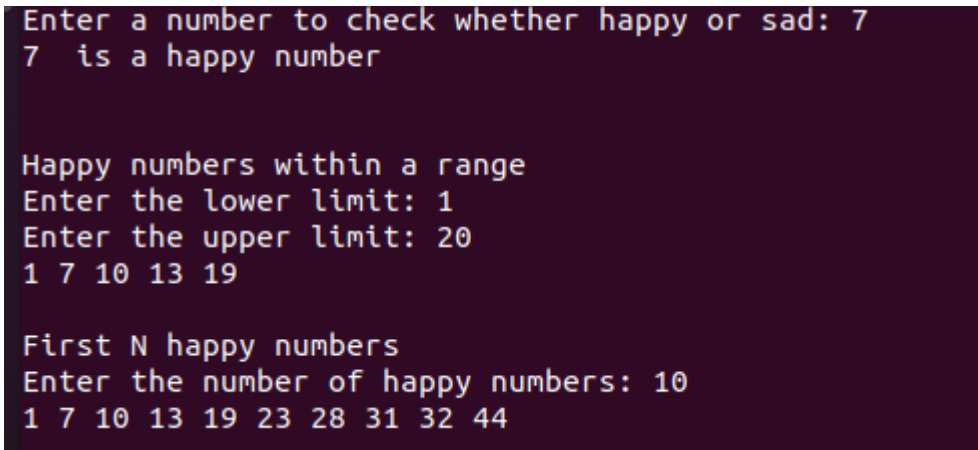
```
    for i in range(0,101):
        sum = 0
        while(n>0):
            digit=n%10
            n=n//10
            sum=sum+digit**2
        n=sum
        if(sum==1):
            return True
    else:
        return False
```

```
def range_happy(l,u): #function to print happy numbers within a range
    for i in range(l,u+1):
        if check_happy(i):
            print(i,end=" ")
```

```
def print_firstN(N): #function to print first n happy numbers
    count=0
    i=0
    while count<N:
        i=i+1
        if check_happy(i):
            print(i,end=" ")
            count=count+1
        else:
            continue

num=int(input("Enter a number to check whether happy or sad: "))
ans = check_happy(num)
if ans:
    print(num," is a happy number")
else:
    print(num," is a sad number")
print("\n")
print("Happy numbers within a range")
lower_limit=int(input("Enter the lower limit: "))
upper_limit=int(input("Enter the upper limit: "))
range_happy(lower_limit,upper_limit) #calling function
print("\n")
print("First N happy numbers")
no_of_terms=int(input("Enter the number of happy numbers: "))
print_firstN(no_of_terms) #calling function
```

SAMPLE INPUT-OUTPUT



```
Enter a number to check whether happy or sad: 7
7 is a happy number

Happy numbers within a range
Enter the lower limit: 1
Enter the upper limit: 20
1 7 10 13 19

First N happy numbers
Enter the number of happy numbers: 10
1 7 10 13 19 23 28 31 32 44
```

TEST CASES

Test case no.	Test description	Input	Expected Outcome	Actual Outcome	Result
1	Check whether number happy or sad	num = 7	7 is a happy number	7 is a happy number	Pass
2	Print happy numbers within a range	Lower limit =1 Upper limit = 20	1 7 10 13 19	1 7 10 13 19	Pass
3	Print N terms of happy numbers	N = 10	1 7 10 13 19 23 28 31 32 44	1 7 10 13 19 23 28 31 32 44	Pass

GITHUB LINK

[Click Here for the Code](#)

RESULT

Program executed successfully. Result obtained .

Operations on strings

AIM

Develop a program to read a string and perform the following operations:

1. Print all possible sub strings.
2. Print all possible sub strings of length K.
3. Print all possible sub strings of length K with N distinct characters.
4. Print sub string(s) of length maximum length with N distinct characters.
5. Print all palindrome sub strings.

Define function for each of the task

THEORY

- Strings-Strings are arrays of bytes representing Unicode characters
- Strings functions-capitalize() function,lower() function,title() function,casefold() function,upper() function,count() function,find() function,replace() function,swapcase() function,join() function
- Slicing-The slice() function returns a slice object. A slice object is used to specify how to slice a sequence

PROGRAM

```
def possible_substrings(str):  
#creating a function to find all the possible strings  
    print("Possible substrings are: ")  
    for i in range(0,len(str)+1):  
        for j in range(i+1,len(str)+1):  
            s=str[i:j]  
            print(s)
```

```
def desiredlen_substring(str,length):  
#creating a function to find all the possible strings  
with desired length  
    print("Strings of desired length are: ")  
    for i in range(0,len(str)+1):  
        for j in range(1,len(str)+1):  
            string=str[i:j]  
            if len(string)==length:  
                print(string)
```

```
def Klength_Ndistinct(str,length,distinct_char): #creating a function to find all the poss
with length K and N distinct characters
    print("Strings with length K and N distinct characters: ")
    for i in range(0,len(str)+1):
        for j in range(1,len(str)+1):
            string=str[i:j]
            if len(string) == length:
                result = set(string)
                if len(result) == distinct_char:
                    print(string)
```

```
def MaxLength_Ndistinct(str,distinct_char):
#creating a function to find all the possible strings
with maximum length and N distinct characters
    print("Strings with maximum length and N distinct characters: ")
    str_list = []
    for i in range(0,len(str)+1):
        for j in range(i+1,len(str)+1):
            string = str[i:j]
            d = set(string)
            if len(d) == distinct_char:
                str_list.append(string)
    length = len(max(str_list,key = len)) #finding the string with maximum length
    for i in str_list:
        if len(i)==length:
            print(i)
```

```
def string_palindrome(str): #creating a function to find all palindrome strings

    print("Paliandrome strings: ")
    for i in range(0,len(str)+1):
        for j in range(i+1,len(str)+1):
            string=str[i:j]
            r = string[::-1]
            if r == string:
                print(string)
```

```
#calling functions
string_input=input("Enter a string: ") #to input a string
possible_substrings(string_input)
print("\n")

K=int(input("Enter the desired length of the substring: "))
desiredlen_substring(string_input,K)
print("\n")

K=int(input("Enter the length: "))
N=int(input("Enter no of distinct characters: "))
Klength_Ndistinct(string_input,K,N)
print("\n")

N=int(input("Enter number of distinct characters: "))
MaxLength_Ndistinct(string_input,N)
print("\n")

string_palindrome(string_input)
```


SAMPLE INPUT-OUTPUT

```
Enter a string: abcacb
Possible substrings are:
a
ab
abc
abca
abcac
abcacb
b
bc
bca
bcac
bcacb
c
ca
cac
cacb
a
ac
acb
c
cb
b

Enter the desired length of the substring: 3
Strings of desired length are:
abc
bca
cac
acb

Enter the length: 4
Enter no of distinct characters: 3
Strings with length K and N distinct characters:
abca
bcac
cacb
```

```
Enter number of distinct characters: 2
Strings with maximum length and N distinct characters:
cac

Paliandrome strings:
a
b
bcacb
c
cac
a
c
b
```

TEST CASES

Test case no.	Test description	Input	Expected Outcome	Actual Outcome	Result
1	Print sub strings	abcacb	a ab abc abca abcac abcacb b bc bca bca bcac bcacb c ca cac cacb ac acb cb	a ab abc abca abcac abcacb b bc bca bca bcac bcacb c ca cac cacb ac acb cb	Pass
2	Print sub strings of length K	K = 3	abc bca cac acb	abc bca cac acb	Pass
3	Print sub strings of length K and N distinct characters	K = 4 N = 3	abca bcac cacb	abca bcac cacb	Pass
4	Print sub strings of maximum length with N distinct characters	N = 2	cac	cac	Pass
5	Print palindromic sub strings	abcacb	a b c bcab cac	a b c bcab cac	Pass

GITHUB LINK

[Click Here for the Code](#)

RESULT

Program executed successfully. Result obtained .

Pair of Rabbits

AIM

Suppose a newly born pair of rabbits, one male and one female, are put in a field. Rabbits can mate at the age of one month so that at the end of its second month, a female has produced another pair of rabbits. Suppose that our rabbits never die and that the female always produces one new pair every month from the second month. Develop a program to show a table containing the number of pairs of rabbits in the first N months

THEORY

- Critical thinking-Critical thinking involves approaching a problem or situation analytically and breaking it into separate components for more efficient problem-solving.
- Loops-The three types of loops in Python programming are: while loop, for loop, nested loops.
- Formatted io

PROGRAM

```
def rabbit_pair(n):
    #initialising the number of pairs of rabbits
    numPair = [1,1]
    print("\nMonth\t\tNumber of pairs ")
    for i in range(0,n):#iteration
        #printing data
        print(i+1,end="\t\t")
        print(numPair[i])
        numPair.append(numPair[i]+numPair[i+1])

#to input number of months
num = int(input("Enter number of months: "))
#calling function
rabbit_pair(num)
```

SAMPLE INPUT-OUTPUT

```
Enter number of months: 20

Month          Number of pairs
1              1
2              1
3              2
4              3
5              5
6              8
7             13
8             21
9             34
10            55
11            89
12           144
13           233
14           377
15           610
16           987
17          1597
18          2584
19          4181
20          6765
```

TEST CASES

Test case no.	Test description	Input	Expected Outcome	Actual Outcome	Result
1	Checking for random input	20	list of pairs for 20 months	list of pairs for 20 months	Pass
2	Checking for random input	10	list of pairs for 10months	list of pairs for 10 months	Pass

GITHUB LINK

[Click Here for the Code](#)

RESULT

Program executed successfully. Result obtained .

Operations on list of integers

AIM

Write a program to read a string containing numbers separated by a space and convert it as a list of integers. Perform the following operations on it.

1. Rotate elements in a list by 'k' position to the right
2. Convert the list into a tuple using list comprehension
3. Remove all duplicates from the tuple and convert them into a list again.
4. Create another list by putting the results of the evaluation of the function $f(x) = x^2 - x$ with each element in the final list
5. After sorting them individually, merge the two lists to create a single sorted list

THEORY

- List : It is a mutable container, that holds any type of data object and is specified using a pair of '['']'
- Tuple : A Tuple is a collection of Python objects separated by commas.
- Set : Sets are used to store multiple items in a single variable.
- List Comprehension : List comprehension offers a shorter syntax when you want to create a new list based on the values of an existing list.

PROGRAM

```
#function to convert string to list of integers
def string_to_int(stringList):
    newList = [int(i) for i in stringList]
    return newList

#function to rotate elements in a list by 'k' position to the right
def rotateElements(rot_list,k):
    print("After rotating",end = ":")
    print(rot_list[-k:]+rot_list[: -k])
    print("\n")

#function to convert the list into a tuple using list comprehension
def tuple_conversion(list_1):
    #converting the list into a tuple using list comprehension
    tuple_1 = tuple(list_1)
    print("Tuple: ",tuple_1)
    print(type(tuple_1))
```

```
    print("\n")
    return tuple_l

#function to remove all duplicates from the tuple and convert them into a list again
def removeDuplicates(tuple_l):
    #3.removing all duplicates
    tuple_l = tuple(set(tuple_l))
    list_l = list(tuple_l)
    print("After removing duplicates : ",list_l)
    print("\n")
    return list_l

#function to create another list by
putting the results of the evaluation
of the function with each element
in the final list
def func_append(list_l):
    #Creating another list by putting the results of the
    evaluation of the function with each element in the final list

    f = []
    for i in list_l:
        f.append((i**2)-i)
    print("List after evaluating function with each element ",end = " = ")
    print(f)
    print("\n")
    return f

#function to sort lists individually,
merge the two lists to create a single sorted list

def final_single_list(list_l,f):
    #sorting individually and merging to form a single list
    finalList = list_l + f
    finalList.sort()
    print("Final list ",end = " = ")
    print(finalList)
    print("\n")

#input string
input_string = input("Enter the numbers separated by space : ")
```

```
list_numbers = list(input_string.split(" "))
print("\n")
```

```
#calling function
list_of_int =string_to_int(list_numbers)
print("After conversion to list of integers: ")
print(list_of_int)
print("\n")
```

```
k_unit=int(input("Enter the position by which you want to rotate: "))
#calling function
rotateElements(list_of_int,k_unit)
#calling function
int_tuple = tuple_conversion(list_of_int)
#calling function
list_of_int = removeDuplicates(int_tuple)
#calling function
function_list = func_append(list_of_int)
#calling function
final_single_list(list_of_int,function_list)
```

SAMPLE INPUT-OUTPUT

```
Enter the numbers seperated by space : 1 2 1 1 3 2

After conversion to list of integers:
[1, 2, 1, 1, 3, 2]

Enter the position by which you want to rotate: 2
After rotating:[3, 2, 1, 2, 1, 1]

Tuple: (1, 2, 1, 1, 3, 2)
<class 'tuple'>

After removing duplicates : [1, 2, 3]

List after evaluating function with each element = [0, 2, 6]

Final list = [0, 1, 2, 2, 3, 6]
```

TEST CASES

Test case no.	Test description	Input	Expected Outcome	Actual Outcome	Result
1	Checking the output of program	string :1 2 1 1 3 2 Position by which you want to rotate : 2	Final sorted list [0,1,2,2,3,6]	Final sorted list [0,1,2,2,3,6]	Pass
2	Checking the output of program	string :5 6 2 2 5 Position by which you want to rotate : 4	Final sorted list [2,2,5,6,20,30]	Final sorted list [2,2,5,6,20,30]	Pass

GITHUB LINK

[Click Here for the Code](#)

RESULT

Program executed successfully. Result obtained .

File Handling

AIM

Read the file 'iris.json' as a text file :

- 1.Create a list having each line of the file as an element
- 2.Convert it into a list of dictionary objects
- 3.Show the details of all flowers whose species is "setosa"
- 4.Print the minimum petal area and max sepal area in each species
- 5.Sort the list of dictionaries according to the total area are sepal and petal

THEORY

- Json-JavaScript Object Notation (JSON) is a standard text-based format for representing structured data based on JavaScript object syntax
- Dictionary-A dictionary key can be almost any Python type, but are usually numbers or strings.

PROGRAM

```
import json

#function to create a list having each line of the file as an element
def read_as_list(filename):
    print("\t\tList having each line of the file as an element\t\t")
    print("\n")
    fp=open(filename,"r")
    data=fp.readlines()          #The file elements as list elements
    fp.close()
    return data

#function to convert it into a list of dictionary objects.
def read_as_dict(filename):
    print("\t\tList of dictionary objects\t\t")
    fp=open(filename,'r')
    dictionary=json.load(fp)
    return dictionary    #list of dictionary

#function to show the details of all flowers whose species is "setosa"
def print_details_setosa(data_dict):
    print("\t\tAll flowers whose species is setosa\t\t")
```

```
for i in data_dict:    #'i' is the dictionary element in the list
    if (i['species']=='setosa'):
        print('Sepal length: %f,%'(i['sepalLength']),
              'Sepal width : %f,  '%(i['sepalWidth']),
              'Petal length : %f,  '%(i['petalLength']),
              'Petal width : %f,  '%(i['petalWidth']))

#function to print the minimum petal area and max sepal area in each species
def get_areas(data_dict):
#to print the minimum petal area and max sepal area in each species
    list_species = list()
    for i in data_dict:
        list_species.append(i['species'])
    list_species = set(list_species)
    print('\n')
    areaSepal = list()
    areaPetal = list()
    for i in list_species:
        print(i.capitalize())
        print("_"*9)
        print(" ")
        for j in data_dict:
            if(j['species']==i):
                areaSepal.append(j['sepalLength']*j['sepalWidth'])
                areaPetal.append(j['petalLength']*j['petalWidth'])
        print("Maximum area of sepal: ",round(max(areaSepal),2))
        print("Minimum area of petal: ",round(min(areaPetal),2))
        print('\n')
        areaSepal.clear()
        areaPetal.clear()
    print('\n')

#function to sort the list of dictionaries according to
the total area are sepal and petal
def get_total_area(data_dict):
    #Sorting the list of dictionaries according to the total area of sepal and petal.
    print("\t\tList of dictionaries sorted according to the
total area of sepal and petal\t\t")
    print("\n")
    copyOfList = list()
    copyOfList = data_dict
```

```
for i in copyOfList:
    petal_area = (i["petalLength"]*i["petalWidth"])
    sepal_area = (i["sepalLength"]*i["sepalWidth"])
    total_area = (petal_area+sepal_area)
    i.update({'total_area':total_area})
sortedList = (sorted(copyOfList, key = lambda i:i['total_area'] ))
for i in sortedList:
    print(i)


#calling functions
data = read_as_list('iris.json')
for line in data:
    print(line)


data_dict = read_as_dict('iris.json')
for row in data_dict:
    print(row)


print_details_setosa(data_dict)
print("\n")
get_areas(data_dict)
get_total_area(data_dict)
```

SAMPLE INPUT-OUTPUT

```
Virginica
-----

Maximum area of sepal:  30.02
Minimum area of petal:  7.5

Setosa
-----

Maximum area of sepal:  25.08
Minimum area of petal:  0.11

Versicolor
-----

Maximum area of sepal:  22.4
Minimum area of petal:  3.3
```

TEST CASES

Test case no.	Test description	Input	Expected Outcome	Actual Outcome	Result
1	Reading each line as element of list	iris.json file	List having each line of the file as an element	List having each line of the file as an element	Pass
2	Converting it into a list of dictionary objects	iris.json file	List of dictionary objects	List of dictionary objects	Pass
3	Check output for setosa species	iris.json file	List of dictionary with key value setosa	List of dictionary with key value setosa	Pass
4	Check output for Minimum sepal area and Maximum petal area	iris.json file	Versicolor Maximum area of sepal:22.4 Minimum area of petal:3.3 Virginica Maximum area of sepal:30.02 Minimum area of petal:7.5 Setosa Maximum area of sepal:25.08 Minimum area of petal:0.11	Versicolor Maximum area of sepal:22.4 Minimum area of petal:3.3 Virginica Maximum area of sepal:30.02 Minimum area of petal:7.5 Setosa Maximum area of sepal:25.08 Minimum area of petal:0.11	Pass
5	Sorting the list of dictionaries according to the total area are sepal and petal	List of dictionary of json data	List sorted according to total area	List sorted according to total area	Pass

GITHUB LINK

[Click Here for the Code](#)

RESULT

Program executed successfully. Result obtained .

List of N Boxes

AIM

Write a program to create a class Box with data members length, breadth, height, area, and volume. Provide constructor that enables initialization with one parameter (for cube), two parameters (for square prism) three parameters (rectangular prism). Also, provide functions to calculate area and volume. Create a list of N boxes with random measurements and print the details of the box with maximum volume: area ratio.

THEORY

- Class : A class is a user-defined blueprint or prototype from which objects are created. Classes provide a means of bundling data and functionality together.
- Objects : An object is simply a collection of data (variables) and methods (functions) that act on those data.
- Constructor : Constructors allow you to create and properly initialize objects of a given class, making those objects ready to use.

PROGRAM

```
import random

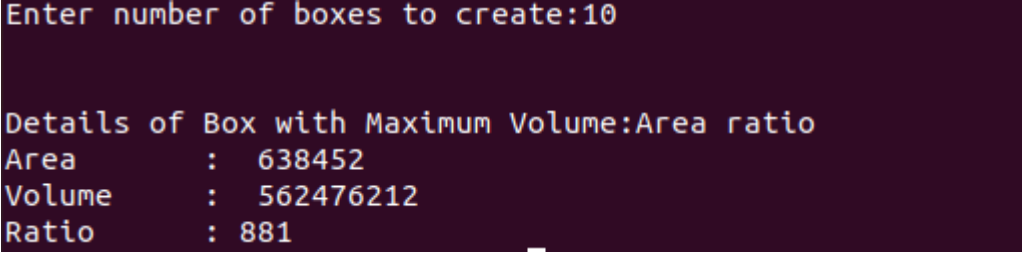
#creating class
class Box:
    def __init__(box,*arg):
        if len(arg) == 1:
            box.length = arg[0]
            box.breadth = arg[0]
            box.height = arg[0]
        elif len(arg) == 2:
            box.length = arg[0]
            box.breadth = arg[0]
            box.height = arg[1]
        else:
            box.length = arg[0]
            box.breadth = arg[1]
            box.height = arg[2]
    #function to find area
    def get_area(box):
        box.area = box.length*box.breadth
        return box.area
```

```
#function to find volume
def get_volume(box):
    box.volume = box.length*box.breadth*box.height
    return box.volume

#function to print details of box
def show(self):
    print("Area      : ",self.area)
    print("Volume     : ",self.volume)

N = int(input("Enter number of boxes to create:"))
print("\n")
box = [Box(random.randint(1,1000),random.randint(1,1000),
random.randint(1,1000)) for i in range(N)]
area = [i.get_area() for i in box]
volume = [i.get_volume() for i in box]
ratio = [x//y for x,y in zip(volume,area)]
index = ratio.index(max(ratio))
#To print the details of the box with maximum volume: area ratio
print("Details of Box with Maximum Volume:Area ratio")
box[index].show()
print("Ratio      :",max(ratio))
```

SAMPLE INPUT-OUTPUT



```
Enter number of boxes to create:10

Details of Box with Maximum Volume:Area ratio
Area      :  638452
Volume     :  562476212
Ratio      :  881
```

TEST CASES

Test case no.	Test description	Input	Expected Outcome	Actual Outcome	Result
1	Check for random numbers	N = 10	Details of box with maximum area:volume ratio printed	Details of box with maximum area:volume ratio printed	Pass
2	Check for another number	N = 8	Details of box with maximum area:volume ratio printed	Details of box with maximum area:volume ratio printed	Pass

GITHUB LINK

Click Here for the Code

RESULT

Program executed successfully. Result obtained .

Calculation of area of two shapes

AIM

Write a program to create a parent class, 3DShapes, with methods printVolume() and printArea(), which prints the Volume and Area, respectively. Create classes Cylinder and Sphere by inheriting 3DShapes class. Using these child classes, calculate and print the volume and area of a cylinder and sphere.

THEORY

- Inheritance : Inheritance refers to defining a new class with little or no modification to an existing class. The new class is called derived (or child) class and the one from which it inherits is called the base (or parent) class.

PROGRAM

```
#creating a class shapes3D(acts as a base class)
class Shapes3D:
#functions to print volume and area respectively
    def printVolume(self):
        print("Volume is ",self.volume)
    def printArea(self):
        print("Area is ",self.area)

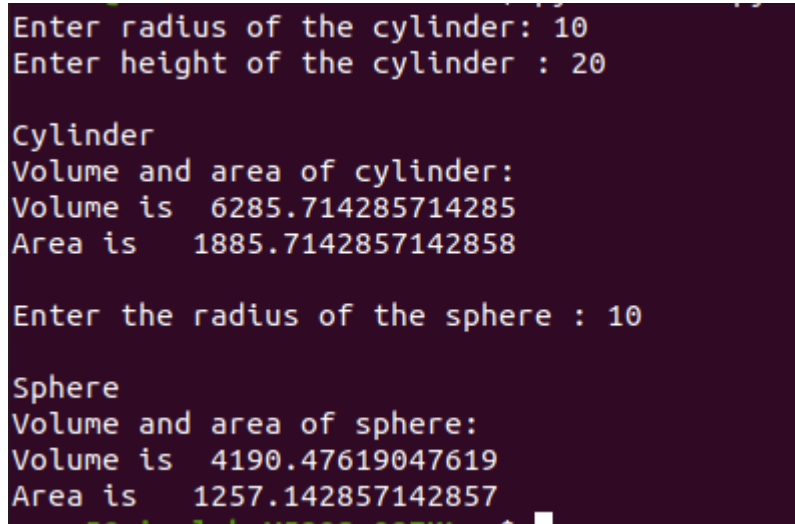
#creating a class cylinder(derived class of shapes3D)
class Cylinder(Shapes3D):
    def __init__(self,r,h):
        self.r = r
        self.h = h
        self.area = (2*(22/7)*r*h) + (2*(22/7)*r*r)
        self.volume = ((22/7)*r*r*h)

#creating a class cylinder(derived class of shapes3D)
class Sphere(Shapes3D):
    def __init__(self,r):
        self.r = r
        self.area = 4*(22/7)*r*r
        self.volume = (4/3)*(22/7)*(r**3)
```

```
#input dimensions of cylinder
r_cyclinder = int(input("Enter radius of the cylinder: "))
h_cylinder= int(input("Enter height of the cylinder : "))
#c_object is an object of Cylinder
c_object = Cylinder(r_cyclinder,h_cylinder)
print(" ")
print("Cylinder")
print("Volume and area of cylinder:")
c_object.printVolume()
c_object.printArea()

print(" ")
#input dimensions of shere
r_sphere = int(input("Enter the radius of the sphere : "))
#s_object is an object of Cylinder
s_object = Sphere(r_sphere)
print(" ")
print("Sphere")
print("Volume and area of sphere:")
s_object.printVolume()
s_object.printArea()
```

SAMPLE INPUT-OUTPUT



```
Enter radius of the cylinder: 10
Enter height of the cylinder : 20

Cylinder
Volume and area of cylinder:
Volume is  6285.714285714285
Area is    1885.7142857142858

Enter the radius of the sphere : 10

Sphere
Volume and area of sphere:
Volume is  4190.47619047619
Area is    1257.142857142857
```

TEST CASES

Test case no.	Test description	Input	Expected Outcome	Actual Outcome	Result
1	Check output for cylinder	Radius = 10 Height = 20	Volume :6285.714 Area:1885.714	Volume :6285.714 Area:1885.714	Pass
2	Check output for sphere	Radius = 10	Volume:4190.476 Area:1257.142	Volume:4190.476 Area:1257.142	Pass

GITHUB LINK

[Click Here for the Code](#)

RESULT

Program executed successfully. Result obtained .