DE PAUL

INSTITUTE OF SCIENCE & TECHNOLOGY

DiST, Angamaly

(*Affiliated to Mahatma Gandhi University, Kottayam & Approved by AICTE,New Delhi*
An ISO 9001:2008 Certified Institution)

SCHOOL OF COMPUTER SCIENCE

Master of Computer Applications

PROGRAMMING LAB RECORD

Name:…….

Reg. No: …….

Course: ……Semester …….

Certified that this is a bonafide record of work done by the above student in the programming lab.....

during the year…….

Lecturer-in-charge Head of the Department

Submitted for the Practical Examination held on…….

Internal Examiner (seal) External Examiner

SI NO	DATE	TOPIC	PAGENO
1		Even or odd number	
2		Increasing or decreasing	
3		Temperature	
4		Number Palindrome	
5		Perfect Number	
6		Armstrong	
7		Prime Numbers	
8		Arithmetic operations	
9		Student details	
10		List operations	
11		User-Password	
12		Employee - List operations	
13		User defined functions	
14		Fibonacci	
15		String operations	
16		Sum of strings	
17		Display sum of odd numbers	
18		String Palindrome	
19		2-D List	
20		Functions	
21		String Slicing	
22		List - Sum,Minimum,Maximum	
23		Substring	
24		Multidimensional list	
25		Anonymous function	
26		Positive and Negative integer list	
27		Most occuring integer	
28		Math function	
29		Relational and Logical operators	
30		MakeDictionary function	
31		Average mark of student using constructor	
32		Multilevel Inheritence	
33		Multiple Inheritence	
34		Operator Overloading	
35		Multiple Exception	
36		User defined Exception	
37		File Operations	
38		Database Operations	
39		Static Page Creation (Django)	
40		Dynamic Page Creation (Django)	
41		Pandas Library	
42		Numpy Library	
43		Matplotlib Library	

PROGRAM NO: 1

AIM: Write a python program to check whether a number is even or odd

PROGRAM

```
print("even or odd")
a=int(input("Enter a number : "))
if(a%2==0):
        print(a," is even number")
else:
        print(a," is odd number")
```

OUTPUT

```
even or odd
Enter a number : 5
5  is odd number
even or odd
Enter a number : 8
8  is even number
```

PROGRAM NO: 2

AIM: Write a program to read three numbers and print “increasing” if they are in increasing order, “decreasing” if they are in decreasing order and “neither” otherwise. Here increasing means strictly increasing with each value larger than its predecessor. The sequence 3 4 4 will be considered as increasing.

PROGRAM

```
n1=int(input("Enter 1st number : "))
n2=int(input("Enter 2nd number : "))
n3=int(input("Enter 3rd number : "))
if(n1>n2 and n2>n3):
        print("decreasing")
elif(n1<=n2 and n2<=n3):
        print("increasing")
else:
        print(“neither”)
```

OUTPUT

```
Enter 1st number : 2
Enter 2nd number : 3
Enter 3rd number : 4
Increasing
Enter 1st number : 5
Enter 2nd number : 4
Enter 3rd number : 3
Decreasing
Enter 1st number : 1
```

```
Enter 2nd number : 5
Enter 3rd number : 2
Neither
```

PROGRAM NO: 3

AIM: Write a Python program called Temperatures that has two tasks: Allows the user to convert a temperature given in degrees from either Celsius to Fahrenheit or Fahrenheit to Celsius. Use the following formulas: • Celsius = 5 * (Fahrenheit − 32) / 9 • Fahrenheit = (9 * (Celsius) / 5) + 32 Prompt the user to enter a temperature and either a C or c for Celsius or an F or f for Fahrenheit. Convert the temperature to Fahrenheit if Celsius is entered or Celsius if Fahrenheit is entered. Display the result in a readable format. If anything, other than C, c, F, or f is entered, print an error message and ask for the correct entry again.

PROGRAM

```
flag=1
while(flag==1):
        print("press c for celsius to fahrenheit")
        print("press f for fahrenheit to celsius")
        op=input("Enter your option : ")
        if(op=='c'):
            temp=float(input("Enter temperature : "))
            temp=5*(temp-32)/9
            print("celsius : ",temp)
            break
        elif(op=='f'):
            temp=float(input("Enter temperature : "))
            temp=9*(temp)/5
            print("fahrenheit : ",temp)
            break
        else:
            print("program exited")
            break
```

OUTPUT

```
Press c for celsius to fahrenheit

Press f for fahrenheit to celsius

Enter your option : c

Enter temperature : 100

Celsius :  37.77777777777778
```

PROGRAM NO: 4

AIM: Write a python program to check whether a number is palindrome or not

PROGRAM

```
n=int(input("Enter a number : "))
temp=n
sum=0
while n>0:
        d=n%10
        sum=(sum*10)+d
        n=n//10
if(temp==sum):
        print(temp," is palindrome")
else:
        print(temp," is not palindrome")
```

OUTPUT

```
Enter a number : 121
121  is palindrome
Enter a number : 123
123  is not palindrome
```

PROGRAM NO: 5

AIM: Program to find whether a number is perfect number or not. A perfect number is one whose sum of divisors is equal to that number. E.g. 6 = (1+2+3)

PROGRAM

```
n=int(input("Enter a number : "))
sum=0
temp=n
for i in range(1,n):
        if(n%i==0):
                sum=sum+i
if sum==temp:
        print(n," is a perfect number")
else:
        print(n," is not a perfect number")
```

OUTPUT

```
Enter a number : 16
16  is not a perfect number
Enter a number : 6
6  is a perfect number
```

PROGRAM NO: 6

AIM: Write a python program to check 3 digit Armstrong or not

PROGRAM

```
n=int(input("Enter a limit : "))
temp=n
sum=0
while n>0:
        d=n%10
        sum=sum+(d*d*d)
        n=n//10
if temp==sum:
        print(temp," is armstrong")
else:
        print(temp," is not armstrong")
```

OUTPUT

```
Enter a limit : 153
153  is Armstrong
Enter a limit : 12
12  is not armstrong
```

PROGRAM NO: 7

AIM: Write a program to display prime numbers for a given limit

PROGRAM

```
limit = int(input("Enter the limit: "))
if limit < 2:
    print("")
else:
    print("Prime numbers up to", limit, "are:")
    print(2, end=" ")
    for num in range(3, limit + 1, 2):
        is_prime = True
        for i in range(2, int(num/2) + 1):
            if num % i == 0:
                is_prime = False
                break
        if is_prime:
            print(num, end=" ")
```

OUTPUT

```
Enter the limit: 10
Prime numbers up to 10 are:
2 3 5 7
```

PROGRAM NO: 8

AIM: Write a program to perform arithmetic operations on two integers(Read two input integers using input(),Addition,Subtraction,Multiplication,Division,Integer Division,Modulus,exponential operation)

PROGRAM

```
def add(a,b):
  c=a+b
  return c
def sub(a,b):
  c=a-b
  return c
  def mul(a,b):
  c=a*b
  return c
  def div(a,b):
  c=a/b
  return c
  def idiv(a,b):
  c=a//b
  return c
  def mod(a,b):
  c=a%b
  return c
  def exp(a,b):
  p=1
  for i in range(1,b+1):
```

```
    p=p*a
  return(p)
  a=int(input("Enter the 1st number : "))
b=int(input("Enter the 2nd number : "))
print("opertions on ",a,"and ",b)
s=add(a,b)
print ("Addition : ",s)
sub=sub(a,b)
print ("Subtraction : ",sub)
m=mul(a,b)
print ("Multiplication : ",m)
d=div(a,b)
print ("Division : ",d)
i=idiv(a,b)
print ("Integer Division : ",i)
mod=mod(a,b)
print ("Modulus : ",mod)
e=exp(a,b)
print("Exponential operation : ",e)
```

OUTPUT

```
Enter the 1st number : 25
Enter the 2nd number : 5
opertions on  25 and  5
Addition :  30
Subtraction :  20
```

```
Multiplication :  125
Division :  5.0
Integer Division :  5
Modulus :  0
Exponential operation :  9765625
```

PROGRAM NO: 9

AIM: Write a program for reading and displaying the student details-name,rollno,internal mark and external mark.Show the details along with grace mark (5%) and final marks(after adding grace mark)

PROGRAM

```
n=input("Enter the name : ")
r=int(input("Enter the Roll no : "))
e=int(input("Enter the External mark out of 75 : "))
i=int(input("Enter the Internal mark out of 20 : "))
print("\n........STUDENT DETAILS.............")
print("NAME : ",n)
print("ROLL NO : ",r)
print("EXTERNAL MARK : ",e)
print("INTERNAL MARK ",i)
f=e+i+5
print("FINAL MARK ON 100 : ",f)
```

OUTPUT

```
Enter the name : alen
Enter the Roll no : 05
Enter the External mark out of 75 : 67
Enter the Internal mark out of 20 : 18
........STUDENT DETAILS.............
NAME :  alen
ROLL NO :  5
EXTERNAL MARK :  67
INTERNAL MARK  18
FINAL MARK ON 100 :  90
```

PROGRAM NO: 10

AIM: Create a list pets with values dog, cat, goldfish, goat, cow and perform the following operations.

i. Find length of list
ii. Concatenate list to itself
iii. Find 3rd element of list
iv. Append a new item ‘bird’ to the list
v. Find the number of times dog is repeated
vi. Find the reverse of the list
vii. Pop out an element from the list

PROGRAM

```
pet = ["dog","cat","goldfish","goat","cow"]

l=len(pet)

print("length : ",l)

con=pet+pet

print("concatenation : ",con)

print("3rd element : ",pet[2])

pet.append("bird")

print("appended item : ",pet[5])

print("count of item repeated : ",pet.count("dog"))

print("reverse : ",pet[5:0: -1])

item=pet.pop()

print("poped item : ",item)
```

OUTPUT

```
length :  5
concatenation :  ['dog', 'cat', 'goldfish', 'goat', 'cow', 'dog', 'cat', 'goldfish', 'goat', 'cow']
3rd element :  goldfish
appended item :  bird
count of item repeated :  1
reverse :  ['bird', 'cow', 'goat', 'goldfish', 'cat']
 poped item :  bird
```

PROGRAM NO: 11

AIM: Write a program that asks the user for a password, with error checking to repeat if the password doesn't meet a minimum length set by a variable. The program should then print asterisks as long as the word. Example: if the user enters Pythonista (10 characters), the program should print **********.

PROGRAM

```
uname=input("Enter username : ")
password=input("Enter password : ")
length=len(password)
if length <11:
        for i in range(length):
                print('*',end=' ')
else:
        print("good password")
```

OUTPUT

```
Enter username : mca22
Enter password : mca22admin12345
good password

Enter username : mca22
Enter password : mca22admin
* * * * * * * * * *
```

PROGRAM NO: 12

AIM: Create a list containing employee names ['Kiran','Rahul','Lakshmi','Priya','Rahul'] and perform the following operations

i. Check the number of times the employee name 'Rahul' present in the list
ii.Remove the employee' Priya' from the list
iii.Add a new employee name 'Raghav' to the beginning of the list
iv.Print the employee name in sorted order
v.Remove the last employee from the list
vi.Print the employee names from the list as last to first order

PROGRAM

```
l=["kiran","rahul","lakshmi","priya","rahul"]

print("list: ",l)


print("\n1. Number of time employee rahul repeat is:",l.count("rahul"))


l.remove("priya")

print("\n2. List after removing the employee priya: ",l)


l.insert(0,"ragav")

print("\n3. List after appending ragav at beginning of list: ",l)


l.sort()

print("\n4. Sorted the list: ",l)


l.pop()

print("\n5. List after removing last employee: ",l)
```

```
l.reverse()
print("\n6. reversed list: ",l)
#print(emp[::-1])
```

OUTPUT

```
list:  ['kiran', 'rahul', 'lakshmi', 'priya', 'rahul']

1. Number of time employee rahul repeat is: 2

2. List after removing the employee priya:  ['kiran', 'rahul', 'lakshmi',
'rahul']

3. List after appending ragav at beginning of list:  ['ragav', 'kiran', 'rahul',
'lakshmi', 'rahul']

4. Sorted the list:  ['kiran', 'lakshmi', 'ragav', 'rahul', 'rahul']

5. List after removing last employee:  ['kiran', 'lakshmi', 'ragav', 'rahul']

6. reversed list:  ['rahul', 'ragav', 'lakshmi', 'kiran']
```

PROGRAM NO: 13

AIM: Create user defined functions for finding the area of a square & cube based on user choice

PROGRAM

```
def square(x):
    s=x*x
    return s
def cube(x):
    c=x*x*x
    return c

print("1. square\n2. cube\n")
op=int(input("Enter the option"))
if(op==1):
    a=int(input("Enter the length of square: "))
    area=square(a)
    print("Area of the square is: ",area)
elif(op==2):
    a=int(input("Enter the length of cube"))
    area=cube(a)
    print("Area of the cube is: ",area)
else:
    print("wrong option..")
```

OUTPUT

```
1. square
2. cube

Enter the option1
Enter the length of square: 4
Area of the square is:  16
1. square
2. cube

Enter the option2
Enter the length of cube3
Area of the cube is:  27
```

PROGRAM NO: 14

AIM: Generate Fibonacci series using recursion.

PROGRAM

```
def recur_fibo(n):
  if n <= 1:
      return n
  else:
      return(recur_fibo(n-1) + recur_fibo(n-2))
num=int(input("Enter the limit : "))
if num <= 0:
  print("Plese enter a positive integer")
else:
  print("Fibonacci sequence:")
  for i in range(num):
      print(recur_fibo(i))
```

OUTPUT

```
Enter the limit : 8
Fibonacci sequence:
0
1
1
2
3
5
8
```

PROGRAM NO: 15

AIM: Read a Name from the user and perform the following operations
a. Find the length of name
b. Check whether name contains a substring entered by the user
c. Capitalize the name
d. Find the count of a substring from the name
e. Display the name concatenated with 'IMCA' to the starting of the name f. Exit
Do it as a menu based options where user exit with option f. [Switch Cases are not available in python]

PROGRAM

```
name=input("Enter your name :")


length=len(name)
print("1. Lenght of the string : ",length)


sub=input("\n2. Enter the string to check : ")
if sub in name:
        print("substring found")
else:
        print("substring not found")


nameUp=name.upper()
print("\n3. Name after captitalizing : ",nameUp)


c=name.count(sub)
print("\n4. Number of times substring repeated : ",c)
```

```
test="imca"
print("\n5. Concatenation of strings - ",test," with ",name,"
:\n",test+name)
```

OUTPUT

```
Enter your name :merzilin
1. Lenght of the string :  8

2. Enter the string to check : z
substring found

3. Name after captitalizing :  MERZILIN

4. Number of times substring repeated :  1

5. Concatenation of strings -  imca  with  merzilin  :
 imcamerzilin
```

PROGRAM NO: 16

AIM: Read a string containing some number characters separated by + and return the sum of these numbers. E.g. For a string “21+32+12”,it should return 65 as the result.

PROGRAM

```
numbers=input("Enter number separated by + : ")
number=numbers.split('+')
s=0
flag=0
for x in number:
        if ( x.isdigit()):
                s = s + int(x)
        else:
                flag=1
if(flag == 0):
        print("sum = ",s)
else:
        print("non-digit item found")
```

OUTPUT

```
Enter number separated by + : 21+32+12
sum =  65

Enter number separated by + : 1+a
non-digit item found
```

PROGRAM NO: 17

AIM: Program to display randomly generated 50 numbers from 1 to 100 and find the sum of odd numbers from it?[import module random & use function randrange( ) to generate 50 random numbers]

PROGRAM

```
from random import randrange
oddsum=evensum=0
for i in range(50):
    num=randrange(1,100)
    print(num)
    if(num%2!=0):
        oddsum=oddsum+num
print("Sum of odd numbers is: ",oddsum)
```

OUTPUT

```
75
28
32
42
92
58
67
62
31
27
34
```

```
49
23
4
61
66
53
99
67
4
20
50
84
65
90
22
95
93
24
18
82
63
60
93
90
58
6
```

```
48
48
90
27
54
58
77
26
53
13
97
81
72
Sum of odd numbers is:  1309
```

PROGRAM NO: 18

AIM: Program to check for string palindrome

PROGRAM

```
s=input("Enter a string: ")
temp=s[::-1]
if(s==temp):
    print("String is palindrome.")
else:
    print("String is not palindrome.")
```

OUTPUT

```
Enter a string: malayalam
String is palindrome.

Enter a string: hindi
String is not palindrome.
```

PROGRAM NO: 19

AIM: Read a 2 dimensional list from user and display the row wise sum for the list

PROGRAM

```
row=int(input("Enter row size : "))
col=int(input("Enter col size : "))
list_2d=[]
for i in range(0,row):
    lst = []
    for j in range(0,col):
        num = int(input("Enter item : "))
        lst.append(num)
    list_2d.append(lst)
for i in range(0,row):
    sum=0
    for j in range(0,col):
        sum=sum+list_2d[i][j]
    print("sum of row ",(i+1)," = ",sum)
```

OUTPUT

```
Enter row size : 2
Enter col size : 2
Enter element : 1
Enter element : 2
Enter element : 3
Enter element : 4
```

```
sum of row  1  =  3
sum of row  2  =  7
```

PROGRAM NO: 20

AIM: Read a list of 10 integers and create a function odd to generate a sublist containing odd numbers from the list and another function even to generate a sublist containing even numbers from the list. Return these sublist to the main program and display the sum of sublist1 and maximum of sublist2

PROGRAM

```
lo=[]
le=[]
def odd(o):
   lo.append(o)
def even(e):
   le.append(e)
l=[]
for x in range(0,10):
 num=int(input("Enter the integer:"))
 l.append(num)
print("List is:",l)
for y in l:
   if(y%2==0):
      even(y)
   else:
      odd(y)
so=0
se=max(le)
for x in lo:
```

```
    so=so+x
print("List of Odd numbers:",lo)
print("List of Even numbers:",le)
print("Sum of item in odd sub-list is :",so)
print("max value of even sub-list is :",se)
```

OUTPUT

```
Enter the integer:1
Enter the integer:2
Enter the integer:3
Enter the integer:4
Enter the integer:5
Enter the integer:6
Enter the integer:7
Enter the integer:8
Enter the integer:9
Enter the integer:10
List is: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
List of Odd numbers: [1, 3, 5, 7, 9]
List of Even numbers: [2, 4, 6, 8, 10]
Sum of item in odd sub-list is : 25
max value of even sub-list is : 10
```

PROGRAM NO: 21

AIM: Create a string s with value "Hello World" and find the following values:

i)s[5] ii)s[0:2] iii)s[1:] iv)s[:2 v)s[0:-2] vi)s[-3:0]

PROGRAM

```
st="Hello World"
print("String : ",st)
print("s[5]: ",st[5])
print("s[0:2] : ",st[0:2])
print("s[1:] : ",st[1:])
print("s[:2] : ",st[:2])
print("s[0:-2] : ",st[0:-2])
print("s[-3:0] : ",st[-3:0])
```

OUTPUT

```
String :  Hello World
s[5]:
s[0:2] :  He
s[1:] :  ello World
s[:2] :  He
s[0:-2] :  Hello Wor
s[-3:0] :
```

PROGRAM NO: 22

AIM: create an empty list and read a range of elements from the user with the help of append function and find the sum of elements from the list, minimum element and maximum element.

PROGRAM

```
str=[]
len=int(input("Enter size : "))
for i in range(0,len):
    temp=int(input("Enter item : "))
    str.append(temp)
print(str)
sum=0
for x in str:
    sum=sum+x
print("sum = ",sum)

min=min(str)
max=max(str)
print("minimum : ",min)
print("maximum : ",max)
```

OUTPUT

```
Enter size : 5
Enter item : 7
Enter item : 3
Enter item : 2
```

```
Enter item : 1
Enter item : 6
[7, 3, 2, 1, 6]
sum =  19
minimum :  1
maximum :  7
```

PROGRAM NO: 23

AIM: Input a string name and a substring from the user and check whether the particular element is present in the list or not using in operator

PROGRAM

```
string_name = input("Enter the string: ")
substring = input("Enter the substring: ")



if substring in string_name:
    print("The substring '{substring}' is present in  '{string_name}'.")
else:
    print("The substring '{substring}' is not present in  '{string_name}'.")
```

OUTPUT

```
Enter the string: hallo world
Enter the substring: o
The substring 'o' is present in  'hallo world'.

Enter the string: hallo world
Enter the substring: v
The substring 'v' is not present in  'hallo world'.
```

PROGRAM NO: 24

AIM: Create a multidimensional list with elements [[3,4,6],[1,5,7],[2,8,0]] and find the row wise total and average of list elements.

PROGRAM

```
lst=[[3,4,6],[1,5,7],[3,8,0]]
l=len(lst)
print("List = ",lst)
for i in range(3):
    rc=0
    for x in lst[i]:
        rc=rc+x
    print("total of ",(i+1)," row : ",rc)
    print("average : ",(rc/l))
```

OUTPUT

```
List =  [[3, 4, 6], [1, 5, 7], [3, 8, 0]]
total of  1  row :  13
average :  4.333333333333333
total of  2  row :  13
average :  4.333333333333333
total of  3  row :  11
average :  3.6666666666666665
```

PROGRAM NO: 25

AIM: Add 10 to argument and display the result using anonymous function.

PROGRAM

```
x=lambda a:a+10
print(x(30))
```

OUTPUT

40

PROGRAM NO: 26

AIM: Write a program that has a list of numbers (both positive and negative). Make a new tuple that has only positive values from the list.

PROGRAM

```
lst=[2,3,5,4,-1,-3,-5]
pos=[]
neg=[]
for x in lst:
    if x>=0:
        pos.append(x)
    else:
        neg.append(x)
pos=tuple(pos)
neg=tuple(neg)
print("positive integers : ")
print(pos)
print("negative integers : ")
print(neg)
```

OUTPUT

```
positive integers :
(2, 3, 5, 4)
negative integers :
(-1, -3, -5)
```

PROGRAM NO: 27

AIM: Create a tuple with some integer value and find the element with the highest number of counts in it.

PROGRAM

```
tup=(1,2,4,4,6,8,8,4)
a=0
b=0
for x in tup:
    a=tup.count(x)
    if a>b:
        b=a
        ele=x
print("Tuple - ",tup)
print("Item with highest occurence : ",x," with count : ",b)
```

OUTPUT

```
Tuple -  (1, 2, 4, 4, 6, 8, 8, 4)
Item with highest occurence :  4  with count :  3
```

PROGRAM NO: 28

AIM: Write a program that prints absolute value, square root of a number (using math module).

PROGRAM

```
import math
n=int(input("Enter a number : "))
n=abs(n)
print("Absolute Value : ",n)
sq=math.sqrt(n)
print("Square route of ",n," : ",sq)
```

OUTPUT

```
Enter a number : -4
Absolute Value :  4
Square route of  4  :  2.0
```

PROGRAM NO: 29

AIM: Create a set phonebook1 and phonebook2 and perform set operations

i. phonebook1 == phonebook2
ii. phonebook1! = phonebook2
iii. phonebook1 <= phonebook2
iv. phonebook1 | phonebook2
v. phonebook1 & phonebook2
vi. phonebook1 - phonebook2
vii. phonebook1 ^ phonebook2

PROGRAM

```
n1=input("Enter set 1 : ")

n2=input("Enter set 2 : ")

n1=n1.split(',')

n2=n2.split(',')

ph1=set(n1)

ph2=set(n2)

print(ph1)

print(ph2)

if ph1==ph2:

        print("phonebook 1 & 2 are equal")


if ph1!=ph2:

        print("phonebook 1 & 2 are not equal")


if ph1<ph2:
```

```
        print("phonebook 1 is less than 2 ")
else:
        print("phonebook 1 is greater than 2")
print("phonebook1 | phonebook2 ",ph1|ph2)
print("phonebook1 & phonebook3 ",ph1&ph2)
print("phonebook1 - phonebook2 ",ph1-ph2)
print("phonebook1^phonebook2 ",ph1^ph2)
```

OUTPUT

```
Enter set 1 : 1,2,3,4,5
Enter set 2 : 7,8,9
{'3', '4', '5', '1', '2'}
{'7', '9', '8'}
phonebook 1 & 2 are not equal
phonebook 1 is greater than 2
phonebook1 | phonebook2  {'3', '4', '7', '1', '9', '2', '5', '8'}
phonebook1 & phonebook3  set()
phonebook1 - phonebook2  {'3', '4', '5', '1', '2'}
phonebook1^phonebook2  {'3', '7', '4', '9', '5', '1', '2', '8'}
```

PROGRAM NO: 30

AIM: The function make Dictionary takes 2 list as arguments that is a name list (string type) and score list (integer type) using which a dictionary is created with key as name and value as score and perform the following operations

i. Print the score of Lisa
ii. Change score of Rahul to 34
iii. Ken dropped the course so remove his details
iv. Calculate the average of scores from the dictionary
v. Print students with their score in alphabetic order

PROGRAM

```
student={}
l_s=0
def makeDictionary(name,score):
    for i in range(len(score)):
        student[name[i]]=score[i]
    return student
def perform(student):
    print("score of lisa : ",student["lisa"])
    student["rahul"]=34
    del student["ken"]
    print(student)
    average=sum(student.values())
    l_s=len(score)
    print("average of values : ",average/l_s)
    s_student=sorted(student.items())
    print("sorted : ",s_student)
```

```
name=['khaif','soba','julia','lisa','ken']
score=[10,23,15,20,48]

student=makeDictionary(name,score)
print(student)
perform(student)
```

OUTPUT

```
{'khaif': 10, 'soba': 23, 'julia': 15, 'lisa': 20, 'ken': 48}
score of lisa :  20
{'khaif': 10, 'soba': 23, 'julia': 15, 'lisa': 20, 'rahul': 34}
average of values :  20.4
sorted :  [('julia', 15), ('khaif', 10), ('lisa', 20), ('rahul', 34), ('soba', 23)]
```

PROGRAM NO: 31

AIM: Create a class Student with student_name, rollno, mark(as list).use functions calculate_avg() for calculation of average and display() for displaying the all details of students.use init()method.

PROGRAM

```
class student:
    def __init__(self,name,roll,mark):
        self.name=name
        self.roll=roll
        self.mark=mark
    def average(self):
        self.avg=sum(self.mark)/len(self.mark)
    def print_all(self):
        print(self.name,self.roll,self.avg)



ls=[34,29,40,36]
c1=student("ann",11,ls)
c1.average()
c1.print_all()
```

OUTPUT

ann 11 34.75

PROGRAM NO: 32

AIM: Create a classGrandparent with properties gname and gage,function GrangparentDetails() inherited to a class Parent with properties pname and page,function ParentDetails() inherited to a class Child with properties cname nd cage,function ChildDetails().Create an object for Child display all details of family members.

PROGRAM

```
class Grandparent:
    def __init__(self,gname,gage):
        self.gname=gname
        self.gage=gage
    def grandParentDetails(self):
        print("grand parent name : ",self.gname)
        print("grand parent age : ",self.gage)
class Parent(Grandparent):
    def __init__(self,gname,gage,pname,page):
        super().__init__(gname,gage)
        self.pname=pname
        self.page=page
    def ParentDetails(self):
        super().grandParentDetails()
        print("parent name : ",self.pname)
        print("parent age : ",self.page)
class Child(Parent):
    def __init__(self,gname,gage,pname,page,cname,cage):
        super().__init__(gname,gage,pname,page)
```

```
        self.cname=cname
        self.cage=cage
    def ChildDetails(self):
        super().ParentDetails()
        print("Child name : ",self.cname)
        print("Child age : ",self.cage)
ch=Child("jose",78,"alex",50,"alen",23)
ch.ChildDetails()
```

OUTPUT

```
grand parent name :  jose
grand parent age :  78
parent name :  alex
parent age :  50
Child name :  alen
Child age :  23
```

PROGRAM NO: 33

AIM: Create a class Personal with properties name,rollno,age with function displaypersonal() and a class Academic with properties attendance and total marks with function displayacademic() and inherit these two classes to another class student with property grade and function calculateGrade() that calculate and display the grade.Using object of student display all details related to student.

PROGRAM

```
class Personal:
    def __init__(self,name,rollno,age):
        self.name=name
        self.rollno=rollno
        self.age=age
    def displayPersonal(self):
        print("name:",self.name)
        print("rollno:",self.rollno)
        print("age:",self.age)
class Academic:
    def __init__(self,attendance,total):
        self.attendance=attendance
        self.total=total
    def displayAcademic(self):
        print("attendance:",self.attendance)
        print("total marks:",self.total)
class Student(Personal,Academic):
    def __init__(self,name,rollno,age,attendance,total):
        Personal.__init__(self,name,rollno,age)
```

```
        Academic.__init__(self,attendance,total)
        self.grade=self.calculateGrade()
    def calculateGrade(self):
        per=(self.total/600)*100
        if per>=90:
            return "A+"
        elif per>=80:
            return "A"
        elif per>=70:
            return "B+"
        elif per>=60:
            return "B"
        elif per>=50:
            return "C"
        else:
            return "F"
    def displayStudent(self):
        Personal.displayPersonal(self)
        Academic.displayAcademic(self)
        print("grade",self.grade)

s=Student("anu",2323,20,85,500)
s.displayStudent()
```

OUTPUT

name: anu

rollno: 2323

age: 20

attendance: 85

total marks: 500

grade A

PROGRAM NO: 34

AIM: Create a class Circle to implement operator overloading of addition,multiplication,not equal to and equal to.

PROGRAM

```
class Circle:
    def __init__(self,radius):
        self.radius=radius
    def __add__(self,other):
        return Circle(self.radius+other.radius)
    def __mul__(self,other):
        return Circle(self.radius*other.radius)
    def __ne__(self,other):
        return self.radius!=other.radius
    def __eq__(self,other):
        return self.radius==other.radius
c1=Circle(5)
c2=Circle(3)
c3=c1+c2
print("radius of c3 after addition:",c3.radius)
c4=c1*c2
print("radius of c4 after multiplication",c4.radius)
print("c1 and  c2 are equal",c1==c2)
print("c1 and  c2 are not equal",c1!=c2)
```

OUTPUT

```
radius of c3 after addition: 8
radius of c4 after multiplication 15
c1 and  c2 are equal False
c1 and  c2 are not equal True
```

PROGRAM NO: 35

AIM: Create a program that display an exception for TypeError and IndexError using multiple except statement

PROGRAM

```
def get_element(lst,idx):
    try:
        print(lst[idx])
    except TypeError:
        print("invalid index type")
    except IndexError:
        print("index out of range")
my_list=[6,8,4,3,2]
get_element(my_list,2)
get_element(my_list,"2")
get_element(my_list,6)
```

OUTPUT

```
4
invalid index type
index out of range
```

PROGRAM NO: 36

AIM: Create a user defined exception to raise an exception if a mark entered by the user is less than zero or greater than 100.

PROGRAM

```
class MarkError(Exception):
    pass
try:
    m=int(input("enter the mark : "))
    if m<0 or m>100:
        raise MarkError
    print("mark:",m)
except MarkError:
    print("Marks should be  between 0 and 100")
```

OUTPUT

```
enter the mark : -1
Marks should be  between 0 and 100

enter the mark : 101
Marks should be  between 0 and 100

enter the mark : 78
mark: 78
```

PROGRAM NO: 37

AIM: Create a file named test.txt with the content
Hello Programmer, Welcome to python tutorial
And perform the following operations
a) Read file content line by line
b) Read five characters from the file
c) Write a new line " Have a great learning experience"

PROGRAM

```
print("python file")
f=open("test.txt",'r')
print("printing line by line")
for line in f:
        print(line)
print("read 5 characters : ",f.read(5))
f.close()
print("appending sentence : Have a great learning experience ")
f=open("test.txt",'a')
f.write("Have a great learning experience")
f.close()
print("After appending")
f=open("test.txt",'r')
for l in f:
        print(l)
```

OUTPUT

```
python file
printing line by line
hello programmer,

welcome to python tutorial

Have a great learning experienceHave a great learning experience
read 5 characters :
appending sentence : Have a great learning experience
After appending
hello programmer,

welcome to python tutorial

Have a great learning experienceHave a great learning experienceHave a great learning experience
```

PROGRAM NO: 38

AIM: Create a database student and perform the basic database operations

a)creation
b)insertion
c)selection
d)updation
e)deletion

using sqlite3

PROGRAM

```
con=sqlite3.connect('py.db')

print("Connection successfull")

cr=con.cursor()

l=1

while l=1:

        print("choose operations \n1.creation\2.insertion\3.selection\n4.updation\n5.deletion\n")

        ch=int(input("Enter your option : "))

        if ch==1:

                try:

                        q1="create table student(rno intint,name text)"

                        cr.execute(q1)

                        print("Table create")

                        con.commint()

                except:

                        print("table already created")

        elif ch==2:
```

```
        a=int(input("Enter the rollno : "))
        b=input("Enter name : ")
        value=(a,b)
        q2="insert into student values(?,?)"
        cr.execute(q2,value)
        print("insertion successfull")
        con.commit()
    elif ch==3:
        q3="select * from student"
        cr.execute(q3)
        res=cr.fetchall()
        for i in res:
            print(i)
        con.commit()
    elif ch==4:
        c=int(input("Enter rollno to be updated : "))
        d=input("Enter name to be updated : ")
        value=(c,d)
        cr.execute("update student set name= ? where rno=?",(value))
        print("updation successful")
        con.commit()
    elif ch==5:
        e=int(input("Enter the rollno to be deleted : "))
        value=(e)
        q5="deleted from student where rno=?"
        cr.execute(q5,(value))
```

```
                print("Deletion successful")
                con.commit()
        else:
                print("Wrong choice")
        print("Do you want to continue : 1-yes || 2-No\n")
        l=int(input("Enter your response : "))
con.commit()
con.close()
```

OUTPUT

```
connection established successfully
----menu---
1.creation
2.insertion
3.selection
4.updation
5.deletion

enter your choice:1
table already created
do you want to continue
1.yes
2.no

enter your response1
----menu---
1.creation
```

```
2.insertion
3.selection
4.updation
5.deletion

enter your choice:2
enter the rollno:40
enter name:merzilin
insertion successfull
do you want to continue
1.yes
2.no

enter your response1
----menu---
1.creation
2.insertion
3.selection
4.updation
5.deletion

enter your choice:2
enter the rollno:16
enter name:Bruce Wayne
insertion successfull
do you want to continue
1.yes
```

```
2.no

enter your response1
----menu---
1.creation
2.insertion
3.selection
4.updation
5.deletion

enter your choice:3
(40, 'merzilin')
(16, 'Bruce Wayne')
do you want to continue
1.yes
2.no

enter your response1
----menu---
1.creation
2.insertion
3.selection
4.updation
5.deletion

enter your choice:4
enter rollno to be updated:40
```

```
enter the name to be updated:clark kent
updation successfully done
do you want to continue
1.yes
2.no

enter your response1
----menu---
1.creation
2.insertion
3.selection
4.updation
5.deletion

enter your choice:3
(40, 'clark kent')
(16, 'Bruce Wayne')
do you want to continue
1.yes
2.no

enter your response1
----menu---
1.creation
2.insertion
3.selection
4.updation
```

```
5.deletion

enter your choice:5
enter the roll no to be deleted:40
deletion done successfully
do you want to continue
1.yes
2.no

enter your response2
```

PROGRAM NO: 39

AIM: Static page creation

PROGRAM

1.create a Django project: create a Django project by following command Django-admin startproject sample

2.create a Django App : Inside project, create a Django app using the following command :

Cd sample

Python3 manage.py startapp student

Then add it , to the INSTALLED_APPS in the settings.py file of project

3.Define a View: in views.py, define a simple view that renders the static page:

```
from django.shortcuts import render


from django.http import HttpResponse

# Create your views here.

def home(request):

    return render(request,'home.html')
```

4.create a template: inside app folder, create a folder named templates and create a static HTML file to display.

```
<html>

<head>

</head>

<body>

<h1>this is a static page</h1>

</body>
```

```
</html>
```

5.configure URLS: define urls in the urls.py file of app to map the view to a specific URL:

```
from django.urls import path
from django.contrib import admin
from . import views

urlpatterns = [
    path(",views.home, name='home')
]
```

Define also in urls.py file of the project folder

```
from django.contrib import admin
from django.urls import path,include

urlpatterns = [
    path(",include('student.urls')),
    path('admin/', admin.site.urls),
]
```

Then make changes in settings.py in your project folder,

Add

Import os and make change in DIRS of templates

```
‘DIRS’:[os.path.join(BASE_DIR,’templates’)],
```

6.Run the development server: Run the server by the command

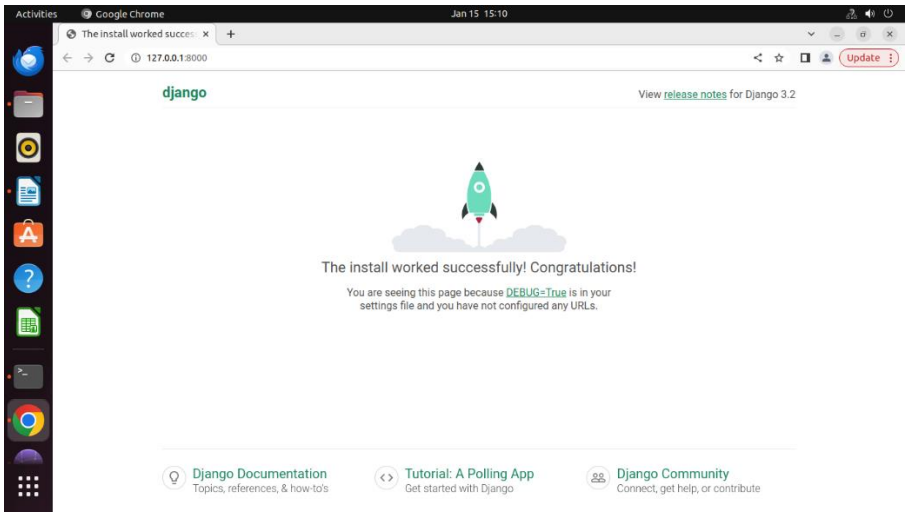
```
Python3 manage.py runserver
```

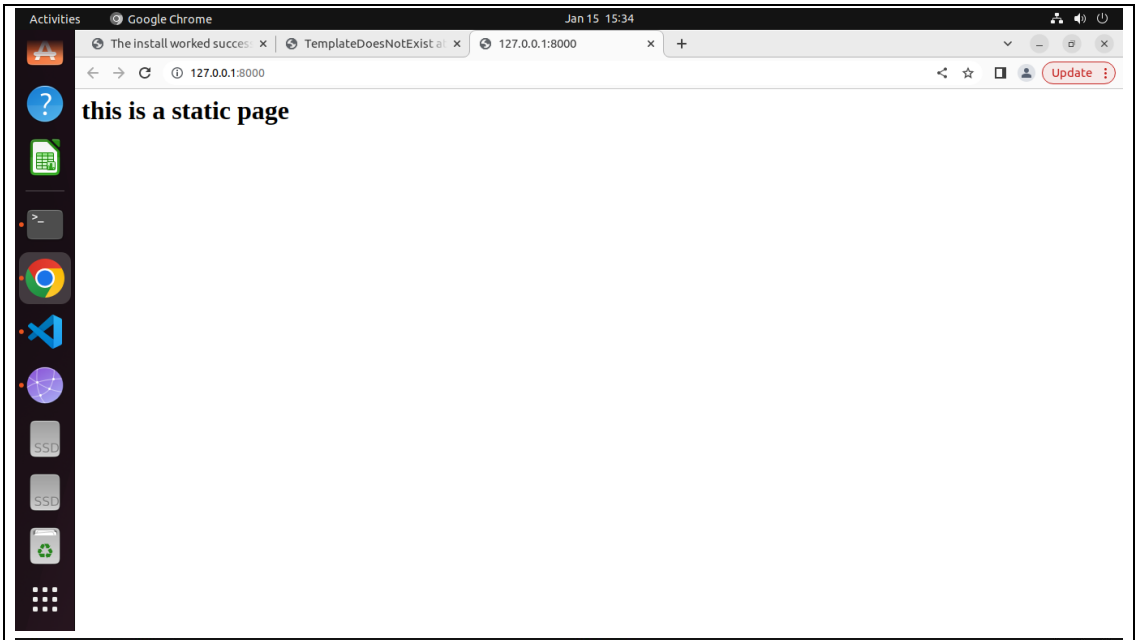
OUTPUT

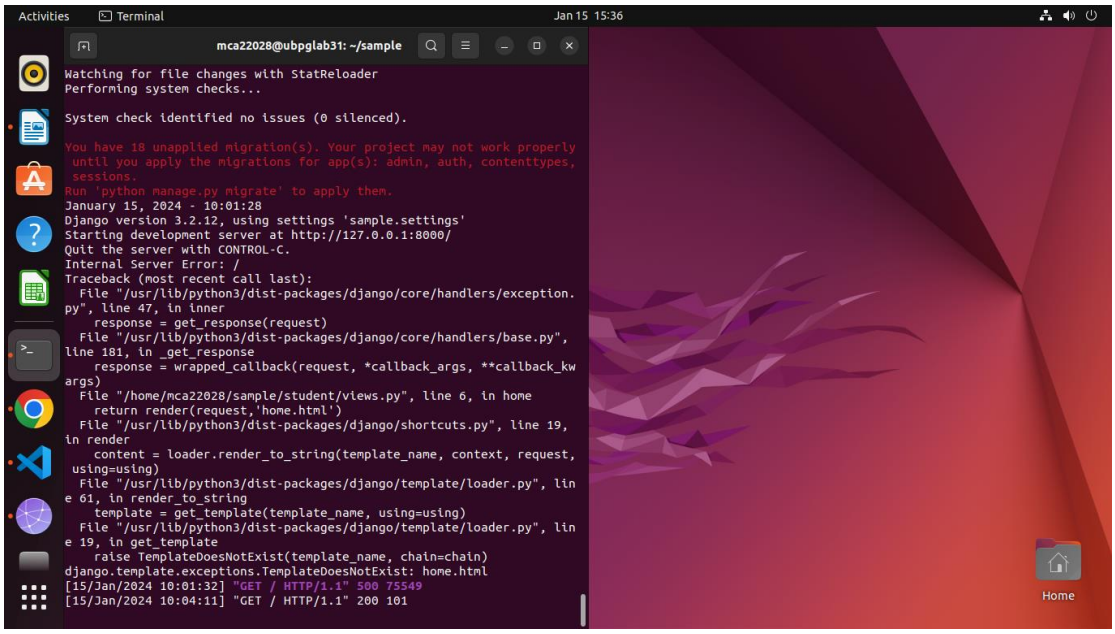
```
mca22028@ubpglab31:~$ django-admin --version
3.2.12
mca22028@ubpglab31:~$ django startproject sample
django: command not found
mca22028@ubpglab31:~$ django-admin  startproject sample
mca22028@ubpglab31:~$ ls
Desktop     mca22028  NetBeansProjects  Public  Templates
Documents   Music     Pictures          sample  Videos
Downloads   mysite    pro1              snap
mca22028@ubpglab31:~$ cd sample
mca22028@ubpglab31:~/sample$ ls
manage.py  sample
mca22028@ubpglab31:~/sample$ python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly
 until you apply the migrations for app(s): admin, auth, contenttypes,
 sessions.
Run 'python manage.py migrate' to apply them.
January 15, 2024 - 09:39:08
Django version 3.2.12, using settings 'sample.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```




Activities
Google Chrome
Jan 15 15:34
The install worked succes
TemplateDoesNotExist a
127.0.0.1:8000
127.0.0.1:8000
Update
this is a static page


Activities
Terminal
Jan 15 15:36
mca22028@ubpglab31: ~/sample
Watching for file changes with StatReloader
Performing system checks...
System check identified no issues (0 silenced).
You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
January 15, 2024 - 10:01:28
Django version 3.2.12, using settings 'sample.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
Internal Server Error: /
Traceback (most recent call last):
File "/usr/lib/python3/dist-packages/django/core/handlers/exception.py", line 47, in inner
response = get_response(request)
File "/usr/lib/python3/dist-packages/django/core/handlers/base.py", line 181, in _get_response
response = wrapped_callback(request, *callback_args, **callback_kwargs)
File "/home/mca22028/sample/student/views.py", line 6, in home
return render(request,'home.html')
File "/usr/lib/python3/dist-packages/django/shortcuts.py", line 19, in render
content = loader.render_to_string(template_name, context, request, using=using)
File "/usr/lib/python3/dist-packages/django/template/loader.py", line 61, in render_to_string
template = get_template(template_name, using=using)
File "/usr/lib/python3/dist-packages/django/template/loader.py", line 19, in get_template
raise TemplateDoesNotExist(template_name, chain=chain)
django.template.exceptions.TemplateDoesNotExist: home.html
[15/Jan/2024 10:01:32] "GET / HTTP/1.1" 500 75549
[15/Jan/2024 10:04:11] "GET / HTTP/1.1" 200 101
Home

PROGRAM NO: 40

AIM: Dynamic page creation

PROGRAM

1.create a Django project: create a Django project by following command Django-admin startproject sample

2.create a Django App : Inside project, create a Django app using the following command :

Cd sample

Python3 manage.py startapp student

Then add it , to the INSTALLED_APPS in the settings.py file of project

3.create a model class:

Model is nothing but the source of information about the data where we can define the type of data, behavior.

Usually, one model class maps to one database table.

In models.py, define

```
from django.db import models


# Create your models here.
class Details(models.Model):
    type = models.CharField(
        max_length = 20,
        default='',
        null = False
    )
    name = models.CharField(
        max_length = 20,
```

```
        default = '',
        null = False
    )
```

Making migrations and setting database

After writing the Django model, we need to create the migration by running the following command

Python3 manage.py makemigrations

To migrate

Run command:

Python3 manage.py migrate

4.create a template:inside the app folder create a html file.

```
<html>
    <head>


    </head>
    <body>
        <H1>Add Content</H1>
        <form action="" method="POST">
            {% csrf_token %}
            <br>
            <input type="text" name="type" placeholder="Enter type"><br>
            <input type="text" name="name" placeholder="Enter name"><br>
            <input type="submit" value="Add Data"><br>
        </form>
    </body>
```

```
</html>
```

5.define a view: open views.py file in the app and define a simple view that renders the dynamic page:

```
from django.shortcuts import render


from django.http import HttpResponse
# Create your views here.
def home(request):
    if request.method == "POST":
        type = request.POST['type']
        name = request.POST['name']
        print(type,name)
    return render(request,'home.html')
```

6.configure urls: define the urls in the urls.py file inside the app folder, to map the view to a specific urls:

```
from django.urls import path
from django.contrib import admin


from . import views


urlpatterns = [
    path('',views.home, name='home')
]
```

Also include these app-specific urls in project urls.py file

```
from django.contrib import admin
from django.urls import path,include
```

```
urlpatterns = [
    path('',include('student.urls')),
    path('admin/', admin.site.urls),
]
```

Then make changes in settings.py in your project folder,

Add

Import os and make change in DIRS of templates

'DIRS':[os.path.join(BASE_DIR,'templates')],

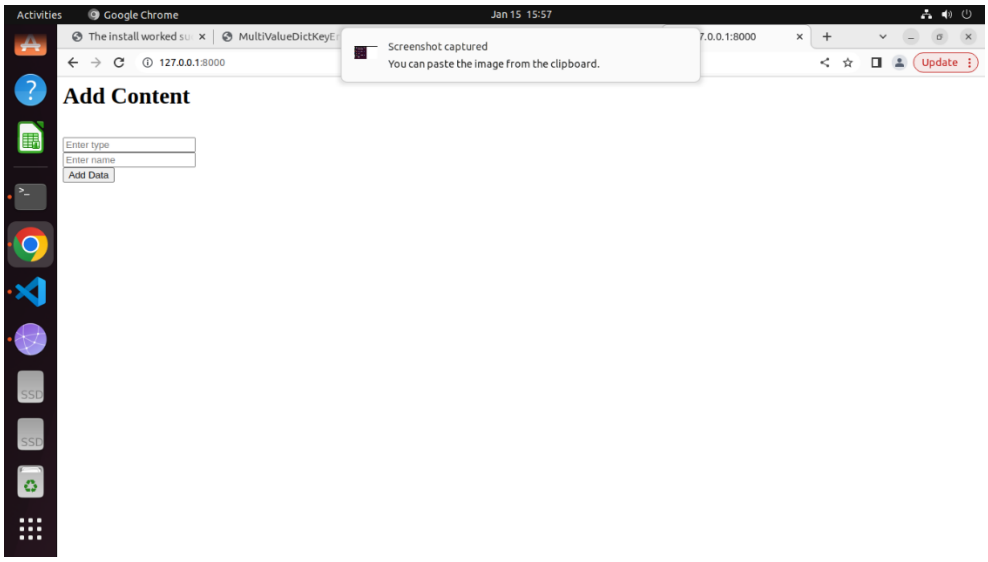
7.run the development server:

Start the development server by running the command:

Python3 manage.py runserver

OUTPUT

```
mca22028@ubpglab31:~/sample$ django-admin startapp student
mca22028@ubpglab31:~/sample$ ls
db.sqlite3  manage.py  sample  student
mca22028@ubpglab31:~/sample$ cd student ; ls
admin.py  __init__.py  models.py  views.py
apps.py   migrations   tests.py
mca22028@ubpglab31:~/sample/student$
```


Activities
Google Chrome
Jan 15 15:57
The install worked su
MultiValueDictKeyE
7.0.0.1:8000
Screenshot captured
You can paste the image from the clipboard.
127.0.0.1:8000
Update
Add Content
Enter type
Enter name
Add Data

```
Activities    Terminal                                                    Jan 15
mca22028@ubpglab31: ~/sample
January 15, 2024 - 10:01:28
Django version 3.2.12, using settings 'sample.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
Internal Server Error: /
Traceback (most recent call last):
  File "/usr/lib/python3/dist-packages/django/core/handlers/exception.
py", line 47, in inner
    response = get_response(request)
  File "/usr/lib/python3/dist-packages/django/core/handlers/base.py",
line 181, in _get_response
    response = wrapped_callback(request, *callback_args, **callback_kw
args)
  File "/home/mca22028/sample/student/views.py", line 6, in home
    return render(request,'home.html')
  File "/usr/lib/python3/dist-packages/django/shortcuts.py", line 19,
in render
    content = loader.render_to_string(template_name, context, request,
 using=using)
  File "/usr/lib/python3/dist-packages/django/template/loader.py", lin
e 61, in render_to_string
    template = get_template(template_name, using=using)
  File "/usr/lib/python3/dist-packages/django/template/loader.py", lin
e 19, in get_template
    raise TemplateDoesNotExist(template_name, chain=chain)
django.template.exceptions.TemplateDoesNotExist: home.html
[15/Jan/2024 10:01:32] "GET / HTTP/1.1" 500 75549
[15/Jan/2024 10:04:11] "GET / HTTP/1.1" 200 101
^Cmca22028@ubpglab31:~/sample$ pwd
/home/mca22028/sample
mca22028@ubpglab31:~/sample$ python3 manage.py makemigrations
No changes detected
mca22028@ubpglab31:~/sample$ python3 manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  No migrations to apply.
mca22028@ubpglab31:~/sample$
```

```
Activities    Terminal                                                    Jan 1
mca22028@ubpglab31: ~/sample
    return Template(self.engine.get_template(template_name), self)
  File "/usr/lib/python3/dist-packages/django/template/engine.py", lin
e 143, in get_template
    template, origin = self.find_template(template_name)
  File "/usr/lib/python3/dist-packages/django/template/engine.py", lin
e 125, in find_template
    template = loader.get_template(name, skip=skip)
  File "/usr/lib/python3/dist-packages/django/template/loaders/base.py
", line 29, in get_template
    return Template(
  File "/usr/lib/python3/dist-packages/django/template/base.py", line
155, in __init__
    self.nodelist = self.compile_nodelist()
  File "/usr/lib/python3/dist-packages/django/template/base.py", line
193, in compile_nodelist
    return parser.parse()
  File "/usr/lib/python3/dist-packages/django/template/base.py", line
472, in parse
    self.invalid_block_tag(token, command, parse_until)
  File "/usr/lib/python3/dist-packages/django/template/base.py", line
531, in invalid_block_tag
    raise self.error(
django.template.exceptions.TemplateSyntaxError: Invalid block tag on l
ine 9: 'csrf'. Did you forget to register or load this tag?
[15/Jan/2024 10:25:01] "GET / HTTP/1.1" 500 118876
^Cmca22028@ubpglab31:~/sample$ python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
January 15, 2024 - 10:26:03
Django version 3.2.12, using settings 'sample.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
[15/Jan/2024 10:26:08] "GET / HTTP/1.1" 200 498
fruit orange
[15/Jan/2024 10:26:25] "POST / HTTP/1.1" 200 498
```

Program NO:41

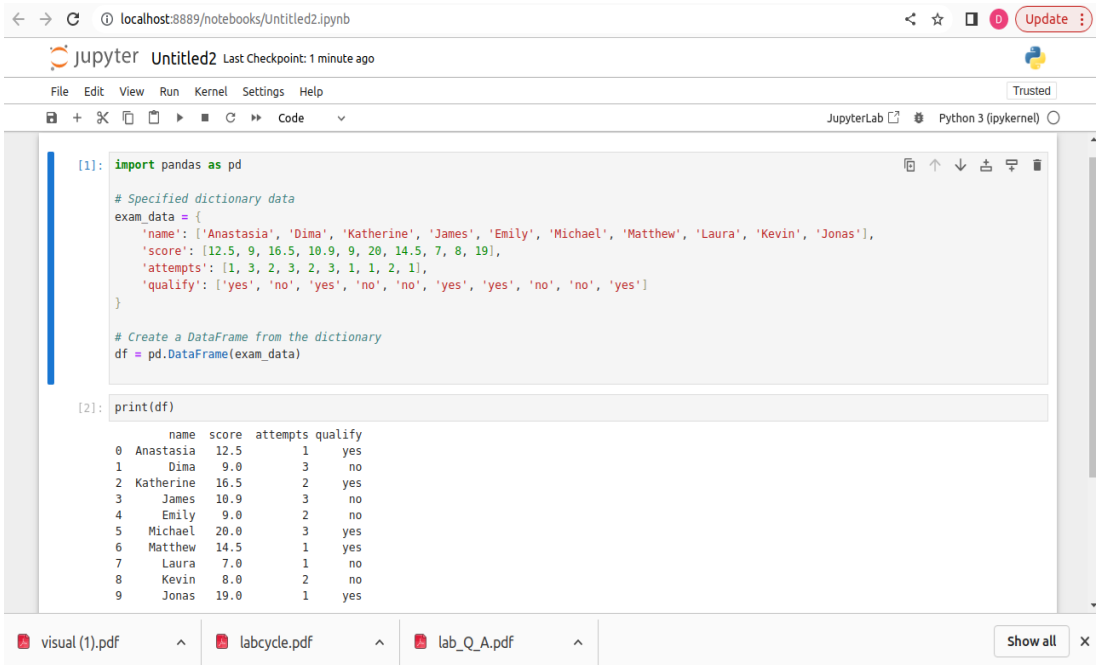
AIM: Write a Pandas program to create and display a DataFrame from

a specified dictionary data
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James',
'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, 10.9, 9, 20, 14.5, 7, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
a) Display the values in column names using DataFrame.[ ]
b) Display names with scores using DataFrame.loc[ ]
c) Display 3rd and 5th rows related with columns 1 and 3 using
iloc [ ] function
d) Set name as the index of the data

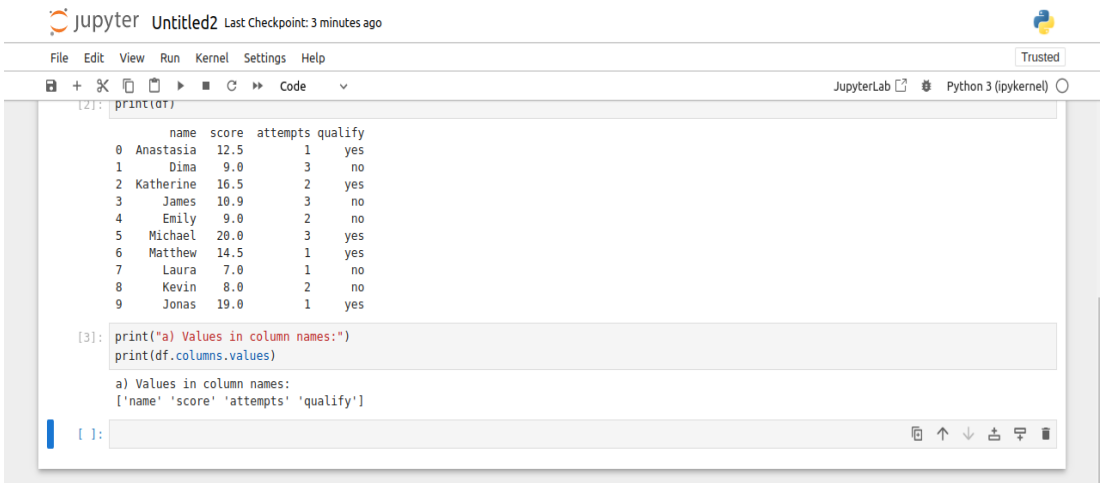
```
import pandas as pd

# Specified dictionary data
exam_data = {
    'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael',
'Matthew', 'Laura', 'Kevin', 'Jonas'],
    'score': [12.5, 9, 16.5, 10.9, 9, 20, 14.5, 7, 8, 19],
    'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
    'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
}

# Create a DataFrame from the dictionary
df = pd.DataFrame(exam_data)
```



# a) Display the values in column names using DataFrame.[ ]
print("a) Values in column names:")
print(df.columns.values)



# b) Display names with scores using DataFrame.loc[ ]
print("\nb) Names with scores:")
print(df.loc[:, ['name', 'score']])

```
[3]: print("a) Values in column names:")
     print(df.columns.values)

a) Values in column names:
['name' 'score' 'attempts' 'qualify']

[4]: print("\nb) Names with scores:")
     print(df.loc[:, ['name', 'score']])

b) Names with scores:
        name  score
0  Anastasia   12.5
1       Dima    9.0
2  Katherine   16.5
3      James   10.9
4      Emily    9.0
5    Michael   20.0
6    Matthew   14.5
7      Laura    7.0
8      Kevin    8.0
9      Jonas   19.0
```

# c) Display 3rd and 5th rows related to columns 1 and 3 using iloc[ ] function
print("\nc) 3rd and 5th rows related to columns 1 and 3:")
print(df.iloc[[2, 4], [0, 2]])

```
[5]: print("\nc) 3rd and 5th rows related to columns 1 and 3:")
     print(df.iloc[[2, 4], [0, 2]])

c) 3rd and 5th rows related to columns 1 and 3:
       name  attempts
2  Katherine        2
4     Emily        2
```

# d) Set name as the index of the data
df.set_index('name', inplace=True)
print("\ne) DataFrame with 'name' as the index:")

```
[7]: df.set_index('name', inplace=True)
     print("\ne) DataFrame with 'name' as the index:")

e) DataFrame with 'name' as the index:

[8]: print(df)

            score  attempts qualify
name
Anastasia    12.5         1     yes
Dima          9.0         3      no
Katherine    16.5         2     yes
James        10.9         3      no
Emily         9.0         2      no
Michael      20.0         3     yes
Matthew      14.5         1     yes
Laura         7.0         1      no
Kevin         8.0         2      no
Jonas        19.0         1     yes
```

print(df)

PROGRAM NO:42
AIM: Using numpy do the following operations
a) Create a 2-D array with the following elements

[1,2,3,4,5][6,7,8,9,10]
b) Print the 2nd element of 2nd row using index operator. Choose the same using negative indexing method` also.
c) Slice elements from columns 2 and 3 related to row2

```
import numpy as np

# a) Create a 2-D array
array_2d = np.array([[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]])

# b) Print the 2nd element of the 2nd row using index operator and
negative indexing
element_2nd_row_index = array_2d[1, 1]
element_2nd_row_neg_index = array_2d[-1, -4]

print("b) 2nd element of 2nd row using index operator:",
element_2nd_row_index)
print("   2nd element of 2nd row using negative indexing:",
element_2nd_row_neg_index)

# c) Slice elements from columns 2 and 3 related to row 2
sliced_elements = array_2d[1, 1:3]

print("\nc) Sliced elements from columns 2 and 3 of row 2:",
sliced_elements)
```

jupyter Untitled2 Last Checkpoint: 10 minutes ago

File Edit View Run Kernel Settings Help

Code JupyterLab Python 3 (ipykernel)

```
NameError: name 'np' is not defined
```

[10]:
```
import numpy as np
array_2d = np.array([[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]])
```

[11]:
```
element_2nd_row_index = array_2d[1, 1]
element_2nd_row_neg_index = array_2d[-1, -4]

print("b) 2nd element of 2nd row using index operator:", element_2nd_row_index)
print("   2nd element of 2nd row using negative indexing:", element_2nd_row_neg_index)
```

```
b) 2nd element of 2nd row using index operator: 7
   2nd element of 2nd row using negative indexing: 7
```

[12]:
```
sliced_elements = array_2d[1, 1:3]

print("\nc) Sliced elements from columns 2 and 3 of row 2:", sliced_elements)
```

```
c) Sliced elements from columns 2 and 3 of row 2: [7 8]
```

PROGRAM NO:43

AIM: Using matplotlib visualize the data

```
from matplotlib import pyplot as plt
x = [1,2,3]
y = [2,4,1]
plt.plot(x,y)
plt.title("Info")
plt.ylabel("Y axis")
plt.xlabel("X axis")
plt.show()
```