

**Xavier Institute of Engineering**  
Mahim, Mumbai 400016

**Department of Information Technology**

(Affiliated to University of Mumbai)

## Certificate

This is to certify that Mr Soham Desai of Second Year Semester IV  
of Information Technology Engineering having Roll No. 11 has performed the experiments  
during the academic year 2021-22.

Prof. Jaychand Upadhyay  
Practical In-charge

Prof. Meena Ugale  
Head of the department

Examiner

Principal

Date: 11/5/22





**Xavier Institute of Engineering**

**Mahim, Mumbai 400016**

**Department of Information Technology**

(Affiliated to University of Mumbai)

**Index of Experiments for Python Lab for the Academic Year 2021-22**

Sr. No.	Title of the Experiment	Date	Remarks
1	Basics of Python:	19/01/2022	
2	Advanced Data types and functions in Python:	29/01/2022	
3	Object Oriented Programming in Python:	10/02/2022	
4	Python applications using modules, packages, multithreading and exception handling	04/03/2022	
5	File handling, GUI and database in Python:	25/03/2022	
6	Latest trends and technologies in Python:	07/04/2022	
	<b>Assignments</b>		
1	Assignment No 1	30/03/2022	
2	Fast Learner Assignment	20/04/2022	



## **Python – Experiment 1**

**AIM :** Basics of python

**LO - 1 :** Basics of python including data types, operator, conditional statements, looping statements, input and output functions in Python

A) Write a python program to swap two numbers and check if the first number is positive, negative or zero.

**Code:**

```
num1 = int(input("Enter first number: "))
num2 = int(input("Enter second number: "))
a = num1
num1 = num2
num2 = a
print("After swapping: ")
print("First number: ", num1)
print("Second number: ", num2)
```

```
if num1 < 0:
    print("First number is negative")
elif num1 > 0:
    print("First number is positive")
else:
    print("First number is zero")
```

```
if num2 < 0:
    print("Second number is negative")
elif num2 > 0:
    print("Second number is positive")
else:
    print("Second number is zero")
```

**Output:**

```
Enter first number: 34
Enter second number: 56
After swapping:
First number: 56
Second number: 34
First number is positive
Second number is positive
```

B) Write a python program to print all the numbers divisible by 4 in the range 1 to n (use for loop).

**Code:**

```
r = int(input("Enter The Range (1-n) To Get The Number's Divisible By 4 : "))
if (r < 4):
    print("No Numbers Found, Enter Range Greater Than 3 To Get Some Output Number's.")
else:
    for i in range(1, r):
        if (i % 4 == 0):
            print(i , end=" ")
```

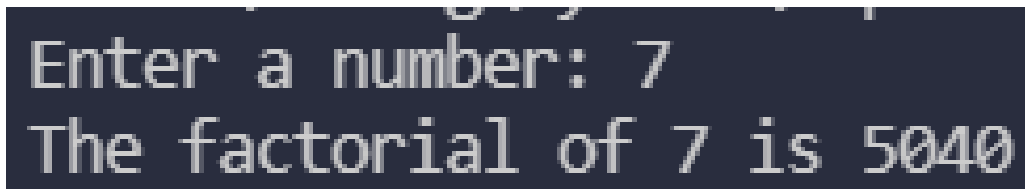
**Output:**

```
Enter The Range (1-n) To Get The Number's Divisible By 4 : 56
4 8 12 16 20 24 28 32 36 40 44 48 52
```

C) Write a python program to find the factorial of an input number (use while loop).

**Code:**

```
n = int(input("Enter a number: "))
factorial = 1
i = 1
while i <= n:
    factorial *= i
    i += 1
print("The factorial of", n, "is", factorial)
```

**Output:**

```
Enter a number: 7
The factorial of 7 is 5040
```

D) Write a menu-driven python program to build simple calculator functions.

**Code:**

```
while True:
    print("Select operation.")
    print("1. Add")
    print("2. Subtract")
    print("3. Multiply")
    print("4. Divide")
    print("5. Modulus")
    print("0. Exit")
    choice = int(input("Enter choice: "))
    if choice == 0:
        break
    num1 = float(input("Enter first number: "))
    num2 = float(input("Enter second number: "))
```

```
if choice == 1:
    print(num1, "+", num2, "=", num1 + num2)
elif choice == 2:
    print(num1, "-", num2, "=", num1 - num2)
elif choice == 3:
    print(num1, "*", num2, "=", num1 * num2)
elif choice == 4:
    print(num1, "/", num2, "=", num1 / num2)
elif choice == 5:
    print(num1, "%", num2, "=", num1 % num2)
else:
    print("Invalid input")
print("Thank you for using calculator.")
```

**Output:**

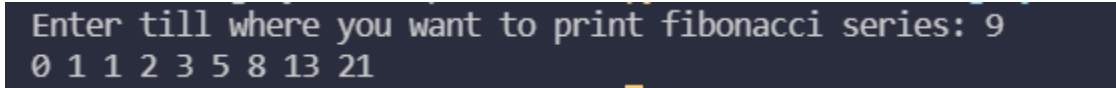
```
Select operation.
1. Add
2. Subtract
3. Multiply
4. Divide
5.Modulus
0. Exit
Enter choice: 1
Enter first number: 4
Enter second number: 6
4.0 + 6.0 = 10.0
Select operation.
1. Add
2. Subtract
3. Multiply
4. Divide
5.Modulus
0. Exit
Enter choice: 2
Enter first number: 3
Enter second number: 2
3.0 - 2.0 = 1.0
```

```
Select operation.
1. Add
2. Subtract
3. Multiply
4. Divide
5.Modulus
0. Exit
Enter choice: 3
Enter first number: 5
Enter second number: 3
5.0 * 3.0 = 15.0
Select operation.
1. Add
2. Subtract
3. Multiply
4. Divide
5.Modulus
0. Exit
Enter choice: 0
Thank you for using calculator.
```

E) Write a python program to display Fibonacci series of n number

**COde:**

```
def fibonacci(n):  
    if n == 0:  
        return 0  
    elif n == 1:  
        return 1  
    else:  
        return fibonacci(n - 1) + fibonacci(n - 2)  
  
num = int(input("Enter till where you want to print fibonacci series: "))  
for i in range(0, num):  
    print(fibonacci(i), end=" ")
```

**Output:**

```
Enter till where you want to print fibonacci series: 9  
0 1 1 2 3 5 8 13 21
```

**CONCLUSION:**

From this experiment I have understood the basics of python like data types, operator, conditional statements, looping statements, input and output functions.

## Python – Experiment 2

**AIM :** Advanced Data types and functions in Python

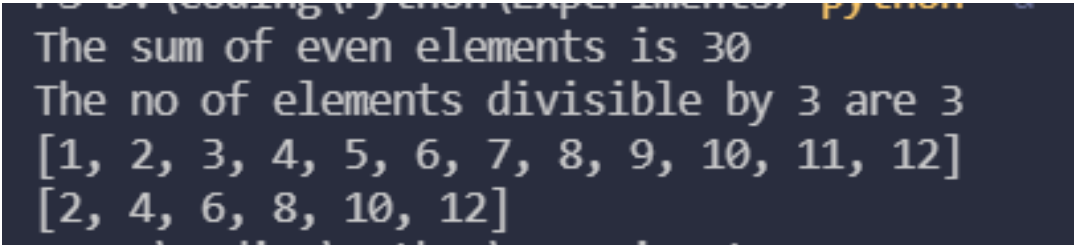
**LO - 2 :** Advanced Data types and functions in Python

A. Write a python program to perform following operations on an array: Sum of all even elements, count the no. of elements divisible by 3, insert 2 elements at the end of the list, delete all the odd elements from the list.

**Code:**

```
a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
sum = sum(i for i in a if i % 2 == 0)
print("The sum of even elements is", sum)
print("The no of elements divisible by 3 are", len([x for x in a if x % 3 == 0]))
a.append(11)
a.append(12)
print(a)
a = [x for x in a if x % 2 == 0]
print(a)
```

**Output:**



```
The sum of even elements is 30
The no of elements divisible by 3 are 3
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
[2, 4, 6, 8, 10, 12]
```

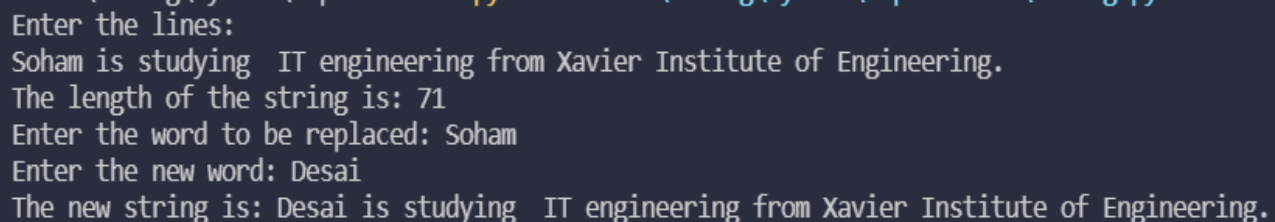


B. Write a python program to input a multiline string or a paragraph & count the no. of words & characters in string. Also check for a substring & replace each of its occurrences by some other string.

**Code:**

```
print("Enter the lines: ")
lines = ""
while not lines.endswith("."):
    lines = str(input())

print("The length of the string is:", len(lines))
rep_word = input("Enter the word to be replaced: ")
new_word = input("Enter the new word: ")
lines = lines.replace(rep_word, new_word)
print("The new string is:", lines)
```

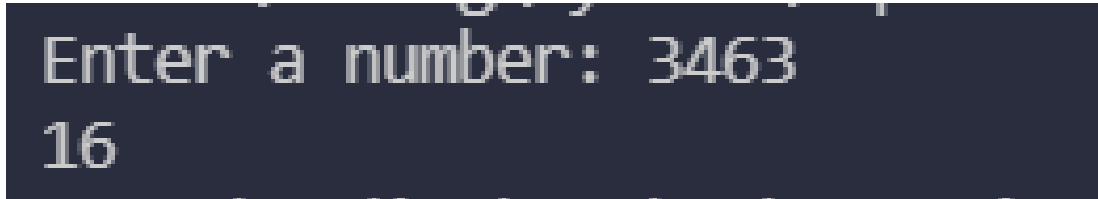
**Output:**

```
Enter the lines:
Soham is studying IT engineering from Xavier Institute of Engineering.
The length of the string is: 71
Enter the word to be replaced: Soham
Enter the new word: Desai
The new string is: Desai is studying IT engineering from Xavier Institute of Engineering.
```

C. Write a recursive function in python to find the Sum of the digits of the numbers

**Code:**

```
def sum_digits(n):
    if n < 10:
        return n
    else:
        return n % 10 + sum_digits(n // 10)
n = int(input("Enter a number: "))
print(sum_digits(n))
```

**Output:**A terminal window with a dark background and light-colored text. The first line shows the prompt 'Enter a number:' followed by the input '3463'. The second line shows the output '16'.

D. Write a menu-driven python program to :

- 1) Add students marks information in terms of tuples. Calculate the total and average marks.
- 2) Display students with specified key.
- 3) Enter students' admission date in the form (dd/mm/yyyy) to introduce nested tuple. Display students having admission in the same year.

**Code:**

```
while True:
    print("1.Calculate average and total marks")
    print("2.Display students with specific key")
    print("3.Enter admission date")
    print("0.Exit")
    choice = int(input("Enter your choice: "))
    a = (78,45,58,69,90)
    if choice == 1:
        print("Students marks: ", a)
        print("Total marks: ", sum(a))
        print("Average marks: ", sum(a)/len(a))
    elif choice == 2:
        key = int(input("Enter the key: "))
        print("The student at given key is:", a[key])
    elif choice == 3:
        student_date = ((12,2,2022),(11,2,2022),(10,2,2022),(9,2,2022),(8,2,2022))
        studentsID = 100
        toCheck = False
```

```

toGetYear = int(input("Enter year (2020 or 2021) to get student's having admission in the
year : "))
if 2020 or 2021 in toGetYear:
    for addDate in a:
        if toGetYear in addDate:
            print(f"Student {studentsID} ", end=" ")
        toCheck = True
        studentsID = studentsID + 1
        if toCheck :
            print("Having Admission In The Year",toGetYear)
    else:
        print("enter valid year!!!!(2020-2021)")
elif choice == 0:
    exit()

```

### Output:

```

1.Calculate average and total marks
2.Display students with specific key
3.Enter admission date
0.Exit
Enter your choice: 1
Students marks: (78, 45, 58, 69, 90)
Total marks: 340
Average marks: 68.0
1.Calculate average and total marks
2.Display students with specific key
3.Enter admission date
0.Exit
Enter your choice: 2
Enter the key: 4
The student at given key is: 90
1.Calculate average and total marks
2.Display students with specific key
3.Enter admission date
0.Exit

```



E. Write a menu-driven program to demonstrate the use of set in python:

- i) Accept two strings from the user.
- ii) Display common letters in two input strings (set intersection).
- iii) Display letters which are in the first string but not in the second string (set difference).
- iv) Display set of all letters from both the strings (set union)
- v) Display set of letters which are in two strings but not common (Symmetric Difference).

**Code:**

```
a = input("Enter first string:")
b = input("Enter second string:")
f = set(a)
f1 = set(b)
for i in range(len(f)):
    f.add(a[i])
for i in range(len(f1)):
    f1.add(b[i])
while True:
    print("1.Display common elements")
    print("2.Display elements in a but not in b")
    print("3.Display union of a and b")
    print("4.Display elements present in both a and b but not common")
    print("0.Exit")
    choice = int(input("Enter your choice: "))
    if choice == 1:
        print("Common elements: ", f.intersection(f1))
    elif choice == 2:
        print("Elements in a but not in b: ", f.difference(f1))
    elif choice == 3:
        print("Union of a and b: ", f.union(f1))
    elif choice == 4:
        print("Elements present in both a and b but not common: ", f.symmetric_difference(f1))
    elif choice == 0:
        exit()
    else:
        print("Invalid choice")
```

**Output:**

```
Enter first string:Soham
Enter second string:Desai
1.Display common elements
2.Display elements in a but not in b
3.Display union of a and b
4.Display elements present in both a and b but not common
0.Exit
Enter your choice: 1
Common elements: {'a'}
1.Display common elements
2.Display elements in a but not in b
3.Display union of a and b
4.Display elements present in both a and b but not common
0.Exit
Enter your choice: 2
Elements in a but not in b: {'h', 's', 'm', 'o'}
1.Display common elements
2.Display elements in a but not in b
3.Display union of a and b
4.Display elements present in both a and b but not common
0.Exit
Enter your choice: 3
Union of a and b: {'e', 's', 'o', 's', 'a', 'i', 'D', 'm', 'h'}
1.Display common elements
2.Display elements in a but not in b
3.Display union of a and b
4.Display elements present in both a and b but not common
0.Exit
Enter your choice: 4
Elements present in both a and b but not common: {'e', 's', 'o', 'h', 's', 'D', 'm', 'i'}
1.Display common elements
2.Display elements in a but not in b
3.Display union of a and b
4.Display elements present in both a and b but not common
0.Exit
Enter your choice: 0
```

F. Write a menu-driven program to demonstrate the use of dictionary in python:

- i) Create key/value pair dictionary.
- ii) Update/concatenate and delete item from existing dictionary.

Find a key and print its value.

**Code:**

```
Keys = int(input("Enter the number of keys you want to add in dictionary : "))
myDict = {}
for i in range(Keys):
    keys = input("Enter the Keys In The Dictionary : ")
    value = input("Enter the Values In The Dictionary : ")
    toAdd = f"{keys} : {value}"
    myDict[keys] = value
while True:
    print("1.Update dictionary")
    print("2.Delete any key from dictionary")
    print("3.Find any value of keys from dictionary")
    print("0.Exit")
    choice = int(input("Enter your choice :"))
    if choice==1:
        uKey = input("Enter key to update :")
        if uKey in myDict:
            uValue = input("Enter the updated value : ")
            myDict[uKey]=uValue
            print(myDict)
        else:
            print("Enter a valid input")
    elif choice==2:
        dKey = input("Enter key to delete :")
        if dKey in myDict:
            del myDict[dKey]
            print(myDict)
        else:
            print("Enter a valid input")
    elif choice==3:
```



```
fKey = input("Enter key to find :")
if fKey in myDict:
    print(myDict[fKey])
else:
    print("Enter a valid input")
elif choice==0:
    exit()
```

### Output:

```
Enter the number of keys you want to add in dictionary : 2
Enter the Keys In The Dictionary : 1
Enter the Values In The Dictionary : S
Enter the Keys In The Dictionary : 2
Enter the Values In The Dictionary : F
1.Update dictionary
2.Delete any key from dictionary
3.Find any value of keys from dictionary
0.Exit
Enter your choice :1
Enter key to update :2
Enter the updated value : VC
{'1': 'S', '2': 'VC'}
1.Update dictionary
2.Delete any key from dictionary
3.Find any value of keys from dictionary
0.Exit
Enter your choice :2
Enter key to delete :1
{'2': 'VC'}
1.Update dictionary
2.Delete any key from dictionary
3.Find any value of keys from dictionary
0.Exit
Enter your choice :3
Enter key to find :2
VC
1.Update dictionary
2.Delete any key from dictionary
3.Find any value of keys from dictionary
0.Exit
Enter your choice :0
```

**Conclusion:** From this experiment we have learned about how to deal with dictionary, tuples and many form of loops.

## Python – Experiment 3

**AIM :** Object Oriented Programming in Python

**LO - 2 :** Illustrate the concepts of object-oriented programming as used in Python

1. Design an person/employee / account class using python for reading & displaying the employee information.

**Code:**

```
class Employee:
    co = 0
    def __init__(self, name, salary,id):
        self.name = name
        self.salary = salary
        self.id = id
        Employee.co += 1

    def Display(self):
        print("Name :", self.name)
        print("Salary : ",self.salary)
        print("ID : ",self.id)
    def displayCount(self):
        print("Total Employee : ",Employee.co)

emp1 = Employee("Soham Desai",50000,1)
emp2 = Employee("Dhruv Agrawal",30000,2)
emp3 = Employee("Falguni Joshi",40000,3)

emp3.displayCount()
emp1.Display()
emp2.Display()
emp3.Display()
```

**Output:**

```
Total Employee : 3
Name : Soham Desai
Salary : 50000
ID : 1
Name : Dhruv Agrawal
Salary : 30000
ID : 2
Name : Falguni Joshi
Salary : 40000
ID : 3
```

2. Write python programs to understand

a) Classes, Objects, Constructors, Inner class and Static method

**Code:**

```
class Vehicle:
```

```
    def __init__(self, name, type):
        self.name = name
        self.kind = type
        self.car = self.Car()
        self.bike = self.Bike()
```

```
    def show(self):
        print("Name : ",self.name)
        print("Kind : ",self.kind)
```

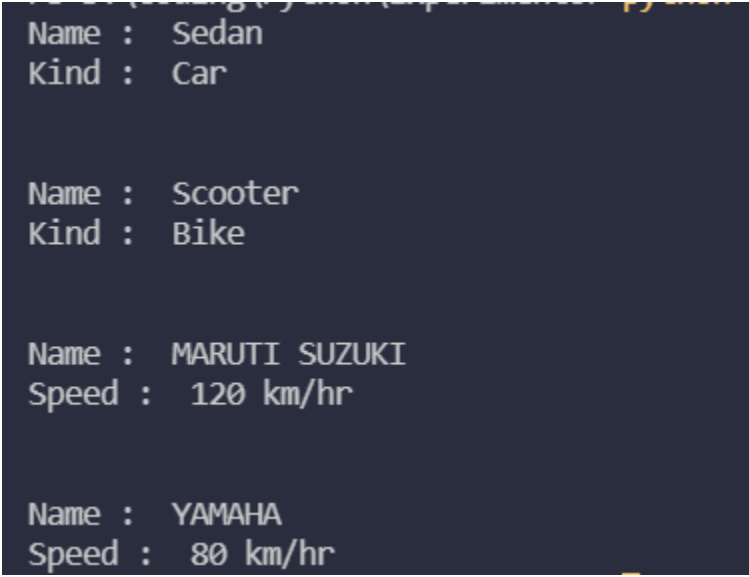
```
class Car:
```

```
    def __init__(self):
        self.name = "MARUTI SUZUKI"
        self.speed = "120 km/hr"
    def show(self):
        print("Name : ",self.name)
        print("Speed : ",self.speed)
```



```
class Bike:
    def __init__(self):
        self.name = "YAMAHA"
        self.speed = "80 km/hr"
    def show(self):
        print("Name : ",self.name)
        print("Speed : ",self.speed)
```

```
a = Vehicle("Sedan","Car")
b = Vehicle("Scooter","Bike")
a.show()
print("\n")
b.show()
print("\n")
c = b.Car()
c1 = b.Bike()
c.show()
print("\n")
c1.show()
```

**Output:**

```
Name : Sedan
Kind : Car

Name : Scooter
Kind : Bike

Name : MARUTI SUZUKI
Speed : 120 km/hr

Name : YAMAHA
Speed : 80 km/hr
```

Write a python program to understand

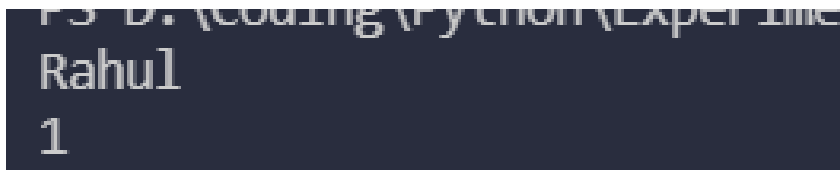
b) Different types of Inheritance

**Code:**

```
class Person():
    def __init__(self, name, idnumber):
        self.name = name
        self.idnumber = idnumber
    def display(self):
        print(self.name)
        print(self.idnumber)

class Employee(Person):
    def __init__(self, name, idnumber, salary, post):
        self.salary = salary
        self.post = post
        Person.__init__(self, name, idnumber)

a = Employee('Rahul', 1, 20000, "Intern")
a.display()
```

**Output:**

```
Rahul
1
```

c) Polymorphism using Operator overloading, Method overloading, Method overriding, Abstract class, Abstract method and Interfaces in Python.


**Code:**

```
class Parent1():
    def show(self):
        print("This is Parent1")

class Parent2():
    def display(self):
        print("This is Parent2")

class Child(Parent1, Parent2):
    def show(self):
        print("This is Child")

obj = Child()
obj.show()
obj.display()
```

**Output:**A screenshot of a terminal window with a dark background. It shows two lines of output: "This is Child" on the first line and "This is Parent2" on the second line, both in a light-colored monospace font.**Code:**

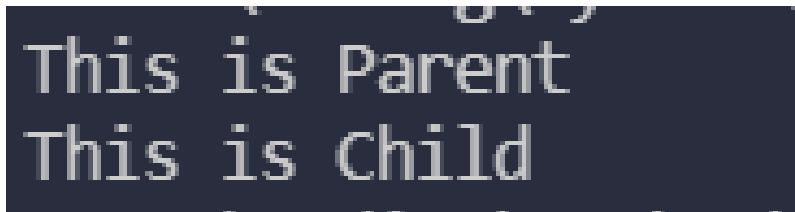
```
class Parent():
    def __init__(self):
        self.value = "This is Parent"

    def show(self):
        print(self.value)

class Child(Parent):
    def __init__(self):
        self.value = "This is Child"
```



```
def show(self):  
    print(self.value)  
obj1 = Parent()  
obj2 = Child()  
obj1.show()  
obj2.show()
```

**Output:**

```
This is Parent  
This is Child
```

**Code:**

```
from abc import ABC, abstractmethod  
class Animal(ABC):  
    @abstractmethod  
    def move(self):  
        pass  
class Human(Animal):  
    def move(self):  
        print("I can walk and run")  
class Snake(Animal):  
    def move(self):  
        print("I can crawl")  
class Dog(Animal):  
    def move(self):  
        print("I can bark")  
class Lion(Animal):  
    def move(self):  
        print("I can roar")  
c=Animal()
```

**Conclusion:** From this Experiment we have learned about classes and the implementation of the classes, method overloading and method overriding and also about abstraction.

## Python – Experiment 4

**AIM :** Python applications using modules, packages, multithreading and exception handling

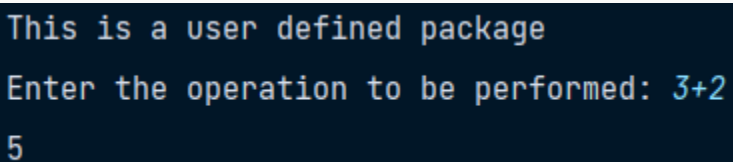
**LO - 4 :** Create Python applications using modules, packages, multithreading and exception handling.

a) Creating User-defined modules/packages and import them in a program

**Code:**

```
def display():  
    print("This is a user defined package")  
  
def calc():  
    c = eval(input("Enter the operation to be performed: "))  
    print(c)  
  
import exp  
exp.display()  
exp.calc()
```

**Output:**

A screenshot of a terminal window with a dark background. It shows the output of the Python program: "This is a user defined package", followed by a prompt "Enter the operation to be performed: " where "3+2" has been entered, and finally the result "5" is printed.

```
This is a user defined package  
Enter the operation to be performed: 3+2  
5
```

b) Creating user defined multithreaded application with thread synchronization and deadlocks

**Code:**

```
import threading  
x = 0  
def increment():  
    global x  
    x += 1
```

```
def thread_task():  
    for _ in range(1000):  
        increment()  
def main_task():  
    global x  
    x = 0  
    t1 = threading.Thread(target=thread_task)  
    t2 = threading.Thread(target=thread_task)  
    t1.start()  
    t2.start()  
    t1.join()  
    t2.join()  
if __name__ == "__main__":  
    for i in range(10):  
        main_task()  
        print("Iteration {0}: x = {1}".format(i, x))
```

**Output:**

```
Iteration 0: x = 2000  
Iteration 1: x = 2000  
Iteration 2: x = 2000  
Iteration 3: x = 2000  
Iteration 4: x = 2000  
Iteration 5: x = 2000  
Iteration 6: x = 2000  
Iteration 7: x = 2000  
Iteration 8: x = 2000  
Iteration 9: x = 2000
```

c) Creating a menu driven application which should cover all the exceptions in python.

**Code:****while True:**

```
    print("1.ZeroDivisionError\n 2.NameError\n 3.SyntaxError\n 4.IndexError\n5.KeyError\n 6.TypeError\n 7.ImportError\n 0.Exit")
    a = int(input("Enter your choice: "))
    if a == 1:
        try:
            print(1/0)
        except ZeroDivisionError as e:
            print("ZeroDivisionError:", e)
    elif a == 2:
        try:
            print(b.name)
        except NameError as e:
            print("NameError")
    elif a == 3:
        try:
            print(1+"1")
        except SyntaxError as e:
            print("SyntaxError :",e)
    elif a == 4:
        try:
            print(a[10])
        except IndexError as e:
            print("IndexError :",e)
    elif a == 5:
        try:
            print(a["name"])
        except KeyError as e:
            print("KeyError :",e)
    elif a == 6:
        try:
            print(int("a"))
        except TypeError as e:
            print("TypeError :",e)
```



```

elif a == 7:
    try:
        import soham
    except ImportError as e:
        print("ImportError :",e)
elif a == 0:
    break
else:
    print("Invalid choice")

```

### Output:

```

Enter your choice: 1
1.ZeroDivisionError
2.NameError
3.SyntaxError
4.IndexError
5.KeyError
6.TypeError
7.ImportError
0.Exit
ZeroDivisionError: division by zero
Enter your choice: 2
1.ZeroDivisionError
2.NameError
3.SyntaxError
4.IndexError
5.KeyError
6.TypeError
7.ImportError
0.Exit
Enter your choice: 3
1.ZeroDivisionError
2.NameError
3.SyntaxError
4.IndexError
5.KeyError
6.TypeError
7.ImportError
0.Exit
Traceback (most recent call last):
  File "d:\Coding\Python\Experiments\error.py", line 17, in <module>
    print(1+"1")
TypeError: unsupported operand type(s) for +: 'int' and 'str'
Enter your choice: 4
1.ZeroDivisionError
2.NameError
3.SyntaxError
4.IndexError
5.KeyError
6.TypeError
7.ImportError
0.Exit
Traceback (most recent call last):
  File "d:\Coding\Python\Experiments\error.py", line 23, in <module>
    print(a[10])
TypeError: 'int' object is not subscriptable
Enter your choice: 5
1.ZeroDivisionError
2.NameError
3.SyntaxError
4.IndexError
5.KeyError
6.TypeError
7.ImportError
0.Exit
Traceback (most recent call last):
  File "d:\Coding\Python\Experiments\error.py", line 28, in <module>
    print(a["name"])
TypeError: 'int' object is not subscriptable
Enter your choice: 6
1.ZeroDivisionError
2.NameError
3.SyntaxError
4.IndexError
5.KeyError
6.TypeError
7.ImportError
0.Exit
Traceback (most recent call last):
  File "d:\Coding\Python\Experiments\error.py", line 33, in <module>
    print(int("a"))
ValueError: invalid literal for int() with base 10: 'a'
Enter your choice: 7
1.ZeroDivisionError
2.NameError
3.SyntaxError
4.IndexError
5.KeyError
6.TypeError
7.ImportError
0.Exit
ImportError: No module named 'soham'

```

### Conclusion:

From this experiment we have learnt the multithreading and also how to import user defined packages.

## Python – Experiment 5

**AIM :** File handling, GUI and database in Python

**LO - 5 :** Gain proficiency in writing File Handling programs ,also create GUI applications and evaluate database operations in python.

a) Different File Handling operations in Python

### Code:

```
file = open("soham.txt","w")
file.write("My name is soham\n")
file.write("This is a sample written text")
file.close()
```

```
f = open("soham.txt")
c = f.read(4)
print(c)
print(f.readline(),end="")
print(f.tell())
print(f.seek(0))
print(f.readline(),end="")
f.close()
```

```
with open("soham.txt") as f:
    print(f.readline(),end="")
    print(f.tell())
    print(f.seek(0))
    print(f.readline(),end="")
```

### Output:

```
My n
ame is soham
18
0
My name is soham
My name is soham
18
0
My name is soham
```

```
My name is soham
This is a sample written text
```

b) Designing Graphical user interface (GUI) using built-in tools PyQt and GUI database connectivity to perform CRUD operations in python.

**Code:**

```
import sys
from PyQt5.QtWidgets import *
dic = {}

def main():
    app = QApplication(sys.argv)
    w = QWidget()
    w.resize(500,500)

    layout = QGridLayout()
    productLabel = QLabel("Item name:")
    costLabel = QLabel("Cost:")
    prodLine = QLineEdit()
    costLine = QLineEdit()
    layout.addWidget(productLabel,1,1)
    layout.addWidget(prodLine,1,2)
    layout.addWidget(costLabel,2,1)
    layout.addWidget(costLine,2,2)

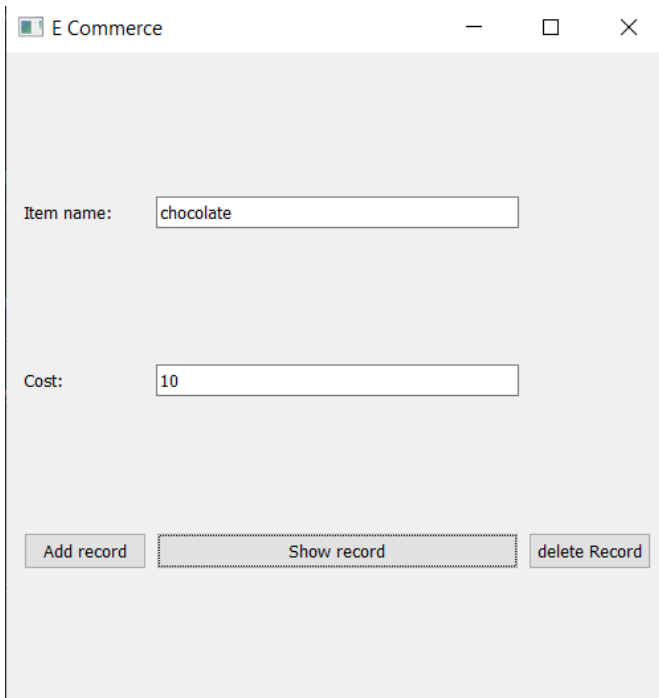
    def add():
        p =prodLine.text()
        c = costLine.text()
        dic[p] = float(c)
        print("item Added")

    def delete():
        if dic == {}:
            print("No record to delete")
        else:
            dic.popitem()
            print("record deleted")

    button = QPushButton("Add record")
    button.clicked.connect(add)
```

```
layout.addWidget(button,3,1)
button1 = QPushButton("Show record")
button1.clicked.connect(lambda:print(dic))
layout.addWidget(button1,3,2)
button2 = QPushButton("delete Record")
button2.clicked.connect(delete)
layout.addWidget(button2,3,3)
w.setLayout(layout)
w.setWindowTitle("E Commerce")
w.show()
sys.exit(app.exec_())
if __name__ == "__main__":
    main()
```

### Output:



```
item Added
item Added
{'bread': 20.0, 'chocolate': 10.0}
record deleted
{'bread': 20.0}
```

**Conclusion:** From this experiment, it is concluded that we have understood the concept of File handling, GUI and database in Python

## Python – Experiment 6

**AIM :** Latest trends and technologies in Python

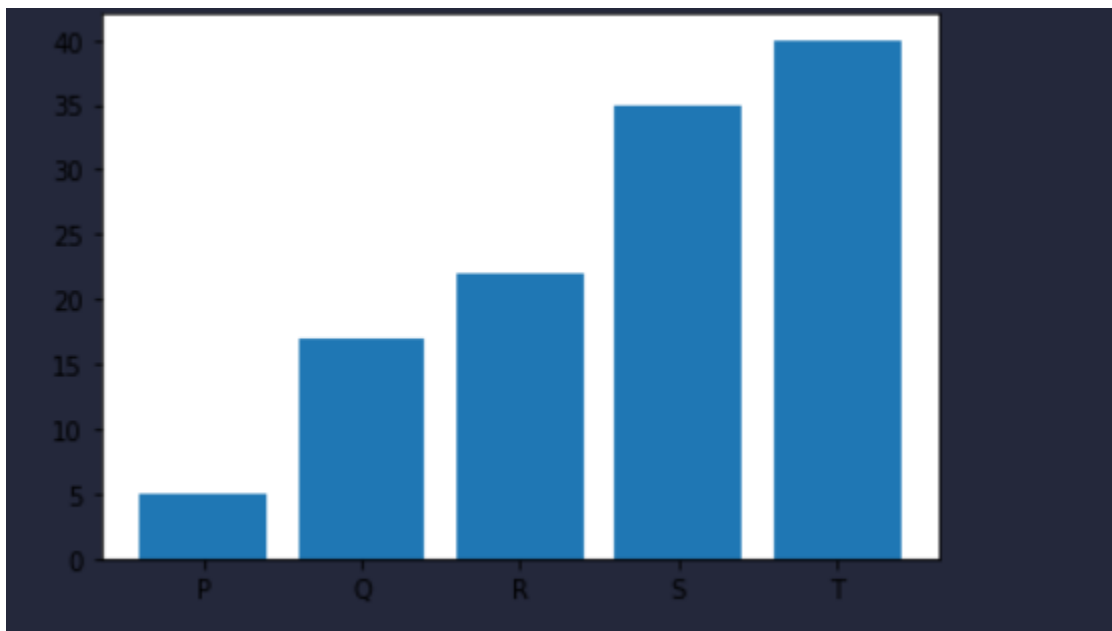
**LO - 5 :** Design and Develop cost-effective robust applications using the latest Python trends and technologies

a) Different types of plots using Numpy and Matplotlib

### Code:

```
import matplotlib.pyplot as plt
import numpy as np
a = np.array(["P", "Q", "R", "S", "T"])
b = np.array([5, 17, 22, 35, 40])
plt.bar(a, b)
plt.show()
```

### Output:

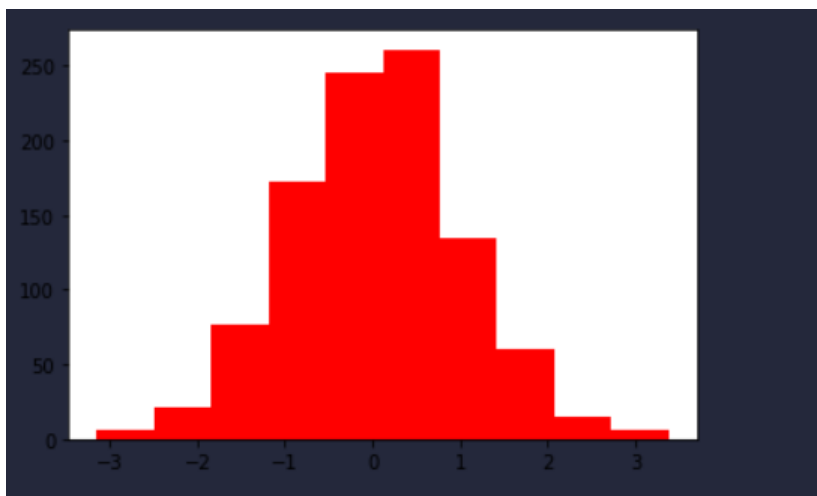




**Code:**

```
import matplotlib.pyplot as plt
import numpy as np
a = np.random.normal(0, 1, 1000)
plt.hist(a,color='red',bins=10)
plt.show()
```

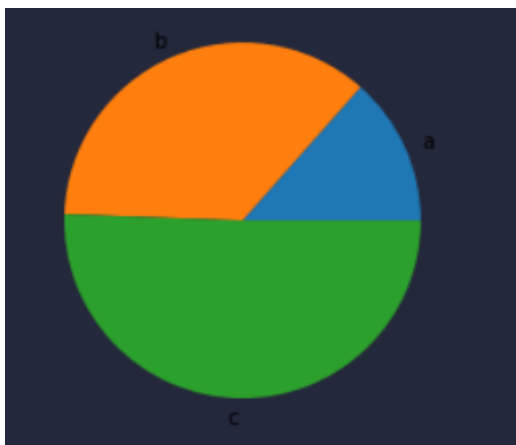
**Output:**



**Code:**

```
import matplotlib.pyplot as plt
import numpy as np
b = np.array([12,32,45])
plt.pie(b,labels=['a','b','c'])
plt.show()
```

**Output:**



b) Basic operations using pandas like series, data frames, indexing, filtering, combining and merging data frames

**Code:**

```
import pandas as pd
food = {1:'pizza',2:'burger',3:'chicken'}
a = pd.Series(food)
print(a)
```

**Output:**

```
1    pizza
2    burger
3    chicken
dtype: object
```

**Code:**

```
import pandas as pd
data = {"Name": ["John", "Anna", "Peter", "Linda"],
        "Location": ["New York", "Paris", "Berlin", "London"]}
data_pandas = pd.DataFrame(data)
print(data_pandas)
```

**Output:**

```
   Name  Location
0  John  New York
1  Anna    Paris
2 Peter    Berlin
3 Linda    London
```

**Code:**

```
import pandas as pd
data = {"Name": ["John", "Anna", "Peter", "Linda"], "Amount": [14, 29, 5, 10]}
data_pandas = pd.DataFrame(data)
print(data_pandas[data_pandas['Amount']>10])
```

**Output:**

	Name	Amount
0	John	14
1	Anna	29

**Code:**

```
import pandas as pd
df1= pd.DataFrame({'X': [10, 0], 'Y': [5, 5]})
df2= pd.DataFrame({'X': [2, 2], 'Y': [4, 4]})
take_smaller = lambda s1, s2: s1 if s1.sum() < s2.sum() else s2
df1.combine(df2, take_smaller)
print(df1)
```

**Output:**

	X	Y
0	10	5
1	0	5

**Conclusion:**

From this experiment, it is concluded that we have understood the concept of different plots using numpy and matplotlib and basic operation of pandas

## Department of Information Technology

**Sem: IV**

**Python Lab**

**2021-22**

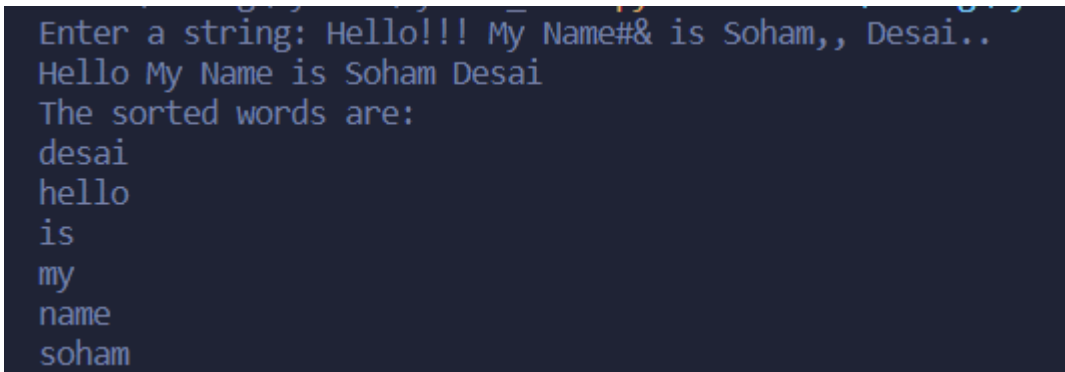
### Assignment

Q.1) Write a python program to input a string, remove punctuation from a string and then sort words in alphabetic order. (LO1)

**Code:**

```
punctuations = ""!()-[]{};:'"\,<>./?@#$%^&*~_""
my_str = input("Enter a string: ")
no_punct = ""
for char in my_str:
    if char not in punctuations:
        no_punct = no_punct + char
print(no_punct)
words = [word.lower() for word in no_punct.split()]
words.sort()
print("The sorted words are:")
for word in words:
    print(word)
```

**Output:**



```
Enter a string: Hello!!! My Name#& is Soham,, Desai..
Hello My Name is Soham Desai
The sorted words are:
desai
hello
is
my
name
soham
```

Q.2) Write the following programs (LO2)

i) Write a python program to count tuples occurrences in given list of tuples and then remove duplicate tuples from list of tuples

**Code:**

```
import collections

x = [(('Mon', 'Wed')), (('Mon')), (('Tue')), (('Mon', 'Wed')) ]

a = collections.defaultdict(int)

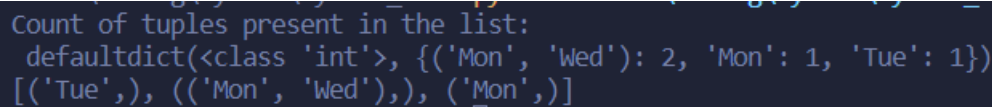
for elem in x:
    a[elem[0]] += 1

print("Count of tuples present in the list:\n",a)

def removeDuplicats(Tuple):
    return [t for t in (set(tuple(i) for i in Tuple))]

print(removeDuplicats(x))
```

**Output:**



```
Count of tuples present in the list:
defaultdict(<class 'int'>, {('Mon', 'Wed'): 2, 'Mon': 1, 'Tue': 1})
[('Tue',), (('Mon', 'Wed',), ('Mon',))]
```

ii) Write a python program to create a sub-dictionary containing all keys from dictionary list

**Code:**

```
from itertools import chain

a = [{'soham': 3, 'is': 7},
      {'soham': 3, 'is': 1, 'best': 5},
      {'soham': 8}]

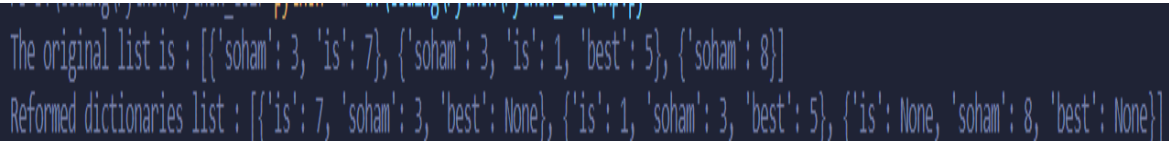
print("The original list is : " + str(a))

all_keys = set(chain.from_iterable(a))

res = [dict((key, sub.get(key, None)) for key in all_keys) for sub in a]

print("Reformed dictionaries list : " + str(res))
```

**Output:**



```
The original list is : [{'soham': 3, 'is': 7}, {'soham': 3, 'is': 1, 'best': 5}, {'soham': 8}]
Reformed dictionaries list : [{'is': 7, 'soham': 3, 'best': None}, {'is': 1, 'soham': 3, 'best': 5}, {'is': None, 'soham': 8, 'best': None}]
```

Q.3) Create a Vehicle class with max\_speed and mileage instance attributes. Create a Bus and Taxi classes that inherit the Vehicle class. Give the capacity argument of Bus. The seating\_capacity() for bus and Taxi a default value of 50 and 3 respectively. The default fare charge of any vehicle is seating capacity \* 100 per 5km. If Vehicle is Bus instance, we need to add an extra 10% on full fare as a maintenance charge. So total fare for bus instance will become the final amount = total fare + 10% of the total fare. Calculate total fare charges spent by group for picnic if both taxi and bus is used for travelling 100km distance one way. (LO3)

**Code:**

```
class Vehicle:
```

```
    def __init__(self, name, mileage, capacity):
```

```
        self.name = name
```

```
        self.mileage = mileage
```

```
        self.capacity = capacity
```

```
    def show(self):
```

```
        print("Name:", self.name, "\nMileage:", self.mileage, "\nCapacity:", self.capacity)
```

```
class Bus(Vehicle):
```

```
    def fare(self, distance):
```

```
        def_fare = 0
```

```
        def_fare = self.capacity * 20 * distance
```

```
        print("Fare:", def_fare)
```

```
        total_bus_fare = def_fare + 0.1 * def_fare
```

```
        print("Total fare:", total_bus_fare)
```

```
class Taxi(Vehicle):
```

```
    def fare(self, distance):
```

```
        def_fare = 0
```

```
        def_fare = self.capacity * 10 * distance
```

```
        print("Fare:", def_fare)
```

```
        total_bus_fare = def_fare + 0.05 * def_fare
```

```
        print("Total fare:", total_bus_fare)
```

```
School_bus = Bus("School Volvo", 12, 50)
```

```
School_bus.show()
```



```
School_bus.fare(100)
taxi = Taxi("Taxi", 10, 10)
taxi.show()
taxi.fare(100)
```

**Output:**

```
Name: School Volvo
Mileage: 12
Capacity: 50
Fare: 100000
Total fare: 110000.0
Name: Taxi
Mileage: 10
Capacity: 10
Fare: 10000
Total fare: 10500.0
```

Q.4) Create module for performing mathematical function and import it to calculate Euclidean distance. Show exception handling to handle the runtime mistake done by user. (LO4)

**Code:**

```
import math
class MyMathLibrary:
    @staticmethod
    def calculateEuclideanDistance(x1, x2, y1, y2):
        xMinus = x2 - x1
        yMinus = y2 - y1
        internalCalc = xMinus**2 + yMinus**2
        euclidDistance = math.sqrt(internalCalc)
        return euclidDistance
```

import mathmodule

try:

```
x1 = float(input("enter the x1 co-ordinate value : "))
x2 = float(input("enter the x2 co-ordinate value : "))
y1 = float(input("enter the y1 co-ordinate value : "))
y2 = float(input("enter the y1 co-ordinate value : "))

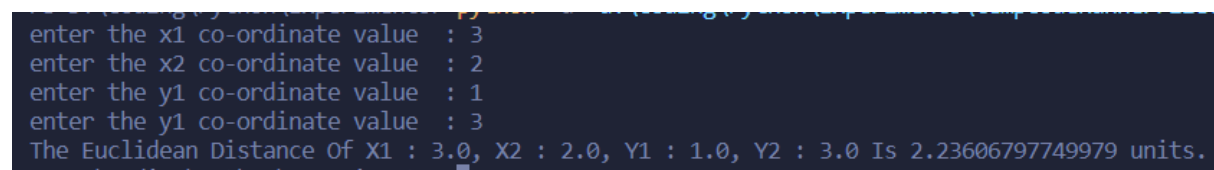
myResult = mathmodule.MyMathLibrary.calculateEuclideanDistance(x1, x2, y1, y2)

print(f"The Euclidean Distance Of X1 : {x1}, X2 : {x2}, Y1 : {y1}, Y2 : {y2} Is {myResult} units.")
```

except:

```
print("kindly enter valid co-ordinates value.")
```

**Output:**



```
enter the x1 co-ordinate value : 3
enter the x2 co-ordinate value : 2
enter the y1 co-ordinate value : 1
enter the y1 co-ordinate value : 3
The Euclidean Distance Of X1 : 3.0, X2 : 2.0, Y1 : 1.0, Y2 : 3.0 Is 2.23606797749979 units.
```

Q.5) Develop GUI Application for E-commerce application use (LO5)

1) file, pickle, dictionary to show add, delete, update operations.

**Code:**

```
import sys, pickle
```

```
from PyQt5.QtWidgets import QApplication, QWidget, QGridLayout, QLabel, QLineEdit,
QPushButton, QTableWidgetItem, QMessageBox
```

```
class dictionary(dict):
```

```
    def init(self):
```

```
        self = dict()
```

```
    def add(self, key, value):
```

```
        self[key] = value
```

```
    def delete(self, key):
```

```
        self.pop(key)
```

```
    def main(self):
```

```
        s = dictionary()
```

```
        def add():
```

```
        name = nameLine.text()
        price = priceLine.text()
        s.add(name, price)
        print(s)
        row = table.rowCount()
        table.setRowCount(row + 1)
        namecell = QTableWidgetItem(name)
        pricecell = QTableWidgetItem(price)
        table.setItem(row, 0, namecell)
        table.setItem(row, 1, pricecell)
    def delete(self):
        selected = table.selectedItems()
        name = selected[0].text()
        selectedIndex = table.selectedIndexes()
        rowNo = selectedIndex[0].row()
        table.removeRow(rowNo)
        s.delete(name)
        print(s)
    def save(self):
        file = open("shopping.pickle", "wb")
        pickle.dump(s, file)
        file.close()
        print("Data saved!")
    def upload(self):
        file = open("shopping.pickle", "rb")
        temp = pickle.load(file)
        for name, price in temp.items():
            print(name, price)
            s.add(name, price)
        row = table.rowCount()
```

```
        table.setRowCount(row + 1)

        namecell = QTableWidgetItem(name)
        pricecell = QTableWidgetItem(price)
        table.setItem(row, 0, namecell)
        table.setItem(row, 1, pricecell)
        file.close()

def bill():
    file = open("shopping.pickle", "rb")
    temp = pickle.load(file)
    sum = 0
    for Name, price in temp.items():
        print(Name, price)
        sum += float(price)
    msg = QMessageBox()
    msg.setWindowTitle("Cash Invoice")
    msg.setText("Your bill amount is " + str(sum))
    x = msg.exec_()
    file.close()

app = QApplication(sys.argv)
w = QWidget()
layout = QGridLayout()
nameLabel = QLabel()
nameLabel.setText("Name of Product :")
nameLine = QLineEdit()
priceLabel = QLabel()
priceLabel.setText("Price :")
priceLine = QLineEdit()
emptyLabel = QLabel()
emptyLabel.setText("****Product Names And Prices****")
addButton = QPushButton()
```

```
addButton.setText("Add Record")

delButton = QPushButton()
delButton.setText("Delete Record")

saveButton = QPushButton()
saveButton.setText("Save Record")

uploadButton = QPushButton()
uploadButton.setText("Upload Record")

billButton = QPushButton()
billButton.setText("View bill")
billButton.clicked.connect()

table = QTableWidgetItem()
table.setColumnCount(2)
table.setHorizontalHeaderLabels(["Name of Product", "Price"])
table.resizeColumnToContents(0)
table.resizeColumnToContents(1)
table.setWordWrap(True)

addButton.clicked.connect(add)
delButton.clicked.connect(delete)
saveButton.clicked.connect(save)
uploadButton.clicked.connect(upload)

w.resize(550, 350)
w.setWindowTitle("Shopping List")

layout.addWidget(nameLabel, 1, 1)
layout.addWidget(nameLine, 1, 2)
layout.addWidget(priceLabel, 2, 1)
layout.addWidget(priceLine, 2, 2)
layout.addWidget(addButton, 3, 1)
layout.addWidget(delButton, 3, 2)
layout.addWidget(saveButton, 3, 3)
layout.addWidget(uploadButton, 3, 4)
```

```
        layout.addWidget(billButton, 3, 5)
        layout.addWidget(emptyLabel, 4, 2)
        layout.addWidget(table, 5, 2)
        w.setLayout(layout)
        w.show()
        sys.exit(app.exec_())
if __name__ == " main ":
    d = dictionary()
    d.main()
```

2) sqlite3 dictionary to show add, delete, update operations.

**Code:**

```
from tkinter import *
import sqlite3
top = Tk()
top.geometry("750x700")
conn = sqlite3.connect('products.db')
print("Database established succesfully!")
cur = conn.cursor()
cur.execute( """CREATE TABLE IF NOT EXISTS PRODUCTS(NAME TEXT,PRICE
TEXT,QUANTITY TEXT)""" ) # write SQL queries in ()
print("Products table created
succesfully")
conn.commit()
conn.close()
s1 = e1.get()
s2 = e2.get()
s3 = e3.get()
print(s1, s2, s3)
conn = sqlite3.connect('products.db')
print("Attempting to open the database")
cur = conn.cursor()
```



```
val = (s1, s2, s3)

cur.execute("INSERT INTO PRODUCTS(NAME,PRICE,QUANTITY) VALUES (?,?,?)",
val)

conn.commit()

print("Values fed into database: ", val)

l_add = Label(top, text="Record inserted successfully")

l_add.place(x=200, y=300)

conn.close()

def view():

    conn = sqlite3.connect('products.db')

    print("Attempting to open the database")

    cur = conn.cursor()

    cur.execute("SELECT * FROM PRODUCTS")

    records = cur.fetchall() # record = str(records)

    print("Records in the database are: ", str(records))

    l_view = Label(top, text="Records in the Table products are: ")

    l_view.place(x=200, y=380)

    T = Text(top, height=10, width=50)

    T.place(x=200, y=440)

    T.insert(INSERT, str(records))

    conn.commit()

    conn.close()

def delete():

    conn = sqlite3.connect('products.db')

    print("Attempting to open the database")

    cur = conn.cursor()

    n1 = e4.get()

    print(n1)

    cur.execute("DELETE FROM PRODUCTS WHERE NAME = ?", (n1,))

    l_del = Label(top, text="Record deleted")

    l_del.place(x=450, y=150)
```

```
conn.commit()

conn.close()

l_main = Label(top, text="Available Products", font=("Helvetica 25 bold "), fg="violet",
bg="orange")

l_main.place(x=20, y=20)

l_name = Label(top, text="Name of product", font=('Helvetica 12 bold'))

l_name.place(x=20, y=80)

e1 = Entry(top)

e1.place(x=200, y=80)

l_price = Label(top, text="Price", font=('Helvetica 12 bold'))

l_price.place(x=20, y=120)

e2 = Entry(top)

e2.place(x=200, y=120)

l_quantity = Label(top, text="Quantity", font=('Helvetica 12 bold'))

l_quantity.place(x=20, y=160)

e3 = Entry(top)

e3.place(x=200, y=160)

l_n = Label(top, text="Product Name to be deleted", font=('Helvetica 12 bold'))

l_n.place(x=350, y=80)

e4 = Entry(top)

e4.place(x=600, y=80)

b1 = Button(top, text="ADD", font=('Helvetica 12 bold'), command=insert)

b1.place(x=20, y=240)

b2 = Button(top, text="VIEW", font=('Helvetica 12 bold'), command=view)

b2.place(x=100, y=240)

top.mainloop()
```

## Output:

tk

### E-commerce application using TKINTER

Product Name

Amount

record inserted successfully

records in db are

```
[('Book', '20'), ('Book', '50'), ('pen', '20')]
```

tk

### E-commerce application using TKINTER

Product Name

Amount

record deleted

record inserted successfully

records in db are

```
[('pen', '20'), ('pen', '20')]
```

Q.6) Prepare a graph showing attendance analysis of SE IT students (attendance sheet is uploaded on Google classroom) (LO6)

**Code:**

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

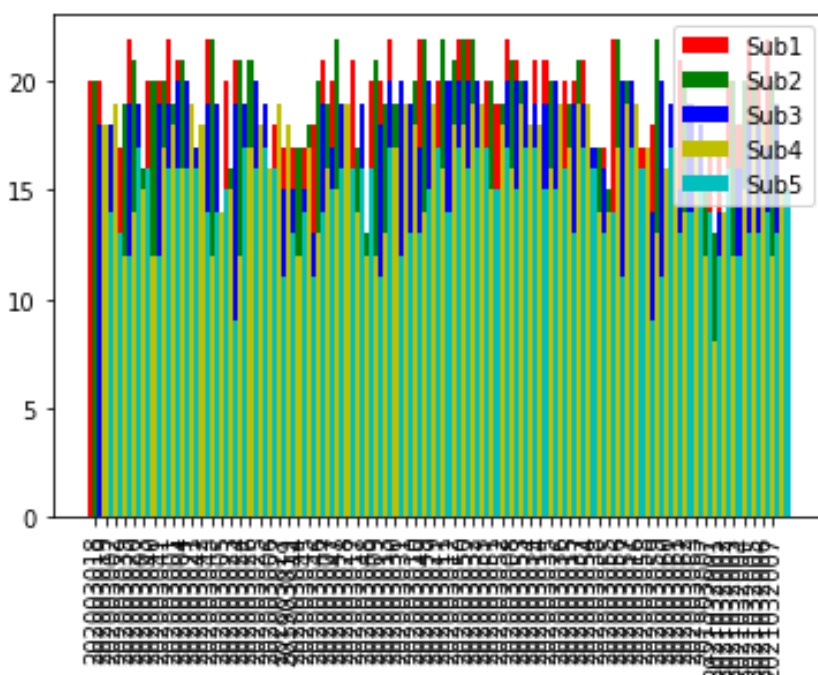
df = pd.read_csv('atten.csv')
rollNo = df['Roll_no'].values
a = np.arange(len(rollNo))
w = 0.5

plt.bar(a, df['Sub1'].values, width=w, color='r',label='Sub1')
plt.bar(a+w, df['Sub2'].values, width=w, color='g',label='Sub2')
plt.bar(a+2*w, df['Sub3'].values, width=w, color='b',label='Sub3')
plt.bar(a+3*w, df['Sub4'].values, width=w, color='y',label='Sub4')
plt.bar(a+4*w, df['Sub5'].values, width=w, color='c',label='Sub5')

plt.xticks(a+w, rollNo)

plt.legend()

plt.show()
```



## Department of Information Technology

**Sem: IV**

**Python Lab**

**2021-22**

### Fast Learner Assignment

**Q. Using Titanic dataset (dataset can be downloaded from ERP) perform the following task of**

a) finding out the information about how many male survived who had cabin and age is less than 50.

**Code:**

```
import mysql.connector as sql
myDatabase = sql.connect(host="localhost", user="root", passwd="", database="soham123")
myQuery= myDatabase.cursor()
query = "select * from fastlearner where Sex='male' and Cabin!='' and Age<50 and survived=1;"
myQuery.execute(query)
myResult= (myQuery.fetchall)
print(" - "*20)
print(f"the information about how many male survived who had cabin and age is less than 50:")
print(" - "*20)
for x in myResult:
    print(x)
print(" - "*20)
```

## Output:

```
- - - - -
the information about how many male survived who had cabin and age is less than 50 :
- - - - -
(22, 1, 2, 'Beesley, Mr. Lawrence', 'male', 34, 0, 0, '248698', 13.0, 'D56', 'S')
- - - - -
(24, 1, 1, 'Sloper, Mr. William Thompson', 'male', 28, 0, 0, '113788', 35.5, 'A6', 'S')
- - - - -
(56, 1, 1, 'Woolner, Mr. Hugh', 'male', 0, 0, 0, '19947', 35.5, 'C52', 'S')
- - - - -
(98, 1, 1, 'Greenfield, Mr. William Bertram', 'male', 23, 0, 1, 'PC 17759', 63.3583, 'D10 D12', 'C')
- - - - -
(184, 1, 2, 'Becker, Master. Richard F', 'male', 1, 2, 1, '230136', 39.0, 'F4', 'S')
- - - - -
(194, 1, 2, 'Navratil, Master. Michel M', 'male', 3, 1, 1, '230080', 26.0, 'F2', 'S')
- - - - -
(210, 1, 1, 'Blank, Mr. Henry', 'male', 40, 0, 0, '112277', 31.0, 'A31', 'C')
- - - - -
(225, 1, 1, 'Hoyt, Mr. Frederick Maxfield', 'male', 38, 1, 0, '19943', 90.0, 'C93', 'S')
- - - - -
(249, 1, 1, 'Beckwith, Mr. Richard Leonard', 'male', 37, 1, 1, '11751', 52.5542, 'D35', 'S')
- - - - -
(299, 1, 1, 'Saalfeld, Mr. Adolphe', 'male', 0, 0, 0, '19988', 30.5, 'C106', 'S')
- - - - -
(306, 1, 1, 'Allison, Master. Hudson Trevor', 'male', 1, 1, 2, '113781', 151.55, 'C22 C26', 'S')
```

```
- - - - -
(306, 1, 1, 'Allison, Master. Hudson Trevor', 'male', 1, 1, 2, '113781', 151.55, 'C22 C26', 'S')
- - - - -
(341, 1, 2, 'Navratil, Master. Edmond Roger', 'male', 2, 1, 1, '230080', 26.0, 'F2', 'S')
- - - - -
(371, 1, 1, 'Harder, Mr. George Achilles', 'male', 25, 1, 0, '11765', 55.4417, 'E50', 'C')
- - - - -
(391, 1, 1, 'Carter, Mr. William Ernest', 'male', 36, 1, 2, '113760', 120.0, 'B96 B98', 'S')
- - - - -
(430, 1, 3, 'Pickard, Mr. Berk (Berk Trembisky)', 'male', 32, 0, 0, 'SOTON/O.Q. 392078', 8.05, 'E10', 'S')
- - - - -
(431, 1, 1, 'Bjornstrom-Steffansson, Mr. Mauritz Hakan', 'male', 28, 0, 0, '110564', 26.55, 'C52', 'S')
- - - - -
(446, 1, 1, 'Dodge, Master. Washington', 'male', 4, 0, 2, '33638', 81.8583, 'A34', 'S')
- - - - -
(454, 1, 1, 'Goldenberg, Mr. Samuel L', 'male', 49, 1, 0, '17453', 89.1042, 'C92', 'C')
- - - - -
(461, 1, 1, 'Anderson, Mr. Harry', 'male', 48, 0, 0, '19952', 26.55, 'E12', 'S')
- - - - -
(485, 1, 1, 'Bishop, Mr. Dickinson H', 'male', 25, 1, 0, '11967', 91.0792, 'B49', 'C')
- - - - -
(513, 1, 1, 'McGough, Mr. James Robert', 'male', 36, 0, 0, 'PC 17473', 26.2875, 'E25', 'S')
- - - - -
(551, 1, 1, 'Thayer, Mr. John Borland Jr', 'male', 17, 0, 2, '17421', 110.883, 'C70', 'C')
- - - - -
(573, 1, 1, 'Flynn, Mr. John Irwin ("Irving")', 'male', 36, 0, 0, 'PC 17474', 26.3875, 'E25', 'S')
```

500

100 2000 2000

```
- - - - -  
(600, 1, 1, 'Duff Gordon, Sir. Cosmo Edmund ("Mr Morgan")', 'male', 49, 1, 0, 'PC 17485', 56.9292, 'A20', 'C')  
- - - - -  
(622, 1, 1, 'Kimball, Mr. Edwin Nelson Jr', 'male', 42, 1, 0, '11753', 52.5542, 'D19', 'S')  
- - - - -  
(633, 1, 1, 'Stahelin-Maeglin, Dr. Max', 'male', 32, 0, 0, '13214', 30.5, 'B50', 'C')  
- - - - -  
(646, 1, 1, 'Harper, Mr. Henry Sleeper', 'male', 48, 1, 0, 'PC 17572', 76.7292, 'D33', 'C')  
- - - - -  
(680, 1, 1, 'Cardeza, Mr. Thomas Drake Martinez', 'male', 36, 0, 1, 'PC 17755', 512.329, 'B51 B53 B55', 'C')  
- - - - -  
(682, 1, 1, 'Hassab, Mr. Hammad', 'male', 27, 0, 0, 'PC 17572', 76.7292, 'D49', 'C')  
- - - - -  
(691, 1, 1, 'Dick, Mr. Albert Adrian', 'male', 31, 1, 0, '17474', 57.0, 'B20', 'S')  
- - - - -  
(702, 1, 1, 'Silverthorne, Mr. Spencer Victor', 'male', 35, 0, 0, 'PC 17475', 26.2875, 'E24', 'S')  
- - - - -  
(708, 1, 1, 'Calderhead, Mr. Edward Pennington', 'male', 42, 0, 0, 'PC 17476', 26.2875, 'E24', 'S')  
- - - - -  
(713, 1, 1, 'Taylor, Mr. Elmer Zebley', 'male', 48, 1, 0, '19996', 52.0, 'C126', 'S')  
- - - - -  
(725, 1, 1, 'Chambers, Mr. Norman Campbell', 'male', 27, 1, 0, '113806', 53.1, 'E8', 'S')  
- - - - -  
(738, 1, 1, 'Lesurer, Mr. Gustave J', 'male', 35, 0, 0, 'PC 17755', 512.329, 'B101', 'C')  
- - - - -
```

```
- - - - -  
(741, 1, 1, 'Hawksford, Mr. Walter James', 'male', 0, 0, 0, '16988', 30.0, 'D45', 'S')  
- - - - -  
(752, 1, 3, 'Moor, Master. Meier', 'male', 6, 0, 1, '392096', 12.475, 'E121', 'S')  
- - - - -  
(803, 1, 1, 'Carter, Master. William Thornton II', 'male', 11, 1, 2, '113760', 120.0, 'B96 B98', 'S')  
- - - - -  
(840, 1, 1, 'Marechal, Mr. Pierre', 'male', 0, 0, 0, '11774', 29.7, 'C47', 'C')  
- - - - -  
(890, 1, 1, 'Behr, Mr. Karl Howell', 'male', 26, 0, 0, '111369', 30.0, 'C148', 'C')  
- - - - -
```

Process finished with exit code 0



b) show graphical representation of male and female survived and dead in the tragedy.

**Code:**

```
import mysql.connector as sql
import matplotlib.pyplot as plt
import numpy as np

maleSurvived = 0
maleDeath = 0
femaleSurvived = 0
femaleDeath = 0

myDatabase = sql.connect(host="localhost", user="root", passwd="", database="soham123")
myQuery= (myDatabase.cursor)
query1="select * from fastlearner where Sex='male' and survived=1"
myQuery.execute(query 1)
myResult1= (myQuery.fetchall)
for x in myResult:
    maleSurvived = maleSurvived+ 1
    print(maleSurvived)
query2="select * from fastlearner where Sex='male' and survived=0"
myQuery.execute(query2)
myResult2= (myQuery.fetchall)
for x in myResult2:
    maleDeath = maleDeath+1
    print(maleDeath)
query3="select * from fastlearner where Sex='female' and survived=1"
myQuery.execute(query3)
myResult3 = (myQuery.fetchall)
for x in myResult3:
    femaleSurvived = femaleSurvived+1
    print(femaleSurvived)
```

```
query4="select * from fastlearner where Sex=female' and survived=0"
myQuery.execute(query4)
myResult4= myQuery.fetchall()
for x in myResult4:
    femaleDeath = femaleDeath+1
    print(femaleDeath)
w = 0.4
xLabel=["Male", "Female"]
Male = [maleSurvived, femaleSurvived]
Female = [maleDeath, femaleDeath]
bar1 = np.arange(len(xLabel))
bar2 = [i+w for i in bar1]
plt.bar(bar1, Male, w, label="Survived")
plt.bar(bar2, Female, w, label="Death")
plt.xlabel("")
plt.ylabel()
plt.xticks(bar1+w/2, xLabel)
plt.legend()
plt.show()
```

