Accredited by NAAC & NBA*

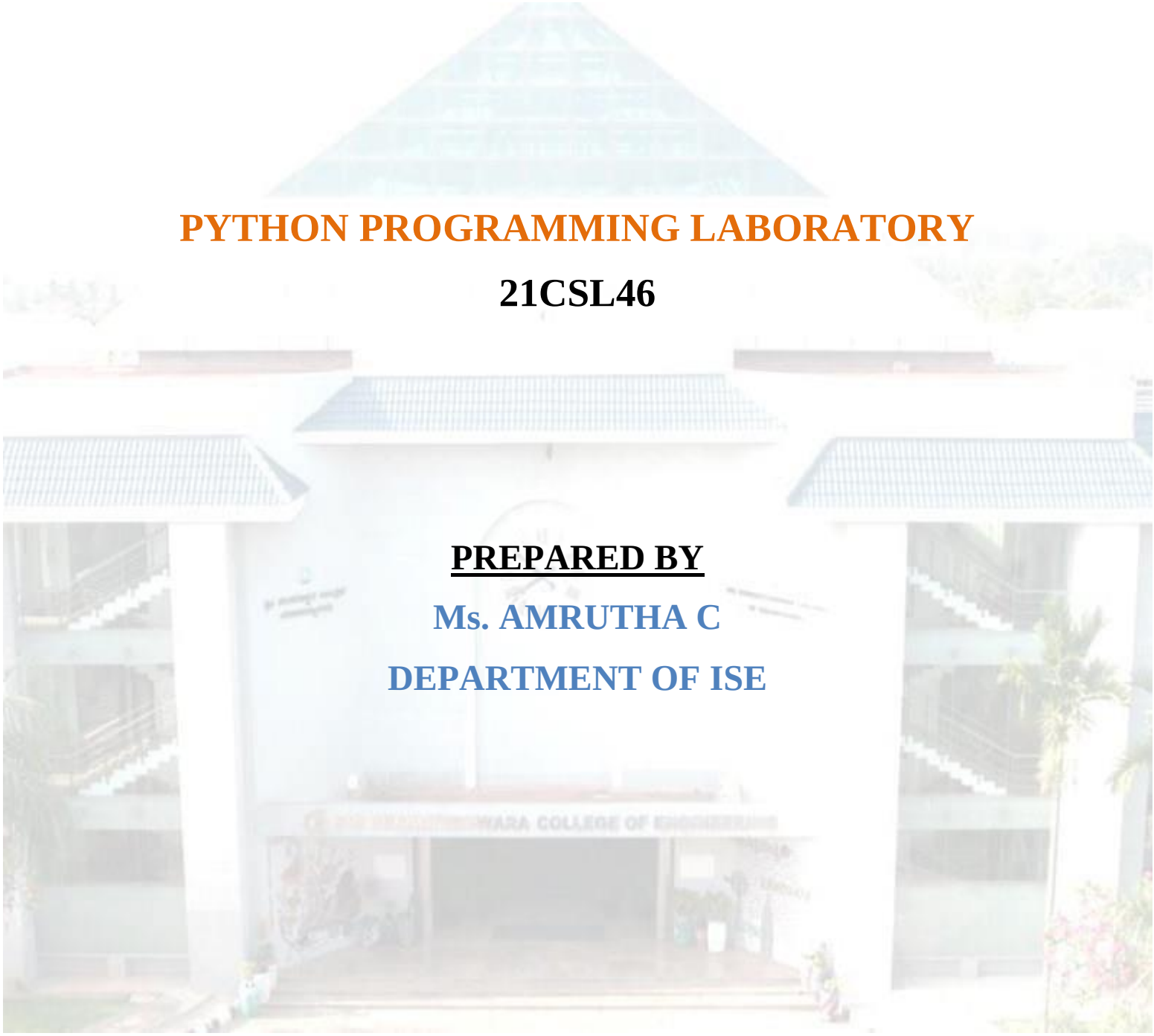
SVCE BENGALURU

SRI VENKATESHWARA COLLEGE OF ENGINEERING

— Affiliated to VTU, Approved by AICTE, Recognised by UGC u/s 2(f) & 12(B)—

PYTHON PROGRAMMING LABORATORY

21CSL46



PREPARED BY

Ms. AMRUTHA C

DEPARTMENT OF ISE

Accredited by NAAC & NBA*

SVCE BENGALURU

SRI VENKATESHWARA COLLEGE OF ENGINEERING

— Affiliated to VTU, Approved by AICTE, Recognised by UGC u/s 2(f) & 12(B)—

PYTHON PROGRAMMING LABORATORY

LABORATORY OVERVIEW

DEGREE:	BE	DEPARTMENT:	ISE
SEMESTER:	4	ACADEMIC YEAR:	2023-24
LABORATORY TITLE:	PYTHON PROGRAMMING	LABORATORY CODE:	21CSL46
LAB MANUAL AUTHOR:	AMRUTHA C ASSISTANT PROFESSOR DEPARTMENT OF ISE		

DESCRIPTION

COURSE OUTCOMES:

The student should be able to:

- Demonstrate proficiency in handling of loops and creation of functions.
- Identify the methods to create and manipulate lists, tuples and dictionaries.
- Discover the commonly used operations involving regular expressions and file system.
- Interpret the concepts of Object-Oriented Programming as used in Python.
- Determine the need for scraping websites and working with PDF, JSON and other file formats.

PROGRAMS LIST

1.	**Aim: Introduce the Python fundamentals, data types, operators, flow control and exception handling in Python** a) Write a python program to find the best of two test average marks out of three test's marks accepted from the user.

<table>
<tr><td></td><td>b) Develop a Python program to check whether a given number is palindrome or not and also count the number of occurrences of each digit in the input number.</td></tr>
<tr><td>2.</td><td>Aim: Demonstrating creation of functions, passing parameters and return values

a) Defined as a function F as Fn = Fn-1 + Fn-2. Write a Python program which accepts a value for N (where N >0) as input and pass this value to the function. Display suitable error message if the condition for input value is not followed.

b) Develop a python program to convert binary to decimal, octal to hexadecimal using functions.</td></tr>
<tr><td>3.</td><td>Aim: Demonstration of manipulation of strings using string methods

a) Write a Python program that accepts a sentence and find the number of words, digits, uppercase letters and lowercase letters.

b) Write a Python program to find the string similarity between two given strings

<table>
<tr><td>Sample Output:</td><td>Sample Output:</td></tr>
<tr><td>Original string:

Python Exercises

Python Exercises</td><td>Original string:

Python Exercises

Python Exercise</td></tr>
<tr><td>Similarity between two said strings: 1.0</td><td>Similarity between two said strings:
0.967741935483871</td></tr>
</table>
</td></tr>
<tr><td>4.</td><td>Aim: Discuss different collections like list, tuple and dictionary

a) Write a python program to implement insertion sort and merge sort using lists

b) Write a program to convert roman numbers in to integer values using dictionaries.</td></tr>
<tr><td>5.</td><td>Aim: Demonstration of pattern recognition with and without using regular expressions

a) Write a function called isphonenumber () to recognize a pattern 415-555-4242</td></tr>
</table>

	without using regular expression and also write the code to recognize the same pattern using regular expression. b) Develop a python program that could search the text in a file for phone numbers (+919900889977) and email addresses (sample@gmail.com)
6.	**Aim: Demonstration of reading, writing and organizing files.** a) Write a python program to accept a file name from the user and perform the following operations 1. Display the first N line of the file 2. Find the frequency of occurrence of the word accepted from the user in the file b) Write a python program to create a ZIP file of a particular folder which contains several files inside it.
7.	**Aim: Demonstration of the concepts of classes, methods, objects and inheritance** a) By using the concept of inheritance write a python program to find the area of triangle, circle and rectangle. b) Write a python program by creating a class called Employee to store the details of Name, Employee_ID, Department and Salary, and implement a method to update salary of employees belonging to a given department
8.	**Aim: Demonstration of classes and methods with polymorphism and overriding** a) Write a python program to find the whether the given input is palindrome or not (for both string and integer) using the concept of polymorphism and inheritance.
9.	**Aim: Demonstration of working with excel spreadsheets and web scraping** a) Write a python program to download the all XKCD comics b) Demonstrate python program to read the data from the spreadsheet and write

	the data in to the spreadsheet
10.	**Aim: Demonstration of working with PDF, word and JSON files** a) Write a python program to combine select pages from many PDFs b) Write a python program to fetch current weather data from the JSON file

INTRODUCTION

Python

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

It is used for:

- web development (server-side),
- software development,
- mathematics,
- system scripting.

Why Python

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-oriented way or a functional way.

Applications of Python

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

Reserved Words

The following list shows the Python keywords. These are reserved words and you cannot use them as constant or variable or any other identifier names. All the Python keywords contain lowercase letters only.

and	as	assert
break	class	continue
def	del	elif
else	except	False
finally	for	from
global	if	import
in	is	lambda
None	nonlocal	not
or	pass	raise
return	True	try
while	with	yield

PYTHON PROGRAMMING LABORATORY

Identifiers

A Python identifier is a name used to identify a variable, function, class, module or other object. An identifier starts with a letter A to Z or a to z or an underscore (_) followed by zero or more letters, underscores and digits (0 to 9).

Python does not allow punctuation characters such as @, $, and % within identifiers.

Naming conventions for Python identifiers:

- Python Class names start with an uppercase letter. All other identifiers start with a lowercase letter.
- Starting an identifier with a single leading underscore indicates that the identifier is private identifier.
- Starting an identifier with two leading underscores indicates a strongly private identifier.
- If the identifier also ends with two trailing underscores, the identifier is a language-defined special name.

Variables

Python variables do not need explicit declaration to reserve memory space or you can say to create a variable. A Python variable is created automatically when you assign a value to it. The equal sign (=) is used to assign values to variables.

Python Variable Names:

Every Python variable should have a unique name like a, b, c. A variable name can be meaningful like color, age, name etc. There are certain rules which should be taken care while naming a Python variable:

- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number or any special character like $, (, * % etc.
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Python variable names are case-sensitive which means Name and NAME are two different variables in Python.

- Python reserved keywords cannot be used naming the variable.

Data Types

Are used to define the type of a variable. It defines what type of data we are going to store in a variable. The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters.

Python has various built-in data types:

- Numeric - int, float, complex
- String – str
- Sequence - list, tuple, range
- Binary - bytes, bytearray, memoryview
- Mapping – dict
- Boolean – bool
- Set - set, frozenset
- None - NoneType

Operators

Python language supports the following types of operators.

- Arithmetic Operators
- Comparison (Relational) Operators
- Assignment Operators
- Logical Operators
- Bitwise Operators
- Membership Operators
- Identity Operators

PYTHON PROGRAMMING LABORATORY

PROGRAMS

EXPERIMENT NO: 1

AIM: Introduce the Python fundamentals, data types, operators, flow control and exception handling in Python

A) Write a python program to find the best of two test average marks out of three test's marks accepted from the user.

SOLUTION:

```
m1 = int(input("Enter marks for test1 : "))
m2 = int(input("Enter marks for test2 : "))
m3 = int(input("Enter marks for test3 : "))
if m1 <= m2 and m1 <= m3:
    avgMarks = (m2+m3)/2
elif m2 <= m1 and m2 <= m3:
    avgMarks = (m1+m3)/2
elif m3 <= m1 and m3 <= m2:
    avgMarks = (m1+m2)/2
print("Average of best two test marks  out of three test's marks is", avgMarks)
```

OUTPUT:

```
Enter marks for test1 :  45
Enter marks for test2 :  38
Enter marks for test3 :  48
Average of best two test marks  out of three test's marks is 46.5
```

B) Develop a Python program to check whether a given number is palindrome or not and also count the number of occurrences of each digit in the input number.

PYTHON PROGRAMMING LABORATORY

SOLUTION:

```
val = int(input("Enter a value : "))
str_val = str(val)
if str_val == str_val[::-1]:
    print("Palindrome")
else:
    print("Not Palindrome")
for i in range(10):
    if str_val.count(str(i)) > 0:
        print(str(i),"appears", str_val.count(str(i)), "times")
```

OUTPUT:

```
Enter a value :  12321
Palindrome
1 appears 2 times
2 appears 2 times
3 appears 1 times
```

EXPERIMENT NO: 2

AIM: Demonstrating creation of functions, passing parameters and return values

A) Defined as a function F as Fn = Fn-1 + Fn-2. Write a Python program which accepts a value for N (where N >0) as input and pass this value to the function. Display suitable error message if the condition for input value is not followed.

SOLUTION:

```
def fn(n):
    if n == 1:
        return 0
```

```
    elif n == 2:
        return 1
    else:
        return fn(n-1) + fn(n-2)
num = int(input("Enter a number : "))
if num > 0:
    print("fn(", num, ") = ",fn(num) , sep ="")
else:
    print("Error in input")
```

OUTPUT:

```
Enter a number :  5
fn(5) = 3
```

B) Develop a python program to convert binary to decimal, octal to hexadecimal using functions.

SOLUTION:

```
def bin2Dec(val):
    rev=val[::-1]
    dec = 0
    i = 0
    for dig in rev:
        dec += int(dig) * 2**i
        i += 1
    return dec
```

PYTHON PROGRAMMING LABORATORY

```
def oct2Hex(val):
    rev=val[::-1]
    dec = 0
    i = 0
    for dig in rev:
        dec += int(dig) * 8**i
        i += 1
    list=[]
    while dec != 0:
        list.append(dec%16)
        dec = dec // 16
    nl=[]
    for elem in list[::-1]:
        if elem <= 9:
            nl.append(str(elem))
        else:
            nl.append(chr(ord('A') + (elem -10)))
    hex = "".join(nl)
    return hex
num1 = input("Enter a binary number : ")
print(bin2Dec(num1))
num2 = input("Enter a octal number : ")
print(oct2Hex(num2))
```

PYTHON PROGRAMMING LABORATORY

OUTPUT:

```
Enter a binary number :  10111001
185
Enter a octal number :  675
1BD
```

EXPERIMENT NO: 3

Aim: Demonstration of manipulation of strings using string methods

A) Write a Python program that accepts a sentence and find the number of words, digits, uppercase letters and lowercase letters.

SOLUTION:

```
sentence = input("Enter a sentence : ")
wordList = sentence.split(" ")
print("This sentence has", len(wordList), "words")
digCnt = upCnt = loCnt = 0

for ch in sentence:
    if '0' <= ch <= '9':
        digCnt += 1
    elif 'A' <= ch <= 'Z':
        upCnt += 1
    elif 'a' <= ch <= 'z':
        loCnt += 1
print("This sentence has", digCnt, "digits", upCnt, "upper case letters", loCnt, "lower case letters")
```

OUTPUT:

```
Enter a sentence :  Total number of Students in 4th sem is 60
This sentence has 9 words
This sentence has 3 digits 2 upper case letters 28 lower case letters
```

B) Write a Python program to find the string similarity between two given strings

Sample Output:	**Sample Output:**

PYTHON PROGRAMMING LABORATORY

Original string: **Python Exercises** **Python Exercises**	**Original string:** **Python Exercises** **Python Exercise**
Similarity between two said strings: **1.0**	**Similarity between two said strings:** **0.967741935483871**

SOLUTION:

```
str1 = input("Enter String 1 \n")
str2 = input("Enter String 2 \n")
if len(str2) < len(str1):
    short = len(str2)
    long = len(str1)
else:
    short = len(str1)
    long = len(str2)
matchCnt = 0
for i in range(short):
    if str1[i] == str2[i]:
        matchCnt += 1
print("Similarity between two said strings:")
print(matchCnt/long)
```

OUTPUT:

```
Enter String 1
 Python Exercises
Enter String 2
```

PYTHON PROGRAMMING LABORATORY

```
 Python Exercises
Similarity between two said strings:
1.0

Enter String 1
 Python Exercises
Enter String 2
 Python Exercise
Similarity between two said strings:
0.9375
```

EXPERIMENT NO: 4

Aim: Discuss different collections like list, tuple and dictionary

A) Write a python program to implement insertion sort and merge sort using lists

SOLUTION:

```
import random

def merge_sort(lst):
    if len(lst) > 1:
        mid = len(lst) // 2
        left_half = lst[:mid]
        right_half = lst[mid:]

        merge_sort(left_half)
        merge_sort(right_half)

        i = j = k = 0

        while i < len(left_half) and j < len(right_half):
            if left_half[i] < right_half[j]:
                lst[k] = left_half[i]
                i += 1
            else:
                lst[k] = right_half[j]
                j += 1
```

```
            k += 1

        while i < len(left_half):
            lst[k] = left_half[i]
            i += 1
            k += 1

        while j < len(right_half):
            lst[k] = right_half[j]
            j += 1
            k += 1

    return lst

def insertion_sort(arr):
    for i in range(1, len(arr)):
        key = arr[i]
        j = i - 1
        while j >= 0 and key < arr[j]:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key

my_list = []

for i in range(10):
    my_list.append(random.randint(0, 999))

print("\nUnsorted List")
print(my_list)
print("Sorting using Insertion Sort")
insertion_sort(my_list)
print(my_list)

my_list = []

for i in range(10):
```

```
    my_list.append(random.randint(0, 999))

print("\nUnsorted List")
print(my_list)
print("Sorting using Merge Sort")
merge_sort(my_list)

print(my_list)
```

OUTPUT:

```
Unsorted List
[872, 867, 143, 639, 455, 276, 633, 148, 57, 955]
Sorting using Insertion Sort
[57, 143, 148, 276, 455, 633, 639, 867, 872, 955]

Unsorted List
[349, 543, 327, 470, 656, 749, 302, 524, 842, 308]
Sorting using Merge Sort
[302, 308, 327, 349, 470, 524, 543, 656, 749, 842]
```

B) Write a program to convert roman numbers in to integer values using dictionaries.

SOLUTION:

```
def roman2Dec(romStr):
    roman_dict ={'I': 1, 'V': 5, 'X': 10, 'L': 50, 'C': 100, 'D': 500, 'M': 1000}
    # Analyze string backwards
    romanBack = list(romStr)[::-1]
    value = 0
    # To keep track of order
    rightVal = roman_dict[romanBack[0]]
```

```
    for numeral in romanBack:
        leftVal = roman_dict[numeral]
        # Check for subtraction
        if leftVal < rightVal:
           value -= leftVal
        else:
            value += leftVal
        rightVal = leftVal
    return value
romanStr = input("Enter a Roman Number : ")
print(roman2Dec(romanStr))
```

OUTPUT:

```
Enter a Roman Number :  VII
7
```

EXPERIMENT NO: 5

Aim: Demonstration of pattern recognition with and without using regular expressions

A) Write a function called isphonenumber () to recognize a pattern 415-555-4242 without using regular expression and also write the code to recognize the same pattern using regular expression.

SOLUTION:

```
import re

def isphonenumber(numStr):
    if len(numStr) != 12:
        return False
    for i in range(len(numStr)):
```

```
        if i==3 or i==7:
            if numStr[i] != "-":
                return False
        else:
            if numStr[i].isdigit() == False:
                return False
    return True

def chkphonenumber(numStr):
    ph_no_pattern = re.compile(r'^\d{3}-\d{3}-\d{4}$')
    if ph_no_pattern.match(numStr):
        return True
    else:
        return False

ph_num = input("Enter a phone number : ")
print("Without using Regular Expression")
if isphonenumber(ph_num):
    print("Valid phone number")
else:
    print("Invalid phone number")

print("Using Regular Expression")
if chkphonenumber(ph_num):
    print("Valid phone number")
else:
    print("Invalid phone number")
```

OUTPUT:

```
Enter a phone number :  098-765-4321
Without using Regular Expression
Valid phone number
Using Regular Expression
Valid phone number
Enter a phone number :  098-A654-321
Without using Regular Expression
```

Invalid phone number
Using Regular Expression
Invalid phone number

B) Develop a python program that could search the text in a file for phone numbers (+919900889977) and email addresses (sample@gmail.com)

SOLUTION:

```
import re

# Define the regular expression for phone numbers

phone_regex = re.compile(r'\+\d{12}')

email_regex = re.compile(r'[A-Za-z0-9._]+@[A-Za-z0-9]+\.[A-Z|a-z]{2,}')

# Open the file for reading

with open('D:/ABC.txt', 'r') as f:

    # Loop through each line in the file

    for line in f:

        # Search for phone numbers in the line

        matches = phone_regex.findall(line)

        # Print any matches found

        for match in matches:

            print(match)

        matches = email_regex.findall(line)

        # Print any matches found

        for match in matches:

            print(match)
```

PYTHON PROGRAMMING LABORATORY

OUTPUT:

```
+111234567890
+120987654321
+115678904321
+120987612345
a@gmail.com
b@gmail.com
```

EXPERIMENT NO: 6

Aim: Demonstration of reading, writing and organizing files.

A) Write a python program to accept a file name from the user and perform the following operations

1. Display the first N line of the file

2. Find the frequency of occurrence of the word accepted from the user in the file

SOLUTION:

```
import os.path
import sys

fname = input("Enter the filename : ")

if not os.path.isfile(fname):
    print("File", fname, "doesn't exists")
    sys.exit(0)

infile = open(fname, "r")

lineList = infile.readlines()

for i in range(4):
    print(i+1, ":", lineList[i])
word = input("Enter a word : ")
```

```
cnt = 0
for line in lineList:
    cnt += line.count(word)
print("The word", word, "appears", cnt, "times in the file")
```

OUTPUT:

```
Enter the filename :  XYZ.txt
1 : Sri Venkateshwara College of Engineering is a private engineering College

2 : Located near Chikkajala, Vidyanagar about 22 km from Bangalore on the National Hi
ghway No.7.

3 : The College is approved by AICTE, New Delhi and affiliated to Visvesvaraya Techno
logical University, Belgaum.

4 : Established in 2001

Enter a word :  College
The word College appears 3 times in the file
```

B) Write a python program to create a ZIP file of a particular folder which contains several files inside it.

SOLUTION:

```
import os

import sys

import pathlib

import zipfile


dirName = input("Enter Directory name that you want to backup : ")

if not os.path.isdir(dirName):
```

```
    print("Directory", dirName, "doesn't exists")
    sys.exit(0)
curDirectory = pathlib.Path(dirName)
with zipfile.ZipFile("myZip.zip", mode="w") as archive:
    for file_path in curDirectory.rglob("*"):
        archive.write(file_path, arcname=file_path.relative_to(curDirectory))
if os.path.isfile("myZip.zip"):
    print("Archive", "myZip.zip", "created successfully")
else:
    print("Error in creating zip archive")
```

OUTPUT:

```
Enter Directory name that you want to backup : zipDemo
Archive myZip.zip created successfully
```

EXPERIMENT NO: 7

Aim: Demonstration of the concepts of classes, methods, objects and inheritance

A) By using the concept of inheritance write a python program to find the area of triangle, circle and rectangle.

SOLUTION:

```
import math

class Shape:
    def __init__(self):
        self.area = 0
        self.name = ""

    def showArea(self):
        print("The area of the", self.name, "is", self.area, "units")
```

PYTHON PROGRAMMING LABORATORY

```
class Circle(Shape):
    def __init__(self,radius):
        self.area = 0
        self.name = "Circle"
        self.radius = radius

    def calcArea(self):
        self.area = math.pi * self.radius * self.radius

class Rectangle(Shape):
    def __init__(self,length,breadth):
        self.area = 0
        self.name = "Rectangle"
        self.length = length
        self.breadth = breadth

    def calcArea(self):
        self.area = self.length * self.breadth

class Triangle(Shape):
    def __init__(self,base,height):
        self.area = 0
        self.name = "Triangle"
        self.base = base
        self.height = height

    def calcArea(self):
        self.area = self.base * self.height / 2

c1 = Circle(5)
c1.calcArea()
c1.showArea()

r1 = Rectangle(5, 4)
r1.calcArea()
r1.showArea()
```

```
t1 = Triangle(3, 4)
t1.calcArea()
t1.showArea()
```

OUTPUT:

```
The area of the Circle is 78.53981633974483 units
The area of the Rectangle is 20 units
The area of the Triangle is 6.0 units
```

B) Write a python program by creating a class called Employee to store the details of Name, Employee_ID, Department and Salary, and implement a method to update salary of employees belonging to a given department

SOLUTION:

```
class Employee:
    def __init__(self):
        self.name = ""
        self.empId = ""
        self.dept = ""
        self.salary = 0
    def getEmpDetails(self):
        self.name = input("Enter Employee name : ")
        self.empId = input("Enter Employee ID : ")
        self.dept = input("Enter Employee Dept : ")
        self.salary = int(input("Enter Employee Salary : "))
    def showEmpDetails(self):
        print("Employee Details")
```

```
        print("Name : ", self.name)
        print("ID : ", self.empId)
        print("Dept : ", self.dept)
        print("Salary : ", self.salary)
    def updtSalary(self):
        self.salary = int(input("Enter new Salary : "))
        print("Updated Salary", self.salary)
e1 = Employee()
e1.getEmpDetails()
e1.showEmpDetails()
e1.updtSalary()
```

OUTPUT:

```
Enter Employee name :  PRIYA
Enter Employee ID :  M10
Enter Employee Dept :  ISE
Enter Employee Salary :  20000
Employee Details
Name :  PRIYA
ID :  M10
Dept :  ISE
Salary :  20000
Enter new Salary :  30000
Updated Salary 30000
```

EXPERIMENT NO: 8

Aim: Demonstration of classes and methods with polymorphism and overriding

A) Write a python program to find the whether the given input is palindrome or not (for both string and integer) using the concept of polymorphism and inheritance.

PYTHON PROGRAMMING LABORATORY

SOLUTION:

```
class PaliStr:
    def __init__(self):
        self.isPali = False

    def chkPalindrome(self, myStr):
        if myStr == myStr[::-1]:
            self.isPali = True
        else:
            self.isPali = False

        return self.isPali

class PaliInt(PaliStr):
    def __init__(self):
        self.isPali = False

    def chkPalindrome(self, val):
        temp = val
        rev = 0
        while temp != 0:
            dig = temp % 10
            rev = (rev*10) + dig
            temp = temp //10

        if val == rev:
            self.isPali = True
        else:
            self.isPali = False

        return self.isPali

st = input("Enter a string : ")

stObj = PaliStr()
if stObj.chkPalindrome(st):
    print("Given string is a Palindrome")
```

```
else:
    print("Given string is not a Palindrome")
val = int(input("Enter a integer : "))

intObj = PaliInt()
if intObj.chkPalindrome(val):
    print("Given integer is a Palindrome")
else:
    print("Given integer is not a Palindrome")
```

OUTPUT:

```
Enter a string :  MADAM
Given string is a Palindrome
Enter a integer :  12354
Given integer is not a Palindrome

Enter a string :  SCIENCE
Given string is not a Palindrome
Enter a integer :  98789
Given integer is a Palindrome
```

EXPERIMENT NO: 9

Aim: Demonstration of working with excel spreadsheets and web scraping

A) Write a python program to download the all XKCD comics

SOLUTION:

```
import requests
import os
from bs4 import BeautifulSoup

# Set the URL of the first XKCD comic
url = 'https://xkcd.com/1/'

# Create a folder to store the comics
if not os.path.exists('xkcd_comics'):
    os.makedirs('xkcd_comics')
```

PYTHON PROGRAMMING LABORATORY

```
# Loop through all the comics
while True:
    # Download the page content
    res = requests.get(url)
    res.raise_for_status()

    # Parse the page content using BeautifulSoup
    soup = BeautifulSoup(res.text, 'html.parser')

    # Find the URL of the comic image
    comic_elem = soup.select('#comic img')
    if comic_elem == []:
        print('Could not find comic image.')
    else:
        comic_url = 'https:' + comic_elem[0].get('src')

        # Download the comic image
        print(f'Downloading {comic_url}...')
        res = requests.get(comic_url)
        res.raise_for_status()

        # Save the comic image to the xkcd_comics folder
        image_file = open(os.path.join('xkcd_comics', os.path.basename(comic_url)), 'wb')
        for chunk in res.iter_content(100000):
            image_file.write(chunk)
        image_file.close()

    # Get the URL of the previous comic
    prev_link = soup.select('a[rel="prev"]')[0]
    if not prev_link:
        break
    url = 'https://xkcd.com' + prev_link.get('href')

print('All comics downloaded.')
```

PYTHON PROGRAMMING LABORATORY

OUTPUT:

```
Downloading https://imgs.xkcd.com/comics/barrel_cropped_(1).jpg...
Downloading https://imgs.xkcd.com/comics/radians_are_cursed.png...
Downloading https://imgs.xkcd.com/comics/presents_for_biologists.png...
Downloading https://imgs.xkcd.com/comics/launch_window.png...
Downloading https://imgs.xkcd.com/comics/obituary_editor.png...
Downloading https://imgs.xkcd.com/comics/fanservice.png...
Downloading https://imgs.xkcd.com/comics/hand_dryers.png...
```

B) Demonstrate python program to read the data from the spreadsheet and write the data in to the spreadsheet

SOLUTION:

```
from openpyxl import Workbook

from openpyxl.styles import Font


wb = Workbook()

sheet = wb.active

sheet.title = "Language"

wb.create_sheet(title = "Capital")

lang = ["Kannada", "Telugu", "Tamil"]

state = ["Karnataka", "Telangana", "Tamil Nadu"]

capital = ["Bengaluru", "Hyderabad", "Chennai"]

code =['KA', 'TS', 'TN']

sheet.cell(row = 1, column = 1).value = "State"

sheet.cell(row = 1, column = 2).value = "Language"

sheet.cell(row = 1, column = 3).value = "Code"
```

PYTHON PROGRAMMING LABORATORY

```
ft = Font(bold=True)
for row in sheet["A1:C1"]:
    for cell in row:
        cell.font = ft
for i in range(2,5):
    sheet.cell(row = i, column = 1).value = state[i-2]
    sheet.cell(row = i, column = 2).value = lang[i-2]
    sheet.cell(row = i, column = 3).value = code[i-2]
wb.save("demo.xlsx")
sheet = wb["Capital"]
sheet.cell(row = 1, column = 1).value = "State"
sheet.cell(row = 1, column = 2).value = "Capital"
sheet.cell(row = 1, column = 3).value = "Code"
ft = Font(bold=True)
for row in sheet["A1:C1"]:
    for cell in row:
        cell.font = ft
for i in range(2,5):
    sheet.cell(row = i, column = 1).value = state[i-2]
    sheet.cell(row = i, column = 2).value = capital[i-2]
    sheet.cell(row = i, column = 3).value = code[i-2]
wb.save("demo.xlsx")
srchCode = input("Enter state code for finding capital ")
```

```
for i in range(2,5):
    data = sheet.cell(row = i, column = 3).value
    if data == srchCode:
        print("Corresponding capital for code", srchCode, "is", sheet.cell(row = i, column = 2).value)
sheet = wb["Language"]
srchCode = input("Enter state code for finding language ")
for i in range(2,5):
    data = sheet.cell(row = i, column = 3).value
    if data == srchCode:
        print("Corresponding language for code", srchCode, "is", sheet.cell(row = i, column = 2).value)
wb.close()
```

OUTPUT:

```
Enter state code for finding capital  KA
Corresponding capital for code KA is Bengaluru
Enter state code for finding language  KA
Corresponding language for code KA is Kannada
```

EXPERIMENT NO: 10

Aim: Demonstration of working with PDF, word and JSON files

A) Write a python program to combine select pages from many PDFs

SOLUTION:

```
from PyPDF2 import PdfWriter, PdfReader

num = int(input("Enter page number you want combine from multiple documents "))

pdf1 = open('birds.pdf', 'rb')
```

```
pdf2 = open('birdspic.pdf', 'rb')

pdf_writer = PdfWriter()
pdf1_reader = PdfReader(pdf1)
page = pdf1_reader.pages[num - 1]
pdf_writer.add_page(page)

pdf2_reader = PdfReader(pdf2)
page = pdf2_reader.pages[num - 1]
pdf_writer.add_page(page)

with open('output.pdf', 'wb') as output:
    pdf_writer.write(output)
```

OUTPUT:

B) Write a python program to fetch current weather data from the JSON file

SOLUTION:

```
import json

# Load the JSON data from file

with open('weather_data.json') as f:

    data = json.load(f)

# Extract the required weather data

current_temp = data['main']['temp']
```

```
humidity = data['main']['humidity']
weather_desc = data['weather'][0]['description']
# Display the weather data
print(f"Current temperature: {current_temp}°C")
print(f"Humidity: {humidity}%")
print(f"Weather description: {weather_desc}")
```

JSON FILE:

```
{
  "coord": {
    "lon": -73.99,
    "lat": 40.73
  },
  "weather": [
    {
      "id": 800,
      "main": "Clear",
      "description": "clear sky",
      "icon": "01d"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 15.45,
```

```
  "feels_like": 12.74,
  "temp_min": 14.44,
  "temp_max": 16.11,
  "pressure": 1017,
  "humidity": 64
},
"visibility": 10000,
"wind": {
  "speed": 4.63,
  "deg": 180
},
"clouds": {
  "all": 1
},
"dt": 1617979985,
"sys": {
  "type": 1,
  "id": 5141,
  "country": "US",
  "sunrise": 1617951158,
  "sunset": 1618000213
},
"timezone": -14400,
```

PYTHON PROGRAMMING LABORATORY

```
  "id": 5128581,
  "name": "New York",
  "cod": 200
}
```

OUTPUT:

Current temperature: 15.45°C

Humidity: 64%

Weather description: clear sky

CONCEPT MAP