**SREE NARAYANA GURUKULAM COLLEGE OF ENGINEERING**

**KADAYIRUPPU, KOLENCHERY 682 311**

**(Affiliated to APJ Abdul Kalam Technological University)**

**ACADEMIC YEAR 2021-22**

****

**20 MCA 132 | PROGRAMMING LABORATORY RECORD**

***Submitted by***

**RAHUL P R**

**REG NO: SNG21MCA-2028**

*in partial fulfillment for the award of the degree in*

***MASTER OF COMPUTER APPLICATIONS***

**SREE NARAYANA GURUKULAM COLLEGE OF ENGINEERING KADAYIRUPPU, KOLENCHERY 682 311   
(Affiliated to APJ Abdul Kalam Technological University)   
**

**20 MCA 132 PROGRAMMING LABORATORY RECORD**

*Certified that this is a Bonafide record of practical work done by* ***Rahul P R*** *to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the Degree in Master of Computer Applications of Sree Narayana Gurukulam College of Engineering done during the Academic year 2021-22.*

Kadayiruppu Course Instructor

Date: ……………….

Head of the Department

**Prof.Dr. SANDHYA R**

Submitted for University Practical Examination

**Reg No:** **SNG21MCA-2028** **on** ------------

External Examiner Internal Examiner

| **SL NO.** | **DATE** | **NAME OF EXPERIMENT** | **PAGE NO.** | **REMARK** |
| --- | --- | --- | --- | --- |
| **I** | **CO1** | | | |
| **1** | **24/11/21** | **Familiarizing Text Editor, IDE, Code Analysis Tools etc** | **1** |  |
| **2** | **24/11/21** | **Leap Year** | **3** |  |
| **3** | **24/11/21** | **List comprehensions** | **4** |  |
| **4** | **24/11/21** | **Occurrences of each word** | **6** |  |
| **5** | **24/11/21** | **Prompt the user for a list of integers.** | **7** |  |
| **6** | **24/11/21** | **Store a list of first names.** | **8** |  |
| **7** | **24/11/21** | **Checking list are of same length,sums to same value,any value occur in both** | **9** |  |
| **8** | **24/11/21** | **Get a string from an input string and replacing a character** | **11** |  |
| **9** | **24/11/21** | **Create a string from given string where first and last characters exchanged.** | **12** |  |
| **10** | **24/11/21** | **Accept the radius from user and find area of circle** | **13** |  |
| **11** | **29/11/21** | **Find biggest of 3 numbers entered** | **14** |  |
| **12** | **29/11/21** | **Accept a file name from user and print extension of that** | **15** |  |
| **13** | **29/11/21** | **Create a list of colors,Display first and last colors.** | **16** |  |
| **14** | **29/11/21** | **Accept an integer n and compute n+nn+nnn** | **17** |  |
| **15** | **29/11/21** | **Print out all colors from color-list1 not contained in color-list2** | **18** |  |
| **16** | **29/11/21** | **Create a single string separated with space from two strings by swapping the character at position 1.** | **19** |  |
| **17** | **29/11/21** | **Sort dictionary in ascending and descending order** | **20** |  |
| **18** | **29/11/21** | **Merge two dictionaries** | **21** |  |
| **19** | **29/11/21** | **Find gcd of 2 numbers.** | **22** |  |
| **20** | **29/11/21** | **From a list of integers, create a list removing even numbers**. | **23** |  |
| **II** | **CO2** | | | |
| **1** | **1/12/21** | **Program to find the factorial of a number** | **24** |  |
| **2** | **1/12/21** | **Generate Fibonacci series of N terms** | **25** |  |
| **3** | **1/12/21** | **Find the sum of all items in a list** | **26** |  |
| **4** | **1/12/21** | **Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square.** | **27** |  |
| **5** | **1/12/21** | **Display the given pyramid with step number accepted from user** | **28** |  |
| **6** | **1/12/21** | **Count the number of characters (character frequency) in a string** | **29** |  |
| **7** | **8/12/21** | **Add ‘ing’ at the end of a given string. If it already ends with ‘ing’, then add ‘ly’** | **30** |  |
| **8** | **8/12/21** | **Accept a list of words and return length of longest word** | **31** |  |
| **9** | **8/12/21** | **Construct pattern using nested loop** | **32** |  |
| **10** | **8/12/21** | **Generate all factors of a number. def print\_factors(x):** | **34** |  |
| **11** | **8/12/21** | **Write lambda functions to find area of square, rectangle and triangle.** | **35** |  |
| **III** | **CO3** | | | |
| **1** | **15/12/21** | **Work with built-in packages** | **36** |  |
| **2** | **15/12/21** | **Creation of packages** | **41** |  |
| **IV** | **CO4** | | | |
| **1** | **9/1/22** | **Compare two Rectangle objects by their area** | **44** |  |
| **2** | **9/1/22** | **Create a Bank account with members account number, name, type of account and balance.** | **46** |  |
| **3** | **9/1/22** | **Overload ‘<’ operator to compare the area of 2 rectangles.** | **49** |  |
| **4** | **9/1/22** | **Overload ‘+’ operator to find sum of 2 time** | **51** |  |
| **5** | **9/1/22** | **Use base class constructor invocation and method overriding.** | **53** |  |
| **V** | **CO5** | | | |
| **1** | **30/1/22** | **Write a Python program to read a file line by line and store it into a list.** | **55** |  |
| **2** | **30/1/22** | **Python program to copy odd lines of one file to other** | **56** |  |
| **3** | **30/1/22** | **Write a Python program to read each row from a given csv file and print a list of strings.** | **57** |  |
| **4** | **30/1/22** | **Write a Python program to read specific columns of a given CSV file** | **58** |  |
| **5** | **30/1/22** | **Write a Python program to write a Python dictionary to a csv file.** | **59** |  |

1. **COURSE OUTCOME 1(CO1)**

**PROGRAM NO:** 1

**DATE**:24/11/2021

**AIM:** Familiarizing Text Editor, IDE, Code Analysis Tools etc // Use any IDE like PyCharm, PyDev…

A text editor is a tool that allows a user to create and revise documents in a computer.

An integrated development environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development.

An IDE normally consists of at least a source code editor,build automation tools a nd a debugger

Source code analysis tools, also known as Static Application Security Testing (SAST) Tools, can help analyze source code or compiled versions of code to help find security flaws.

**Top Python IDE’s**

* + PyCharm
  + Spyder
  + Eclipse PyDev
  + Wing
  + IDLE

**PyCharm**

In industries most of the professional developers use PyCharm and it has been considered the best IDE for python developers.It was developed by the Czech company JetBrains and it’s a cross-platform IDE.

* It is considered as an intelligent code editor, fast and safe refactoring, and smart code.
* Features for debugging, profiling, remote development, testing the code, auto code completion, quick fixing, error detection and tools of the database.
* Support for Popular web technologies, web frameworks, scientific libraries and version control.

**PROGRAM NO:** 2

**DATE:**24/11/2021

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

**PROGRAM**

print("Future Leap years")

print("enter the starting year")

start=int(input())

print("enter the final year")

end=int(input())

print("start year",start,"\nend year",end)

for y in range(start,end):

if (y % 4 == 0) and (y % 100 != 0) or (y % 400 == 0):

print(y)

**OUTPUT**

Future Leap years

enter the starting year

1692

enter the final year

1710

start year 1692

end year 1710

1692

1696

1704

1708

**PROGRAM NO:** 3

**DATE:24/11/2021**

**AIM:** List comprehensions:

* **Generate positive list of numbers from a given list of integers**

**PROGRAM**

list1=[2,3,-4,-5,-7,8]

list2=[]

for i in list1:

if i>0:

list2.append(i)

print("Resultant list",list2)

**OUTPUT**

Resultant list [2, 3, 8]

* **Square of N number**

**PROGRAM**

n=int(input("Enter the limit:"))

list1=[]

sq=1

for i in range(1,n+1):

sq=i\*i

list1.append(sq)

print("Result:",list1)

**OUTPUT**

Enter the limit:4

Result: [1, 4, 9, 16]

* **Form a list of vowels selected from a given word**

**PROGRAM**

list1=[]

w1=input("Enter the word:")

for i in w1:

if i in 'aeiouAEIOU':

list1.append(i)

print("Resultant list:",list1)

**OUTPUT**

Enter the word:hello world

Resultant list: ['e', 'o', 'o']

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

**PROGRAM**

w1=input("Enter the word:")

for i in w1:

print(i," ")

print(ord(i),"\n")

**OUTPUT**

Enter the word:hai

h

104

a

97

i

105

**PROGRAM NO:** 4

**DATE:24/11/2021**

**AIM:**Count the occurrences of each word in a line of text

**PROGRAM**

str1=input("Enter the text:")

list1=str1.split(" ")

print(list1)

for i in list1:

count=list1.count(i)

print(" ",i," ",count)

**OUTPUT**

Enter the text:how is is

['how', 'is', 'is']

how 1

is 2

is 2

**PROGRAM NO:** 5

**DATE:24/11/2021**

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

**PROGRAM**

list1=[]

n1=int(input("Enter the limit:"))

for i in range(n1):

n2=int(input("Enter the number:"))

if n2>100:

list1.append("over")

else:

list1.append(n2)

print(list1)

**OUTPUT**

Enter the limit:3

Enter the number:300

Enter the number:3

Enter the number:5

['over', 3, 5]

**PROGRAM NO:** 6

**DATE:24/11/2021**

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

**PROGRAM**

list1=["a","b","c","a","a"]

print("List:\n",list1)

print("Occurence of a:",list1.count('a'))

**OUTPUT**

List:

['a', 'b', 'c', 'a', 'a']

Occurence of a: 3

**PROGRAM NO:** 7

**DATE:24/11/2021**

**AIM:**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

**PROGRAM**

list1=[]

list2=[]

s1=0

s2=0

n1=int(input("Enter the number of elements in list1:"))

for i in range(n1):

x=int(input("Enter the elements:"))

s1=s1+x

list1.append(x)

n2=int(input("\nEnter the number of elements in list2:"))

for i in range(n2):

x=int(input("Enter the elements:"))

s2=s2+x

list2.append(x)

print("\nlist1\n",list1)

print("\nlist2\n",list2)

if len(list1)==len(list2):

print("\nLength of 2 list is same")

else:

print("\nLength of 2 list is not same")

if s1 == s2:

print("Sum is equal")

else:

print("Sum is not equal")

print("common elements:",set(list1).intersection(set(list2)))

**OUTPUT**

Enter the number of elements in list1:3

Enter the elements:3

Enter the elements:5

Enter the elements:7

Enter the number of elements in list2:3

Enter the elements:6

Enter the elements:7

Enter the elements:3

list1

[3, 5, 7]

list2

[6, 7, 3]

Length of 2 list is same

Sum is not equal

common elements: {3, 7}

**PROGRAM NO:** 8

**DATE:24/11/2021**

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

**PROGRAM**

string1=input("Enter the string:")

print("Old string:",string1)

first=string1[0]

print("New String:",first+string1[1:].replace(first,'$'))

**OUTPUT**

Enter the string:hai hello

Old string: hai hello

New String: hai $ello

**PROGRAM NO:** 9

**DATE:24/11/2021**

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

**PROGRAM**

string1=input("Enter a string:")

print("Old string:",string1)

first=string1[0]

last=string1[-1]

print("New string:",last+string1[1:-1]+first)

**OUTPUT**

Enter a string:hello world

Old string: hello world

New string: dello worlh

**PROGRAM NO:** 10

**DATE:24/11/2021**

**AIM:**Accept the radius from user and find area of circle.

**PROGRAM**

r=float(input("Enter the radius:"))

print("radius=",r)

area=3.14\*r\*r

print("Area=",'%.2f'%area)

**OUTPUT**

Enter the radius:2

radius= 2.0

Area= 12.56

**PROGRAM NO:** 11

**DATE:29/11/2021**

**AIM:**Find biggest of 3 numbers entered

**PROGRAM**

n1=int(input("Enter the first number:"))

n2=int(input("Enter the second number:"))

n3=int(input("Enter the third number:"))

large=n1

if(n2>large):

large=n2

if(n3>large):

large=n3

print("Largest number is:",large)

**OUTPUT**

Enter the first number:3

Enter the second number:44

Enter the third number:22

Largest number is: 44

**PROGRAM NO:** 12

**DATE:29/11/2021**

**AIM:**Accept a file name from user and print extension of that

**PROGRAM**

file1=input("Enter the file name:")

ext=file1.split('.')

print("File Name=",file1)

print("File Extension=",ext[-1])

**OUTPUT**

Enter the file name:hello.java

File Name= hello.java

File Extension= java

**PROGRAM NO:** 13

**DATE:29/11/2021**

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

**PROGRAM**

list1=[]

n=int(input("Enter the number of colors:"))

for i in range(n):

x=input("Enter the color:")

list1.append(x)

print("\nList:",list1)

print("First color:",list1[0])

print("last color:",list1[n-1])

**OUTPUT**

Enter the number of colors:3

Enter the color:blue

Enter the color:black

Enter the color:yellow

List: ['blue', 'black', 'yellow']

First color: blue

last color: yellow

**PROGRAM NO:** 14

**DATE:29/11/2021**

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

**PROGRAM**

n=input("Enter the number:")

print("Number is",n)

print("Result=",int(n)+ int(n\*2) + int(n\*3))

**OUTPUT**

Enter the number:4

Number is 4

Result= 492

**PROGRAM NO:** 15

**DATE:29/11/2021**

**AIM:**Print out all colors from color-list1 not contained in color-list2

**PROGRAM**

list1=["black","green","red"]

list2=["black","blue","red"]

print("Difference:",set(list1) ^ set(list2))

**OUTPUT**

Difference: {'green', 'blue'}

**PROGRAM NO:** 16

**DATE:29/11/2021**

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

**PROGRAM**

str1=input("Enter the first string:")

str2=input("Enter the second string:")

c1=str1[0]

c2=str2[0]

print("\nString 1:",str1)

print("string 2:",str2)

print("Resultant string:",c2+str1[1:]+" "+c1+str2[1:])

**OUTPUT**

Enter the first string:hello

Enter the second string:world

String 1: hello

string 2: world

Resultant string: wello horld

**PROGRAM NO:** 17

**DATE:29/11/2021**

**AIM:**Sort dictionary in ascending and descending order.

**PROGRAM**

import operator

d={5:22,7:4,3:5}

print("Old Dictionary:\n",d)

print("\nSorted dictionary by value in ascending order:")

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

print("\nSorted dictionary by value in descending order:")

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

print(rev)

**OUTPUT**

Old Dictionary:

{5: 22, 7: 4, 3: 5}

Sorted dictionary by value in ascending order:

[(7, 4), (3, 5), (5, 22)]

Sorted dictionary by value in descending order:

{5: 22, 3: 5, 7: 4}

**PROGRAM NO:** 18

**DATE:29/11/2021**

**AIM:**Merge two dictionaries

**PROGRAM**

d1={1:4,2:5,3:8}

print("First dictionary:",d1)

d2={1:6,2:8,4:9}

print("Second dictionary:",d2)

d3=d1.copy()

d3.update(d2)

print("Merged dictionary:",d3)

**OUTPUT**

First dictionary: {1: 4, 2: 5, 3: 8}

Second dictionary: {1: 6, 2: 8, 4: 9}

Merged dictionary: {1: 6, 2: 8, 3: 8, 4: 9}

**PROGRAM NO:** 19

**DATE:29/11/2021**

**AIM:**Find gcd of 2 numbers.

**PROGRAM**

n1=int(input("Enter the first number:"))

n2=int(input("Enter the second number:"))

i=1

while i<=n1 and i<=n2:

if(n1%i==0 and n2%i==0):

gcd=i

i=i+1

print("Gcd=",gcd)

**OUTPUT**

Enter the first number:20

Enter the second number:40

Gcd= 20

**PROGRAM NO:20**

**DATE:29/11/2021**

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

**PROGRAM**

list1=[]

list2=[]

n=int(input("Enter the limit:"))

for i in range(n):

x=int(input("Enter the element:"))

list1.append(x)

if x%2!=0:

list2.append(x)

print("\nEntered List:",list1)

print("\nResultant List:",list2)

**OUTPUT**

Enter the limit:4

Enter the element:11

Enter the element:22

Enter the element:33

Enter the element:44

Entered List: [11, 22, 33, 44]

Resultant List: [11, 33]

1. **COURSE OUTCOME 2(CO2)**

**PROGRAM NO:** 1

**DATE:1/12/2021**

**AIM:**Program to find the factorial of a number

**PROGRAM**

n1=int(input("Enter the number:"))

f=1

for i in range(1,n1+1):

f=f\*i

print("Factorial of",n1,"is:",f)

**OUTPUT**

Enter the number:6

Factorial of 6 is: 720

**PROGRAM NO:** 2

**DATE:1/12/2021**

**AIM:**Generate Fibonacci series of N terms

**PROGRAM**

n=int(input("Enter the limit:"))

f=0

s=1

r=0

c=1

print("Fibonacci series:")

while c<=n:

print(r,end=" ")

c=c+1

f=s

s=r

r=f+s

**OUTPUT**

Enter the limit:4

Fibonacci series:

0 1 1 2

**PROGRAM NO:** 3

**DATE:1/12/2021**

**AIM:**Find the sum of all items in a list

**PROGRAM**

list1=[]

s=0

n=int(input("Enter the limit:"))

for i in range(n):

x=int(input("Enter the number:"))

list1.append(x)

s=s+x

print("List:",list1)

print("Sum=",s)

**OUTPUT**

Enter the limit:4

Enter the number:2

Enter the number:3

Enter the number:2

Enter the number:1

List: [2, 3, 2, 1]

Sum= 8

**PROGRAM NO:** 4

**DATE:1/12/2021**

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

**PROGRAM**

import math

n=int(input("Enter the limit:"))

print("Perfect numbers:\n")

for i in range(1000,n+1):

t=int(math.sqrt(i))

if t\*t==i and i%2==0:

print(i,end=" ")

**OUTPUT**

Enter the limit:1200

Perfect numbers:

1024 1156

**PROGRAM NO:**5

**DATE:1/12/2021**

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

**PROGRAM**

n=int(input("Enter the limit:"))

for i in range(1,n+1):

for j in range(1,i+1):

print(i\*j,end=" ")

print("\n")

**OUTPUT**

Enter the limit:4

1

2 4

3 6 9

4 8 12 16

**PROGRAM NO:**6

**DATE:1/12/2021**

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

**PROGRAM**

str1=input("Enter the string:")

f={}

for i in str1:

if i in f:

f[i]=f[i]+1

else:

f[i]=1

print(f)

**OUTPUT**

Enter the string:messi

{'m': 1, 'e': 1, 's': 2, 'i': 1}

**PROGRAM NO:**7

**DATE:8/12/2021**

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

**PROGRAM**

str=input("Enter the string:")

print("Entered string:",str)

if(str.endswith("ing")):

str=str+"ly"

else:

str=str+"ing"

print("Resultant string:",str)

**OUTPUT**

Enter the string:play

Entered string: play

Resultant string: playing

**PROGRAM NO:**8

**DATE:8/12/2021**

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

**PROGRAM**

list1=[]

n=int(input("Enter the number of strings:"))

for i in range(n):

str=input("Enter the string:")

list1.append(str);

lword=list1[0]

max=len(list1[0])

for i in list1:

if(len(i)>max):

max=len(i)

lword=i

print("Longest word:",lword)

print("Length:",max)

**OUTPUT**

Enter the number of strings:3

Enter the string:hai

Enter the string:hello

Enter the string:hi

Longest word: hello

Length: 5

**PROGRAM NO:**9

**DATE:8/12/2021**

**AIM:**Construct following pattern using nested loop

**\***

**\* \***

**\* \* \***

**\* \* \* \***

**\* \* \* \***

**\* \* \***

**\* \***

**\***

**PROGRAM**

n=int(input("Enter the limit:"))

for i in range(n+1):

for j in range(1,i+1):

print("\*",end=" ")

print("\n")

for i in range(n,0,-1):

for j in range(i):

print("\*",end=" ")

print("\n")

**OUTPUT**

**Enter the limit:4**

**\***

**\* \***

**\* \* \***

**\* \* \* \***

**\* \* \* \***

**\* \* \***

**\* \***

**\***

**PROGRAM NO:**10

**DATE:8/12/2021**

**AIM:**Generate all factors of a number. def print\_factors(x):

**PROGRAM**

def fact(n):

print("Factors of",n,":")

for i in range(1,n+1):

if n%i==0:

print(i)

n=int(input("Enter the number:"))

fact(n)

**OUTPUT**

Enter the number:16

Factors of 16 :

1

2

4

8

16

**PROGRAM NO:**11

**DATE:8/12/2021**

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

**PROGRAM**

a\_sq=lambda a:a\*a

a\_rec=lambda l,b:l\*b

a\_tri=lambda b,h:1/2\*b\*h

print("Area of square=",a\_sq(2))

print("Area of rectangle=",a\_rec(2,2))

print("Area of triangle=",a\_tri(2,5))

**OUTPUT**

Area of square= 4

Area of rectangle= 4

Area of triangle= 5.0

1. **COURSE OUTCOME 3(CO3)**

**PROGRAM NO:** 1

**DATE:15/12/2021**

**AIM:**Work with built-in packages

**Time Module**

**PROGRAM**

import time

print("Current time in sec:",time.time())

print("Current time:",time.ctime())

print("Time After 30 sec:",time.ctime(time.time()+30))

t=time.localtime()

print("Time:",t)

print("Time-current year:",t.tm\_year)

print("Time:-current month",t.tm\_mon)

print("Time:-current day",t.tm\_mday)

print("Time:-current hour",t.tm\_hour)

print("Time:-current minute",t.tm\_min)

print("Time:-current sec",t.tm\_sec)

print("Time:-current week day",t.tm\_wday)

print("Time:-current year day",t.tm\_yday)

**OUTPUT**

Current time in sec: 1639915265.630671

Current time: Sun Dec 19 17:31:05 2021

Time After 30 sec: Sun Dec 19 17:31:35 2021

Time: time.struct\_time(tm\_year=2021, tm\_mon=12, tm\_mday=19, tm\_hour=17, tm\_min=31, tm\_sec=5, tm\_wday=6, tm\_yday=353, tm\_isdst=0)

Time-current year: 2021

Time:-current month 12

Time:-current day 19

Time:-current hour 17

Time:-current minute 31

Time:-current sec 5

Time:-current week day 6

Time:-current year day 353

**Math module**

**PROGRAM**

import math

print(math.factorial(4))

print (math.gcd(3, 6))

print (math.sqrt(9))

**OUTPUT**

24

3

3.0

**Calendar module**

**PROGRAM**

import calendar

mm=int(input("Enter month:"))

yy=int(input("Enter year:"))

print("\n")

print(calendar.month(yy,mm))

**OUTPUT**

Enter month:1

Enter year:2022

January 2022

Mo Tu We Th Fr Sa Su

1 2

3 4 5 6 7 8 9

10 11 12 13 14 15 16

17 18 19 20 21 22 23

24 25 26 27 28 29 30

31

**DateTime module**

**PROGRAM**

import datetime

t=datetime.time(22,56,20,67)

print(t)

print("Hour",t.hour)

print("Minutes",t.minute)

print("Seconds",t.second)

print("Microsecond:",t.microsecond)

print("\n")

d=datetime.date.today()

print(d)

print("Year:",d.year)

print("Month:",d.month)

print("Day:",d.day)

d1=datetime.date.today()

print(d1)

td=datetime.timedelta(days=2)

print(td)

d2=d1+td

print(d2)

dt=datetime.datetime.combine(d1,t)

print(dt)

**OUTPUT**

22:56:20.000067

Hour 22

Minutes 56

Seconds 20

Microsecond: 67

2021-12-19

Year: 2021

Month: 12

Day: 19

2021-12-19

2 days, 0:00:00

2021-12-21

2021-12-19 22:56:20.000067

**Statistics module**

**PROGRAM**

import statistics

print(statistics.mean([3,4,3]))

print(statistics.median([1, 3, 5, 7, 9, 11, 13]))

print(statistics.mode([1, 1, -3, 3, 7, -9]))

print(statistics.variance([1, 3, 5, 7, 9, 11]))

print(statistics.stdev([1, 3, 5, 7, 9, 11]))

**OUTPUT**

3.3333333333333335

7

1

14

3.7416573867739413

**Random module**

**PROGRAM**

import random

random.seed(10)

print(random.random())

mylist = ["apple", "banana", "cherry"]

print(random.sample(mylist, k=2))

print(random.random())

mylist2 = ["apple", "banana", "cherry"]

random.shuffle(mylist2)

print(mylist2)

mylist3 = ["apple", "banana", "cherry"]

print(random.choice(mylist3))

**OUTPUT**

0.5714025946899135

['banana', 'cherry']

0.5780913011344704

['cherry', 'banana', 'apple']

banana

**PROGRAM NO: 2**

**DATE:15**/12/2021

**AIM:**

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)

**PROGRAM**

* **graphics package**

**circle module**

def area(r):

return(3.14\*r\*r)

def perimeter(r):

return(2\*3.14\*r)

**rectangle module**

def area(l,b):

return(l\*b)

def perimeter(l,b):

return(2\*(l+b))

* **3dgraphics package**

**sphere module**

def area(r):

return(4\*3.14\*r\*r)

def perimeter(r):

return(2\*3.14\*r)

**cuboid module**

def area(l,w,h):

return(2\*l\*w+2\*l\*h+2\*h\*w)

def perimeter(l,b,h):

return(4\*(l+b+h))

from graphics import rectangle

from graphics import circle

from dgraphics import cuboid

from dgraphics import sphere

print("Rectangle:")

l=int(input("Enter the length:"))

b=int(input("Enter the breadth:"))

print("Area=",rectangle.area(l,b))

print("Perimeter=",rectangle.perimeter(l,b))

print("\nCircle:")

r=int(input("Enter the radius:"))

print("Area=",circle.area(r))

print("Perimeter=",circle.perimeter(r))

print("\nCuboid:")

l=int(input("Enter the length:"))

w=int(input("Enter the width:"))

h=int(input("Enter the height:"))

b=int(input("Enter the breadth:"))

print("Area=",cuboid.area(l,w,h))

print("perimeter=",cuboid.perimeter(l,b,h))

print("\nSphere:")

r=int(input("Enter the radius:"))

print("Area=",sphere.area(r))

print("perimeter=",sphere.perimeter(r))

**OUTPUT**

Rectangle:

Enter the length:2

Enter the breadth:2

Area= 4

Perimeter= 8

Circle:

Enter the radius:2

Area= 12.56

Perimeter= 12.56

Cuboid:

Enter the length:2

Enter the width:2

Enter the height:1

Enter the breadth:2

Area= 16

perimeter= 20

Sphere:

Enter the radius:2

Area= 50.24

perimeter= 12.56

1. **COURSE OUTCOME 4(CO4)**

**PROGRAM NO:** 1

**DATE:9**/1/2022

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

**PROGRAM**

class rectangle:

def \_\_init\_\_(self,length,breadth):

self.length=length

self.breadth=breadth

def area(self):

area=self.length\*self.breadth

print("Area=",area)

return(area)

def perimeter(self):

per=2\*(self.length+self.breadth)

print("Perimeter=",per)

print("First Rectangle:")

b1=rectangle(2,2)

a1=b1.area()

b1.perimeter()

print("\nSecond Rectangle:")

b2=rectangle(3,3)

a2=b2.area()

b2.perimeter()

if a1 > a2:

print("\nArea of first rectangle is larger")

else:

print("\nArea of second rectangle is larger")

**OUTPUT**

First Rectangle:

Area= 4

Perimeter= 8

Second Rectangle:

Area= 9

Perimeter= 12

Area of second rectangle is larger

**PROGRAM NO:** 2

**DATE:9/1/2022**

**AIM:** 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.

**PROGRAM**

class bank:

bal=0

def \_\_init\_\_(self,accno,name,ac\_type,bal):

self.accno=accno

self.name=name

self.ac\_type=ac\_type

self.bal=bal

def display(self):

print("\nAccount Info:")

print("Account Number:",self.accno)

print("Account Name:",self.name)

print("Account Type:",self.ac\_type)

print("Account Balance:",self.bal)

def deposit(self):

dep=int(input("Enter the amount to deposit:"))

self.bal=self.bal+dep

def withdraw(self):

w=int(input("Enter the amount to withdraw:"))

if w > self.bal:

print("Insufficient Balance")

else:

self.bal=self.bal-w

print("RS-",w,"Withdrawn successfully")

acc\_no=int(input("Enter the Account Number:"))

acc\_name=input("Enter the name:")

acc\_type=input("Enter the account type-(savings/current):")

balance=int(input("Enter the initial balance:"))

b1=bank(acc\_no,acc\_name,acc\_type,balance)

while(1):

print("\n1.Account Info\n2.Deposit\n3.Withdraw\n4.Exit")

opt=int(input("Select your option:"))

if opt == 1:

b1.display()

elif opt == 2:

b1.deposit()

elif opt == 3:

b1.withdraw()

elif opt == 4:

print("Exited")

break

else:

print("Invalid Option")

**OUTPUT**

Enter the Account Number:1233

Enter the name:Alan

Enter the account type-(savings/current):savings

Enter the initial balance:2000

1.Account Info

2.Deposit

3.Withdraw

4.Exit

Select your option:2

Enter the amount to deposit:200

1.Account Info

2.Deposit

3.Withdraw

4.Exit

Select your option:1

Account Info:

Account Number: 1233

Account Name: Alan

Account Type: savings

Account Balance: 2200

1.Account Info

2.Deposit

3.Withdraw

4.Exit

Select your option:3

Enter the amount to withdraw:100

RS- 100 Withdrawn successfully

1.Account Info

2.Deposit

3.Withdraw

4.Exit

Select your option:1

Account Info:

Account Number: 1233

Account Name: Alan

Account Type: savings

Account Balance: 2100

1.Account Info

2.Deposit

3.Withdraw

4.Exit

Select your option:4

Exited

**PROGRAM NO:** 3

**DATE:9/1/2022**

**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,b):

self.\_\_length=l

self.\_\_breadth=b

def area(self):

self.area=self.\_\_length\*self.\_\_breadth

print("Area=",self.area)

def \_\_lt\_\_(self,second):

if self.area < second.area:

return True

else:

return False

print("first Rectangle:")

len1=int(input("Enter the length:"))

bread1=int(input("Enter the breadth:"))

obj1=rectangle(len1,bread1)

obj1.area()

print("\nSecond Rectangle:")

len2=int(input("Enter the length:"))

bread2=int(input("Enter the breadth:"))

obj2=rectangle(len2,bread2)

obj2.area()

if obj1 < obj2 :

print("\nArea of second rectangle is larger:")

else:

print("\nArea of first rectangle is larger:")

**OUTPUT**

first Rectangle:

Enter the length:2

Enter the breadth:3

Area= 6

Second Rectangle:

Enter the length:2

Enter the breadth:2

Area= 4

Area of first rectangle is larger:

**PROGRAM NO:** 4

**DATE:9/1/2022**

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

**PROGRAM**

class time:

def \_\_init\_\_(self,hour,minute,second):

self.\_\_hour=hour

self.\_\_minute=minute

self.\_\_second=second

def \_\_add\_\_(self,second):

print("\nHour:",self.\_\_hour + second.\_\_hour)

if self.\_\_minute + second.\_\_minute > 60:

h1=(self.\_\_minute + second.\_\_minute)//60

m1=(self.\_\_minute + second.\_\_minute)%60

print("Minutes:",h1," hour ",m1," minutes")

else:

print("Minutes:",self.\_\_minute + second.\_\_minute)

if self.\_\_second+second.\_\_second > 60:

m1=(self.\_\_second+second.\_\_second)//60

s1=(self.\_\_second+second.\_\_second)%60

print("seconds:",m1," minutes ",s1," seconds")

else:

print("Seconds:",self.\_\_second + second.\_\_second)

hour1=int(input("Enter the hour:"))

minute1=int(input("Enter the minutes:"))

sec1=int(input("Enter the second:"))

obj1=time(hour1,minute1,sec1)

hour2=int(input("\nEnter the hour:"))

minute2=int(input("Enter the minutes:"))

sec2=int(input("Enter the second:"))

obj2=time(hour2,minute2,sec2)

obj1 + obj2

**OUTPUT**

Enter the hour:3

Enter the minutes:44

Enter the second:35

Enter the hour:4

Enter the minutes:55

Enter the second:45

Hour: 7

Minutes: 1 hour 39 minutes

seconds: 1 minutes 20 seconds

**PROGRAM NO:**5

**DATE:9/1/2022**

**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,pname):

self.pname=pname

def display(self):

print("Publisher Name:",self.pname)

class book(publisher):

def get(self,title,author):

self.title=title

self.author=author

def display(self):

print("Title Name:",self.title)

print("Author Name:",self.author)

class python(book):

def \_\_init\_\_(self,price,nop,pname):

super().\_\_init\_\_(pname)

self.price=price

self.nop=nop

def details(self):

print("Price:",self.price)

print("No of pages:",self.nop)

s1=python(450,72,"K D")

s1.get("Flames","K D")

s1.display()

s1.details()

**OUTPUT**

Title Name: Flames

Author Name: K D

Price: 450

No of pages: 72

1. **COURSE OUTCOME 5(CO5)**

**PROGRAM NO:** 1

**DATE:30/1/2022**

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

**PROGRAM**

f1=open("sample.txt","w")

f1.write("This is my first line.\n This is my second line \n This is my third line")

f1=open("sample.txt","r")

ff=f1.readlines()

print(ff)

**OUTPUT**

['This is my first line.\n', ' This is my second line \n', ' This is my third line']

This is my first line.

This is my second line

This is my third line

**PROGRAM NO:** 2

**DATE:30/1/2022**

**AIM:**Python program to copy odd lines of one file to other

**PROGRAM**

f1=open("sample.txt","r")

ff=f1.readlines()

with open("odd.txt","w") as f2:

for x in range(0,len(ff)):

if(x%2!=0):

f2.write(ff[x])

**OUTPUT**

This is my second line

**PROGRAM NO:** 3

**DATE:30/1/2022**

**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 r in data:

print(', '.join(r))

**department.csv**

department\_id,department\_name,manager\_id,location\_id

10,Administration,200,1700

20,Marketing,201,1800

30,Purchasing,114,1700

**OUTPUT**

department\_id,department\_name,manager\_id,location\_id

10,Administration,200,1700

20,Marketing,201,1800

30,Purchasing,114,1700

**PROGRAM NO:** 4

**DATE:30/1/2022**

**AIM:** Write a Python program to read specific columns 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")

print("---------------------------------")

for r in data:

print(r['department\_id']," ",r['department\_name']

**department.csv**

department\_id,department\_name,manager\_id,location\_id

10,Administration,200,1700

20,Marketing,201,1800

30,Purchasing,114,1700

**OUTPUT**

id Department

---------------------------------

10 Administration

20 Marketing

30 Purchasing

**PROGRAM NO:** 5

**DATE:30/1/2022**

**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

field\_names = ['No', 'Company', 'Model']

cars = [

{'No': 1, 'Company': 'Ferrari', 'Model': '488 GTB'},

{'No': 2, 'Company': 'Porsche', 'Model': '918 Spyder'},

{'No': 3, 'Company': 'Bugatti', 'Model': 'La Voiture Noire'},

]

with open('cars.csv', 'w') as csvfile:

writer = csv.DictWriter(csvfile, fieldnames=field\_names)

writer.writeheader()

writer.writerows(cars)

with open('cars.csv', newline='') as csvfile:

data = csv.reader(csvfile, delimiter=' ', quotechar='|')

for r in data:

print(', '.join(r))

**cars.csv**

No,Company,Model

1,Ferrari,488 GTB

2,Porsche,918 Spyder

3,Bugatti,La Voiture Noire

**OUTPUT**

No,Company,Model

1,Ferrari,488, GTB

2,Porsche,918, Spyder

3,Bugatti,La, Voiture, Noire