Date: 13/01/2021

Program NO: 1

AIM: Write a python program to find the biggest of three numbers entered by the user.

Program:

```
x = int(input("Enter the First Number: "))
y = int(input("Enter the second Number: "))
z = int(input("Enter the third Number: "))
if (x>y) and (x>z)
        largest = x
elif (y>x) and (y>z)
        largest = y
else:
        largest = z
Print("The largest Number is", largest)
```

Result: The program has been executed and the output is verified.

Output

```
Enter the First Number : 5
Enter the second Number : 8
Enter the third Number : 4
The largest Number is : 8
```

Date: 13/01/2021

Program NO: 2

AIM: Write a python program to find a square of a number entered by the user.

Program:

```
a= int(input("Enter the Number :::"))
sqa = a*a
Print("the square of the number is :: {0} ={1}".Format(a,sqa))
```

Result: The program has been excecuted and the output was verified.

Output

Enter the Number ::: 5

the square of the number is :: 5=25

Date: 13/01/2021

Program Number: 3

AIM: Write a python program to return the area of a circle using a Function.

Program:

```
import math
r=float(input("Enter the radius of the
        circle:"))
area=math.pi*r*r
print("%2f"%area)
```

Result: The program has been executed and the output was verified.

Output

Enter the radius of the circle: 5

78.54

Date: 25/01/2021

Program NO: 4

AIM: Form a list of vowels selected from a given word.

Program:

```
def checkvow(string, vowels)
    Final = [each for each in string if
             each in vowels]
    Print(len(final))
    Print(final)
string = "I am a Student"
vowels = "AaEeIiOoUu"
checkvow(string, vowels)
```

Result: The program has been executed and the output was verified.

Output

5

['I', 'a', 'a', 'U', 'e']

Date: 25/1/2021

Program NO: 5

AIM: Count the occurences of each word in a line of text.

Program:

```
def wordcount(str):
    counts = dict()
    words = str.split()
    for word in words:
        if word in counts:
            counts[word] += 1
        else:
            counts[word] = 1
    return counts

print(wordcount('I am a student', 'I am a programmer'))
```

Result: The program has been executed and the output was verified.

Output

{'i': 2, 'am': 2, 'a': 1, 'student': 1, ':': 1,
'programmer': 1}

Date: 25/1/2021

Program NO: 6

AIM: Store a list of first names. Count the occurances of 'a' within the list.

Program:

```
def countx(lst,x):
    count=0
    for ele in lst:
        if (ele==x):
            count=count+1
    return count
lst=['a','v','c','a']
x='a'
print('{} has occured {} times'.format(x,countx(lst,x)))
```

Result: The program has been executed and the output was verified.

output

a has occured 2 times.

Date: 25/1/2021

Program NO: 7

AIM: Enter two lists of integer. check whether list are of same length, check whether lists sums to same value, check whether any value occur in both.

Program:

```
list1 = [10,11,12,13,14,15]
list2 = [1,2,3,4,5]

len1 = len(list1)
len2 = len(list2)
if (len1==len2):
        print('both the list is are equal length')

else:
        Print('both the list is are not equal
                length')

tot1 = sum(list1)
tot2 = sum(list2)
if (tot1== tot2):
        Print ("sum of the both the list equal")

else:
        Print("sum of the both the list not equal")
```

```
For value in list1:
    IF value in list1:
        common=1
IF common ==0:
        Print("there are common element")
else:
        print("no common element")
```

Result: The program has been executed and the output was verified.

Output

Both the list is equal length

Sum of the both list not equal

no common element.

Program NO: 8 Date: 25/01/2021

AIM: Get a string from an input string where all occurences of first character replaced with '$', except first character [Eg: onion → oni$n]

Program:

```
def charchange(str1):
    cha = str1[0]
    str1 = str1.replace(cha,'$')
    str1 = cha + str1[1:]
    return str1
print(charchange('onion'))
```

Result: Program has been executed and the output was verified.

Output

Oni$h

Date: 25/1/2021

Program NO: 9

AIM: Create a string from given string where first and last character changed

[Eg: Python → nythop]

Program:

```
def changestring(str1):
    return str1[-1:] + str1[1:-1] + str1[:1]
Print (changestring('Python'))
```

Result: The program has been executed and output was verified.

output

python

Program NO: 10 Date: 25/1/2021

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

```
a = int(input("Input an integer : "))
n1 = int ("%s" %a)
n2 = int ("%s%s" %(a,a))
n3 = int ("%s%s%s" %(a,a,a))
print (n1+n2+n3)
```

Result: Program has been executed and the output was verified.

Output.

Input an integer: 6

738

Program NO: 11 Date: 25/1/2021

AIM: Sort dictionary in ascending and descending order.

Pragam:

```
y= {'orange':3, 'apple':2, 'mango':1}
l= list (y.items())
l.sort()
Print ('Ascending order is', l)
l=list(y.items())
l.sort (reverse = true)
Print ('Descending order is', l)
```

Result: The program has been executed and the output is verified.

Output :

Ascending order is [('apple', 2), ('mango', 1)
('orange', 3)]

Descending order is [('orange', 3) ('mango', 1)
('apple', 2)] .

Date: 25/11/2021

Program NO: 12

AIM: Merge two dictionaries.

Program:

```
fruits = {"apple", "orange", "tangerine"}
dryfruits = {"cashew", "almond", "pistachio"}
fruits.update(dryfruits)
Print (fruits)
```

Result: The program has been executed and the output was verified.

Output

{'cashew', 'almond', 'apple', 'orange',
'tangerine', 'pistachio'}

Program NO: 13 Date: 25/1/2021

AIM: Find gcd of two numbers.

Program:

```
def gcd(a,b):
    IF (b==0):
        return a
    else:
        return gcd(b, a%b)

a=60
b=48
Print ("the gcd of 60 and 48 is:", end=" ")
Print (gcd(60,48))
```

Result: The program has been executed and output was verified.

Output

The gcd of 60 and 48 is: 12

Program NO: 14

Date: 25/1/2021

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

Program

```
list = [11,22,33,44,55]
Print (list)
For i in list:
        if (i%2==0):
                list.remove(i)
Print ("list after removing even numbers:",
                                list)
```

Result: The program has been executed and the output was verified.

Output

[11,22,33,44,55]

list after removing even numbers:

[11,33,55]

Date: 27/1/2021

Program No: 15

AIM: Program to Find the factorial of a number.

Program:

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

num=5
Print("Factorial of", num, "is", factorial(num))
```

Result: The program has been executed and the output was verified.

Output

Factorial of 5 is 120

Program NO: 15

Date: 27/1/2021

AIM: Generate Fibonacci series of N terms.

Program:

```
def fibo(n):
    if n<=1:
        return n
    else:
        return (fibo(n-1) + fibo(n-2))

nterms = int(input("Enter the value of n:"))

if nterms <=0:
    Print("No Fibonacci sequence:")

else:
    print("fibonacci sequence:")
    for i in range(nterms):
        print(fibo(i))
```

Result: The program has been executed and output was verified.

Output

Enter the value of 'n' : 5

Fibonacci sequence :

0
1
1
2
3

Program NO: 14 Date: 27/1/2021

AIM: Find the sum of all items in a list

Program:

```
def sumlist(items):
    sumnumbers = 0
    for x in items:
        sumnumbers = sumnumbers + x
    return sumnumbers
print(sumlist([1,2,-12]))
```

Result: The program has been executed and the output is verified.

Output :

-9

Program NO: 18

Date: 27/1/2021

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

```
for i in range(1000,10000):
    for j in range(32,99):
        if i == j*j:
            num=str(i)
            if(int(num[0])%2==0 and int
            (num[1])%2==0 and int(num[2])
            %2==0 and int(num[3])%2==0):
                print(i)
```

Result: The program has been executed and the output was verified.

Output

4624
6084
6400
8464

Program NO: 1819

Date: 27/1/2021

AIM: Display the given pyramid with step number accepted from user.

Eg: N=4

```
1
2 4
3 6 9
4 8 12 16
```

Program:

```
rows=5
for i in range(1, rows)
    for j in range(1, i+1):
        Print(i*j, end=' ')
    Print()
```

Result: The program has been executed and the output was verified.

output

```
1
2 4
3 6 9
4 8 12 16
```

Program NO: 1920

Date: 27/1/2021

AIM: Count the numbe of characters (character frequency) in a string.

Program:

```
def charfrequency(str1):
    dict = {}
    for n in str1:
        keys = dict.keys()
        if n in keys:
            dict[n] = dict[n]+1
        else:
            dict[n]=1
    return dict.
print(charfrequency('frequency of character'))
```

Result: The program has beenexecuted and the output was verified.

output

{'F':2,'r':3,'e':3,'q':1,'u':1,'n'=1,
'c':3,'y':1,' ':2,'o':1,'h':1,'a':2,
't':1}

Program NO: 2021 Date: 27/1/2021

Aim: Add 'ing' at the end of a given string. If it already ends with 'ing', then add 'ly'.

Program:

```
def addstr(str1):
    length = len(str1)
    if length>2:
        if str1[-3:] == 'ing':
            str1 += 'ly'
        else:
            str1 += 'ing'
    return str1
Print (addstr ("good"))
Print (addstr ("morning"))
```

Result: The program has been executed and the output was verified.

Output

googing

Morningly

Program No: 22 Date: 27/1/2021

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

Program:

```
def longestword(wolist):
    word_len=[]
    for n in wolist:
        word_len.append([len(n),n])
    word_len.sort()
    return word_len[-1][0],word_len[-1][1]
result = longestword(["Apple","mango","Banana"])
Print("In Longest word:",result[1])
Print("length of the longest word:",
                    result[0])
```

Result: the program has been executed and the output was verified.

Output

longest word: Banana
length of the longest word: 6

Program NO: 23 Date: 27/1/2021

AIM: Construct Following pattern using nested loop:

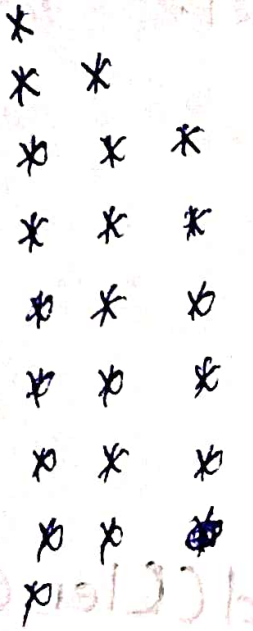
```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
* *
```

Program:

```
n=5
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().
```

Result: The program has been executed and the output was verified.

output



Date: 27/1/2021

Program NO: 24

AIM: Generate all factors of a number.

Program:

```
def factors(x):
    Print("the factor of ", x, "are:")
    For i in range(1, x+1):
        IF x % P == 0:
            Print(P)

num= int(input("Enter the number:"))
Factors(num)
```

Result: The program has been executed and the output was verified.

Output

Enter the number : 10

Factor of 10 are :

1
2
5
10

Date: 27/1/2021

Program NO: 25

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

Program:

```
square = lambda a : a*a
Print (square(5))

rectangle = lambda w, h : w*h
Print (rectangle (5, 3))

triangle = lambda b, h : .5*b*h
Print (triangle (5, 3))
```

Result: The program has been executed and the output was verified.

Output

25

15

7.5

Date: 27/2/21

Program NO: 26.

AIM: Python program to 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 find area and perimeter of figures by different importing statements.

Program:

Circle.py:

```
def area(r):
    print('Area of circle', r, '%.2f' %(3.14*
                                         r*r))

def circumference(r):
    print('circumference of circle', r,
          '%.2f' % (3.14*2*r))
```

rect.py

```
def area(a,b):
    print('Area of rectangle',a,'and',b,'%2f
                                    %(a*b))

def perimeter(a,b):
    Print('perimeter of rectangle',a,'and',
        b,'%.2F' % (2*(a+b)))
```

Sphere.py

```
def area(r):
    print('Area of sphere',r,'%.2f'%(4*(
                3.14*r*r)))

def perimeter(r):
    Print('Perimeter of sphere',r,'is','%
                .2f'%(2*3.14*r))
```

cuboid.py

```
def area(l,b,h):
    Print('area of cuboid',l,',',b,',',h,'is:',
          '%.2F' %(2*((l*b)+(b*h)+(l*h))))

def perimeter(l,b,h):
    Print('Perimeter of cuboid',l,',',b,
          ',',h,'%.2f' % (4*(l+b+h)))
```

Perimeter.py

```
import circle
From rect import *
From Graphics.--3Dgraphics import cuboid,sphe
l1 = float(input('Enter length of the rectangle :'))
b1 = float(input('Enter breadth of the rectangle:'))
 perimeter(l1,b1)
r = Float(input('Enter the radius of the circle :'))
cir.circumference(r)
l2=float(input('Enter the length of cuboid:'))
b2=float(input('Enter the breadth of cuboid :'))
```

```
b2 = float(input('Enter height of the cuboid:'))
cuboid.perimeter(l2,b2,b2)
r=float(input('Enter the radius of the
                sphere:'))
sphere.perimeter(r)
```

area.py

```
import circle
From rect import *
From Graphics.--3Dgraphics import cuboid,
                                  sphere
l1 = float(input('Enter the length of the rectangle:'))
b1=float(input('Enter the breadth of the rectangle:'))
  area(l1,b1)
r=Float(input('Enter the radius of the circle:'))
circleArea(r)
l2=float(input('Enter the length of the cuboid:'))
b2=float(input('Enter the breadth of the cuboid:'))
h2=float(input('Enter the height of the cuboid:'))
 cuboid.area(l2,b2,b2)
```

```
r=float(input('Enter the radius of the sphere:'))
sphere.area(r)
```

Result: The program has been executed and the output was verified.

Output

```
Enter length of the rectangle : 4
Enter breadth of the rectangle : 3
Perimeter of rectangle 4.0 and 3.0 is
14.00
Enter the radius of the circle : 2
Circumference of circle 2.0 is 12.56
Enter length of the cuboid : 5
Enter breadth of the cuboid : 4
Enter height of the cuboid : 3
Perimeter of cuboid 5.0, 4.0, 3.0 is
48.00
Enter the radius of the sphere : 2
Perimeter of sphere is 2.0 is 12.56
```

Enter length of the rectangle : 2
Enter breadth of the rectangle : 3
Area of rectangle 2.0 and 3.0 is 6.0
Enter the radius of the circle : 4
Area of circle 4.0 is 50.24
Enter length of the cuboid : 4
Enter breadth of the cuboid : 7
Enter height of the cuboid : 2
Area of cuboid 4.0, 7.0, 2.0 is 100.00
Enter the radius of the sphere : 1
Area of sphere 1.0 is 12.56

Program NO: 27 Date: 17/2/2021

AIM: Display Future leap years from current year to a Final year entered by the user.

Program:

```
import datetime
a=datetime datetime.now()
a= int (a.year)
b= int(input("enter the Final year"))
print ("leap years")
for i in range(a,b+1):
    IF(i%4==0)
    print (i)
```

Result: The program has been executed and the output was verified.

Output

Enter the final year : 2040

leap years:

2024
2028
2032
2036
2040.

Program No: 28 Date: 27/2/21

AIM: Generate positive list of numbers from a given list of integers.

Program:

```
list = [10,-20,0,30,-40,50]
for num in list:
    if num >=0:
        Print(num)
```

Result: The program has been executed and the output was verified.

Output

10
0
30
50

Program NO: 28 Date: 27/2/2021

AIM: Accept a file name from usa and print extension of that.

Program:

```
Filename = Input ("input the Filename:")
F_extns = Filename.split (".")
print ("The extension of the file is:" +
repr (F_extns[-1]))
```

Result: The program has been executed and the output was verified

Output

Input the file name: file.txt

The extension of the file is: 'txt'

Date: 27/2/2021

Program NO: 30

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

Program:

```
values = input("input some comma
        separated color :")
list = values.split(",")
Print("%s %s" %(list[0], list[-1]))
```

Result: The program has been executed and the output was verified.

Output

Input some comma seperated colors:

red, black, orange.

red, orange.

Program NO: 31

Date: 14/2/2021

AIM: Print out all colors From color_list1 not contained in color list 2.

Program:

```
color_list_1 = set(["white", "Black", "Red",
                    "gray"])
color_list_2 = set(["Red", "Green", "orange"])
Print(color_list_1.difference(color_list_2))
```

Result: The program has been executed and the output was verified.

Output

{'Black', 'gray', 'White'}

Date: 27/2/2021

Program NO: 32

AIM: Create a single string seperated with space from two strings by swapping the character position.1.

Program:

```
a = input("Enter the fast string:")
b = input("Enter the second string:")
x = a[0:1]
a = a.replace(a[0:1], b[0:1])
b = b.replace(b[0:1], x)
Print(a,b)
```

Result: The program has been executed. and the output was verified.

output

enter the first string: good

Enter the second string: Morning

mood gaining

Program NO: 33 Date: 27/2/2021

AIM: Create Rectangle class with attributes length and breadth and Methods to Find area and peimeter. Compare two rectangle object by their area.

Program:

```
Class Rectangle:
    def __init__(self,l,b):
        self.length = l
        self.breadth = b
    def area(self):
        return self.length * self.
                                breadth

    def peimeter(self):
        return 2 * (self.length +
                    self.breadth)

    def comparison(self, obj):
        if self.area() > obj.area():
            Print("The highest
        area of rectangle with
```

```
length=', self.length,' and breadth=
', self.breadth)

    elif self.area()<obj.area():
        Print("The highest area
    of rectangle with length=',obj.
    length,' and breadth=',obj.
    breadth)
    else:
        Print('They have equal area')

r1 = Rectangle(10,20)
r2 = Rectangle(50,60)
r1.Comparison(r2)
```

Result: The program has been executed and the output was verified.

output

The highest area of Rectangle with length = 50 and breadth = 60.

Date: 27/2/21

Program NO: 34

AIM: Create a bank account with member account number, name, type of account and Balance. write constructor and Methods to deposit at the bank and withdraw an amount from the bank.

Program:

```
class Account:
    def __init__(self,a,n,t,b):
        self.no=a
        self.name=n
        self.type=t
        self.balance=b
    def deposit(self,a):
        self.balance=self.balance+a
        Print('Rs.',a,'deposited,
        current balance is:Rs.',
```

```
        self.b  .  )
    def withdraw(self,a):
        IF self.balance >= a:
            Self.balance = a
            Print('Rs.',a,' withdraw.
            current balance is: Rs.', self.
            a !  .  )
        else:
            Print('insufficient balance
            to make this transaction')
a = int(input('Enter account number:'))
n = input('Enter the name of the
        account holder:')
t = input('Enter account type')
b = float(input('Enter your balance:'))
ac1 = Account(a,n,t,b)
ac1.deposit(float(input('Enter amount
        to deposit:')))
ac1.withdraw(float(input('Enter amount
        to withdraw:')))
```

Result: The program has been executed and the output was verified.

Output

Enter account number: 1200637

Enter name of the account holder: Aa

Enter account type: current

Enter your balance: 20000

Enter amount to deposit: 1500

Rs. 1500.0 deposited, current balance is:

Rs: 21500

Enter amount to withdraw: 1500

Rs. 1500 is withdraw, current balance:

Rs. 20000

Program NO: 35 Date: 17/2/2021

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

Program:

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

    def __lt__(self, other):
        If self.area < other.area:
            Print("The lesser area
        rectangle with length=', self.
        length, 'and width=', self.
        width)
        elif other.area < self.area:
            Print('The lesser area
        Rectangle with length=',
```

```
        other.length, 'and width=', other.
          width)
    else:
            Print('They have equal area')
Rect1 = Rectangle(l,w)
l = float(input('Enter length of the
    first rectangle:'))
w = float(input('Enter width of the
    second rectangle'))
Rect2 = Rectangle(l,w)
Rect1 < Rect2
```

Result : The program has been executed and the output was verified.

Output

Enter length of the first rectangle: 30
Enter width of the first rectangle: 20

Enter length of the second rectangle: 50
Enter width of the second rectangle: 10

The lesser area rectangle with length = 30 and width = 20

Date: 27/2/21

Program NO: 35

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

Program:

```
class Time:
    def __init__(self, h, m, s):
        self.hour = h
        self.minute = m
        self.second = s
    def __add__(self, other):
        hour = int((self.hour + other.hour)
        % 24 + (self.minute + other.minute
        /60) minute = int ((self.minute
        + other.minute) %60 + ((self.
        second + other.second) /60))
        second = int ((self.second +
        other.second) %60)
```

```
Print ('Time=', hour, 'hour:', minute,
       'minut:', second, 'second')

T1 = Time(10,20,15)
T2 = Time(16,45,56)
T1 + T2
```

Result: The program has been executed and the output was verified.

output

Time= 3 hour : 6 minute : 11 second

Date: 4/2/2021

Program NO: 34

AIM: 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.

Program:

```
class publisher:
    def __init__(self,name):
        self.name=name
    def show(self):
        pass
```

```
class Book(publisher):
    def __init__(self, title, author, name):
        self.title = title
        self.author = author
        Publisher.__init__(self, name)
    def show(self):
        pass.

class py-thon (Book):
    def __init__(self, P, no, title, author,
                 name):
        self.price = P
        self.no_of_pages = no
        Book.__init__(self, title,
            author, name)

    def show(self):
        Print('Title of the book:',
              self.title)
        Print('Author of the book:',
              self.author)
        Print('publisher of the
              book:', self.name)
```

```
        print('price of the book:'self.price)
        print('no. of pages in the book:'
              self. no_of_page)
P1 = Python (400, 100, 'python', 'G V Rossum',
     'books pvt')
P1. show()
```

Result: The program has been executed and the output was verified.

output

Title of the book: Python

Author of the book: G V Rossum

Publisher of the book: book pvt.

Price of the book = 400

No. of pages in the book: 100

Date: 21/2/2021

Program NO: 38

AIM: Write a python program to read a file line by line and store it into a list.

Program:

```
def file_read(fname):
    with open(fname) as f:
        c=f.readlines()
        print(c)
        print(len(c))
file_read("demo.txt")
```

Text file: demo.txt

```
This is a python program
to read line by line
and store it into a list
```

Result: Program has been executed and the output was verified.

Output

['This is a python program \n', 'to read a file line by line \n', 'and store it into a list \n']

3.

Program NO: 39

Date: 21/2/2021

AIM: Python program to copy odd lines of one file to another.

Program:

```
a= open ('demo.txt', 'r')
b= open ('nextdemo.txt', 'w')
c = a.readlines ()
type (c)
For i in range (0, len(c)):
        IF (i%2 != 0):
            b.write(c[i])
        else:
            pass

b. close()
b= open ('next demo.txt', 'r')
cont1 = b.read ()
Print (cont1)
a. close ()
b. close()
```

Text file: demo.txt

This is a python program
to read a file line by line
and store it into a list

next demo.txt

To read a file line by line.

Result: The program has been executed
and output is verified.

Output

To read a file line by line.

Program NO: 40

Date: 21/2/2021

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

Program:

```
import csv
with open('departments.csv', newline='')
  as csvfile:
  data = csv.reader(csvfile, delimiter=',',
  quotechar='|')
    For row in data:
        print(', '.join(row))
```

departments.csv:

department_id, department_name, Manager_id, location_id

10, Administration, 200, 1700

20, Marketing, 114, 1800

30, Human Resources, 203, 1700

40, Purchasing, 121, 2400

50, shipping, 103, 1500
60, IT, 204, 1400
70, Public Relation, 201, 2700
80, Sales, 145, 2500.

Result: The program has been executed and output was verified.

Output

department_id, department_name, manager_id, loca

10, Administration, 200, 17

20, Marketing, 114, 1800

30, Human_resources, 203, 1700

40, Purchasing, 121, 2400

50, Shipping, 103, 1500

60, IT, 204, 1400

70, Public_relation, 201, 2700

80, Sales, 145, 2500

Program NO: 41 Date: 20/2/2021

AIM: write a python program to read specific column of a given csv file and print the content of the columns.

Program:

```
import csv
with open('departments.csv', newline='')
as csvfile:
    data = csv.DictReader(csvfile)
    Print(" ID Department name")
    For row in data:
        Print(row['department_id'],
        row['department_name'])
```

departments.csv:

department_id, department_name, manager_id, location_id

10, Administration, 200, 1700

20, Marketing, 201, 1800

30, Human Resource, 203, 2400

40, Purchasing, 114, 1700

50, Shipping, 121, 1500

60, IT, 103, 1400

70, Public Relations, 204, 2700

80, Sales, 145, 2500.

Result: The program has been executed and the output was verified.

Output

10	Administration
20	Marketing
30	Human Resources
40	Purchasing
50	Shipping
60	IT
70	Public Relations
80	Sales.

Program NO: 42

Date: 21/2/2021

AIM: 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.

Program:

```
import csv
csv_columns = ['id','column1','column2',
               'column3']
dict_data = {'id':['1','2','3'],
             'column1':[33,25,56],
             'column2':[35,30,30],
             'column3':[21,40,55]}
csv_file = "temp.csv"
try:
    with open(csv_file,'w') as csvfile:
        writer = csv.DictWriter(csvfile,
        fieldnames = csv_columns)
```

```
        writer.writeheader()
        For data in dictdata:
            writer.writerow(dictdata)

except IOError:
        print("I/O Error")

data = csv.DictReader(open(csv_file))
print("csv file dictionary:\n")
For row in data
        print(row)
```

Department.csv:

department_id, department_name, manager_id, location id

10 , Administration, 200, 1700

20 , Marketing , 201, 1800

30 , Human Resources, 203, 2400

40 , Purchasing , 114, 1700

50, Shipping, 121, 1500

60, IT, 103, 1400

70, Public Relation, 204, 2700

80, Sales, 145, 2500

Temp.csv :

id, column1, column2, column3

"['1','2','3']", "[33,25,56]", "[35,30,30]", "[21,40,55]"

"['1','2','3']", "[33,25,56]", "[35,30,30]", "[21,40,55]"

"['1','2','3']", "[33,25,56]", "[35,30,30]", "[21,40,55]"

Result: The program has been executed and the output is verified.

Output

csv file dictionary:

{'id': "['1','2','3']", 'column1': [33,25,56],
'column2': '[35,30,30]', 'column3': '[21,40,

{'id': "['1','2','3']", 'column1': '[33,25,56]',
'column2': '[35,30,30]', 'column3': '[21,40,55

{'id': "['1','2','3']", 'column1': '[33,25,56]',
'column2': '[35,30,30]', 'column3': '[21,40,55]

{'id': "['1','2','3']", 'column1': '[33,25,56]',
'column2': '[35,30,30]', 'column3': '[21,40,55