Practical 1

Develop programs to understand the decision control structures of python.

A] Write a program to calculate the electricity bill (accept number of units from user) according to the following criteria: (if statement)

Unit amt per unit

First 100 units no charge

Next 100 units Rs 5 per unit

After 200 units Rs 10 per unit

(For example, if input unit is 350 than total bill amount is Rs2000)

Program:

```
unit = int(input("Enter the Biling unit: "))
if (unit<=100):
  print("No Charge")
if (unit>100 and unit<201):
  unit2=unit-100
  print("Bill amout Rs: ")
  print(unit2*5)
if (unit>100):
  u1 = unit-200
  u2 = 100*5
  u3 = u1*10
  print("Bill amout Rs: ")
  print(u3+u2)
```

B] Write a program to check whether the last digit of a number (entered by user) is divisible by 3 or not. (if else statement)

Program:

```
num=int(input("Enter a Number: "))
num1=num%10
if (num1%3==0):
  print("{} is Divisible by 3".format(num1))
else:
  print("{} is Not Divisible by 3".format(num1))
```

C] Write a program to accept percentage from the user and display the grade according to the following criteria: (elif statement)

Marks Grade

> 90 A

> 80 and <= 90 B

>= 60 and <= 80 C

below 60 D

Program:

```
p = int(input("Enter Your Percentage: "))
if (p>90 and p<=100):
  print("Your Grade is A")
elif (p>80 and p<=90):
  print("Your Grade is B")
elif (p>=60 and p<=80):
  print("Your Grade is C")
elif (p<60):
  print("Your Grade is D")
else:
  print("Enter corrrect percentage")
```

Practical 2

Develop programs to understand the looping statement

A. Read the string from the user and count the number of vowels in that string. (For using sequence)

Program:

```
str = input("Enter a string: ")
vowel=0
for i in str:
    if i=='a' or  i=='e' or  i=='i' or  i=='o' or  i=='u' or  i=='A' or  i=='E' or  i=='I' or  i=='O' or  i=='U':
      vowel = vowel+1
print("The no of vowels in string: ", vowel)
```

using list:

```
str = input("Enter a string: ")
l = ['a','e','i','o','u','A','E','I','O','U']
vowel=0
for i in str:
    if i in l:
      vowel = vowel+1
print("The no of vowels in string: ", vowel)
```

B. Write a program to find the factorial of a number. (for using range())

Program:

```
num=int(input("Enter a no.: "))
factorial=1
if num < 0:
   print("Sorry, factorial does not exist for negative numbers")
elif num == 0:
   print("The factorial of 0 is 1")
```

```
else:
  for i in range(1,num + 1):
     factorial = factorial*i
  print("The factorial of",num,"is",factorial)
```

output:

Enter a no.: 5

The factorial of 5 is 120

C. Write a program to find the sum of the digits of a number accepted from user

(while loop)

Program:

```
n=int(input("Enter a number:"))
tot=0
while(n>0):
   dig=n%10
   tot=tot+dig
   n=n//10
print("The total sum of digits is:",tot)
```

Practical 3

A] Write a program to demonstrate List sequence.

```
# Create an empty list

lst = []
print(type(lst))


# Read the elements of the list from the user

a = int((input("Enter the size of list")))
for i in range(0,a):
   b = (input("Add list element"))
```

```
    lst.append(b)
print(lst)

# Print elements using slice operator.

lst=[' apple', 'a', '1', 'internet']
print(lst[:])
print(lst[:3])
print(lst[3:])
print(lst[::-1])

# Update the elements of the list

a = ['Ansh', 2, 35, 'code', 2.5]
print("Length of the list", len(a))
print("Before list",a)
a[0] = "ash"
print("After update list",a)

# Add new elements in to the list – append() , insert()

a = ['Ansh', 2, 35, 'code', 2.5]
print("list",a)
print("Length of the list", len(a))
a.append("kash")
print("list",a)
print("Length of the list", len(a))
a.insert(1,25)
print("list",a)
print("Length of the list", len(a))

# Removing elements from the list – remove() del

a = ['Ansh', 2, 35, 'code', 2.5]
print("list",a)
print("Length of the list", len(a))
b = a.pop(1)
print("element deleted using pop()",b)
print("list",a)
a.remove(35)
print("element deleted using remove()",a)
del a[1:3]
print("element deleted using del keyword",a)

# Find out length of the list

a = ['Ansh', 2, 35, 'code', 2.5]
```

```
print("list",a)
print("Length of the list", len(a))

# Find out maximum between list

a = [1,5,8,6,2]
print("Largest element of list", max(a))


# Find out minimum between list

a = [1,5,8,6,2]
print("Smallest element of list", min(a))
```

B] Write a program to demonstrate Tuple sequence.

```
# Creating Tuple

a = ()
tup = tuple()
print(type(a))
print(type(tup))

# Read the elements of the tuple from the user

lst=[]
a = int((input("Enter the size of tuple")))
for i in range(0,a):
    b = (input("Add tuple element"))
    lst.append(b)
tup = tuple(lst)
print(tup)
print(type(tup))

# Print the elements using slice operator.

tup=('apple', 'realme', 'redmi', 'moto')
print(tup[:])
print(tup[:3])
print(tup[3:])
print(tup[::-1])
print(tup[-1:])

# Deleting Tuple

tup = ('apple', 'realme', 'redmi', 'moto')
```

```
print("Tuple",tup)
del tup

# Find out length of the Tuple

tup = ('apple', 'realme', 'redmi', 'moto')
print(" Length of Tuple",len(tup))

# Find out maximum between list

tup = (25,80,74,62,250)
print(tup)
print(" Maximum of Tuple",max(tup))


# Find out minimum between list

tup = (25,80,74,62,250)
print(tup)
print(" Minimum of Tuple",min(tup))
```

C] Demonstrate the dictionary

```
# Creating Dictionary

dic={}
my_dict={"Car1": "Audi", "Car2":"BMW",
      "Car3":"Mercidies Benz","Car4":"Range Rover"}
print(type(dic))
print(my_dict)
print(type(my_dict))

# Sort dictionary by values(Ascending and descending)

import operator

d = {1: 2, 3: 4, 4: 3, 2: 1, 0: 0}

s= sorted(d.items(), key=operator.itemgetter(1))

print('ascending order : ',s)

s1= dict( sorted(d.items(), key=operator.itemgetter(1),reverse=True))

print('descending order : ',s1)


# concatenate two dictionaries to create one

car1_model={'Mercedes':1960}
car2_model={'Audi':1970}
```

```
car={}
car.update(car1_model)
car.update(car2_model)
print(car)


# check whether the key exist or not

my_dict={"Car1": "Audi", "Car2":"BMW",
       "Car3":"Mercidies Benz","Car4":"Range Rover"}

key = input("Enter the key you want to search:\n")

if key in my_dict.keys():
   print("Present")
else:
   print("Not Present")


# iterate the keys of dictionary

my_dict={"Car1": "Audi", "Car2":"BMW",
       "Car3":"Mercidies Benz","Car4":"Range Rover"}
for x in my_dict:
   print(x)


# iterate the values of dictionary

my_dict={"Car1": "Audi", "Car2":"BMW",
       "Car3":"Mercidies Benz","Car4":"Range Rover"}
for x in my_dict.values():
   print(x)


# iterate the items of dictionary

my_dict={"Car1": "Audi", "Car2":"BMW",
       "Car3":"Mercidies Benz","Car4":"Range Rover"}
for x in my_dict.items():
   print(x)


# remove the specific values

my_dict={"Car1": "Audi", "Car2":"BMW",
       "Car3":"Mercidies Benz","Car4":"Range Rover"}
print("Original Dict \n",my_dict)
```

```
my_dict.pop('Car3')
print("Element removed using pop \n",my_dict)
del my_dict['Car2']
print("Element removed using del keyword \n",my_dict)
```

Practical 4

Function Scoping

```
#Local Scope

x=200

def myfunc():

  x = 300

  print("The local scope value of x:",x)

myfunc()

print("The global scope value of x:",x)Output:

The local scope value of x: 300

The global scope value of x: 200


# Global Scope

def demo():

    global S

    S="You are local but now you are global"

    print(S)


S = "You are Global"

demo()

print(S)


#Recursion

def factorial(x):

    """This is a recursive function
```

```
    to find the factorial of an integer"""
    if x == 1:
        return 1
    else:
        return (x * factorial(x-1))
num = 3
print("The factorial of", num, "is", factorial(num))
```

Practical 5

A] Class:

```
c=input("Enter colour of flower")


class flower:
    def colour(self,c):
        if c=="red":
            print("Colour is ",c,"then flower is rose")
        elif c=="yellow":
             print("Colour is ",c,"then flower is sunflower")
        elif c=="white":
             print("Colour is ",c,"then flower is lily")
        else:
            print("Colour not define in database")


C = flower()
C.colour(c)
```

B] Constructer:

```
class Addition:
    first = 0
```

```
    second = 0
    answer = 0

    def __init__(self,f,s):
        self.first = f
        self.second = s

    def display(self):
        print("First no = " , self.first)
        print("Second no = " , self.second)
        print("Addition = " , self.answer)

    def calculate(self):
        self.answer= self.first + self.second

obj = Addition(1000,2000)
obj.calculate()
obj.display()
```

C] Inheritance

```
class Universal:
    def surname(self):
        print("We are Indian coder community")

class Parent:
    def hair(self):
        print("Family have black hair")
```

```
class Child(Parent):
    def eyes(self):
        print("Eyes are dark brown")


class Grandchild(Child,Universal):
    def height(self):
        print("Height is 5.8")


a = Grandchild()
a.surname()
a.hair()
a.eyes()
a.height()
```

Practical 6

A] Search using DS:

```
#Linear search
class linear:
    ele=[]
    def get(self):
        self.a=int(input("Enter no of element you want to insert"))
        for i in range(0,self.a):
            b = int((input("Add list element")))
            self.ele.append(b)
    def search(self):
        c=int(input("Enter Element You want to search"))
        for i in range(0,self.a):
```

```
            if self.ele[i]==c:
                break;
        if i<self.a:
            print("Element found at index: ",i+1)
        else:
            print("Not Found!!!!")


s = linear()
s.get()
s.search()
```

B] Sort using DS:

```
#Bubble sort
class bubble:
    ele=[]
    def get(self):
        self.a=int(input("Enter no of element you want to insert"))
        for i in range(0,self.a):
            b = int((input("Add list element")))
            self.ele.append(b)
        print(self.ele)
    def show(self):
        print("Sorted list")
        print(self.ele)
    def sort(self):
        temp=0
        for i in range(0,self.a):
            for j in range(self.a-i-1):
```

```
            if self.ele[j] > self.ele[j+1]:
                temp=self.ele[j]
                self.ele[j]=self.ele[j+1]
                self.ele[j+1]=temp

b = bubble()
b.get()
b.sort()
b.show()
```

Practical 9

A] Try except:

```
#try & except:

n = int(input("enter 1st no: "))
m = int(input("enter 1st no: "))

try:
   x = n/m
except ZeroDivisionError:
      print("Sorry ! You are dividing by zero ")
else:
   print("Division of two nos. is : ",x)
```

B] Try & Finally:

```
#try & finally:

n = int(input("enter 1st no: "))
m = int(input("enter 1st no: "))
```

```
try:
   x = n/m
   print("Division of two nos. is : ",x)
except ZeroDivisionError:
      print("Sorry ! You are dividing by zero ")
finally:
   print("This statement execute anyways")
```

Practical no 10

Demonstrate implementation of the Anonymous Function Lambda.

A

```
# Function definition is here
sum = lambda arg1, arg2: arg1 + arg2;
# Now you can call sum as a function
print "Value of total : ", sum( 10, 20 )
print "Value of total : ", sum( 20, 20 )
```

B

```
# Python code to illustrate cube of a number
# showing difference between def() and lambda().
def cube(y):
return y*y*y
lambda_cube = lambda y: y*y*y
# using the normally
# defined function
```

```
print(cube(5))
# using the lambda function
print(lambda_cube(5))
```

Practical 11

Demonstrate implementation Mapping Functions over Sequences.

```
def mul(i):
return i * i
num = (3, 5, 7, 11, 13)
resu = map(mul, num)
print(resu)
# making the map object readable
mul_output = list(resu)
print(mul_output)
```

Practical 12: Demonstrate implementation functional programming tools such as filter and reduce.

A

```
scores = [66, 90, 68, 59, 76, 60, 88, 74, 81, 65]
def is_A_student(score):
return score > 75
over_75 = list(filter(is_A_student, scores))
print(over_75)
```

B

```
dromes = ("demigod", "rewire", "madam", "freer", "anutforajaroftuna", "kiosk")
```

```
palindromes = list(filter(lambda word: word == word[::-1], dromes))
print(palindromes)
```

C

```
# Python 3
from functools import reduce
numbers = [3, 4, 6, 9, 34, 12]
def custom_sum(first, second):
return first + second
result = reduce(custom_sum, numbers)
print(result)
```

D

```
from functools import reduce
numbers = [3, 4, 6, 9, 34, 12]
def custom_sum(first, second):
return first + second
result = reduce(custom_sum, numbers, 10)
print(result)
```

Practical 13

Demonstrate the Module Creation, Module usage, Module Namespaces,

Reloading Modules, Module Packages, Data Hiding in Modules.

A Module Creation

```
def add(a, b):
"""This program adds two
numbers and return the result"""
```

```
result = a + b

return result
```

```
import module

import math

print("The value of pi is", math.pi)
```

Import with renaming

```
import math as m

print("The value of pi is", m.pi)
```

from...import statement

```
from math import pi

print("The value of pi is", pi)
```

Run Code

Import all names

```
from math import *

print("The value of pi is", pi)
```