

COMPUTER SCIENCE

PRACTICAL PROGRAMS

FOR GRADE – XII
[2023-2024]

Prepared by:

Mr. Appan Raj D M.C.A., B.Ed., – PGT CS/IP

TABLE OF CONTENTS

S.No	Name of the Exercise	Page.No
Python Programs		
1.	Creating a menu driven program to perform arithmetic operations.	3
2.	Creating a python program to display Fibonacci series	5
3.	Creating a menu driven program to find factorial and sum of list of numbers using function.	7
4.	Creating a python program to implement returning value(s) from function.	9
5.	Creating a python program to implement mathematical functions.	10
6.	Creating a python program to generate random number between 1 to 6	11
7.	Creating a python program to read a text file line by line and display each word separated by '#'. separated by '#'.	12
8.	Creating a python program to read a text file and display the number of vowels/consonants/lower case/ upper case characters.	13
9.	Creating python program to display short words from a text file	15
10.	Creating a python program to copy particular lines of a text file into another text file.	16
11.	Creating a python program to create and search records in binary file.	17
12.	Creating a python program to create and update/modify records in binary file.	19
13.	Creating a python program to create and search employee's record in csv file.	21
14.	Creating a python program to implement stack operations (List).	23
15.	Creating a python program to implement stack operations (Dictionary).	26
Python – SQL connectivity programs		
16.	Creating a python program to integrate MYSQL with Python (Creating database and table)	28
17.	Creating a python program to integrate MYSQL with Python (Inserting records and displaying records)	30
18.	Creating a python program to integrate MYSQL with Python (Searching and displaying records)	32
19.	Creating a python program to integrate MYSQL with Python (Updating records)	33
SQL Queries		
20.	SQL COMMANDS EXERCISE – 1	35
21.	SQL COMMANDS EXERCISE – 2	36
22.	SQL COMMANDS EXERCISE – 3	38
23.	SQL COMMANDS EXERCISE – 4	40
24.	SQL COMMANDS EXERCISE – 5	42

EX.NO: 1

DATE:

CREATING A MENU DRIVEN PROGRAM TO PERFORM ARITHMETIC OPERATIONS

AIM:

To write a menu driven Python Program to perform Arithmetic operations (+,-*,/) based on the user's choice.

SOURCE CODE:

```
print("1. Addition")
print("2. Subtraction")
print("3. Multiplication")
print("4. Division")
opt=int(input("Enter your choice:"))
a=int(input("Enter the First Number:"))
b=int(input("Enter the Second Number:"))

if opt==1:
    c=a+b
    print("The Addition of two number is:",c)

elif opt==2:
    c=a-b
    print("The Subtraction of two number is:",c)

elif opt==3:
    c=a*b
    print("The Multiplication of two number is:",c)

elif opt==4:
    if b==0:
        print("Enter any other number other than 0")
    else:
        c=a/b
        print("The Division of two number is:",c)
else:
    print("Invalid Option")
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUTPUT:

Python Program Executed Output:

```
1. Adddition
2. Subtraction
3. Multiplication
4. Division
Enter your choice:1
Enter the First Number:10
Enter the Second Number:23
The Addition of two number is: 33
```

EX.NO: 2

DATE:

CREATING A PYTHON PROGRAM TO DISPLAY FIBONACCI SERIES

AIM:

To write a Python Program to display Fibonacci Series up to 'n' numbers.

SOURCE CODE:

```
First=0
Second=1
no=int(input("How many Fibonacci numbers you want to display?"))
if no<=0:
    print("Please Enter Positive Integer")
else:
    print(First)
    print(Second)
    for i in range(2,no):
        Third=First+Second
        First=Second
        Second=Third
        print(Third)
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUTPUT:

Python Executed Program Output:

How many Fibonacci numbers you want to display?8

0

1

1

2

3

5

8

13

EX.NO: 3

DATE:

CREATING A MENU DRIVEN PROGRAM TO FIND FACTORIAL AND SUM OF LIST OF NUMBERS USING FUNCTION.

AIM:

To write a menu driven Python Program to find Factorial and sum of list of numbers using function.

SOURCE CODE:

```
def Factorial(no):
    F=1
    if no<0:
        print("Sorry, we cannot take Factorial for Negative number")
    elif no==0:
        print("The Factorial of 0 is 1")
    else:
        for i in range(1,no+1):
            F=F*i
        print("The Factorial of",no,"is:",F)

def Sum_List(L):
    Sum=0
    for i in range(n):
        Sum=Sum+L[i]
    print("The Sum of List is:",Sum)

# Main Program

print("1. To Find Factorial")
print("2. To Find sum of List elements")
opt=int(input("Enter your choice:"))

if opt==1:
    n=int(input("Enter a number to find Factorial:"))
    Factorial(n)

elif opt==2:
    L=[]
    n=int(input("Enter how many elements you want to store in List?:"))
    for i in range(n):
        ele=int(input())
        L.append(ele)
    Sum_List(L)
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUTPUT:

Python Executed Program Output:

```
1. To Find Factorial
2. To Find sum of List elements
Enter your choice:1
Enter a number to find Factorial:5
The Factorial of 5 is: 120
```

EX.NO: 4

DATE:

**CREATING A PYTHON PROGRAM TO IMPLEMENT RETURNING VALUE(S)
FROM FUNCTION**

AIM:

To Write a Python program to define the function Check(no1,no2) that take two numbers and Returns the number that has minimum ones digit.

Source Code:

```
def Check(no1,no2):  
    if (no1%10) < (no2%10):  
        return no1  
    else:  
        return no2  
  
a=int(input("Enter the First number:"))  
b=int(input("Enter the Second number:"))  
r=Check(a,b)  
print("The Number that has minimum one's digit is:",r)
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

Sample Output:

```
Enter the First number:245  
Enter the Second number:342  
The Number that has minimum one's digit is: 342
```

EX.NO: 5

DATE:

CREATING A PYTHON PROGRAM TO IMPLEMENT MATHEMATICAL FUNCTIONS

AIM:

To write a Python program to implement python mathematical functions to find:

- (i) To find Square of a Number.
- (ii) To find Log of a Number(i.e. Log_{10})
- (iii) To find Quad of a Number

SOURCE CODE:

```
import math

def Square(num):
    S=math.pow(num,2)
    return S

def Log(num):
    S=math.log10(num)
    return S

def Quad(X,Y):
    S=math.sqrt(X**2 + Y**2)
    return S

print("The Square of a Number is:",Square(5))
print("The Log of a Number is:",Log(10))
print("The Quad of a Number is:",Quad(5,2))
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUTPUT:

Python Executed Program Output:

```
The Square of a Number is: 25.0
The Log of a Number is: 1.0
The Quad of a Number is: 5.385164807134504
```

EX.NO: 6

DATE:

**CREATING A PYTHON PROGRAM TO GENERATE RANDOM NUMBER
BETWEEN 1 TO 6**

AIM:

To write a Python program to generate random number between 1 to 6 to simulate the dice.

SOURCE CODE:

```
import random
while True:
    Choice=input("\nDo you want to roll the dice(y/n):")
    no=random.randint(1,6)
    if Choice=='y':
        print("\nYour Number is:",no)
    else:
        break
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUTPUT:

Python Executed Output Program:

```
Do you want to roll the dice(y/n):y
Your Number is: 5
Do you want to roll the dice(y/n):y
Your Number is: 2
Do you want to roll the dice(y/n):y
Your Number is: 1
Do you want to roll the dice(y/n):n
```

EX.NO: 7

DATE:

**CREATING A PYTHON PROGRAM TO READ A TEXT FILE LINE BY LINE AND
DISPLAY EACH WORD SEPARATED BY '#'**

AIM:

To write a Python Program to Read a text file "Story.txt" line by line and display each word separated by '#'.

SOURCE CODE:

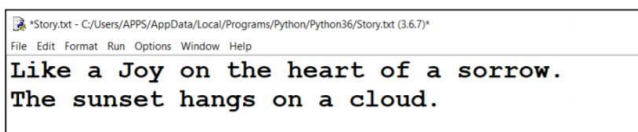
```
f=open("Story.txt",'r')
Contents=f.readlines()
for line in Contents:
    words=line.split()
    for i in words:
        print(i+'#',end=' ')
    print("")
f.close()
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

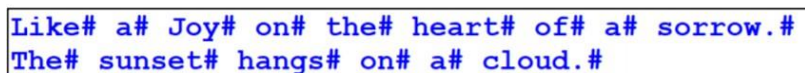
SAMPLE OUTPUT:

Story.txt:

A screenshot of a text editor window titled "*Story.txt - C:/Users/APP/APPData/Local/Programs/Python/Python36/Story.txt (3.6.7)". The window contains two lines of text: "Like a Joy on the heart of a sorrow." and "The sunset hangs on a cloud." The text is in a monospaced font.

Like a Joy on the heart of a sorrow.
The sunset hangs on a cloud.

Python Program Executed Output:

A screenshot of the output from running the Python program. It shows two lines of text where each word is followed by a hash symbol (#). The text is: "Like# a# Joy# on# the# heart# of# a# sorrow.#" and "The# sunset# hangs# on# a# cloud.#".

Like# a# Joy# on# the# heart# of# a# sorrow.#
The# sunset# hangs# on# a# cloud.#

EX.NO: 8

DATE:

**CREATING A PYTHON PROGRAM TO READ A TEXT FILE AND DISPLAY THE
NUMBER OF VOWELS/CONSONANTS/LOWER CASE/ UPPER CASE CHARACTERS.**

AIM:

To write a Python Program to read a text file "Story.txt" and displays the number of Vowels/ Consonants/ Lowercase / Uppercase/characters in the file.

SOURCE CODE:

```
f=open("Story.txt",'r')
Contents=f.read()
Vowels=0
Consonants=0
Lower_case=0
Upper_case=0

for ch in Contents:
    if ch in 'aeiouAEIOU':
        Vowels=Vowels+1
    if ch in 'bcdfghjklmnpqrstvwxyzBCDFGHJKLMNPQRSTVWXYZ':
        Consonants=Consonants+1
    if ch.islower():
        Lower_case=Lower_case+1
    if ch.isupper():
        Upper_case=Upper_case+1

f.close()
print("The total numbers of vowels in the file:",Vowels)
print("The total numbers of consonants in the file:",Consonants)
print("The total numbers of uppercase in the file:",Upper_case)
print("The total numbers of lowercase in the file:",Lower_case)
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUTPUT:

Story.txt:

```
*Story.txt - C:/Users/APPS/AppData/Local/Programs/Python/Python36/Story.txt (3.6.7)*  
File Edit Format Run Options Window Help  
Like a Joy on the heart of a sorrow.  
The sunset hangs on a cloud.
```

Python Program Executed Output:

```
The total numbers of vowels in the file: 20  
The total numbers of consonants in the file: 29  
The total numbers of uppercase in the file: 3  
The total numbers of lowercase in the file: 46
```

EX.NO: 9

DATE:

CREATING PYTHON PROGRAM TO DISPLAY SHORT WORDS FROM A TEXT FILE

AIM:

To Write a method Disp() in Python, to read the lines from **poem.txt** and display those words which are less than 5 characters.

Source Code:

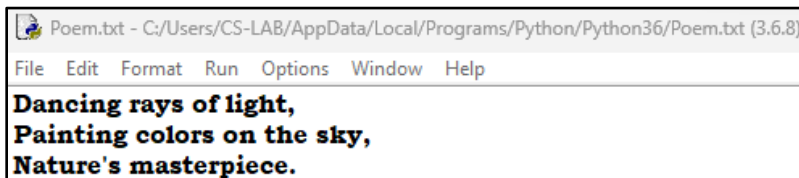
```
def Disp():  
    F=open("Poem.txt")  
    S=F.read()  
    W=S.split()  
    print("The follwoing words are less than 5 characters")  
    for i in W:  
        if len(i)<5:  
            print(i,end=' ')  
    F.close()  
  
Disp()
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

Sample Output:

Poem.txt:



Poem.txt - C:/Users/CS-LAB/AppData/Local/Programs/Python/Python36/Poem.txt (3.6.8)
File Edit Format Run Options Window Help
**Dancing rays of light,
Painting colors on the sky,
Nature's masterpiece.**

Python Executed Program Output:

**The follwoing words are less than 5 characters
rays of on the sky,**

EX.NO: 10

DATE:

**CREATING A PYTHON PROGRAM TO COPY PARTICULAR LINES OF A TEXT FILE
INTO AN ANOTHER TEXT FILE**

AIM:

To write a python program to read lines from a text file "**Sample.txt**" and copy those lines into another file which are starting with an alphabet 'a' or 'A'.

SOURCE CODE:

```
F1=open("Story.txt",'r')
F2=open("New.txt",'w')
S=F1.readlines()
for i in S:
    if i[0]=='A' or i[0]=='a':
        F2.write(i)
print("Written Successfully")
F1.close()
F2.close()
```

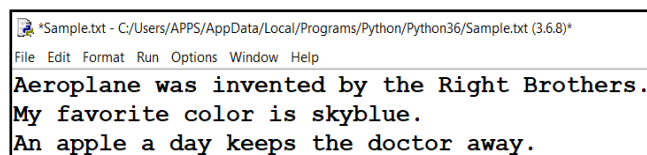
Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUTPUT:

Python Executed Program output:

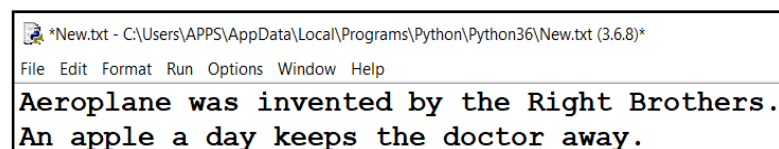
Sample.txt:



Python Executed Program Output:

```
>>>
RESTART: C:\Users\APPS\AppData\Local\Programs\Python\Python36\PR_COPY_TEXT_FILE.py
All lines which are starting with character 'a' or 'A' has been copied successfully into New.txt
```

New.txt:



EX.NO: 11

DATE:

**CREATING A PYTHON PROGRAM TO CREATE AND SEARCH RECORDS IN
BINARY FILE**

AIM:

To write a Python Program to Create a binary file with roll number and name. Search for a given roll number and display the name, if not found display appropriate message.

SOURCE CODE:

```
import pickle
def Create():
    F=open("Students.dat", 'ab')
    opt='y'
    while opt=='y':
        Roll_No=int(input('Enter roll number:'))
        Name=input("Enter Name:")
        L=[Roll_No,Name]
        pickle.dump(L,F)
        opt=input("Do you want to add another student detail(y/n):")
    F.close()

def Search():
    F=open("Students.dat", 'rb')
    no=int(input("Enter Roll.No of student to search:"))
    found=0
    try:
        while True:
            S=pickle.load(F)
            if S[0]==no:
                print("The searched Roll.No is found and Details are:",S)
                found=1
                break
    except:
        F.close()

    if found==0:
        print("The Searched Roll.No is not found")

#Main Program
Create()
Search()
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUPUT:

PYTHON PROGRAM EXECUTED OUTPUT:

```
Enter roll number:1
Enter Name:Arun
Do you want to add another student detail(y/n):y
Enter roll number:2
Enter Name:Bala
Do you want to add another student detail(y/n):y
Enter roll number:3
Enter Name:Charan
Do you want to add another student detail(y/n):y
Enter roll number:4
Enter Name:Dinesh
Do you want to add another student detail(y/n):y
Enter roll number:5
Enter Name:Divya
Do you want to add another student detail(y/n):n

Enter Roll.No of student to search:3
The searched Roll.No is found and Details are: [3, 'Charan']
```

EX.NO: 12

DATE:

CREATING A PYTHON PROGRAM TO CREATE AND UPDATE/MODIFY RECORDS IN BINARY FILE

AIM:

To write a Python Program to Create a binary file with roll number, name, mark and update/modify the mark for a given roll number.

SOURCE CODE:

```
import pickle
def Create():
    F=open("Marks.dat", 'ab')
    opt='y'
    while opt=='y':
        Roll_No=int(input('Enter roll number:'))
        Name=input("Enter Name:")
        Mark=int(input("Enter Marks:"))
        L=[Roll_No, Name, Mark]
        pickle.dump(L, F)
        opt=input("Do you want to add another student detail(y/n):")
    F.close()

def Update():
    F=open("Marks.dat", 'rb+')
    no=int(input("Enter Student Roll.No to modify marks:"))
    found=0
    try:
        while True:
            Pos=F.tell()
            S=pickle.load(F)
            if S[0]==no:
                print("The searched Roll.No is found and Details are:", S)
                S[2]=int(input("Enter New Mark to be update:"))
                F.seek(Pos)
                pickle.dump(S, F)
                found=1
                F.seek(Pos) #moving the file pointer again beginning of the current record.
                print("Mark updated Successfully and Details are:", S)
                break
    except:
        F.close()

    if found==0:
        print("The Searched Roll.No is not found")

#Main Program
Create()
Update()
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUTPUT:

PYTHON PROGRAM EXECUTED OUTPUT:

```
Enter roll number:1
Enter Name:Arun
Enter Marks:450
Do you want to add another student detail(y/n):y
Enter roll number:2
Enter Name:Bala
Enter Marks:342
Do you want to add another student detail(y/n):y
Enter roll number:3
Enter Name:Charan
Enter Marks:423
Do you want to add another student detail(y/n):y
Enter roll number:4
Enter Name:Dinesh
Enter Marks:356
Do you want to add another student detail(y/n):y
Enter roll number:5
Enter Name:Divya
Enter Marks:476
Do you want to add another student detail(y/n):n

Enter Student Roll.No to modify marks:3
The searched Roll.No is found and Details are: [3, 'Charan', 423]
Enter New Mark to be update:470
Mark updated Successfully and Details are: [3, 'Charan', 470]
```

EX.NO: 13

DATE:

**CREATING A PYTHON PROGRAM TO CREATE AND SEARCH EMPLOYEE'S RECORD
IN CSV FILE.**

AIM:

To write a Python program Create a CSV file to store Empno, Name, Salary and search any Empno and display Name, Salary and if not found display appropriate message.

SOURCE CODE:

```
import csv
def Create():
    F=open("Emp.csv",'a',newline='')
    W=csv.writer(F)
    opt='y'
    while opt=='y':
        No=int(input("Enter Employee Number:"))
        Name=input("Enter Employee Name:")
        Sal=float(input("Enter Employee Salary:"))
        L=[No,Name,Sal]
        W.writerow(L)
        opt=input("Do you want to continue(y/n)?:")
    F.close()

def Search():
    F=open("Emp.csv",'r',newline='')
    no=int(input("Enter Employee number to search"))
    found=0
    row=csv.reader(F)
    for data in row:
        if data[0]==str(no):
            print("\nEmployee Deatils are:")
            print("=====")
            print("Name:",data[1])
            print("Salary:",data[2])
            print("=====")
            found=1
            break

    if found==0:
        print("The searched Employee number is not found")
    F.close()

#Main Program
Create()
Search()
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUTPUT:

PYTHON PROGRAM EXECUTED OUTPUT:

```
Enter Employee Number:1
Enter Employee Name:Anand
Enter Employee Salary:23000
Do you want to continue(y/n)? :y
Enter Employee Number:2
Enter Employee Name:Akash
Enter Employee Salary:25000
Do you want to continue(y/n)? :y
Enter Employee Number:3
Enter Employee Name:Balu
Enter Employee Salary:27000
Do you want to continue(y/n)? :y
Enter Employee Number:4
Enter Employee Name:Bavya
Enter Employee Salary:29000
Do you want to continue(y/n)? :y
Enter Employee Number:5
Enter Employee Name:Manoj
Enter Employee Salary:35000
Do you want to continue(y/n)? :n
```

Enter Employee number to search:3

Employee Deatils are:

```
=====
Name: Balu
Salary: 27000.0
=====
```

EX.NO: 14

DATE:

CREATING A PYTHON PROGRAM TO IMPLEMENT STACK OPERATIONS(LIST)

AIM:

To write a Python program to implement Stack using a list data-structure, to perform the following operations:

- (i) To Push an object containing Doc_ID and Doc_name of doctors who specialize in "ENT" to the stack.
- (ii) To Pop the objects from the stack and display them.
- (iii)
- (iv) To display the elements of the stack (after performing PUSH or POP)

SOURCE CODE:

```
def Push():
    Doc_ID=int(input("Enter the Doctor ID:"))
    Doc_Name=input("Enter the Name of the Doctor:")
    Mob=int(input("Enter the Mobile Number of the Doctor:"))
    Special=input("Enter the Specialization:")
    if Special=="ENT":
        Stack.append([Doc_ID,Doc_Name])

def Pop():
    if Stack==[ ]:
        print("Stack is empty")
    else:
        print("The deleted doctor detail is:",Stack.pop())

def Peek():
    if Stack==[ ]:
        print("Stack is empty")
    else:
        top=len(Stack)-1
        print("The top of the stack is:",Stack[top])

def Disp():
    if Stack==[ ]:
        print("Stack is empty")
    else:
        top=len(Stack)-1
        for i in range(top,-1,-1):
            print(Stack[i])
```

```

Stack=[ ]
ch='y'
print("Performing Stack Operations Using List\n")
while ch=='y' or ch=='Y':
    print()
    print("1.PUSH")
    print("2.POP")
    print("3.PEEK")
    print("4.Disp")
    opt=int(input("Enter your choice:"))
    if opt==1:
        Push()
    elif opt==2:
        Pop()
    elif opt==3:
        Peek()
    elif opt==4:
        Disp()
    else:
        print("Invalid Choice, Try Again!!!")
    ch=input("\nDo you want to Perform another operation(y/n):")

```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUTPUT:

Python Program Executed Output:

Performing Stack Operations Using List

```

1.PUSH
2.POP
3.PEEK
4.Disp
Enter your choice:1
Enter the Doctor ID:1
Enter the Name of the Doctor:Arun
Enter the Mobile Number of the Doctor:9858
Enter the Specialization:ENT

Do you want to Perform another operation(y/n):y

1.PUSH
2.POP
3.PEEK
4.Disp
Enter your choice:1
Enter the Doctor ID:2
Enter the Name of the Doctor:Usha
Enter the Mobile Number of the Doctor:8785
Enter the Specialization:Cardio

Do you want to Perform another operation(y/n):y

1.PUSH
2.POP
3.PEEK
4.Disp
Enter your choice:1
Enter the Doctor ID:3
Enter the Name of the Doctor:Murali
Enter the Mobile Number of the Doctor:7854
Enter the Specialization:ENT

Do you want to Perform another operation(y/n):y

```

1.PUSH
2.POP
3.PEEK
4.Disp
Enter your choice:1
Enter the Doctor ID:4
Enter the Name of the Doctor:Rani
Enter the Mobile Number of the Doctor:8778
Enter the Specialization:Anesthesia

Do you want to Perform another operation(y/n):y

1.PUSH
2.POP
3.PEEK
4.Disp
Enter your choice:4
[3, 'Murali']
[1, 'Arun']

Do you want to Perform another operation(y/n):y

1.PUSH
2.POP
3.PEEK
4.Disp
Enter your choice:3
The top of the stack is: [3, 'Murali']

Do you want to Perform another operation(y/n):y

1.PUSH
2.POP
3.PEEK
4.Disp
Enter your choice:2
The deleted doctor detail is: [3, 'Murali']

Do you want to Perform another operation(y/n):y

1.PUSH
2.POP
3.PEEK
4.Disp
Enter your choice:4
[1, 'Arun']

Do you want to Perform another operation(y/n):n

EX.NO: 15

DATE:

CREATING A PYTHON PROGRAM TO IMPLEMENT STACK OPERATIONS(Dictionary)

AIM:

To Write a program, with separate user-defined functions to perform the following operations:

- (i) To Create a function Push(Stk,D) Where Stack is an empty list and D is Dictionary of Items. from this Dictionary **Push the keys (name of the student) into a stack**, where the corresponding **value (marks) is greater than 70**.
- (ii) To Create a Function Pop(Stk) , where Stk is a Stack implemented by a list of student names. The function returns the items deleted from the stack.
- (iii) To display the elements of the stack (after performing PUSH or POP).

Source Code:

```
def Push(Stk,D):
    for i in D:
        if D[i]>70:
            Stk.append(i)

def Pop(Stk):
    if Stk==[ ]:
        return "Stack is Empty"
    else:
        print("The deleted element is:",end=' ')
        return Stk.pop()

def Disp():
    if Stk==[ ]:
        print("Stack is empty")
    else:
        top=len(Stk)-1
        for i in range(top,-1,-1):
            print(Stk[i])

ch='y'
D={}
Stk=[ ]
print("Performing Stack Operations Using Dictionary\n")
while ch=='y' or ch=='Y':
    print()
    print("1.PUSH")
    print("2.POP")
    print("3.Disp")
    opt=int(input("Enter your choice:"))
    if opt==1:
        D['Arun']=int(input("Enter the Mark of Arun:"))
        D['Anu']=int(input("Enter the Mark of Anu:"))
        D['Vishal']=int(input("Enter the Mark of Vishal:"))
        D['Priya']=int(input("Enter the Mark of Priya:"))
        D['Mano']=int(input("Enter the Mark of Mano:"))
        Push(Stk,D)
    elif opt==2:
        r=Pop(Stk)
        print(r)
    elif opt==3:
        Disp()
    opt=input("Do you want to perform another operation(y/n):")
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

Sample Output:

```
Performing Stack Operations Using Dictionary

1.PUSH
2.POP
3.Disp
Enter your choice:1
Enter the Mark of Arun:67
Enter the Mark of Anu:56
Enter the Mark of Vishal:89
Enter the Mark of Priya:45
Enter the Mark of Mano:92
Do you want to perform another operation(y/n):y

1.PUSH
2.POP
3.Disp
Enter your choice:3
Mano
Vishal
Do you want to perform another operation(y/n):y

1.PUSH
2.POP
3.Disp
Enter your choice:2

The deleted element is: Mano
Do you want to perform another operation(y/n):n
```

EX.NO: 16

DATE:

CREATING A PYTHON PROGRAM TO INTEGRATE MYSQL WITH PYTHON
(CREATING DATABASE AND TABLE)

AIM:

To write a Python Program to integrate MYSQL with Python to create Database and Table to store the details of employees.

Source Code:

```
import mysql.connector

def Create_DB():
    Con=mysql.connector.connect(host='localhost',user='root',password='root')
    try:
        if Con.is_connected():
            cur=Con.cursor()
            Q="CREATE DATABASE employees"
            cur.execute(Q)
            print("Employees database created sucessfully")
    except:
        print("Database name already exists")
        Con.close()

def Create_Table():
    Con=mysql.connector.connect(host='localhost',user='root',password='root',database='employees')
    if Con.is_connected():
        cur=Con.cursor()
        Q="CREATE TABLE EMP(ENO INT PRIMARY KEY,ENAME VARCHAR(20),GENDER VARCHAR(3),SALARY INT)"
        cur.execute(Q)
        print("Emp Table created sucessfully")
    else:
        print("Table Name already exists")
        Con.close()

ch='y'
while ch=='y' or ch=='Y':
    print("\nInterfacing Python with Mysql")
    print("1. To Create Database")
    print("2. To Create Table")
    opt=int(input("Enter your choice:"))
    if opt==1:
        Create_DB()
    elif opt==2:
        Create_Table()
    else:
        print("Invalid Choice")
    opt=input("Do you want to perform another operation(y/n):")
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

Sample Output:

Interfacing Python with Mysql

1. To Create Database

2. To Create Table

Enter your choice:1

Employees database created sucessfully

Do you want to perform another operation(y/n):y

Interfacing Python with Mysql

1. To Create Database

2. To Create Table

Enter your choice:2

Emp Table created sucessfully

Do you want to perform another operation(y/n):n

EX.NO: 17

DATE:

CREATING A PYTHON PROGRAM TO INTEGRATE MYSQL WITH PYTHON

(INSERTING RECORDS AND DISPLAYING RECORDS)

AIM:

To write a Python Program to integrate MYSQL with Python by inserting records to Emp table and display the records.

SOURCE CODE:

```
import mysql.connector
con=mysql.connector.connect(host='localhost',username='root',password='root',database='employees')
if con.is_connected():
    cur=con.cursor()
    opt='y'
    while opt=='y':
        No=int(input("Enter Employee Number:"))
        Name=input("Enter Employee Name:")
        Gender=input("Enter Employee Gender(M/F):")
        Salary=int(input("Enter Employee Salary:"))
        Query="INSERT INTO EMP VALUES({},'{}','{}',{})".format(No,Name,Gender,Salary)
        cur.execute(Query)
        con.commit()
        print("Record Stored Successfully")
        opt=input("Do you want to add another employee details(y/n):")

    Query="SELECT * FROM EMP";
    cur.execute(Query)
    data=cur.fetchall()
    for i in data:
        print(i)
    con.close()
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUTPUT:

Python Executed Program Output:

```
Enter Employee Number:1
Enter Employee Name:Arun
Enter Employee Gender(M/F):M
Enter Employee Salary:20000
Record Stored Successfully
Do you want to add another employee details(y/n):y
Enter Employee Number:2
Enter Employee Name:Bala
Enter Employee Gender(M/F):M
Enter Employee Salary:25000
Record Stored Successfully
Do you want to add another employee details(y/n):y
Enter Employee Number:3
Enter Employee Name:Bavya
Enter Employee Gender(M/F):F
Enter Employee Salary:27000
Record Stored Successfully
Do you want to add another employee details(y/n):y
Enter Employee Number:4
Enter Employee Name:Saravanan
Enter Employee Gender(M/F):M
Enter Employee Salary:29000
Record Stored Successfully
Do you want to add another employee details(y/n):n
(1, 'Arun', 'M', 20000)
(2, 'Bala', 'M', 25000)
(3, 'Bavya', 'F', 27000)
(4, 'Saravanan', 'M', 29000)
```

EX.NO: 18

DATE:

CREATING A PYTHON PROGRAM TO INTEGRATE MYSQL WITH PYTHON

(SEARCHING AND DISPLAYING RECORDS)

AIM:

To write a Python Program to integrate MYSQL with Python to search an Employee using EMPID and display the record if present in already existing table EMP, if not display the appropriate message.

SOURCE CODE:

```
import mysql.connector
con=mysql.connector.connect(host='localhost',username='root',password='root',database='employees')
if con.is_connected():
    cur=con.cursor()
    print("*****")
    print("Welcome to Employee Search Screen")
    print("*****")
    No=int(input("Enter the employee number to search:"))
    Query="SELECT * FROM EMP WHERE EMPID={}".format(No)
    cur.execute(Query)
    data=cur.fetchone()
    if data!=None:
        print(data)
    else:
        print("Record not Found!!!")
con.close()
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUTPUT:

Python Executed Program Output:

```
*****
Welcome to Employee Search Screen
*****
Enter the employee number to search:2
(2, 'Bala', 'M', 25000)
```

EX.NO: 19

DATE:

CREATING A PYTHON PROGRAM TO INTEGRATE MYSQL WITH PYTHON
(UPDATING RECORDS)

AIM:

To write a Python Program to integrate MYSQL with Python to search an Employee using EMPID and update the Salary of an employee if present in already existing table EMP, if not display the appropriate message.

SOURCE CODE:

```
import mysql.connector
con=mysql.connector.connect(host='localhost',username='root',password='root',database='employees')
if con.is_connected():
    cur=con.cursor()
    print("*****")
    print("Welcome to Employee detail update Screen")
    print("*****")
    No=int(input("Enter the employee number to Update:"))
    Query="SELECT * FROM EMP WHERE EMPID={}".format(No)
    cur.execute(Query)
    data=cur.fetchone()
    if data!=None:
        print("Record found details are:")
        print(data)
        ans=input("Do you want to update the Salary of the above employee(y/n)?:")
        if ans=='y' or ans=='Y':
            New_Sal=int(input("Enter the New Salary of an Employee:"))
            Q1="UPDATE EMP SET SALARY={} WHERE EMPID={}".format(New_Sal,No)
            cur.execute(Q1)
            con.commit()
            print("Employee Salary Updated Successfully")
            Q2="SELECT * FROM EMP"
            cur.execute(Q2)
            data=cur.fetchall()
            for i in data:
                print(i)
    else:
        print("Record not Found!!!")
```

Result:

Thus, the above Python program has been executed and the output is verified successfully.

SAMPLE OUTPUT:

Python Executed Program Output:

```
*****
Welcome to Employee detail update Screen
*****
Enter the employee number to search:3
Record found details are:
(3, 'BAVYA', 'F', 27000)
Do you want to update the Salary of the above employee(y/n)?:y
Enter the New Salary of an Employee:30000
Employee Salary Updated Successfully
(1, 'Arun', 'M', 20000)
(2, 'Bala', 'M', 25000)
(3, 'BAVYA', 'F', 30000)
(4, 'Saravanan', 'M', 29000)
```

SQL COMMANDS EXERCISE - 1

Ex.No: 20

DATE:

AIM:

To write Queries for the following Questions based on the given table:

Rollno	Name	Gender	Age	Dept	DOA	Fees
1	Arun	M	24	COMPUTER	1997-01-10	120
2	Ankit	M	21	HISTORY	1998-03-24	200
3	Anu	F	20	HINDI	1996-12-12	300
4	Bala	M	19	NULL	1999-07-01	400
5	Charan	M	18	HINDI	1997-09-05	250
6	Deepa	F	19	HISTORY	1997-06-27	300
7	Dinesh	M	22	COMPUTER	1997-02-25	210
8	Usha	F	23	NULL	1997-07-31	200

(a) Write a Query to Create a new database in the name of "**STUDENTS**".

CREATE DATABASE STUDENTS;

(b) Write a Query to Open the database "**STUDENTS**".

USE STUDENTS;

(c) Write a Query to create the above table called: "**STU**"

**CREATE TABLE STU(ROLLNO INT PRIMARY KEY,NAME VARCHAR(10),
GENDER VARCHAR(3), AGE INT,DEPT VARCHAR(15),
DOA DATE,FEES INT);**

(d) Write a Query to list all the existing database names.

SHOW DATABASES;

```
+-----+
| Database |
+-----+
| employees |
| hospital  |
| hospitals |
| information_schema |
| ip_practicals |
| mysql     |
| performance_schema |
| school    |
| students  |
| sys       |
+-----+
```

(e) Write a Query to List all the tables that exists in the current database.

SHOW TABLES;

Output:

```
+-----+
| Tables_in_students |
+-----+
| stu                 |
+-----+
```

SQL COMMANDS EXERCISE - 2

Ex.No: 21

DATE:

AIM:

To write Queries for the following Questions based on the given table:

Rollno	Name	Gender	Age	Dept	DOA	Fees
1	Arun	M	24	COMPUTER	1997-01-10	120
2	Ankit	M	21	HISTORY	1998-03-24	200
3	Anu	F	20	HINDI	1996-12-12	300
4	Bala	M	19	NULL	1999-07-01	400
5	Charan	M	18	HINDI	1997-09-05	250
6	Deepa	F	19	HISTORY	1997-06-27	300
7	Dinesh	M	22	COMPUTER	1997-02-25	210
8	Usha	F	23	NULL	1997-07-31	200

(a) Write a Query to insert all the rows of above table into Info table.

INSERT INTO STU VALUES (1,'Arun','M', 24,'COMPUTER','1997-01-10', 120);

INSERT INTO STU VALUES (2,'Ankit','M', 21,'HISTORY','1998-03-24', 200);

INSERT INTO STU VALUES (3,'Anu','F', 20,'HINDI','1996-12-12', 300);

INSERT INTO STU VALUES (4,'Bala','M', 19, NULL,'1999-07-01', 400);

INSERT INTO STU VALUES (5,'Charan','M', 18,'HINDI','1997-06-27', 250);

INSERT INTO STU VALUES (6,'Deepa','F', 19,'HISTORY','1997-06-27', 300);

INSERT INTO STU VALUES (7,'Dinesh','M', 22,'COMPUTER','1997-02-25', 210);

INSERT INTO STU VALUES (8,'Usha','F', 23, NULL,'1997-07-31', 200);

(b) Write a Query to display all the details of the Employees from the above table 'STU'.

SELECT * FROM STU;

Output:

Rollno	Name	Gender	Age	Dept	DOA	Fees
1	Arun	M	24	COMPUTER	1997-01-10	120
2	Ankit	M	21	HISTORY	1998-03-24	200
3	Anu	F	20	HINDI	1996-12-12	300
4	Bala	M	19	NULL	1999-07-01	400
5	Charan	M	18	HINDI	1997-06-27	250
6	Deepa	F	19	HISTORY	1997-06-27	300
7	Dinesh	M	22	COMPUTER	1997-02-25	210
8	Usha	F	23	NULL	1997-07-31	200

(c) Write a query to Rollno, Name and Department of the students from **STU** table.

SELECT ROLLNO,NAME,DEPT FROM STU;

Output:

ROLLNO	NAME	DEPT
1	Arun	COMPUTER
2	Ankit	HISTORY
3	Anu	HINDI
4	Bala	NULL
5	Charan	HINDI
6	Deepa	HISTORY
7	Dinesh	COMPUTER
8	Usha	NULL

(d) Write a Query to select distinct Department from **STU** table.

SELECT DISTICT(DEPT) FROM STU;

Output:

DEPT
COMPUTER
HISTORY
HINDI
NULL

(e) To show all information about students of History department.

SELECT * FROM STU WHERE DEPT='HISTORY';

Output:

Rollno	Name	Gender	Age	Dept	DOA	Fees
2	Ankit	M	21	HISTORY	1998-03-24	200
6	Deepa	F	19	HISTORY	1997-06-27	300

DATE: _____**AIM:**

To write Queries for the following Questions based on the given table:

Rollno	Name	Gender	Age	Dept	DOA	Fees
1	Arun	M	24	COMPUTER	1997-01-10	120
2	Ankit	M	21	HISTORY	1998-03-24	200
3	Anu	F	20	HINDI	1996-12-12	300
4	Bala	M	19	NULL	1999-07-01	400
5	Charan	M	18	HINDI	1997-09-05	250
6	Deepa	F	19	HISTORY	1997-06-27	300
7	Dinesh	M	22	COMPUTER	1997-02-25	210
8	Usha	F	23	NULL	1997-07-31	200

(a) Write a Query to list name of female students in Hindi Department.**SELECT NAME FROM STU WHERE DEPT='HINDI' AND GENDER='F';****Output:**

```

+-----+
|  NAME  |
+-----+
|  Anu   |
+-----+

```

(b) Write a Query to list name of the students whose ages are between 18 to 20.**SELECT NAME FROM STU WHERE AGE BETWEEN 18 AND 20;****Output:**

```

+-----+
|  NAME  |
+-----+
|  Anu   |
|  Bala  |
|  Charan|
|  Deepa |
+-----+

```

(c) Write a Query to display the name of the students whose name is starting with 'A'.

SELECT NAME FROM STU WHERE NAME LIKE 'A%';

Output:

NAME
Arun
Ankit
Anu

(d) Write a query to list the names of those students whose name have second alphabet 'n' in their names.

SELECT NAME FROM STU WHERE NAME LIKE '_N%';

Output:

NAME
Ankit
Anu

DATE:**AIM:**

To write Queries for the following Questions based on the given table:

Rollno	Name	Gender	Age	Dept	DOA	Fees
1	Arun	M	24	COMPUTER	1997-01-10	120
2	Ankit	M	21	HISTORY	1998-03-24	200
3	Anu	F	20	HINDI	1996-12-12	300
4	Bala	M	19	NULL	1999-07-01	400
5	Charan	M	18	HINDI	1997-09-05	250
6	Deepa	F	19	HISTORY	1997-06-27	300
7	Dinesh	M	22	COMPUTER	1997-02-25	210
8	Usha	F	23	NULL	1997-07-31	200

(a) Write a Query to delete the details of Roll number is 8.**DELETE FROM STU WHERE ROLLNO=8;****Output (After Deletion):**

Rollno	Name	Gender	Age	Dept	DOA	Fees
1	Arun	M	24	COMPUTER	1997-01-10	120
2	Ankit	M	21	HISTORY	1998-03-24	200
3	Anu	F	20	HINDI	1996-12-12	300
4	Bala	M	19	NULL	1999-07-01	400
5	Charan	M	18	HINDI	1997-06-27	250
6	Deepa	F	19	HISTORY	1997-06-27	300
7	Dinesh	M	22	COMPUTER	1997-02-25	210

(b) Write a Query to change the fess of Student to 170 whose Roll number is 1, if the existing fess is less than 130.**UPDATE STU SET FEES=170 WHERE ROLLNO=1 AND FEES<130;****Output(After Update):**

Rollno	Name	Gender	Age	Dept	DOA	Fees
1	Arun	M	24	COMPUTER	1997-01-10	170

(c) Write a Query to add a new column **Area** of type varchar in table STU.

ALTER TABLE STU ADD AREA VARCHAR(20);

Output:

```
Query OK, 0 rows affected
Records: 0 Duplicates: 0
```

```
mysql> SELECT * FROM STU;
```

ROLLNO	NAME	GENDER	AGE	DEPT	DOA	FEES	AREA
1	Arun	M	24	COMPUTER	1997-01-10	120	NULL
2	Ankit	M	21	HISTORY	1998-03-24	200	NULL
3	Anu	F	20	HINDI	1996-12-12	300	NULL
4	Bala	M	19	NULL	1999-07-01	400	NULL
5	Charan	M	18	HINDI	1997-09-05	250	NULL
6	Deepa	F	19	HISTORY	1997-06-27	300	NULL
7	Dinesh	M	22	COMPUTER	1997-02-25	210	NULL
8	Usha	F	23	NULL	1997-07-31	200	NULL

(d) Write a Query to Display Name of all students whose Area Contains NULL.

SELECT NAME FROM STU WHERE AREA IS NULL;

Output:

NAME
Arun
Ankit
Anu
Bala
Charan
Deepa
Dinesh
Usha

(e) Write a Query to delete Area Column from the table STU.

ALTER TABLE STU DROP AREA;

Output:

```
Query OK, 0 rows affected
Records: 0 Duplicates: 0
```

(f) Write a Query to delete table from Database.

DROP TABLE STU;

Output:

```
Query OK, 0 rows affected
```

Ex.No: 24

DATE:

SQL COMMANDS EXERCISE - 5

AIM:

To write Queries for the following Questions based on the given table:

TABLE: UNIFORM

Ucode	Uname	Ucolor	StockDate
1	Shirt	White	2021-03-31
2	Pant	Black	2020-01-01
3	Skirt	Grey	2021-02-18
4	Tie	Blue	2019-01-01
5	Socks	Blue	2019-03-19
6	Belt	Black	2017-12-09

TABLE: COST

Ucode	Size	Price	Company
1	M	500	Raymond
1	L	580	Mattex
2	XL	620	Mattex
2	M	810	Yasin
2	L	940	Raymond
3	M	770	Yasin
3	L	830	Galin
4	S	150	Mattex

(a) To Display the average price of all the Uniform of Raymond Company from table COST.

SELECT AVG(PRICE) FROM COST WHERE COMPANY='RAYMOND';

Output:

AVG(PRICE)
720.0000

(b) To display details of all the Uniform in the Uniform table in descending order of Stock date.

SELECT * FROM UNIFORM ORDER BY STOCKDATE DESC;

Output:

Ucode	Uname	Ucolor	StockDate
1	Shirt	White	2021-03-31
3	Skirt	Grey	2021-02-18
2	Pant	Black	2020-01-01
5	Socks	Blue	2019-03-19
4	Tie	Blue	2019-01-01
6	Belt	Black	2017-12-09

(c) To Display max price and min price of each company.

SELECT COMPANY,MAX(PRICE),MIN(PRICE) FROM COST GROUP BY COMPANY;

Output:

COMPANY	MAX(PRICE)	MIN(PRICE)
RAYMOND	940	500
MATTEX	620	150
YASIN	810	770
GALIN	830	830

(d) To display the company where the number of uniforms size is more than 2.

SELECT COMPANY, COUNT(*) FROM COST GROUP BY COMPANY HAVING COUNT(*)>2;

Output:

COMPANY	COUNT(*)
MATTEX	3

(e) To display the Ucode, Uname, Ucolor, Size and Company of tables uniform and cost.

SELECT U.UCODE,UNAME,UCOLOR,SIZE,COMPANY FROM UNIFORM U,COST C WHERE U.UCODE=C.UCODE;

Output:

UCODE	UNAME	UCOLOR	SIZE	COMPANY
1	Shirt	White	M	RAYMOND
1	Shirt	White	L	MATTEX
2	Pant	Black	XL	MATTEX
2	Pant	Black	M	YASIN
2	Pant	Black	L	RAYMOND
3	Skirt	Grey	M	YASIN
3	Skirt	Grey	L	GALIN
4	Tie	Blue	S	MATTEX
