DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND MANAGEMENT

Kanakapura Road, Udayapura, Bangalore -560 082,
Karnataka

(Affiliated to VTU, Belagavi, Approved by AICTE, New Delhi,)

(CE, CSE, ECE, EEE, ISE, ME Courses Accredited by NBA, New Delhi Accredited by NAAC, A+)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

2022-2023

PYTHON PROGRAMMING LABORATORY MANUAL

(21CSL46)

Compiled by:

Dr.Shalini.S

Ms. Keerthi Mohan

Vision of the Department

Epitomize CSE graduate to carve a niche globally in the field of computer science to excel in the world of information technology and automation by imparting knowledge to sustain skills for the changing trends in the society and industry.

Mission of the Department

M1: To educate students to become excellent engineers in a confident and creative environment through world-class pedagogy.

M2: Enhancing the knowledge in the changing technology trends by giving hands-on experience through continuous education and by making them to organize & participate in various events.

M3: Impart skills in the field of IT and its related areas with a focus on developing the required competencies and virtues to meet the industry expectations.

M4: Ensure quality research and innovations to fulfill industry, government & social needs.

M5: Impart entrepreneurship and consultancy skills to students to develop self-sustaining life skills in multi-disciplinary areas.

Program Specific Outcomes (PSO)

PSO1: **Foundation of Mathematical Concepts:** Ability to use mathematical methodologies to solve the problem using suitable mathematical analysis, data structure and suitable algorithm.

PSO2: **Foundation of Computer System:** Ability to interpret the fundamental concepts and methodology of computer systems. Students can understand the functionality of hardware and software aspects of computer systems.

PSO3: **Foundations of Software Development:** Ability to grasp the software development lifecycle and methodologies of software systems. Possess competent skills and knowledge of software design process. Familiarity and practical proficiency with a broad area of programming concepts and provide new ideas and innovations towards research.

PSO4: **Foundations of Multi-Disciplinary Work:** Ability to acquire leadership skills to perform professional activities with social responsibilities, through excellent flexibility to function in multi-disciplinary work environment with self-learning skills

Program Outcomes:

Sl. No.	Description	POs
1	Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and computer science and business systems to the solution of complex engineering and societal problems.	PO1
2	Problem analysis: Identify, formulate, review research literature, and analyze complex engineering and business problems reaching substantiated conclusions using firstprinciples of mathematics, natural sciences, and engineering sciences.	PO2
3	Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.	PO3
4	Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesisof the information to provide valid conclusions.	PO4
5	Modern tool usage: Create, select, and apply appropriate techniques, resources, andmodern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations	PO5
6	The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering and business practices.	PO6
7	Environment and sustainability: Understand the impact of the professional engineering solutions in business societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.	PO7
8	Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering and business practices.	PO8
9	Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.	PO9
10	Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.	PO10
11	Project management and finance: Demonstrate knowledge and understanding of the engineering, business and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.	PO11
12	Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.	PO12

PYTHON PROGRAMMING LABORATORY			
Course Code	**21CSL46**	CIE Marks	50
Teaching Hours/Weeks (L: T: P: S)	0: 0: 2: 0	SEE Marks	50
Total Hours of Pedagogy	24	Total Marks	100
Credits	01	Exam Hours	03

Course Objectives:

CLO 1. Demonstrate the use of IDLE or PyCharm IDE to create Python Applications

CLO 2. Using Python programming language to develop programs for solving real-world problemsCLO 3. Implement the Object-Oriented Programming concepts in Python.

CLO 4. Appraise the need for working with various documents like Excel, PDF, Word and Others

CLO 5. Demonstrate regular expression using python programming

Note: two hours tutorial is suggested for each laboratory sessions.

Prerequisite

- Students should be familiarized about Python installation and setting Python environment
- Usage of IDLE or IDE like PyCharm should be introduced

Python Installation: https://www.youtube.com/watch?v=Kn1HF3oD19c PyCharm Installation: https://www.youtube.com/watch?v=SZUNUB6nz3g

Sl. No.	***PART A – List of problems for which student should develop program and execute inthe Laboratory***
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. b) Develop a Python program to check whether a given number is palindrome or not andalso count the number of occurrences of each digit in the input number. Datatypes: https://www.youtube.com/watch?v=gCCVsvgR2KU Operators: https://www.youtube.com/watch?v=v5MR5JnKcZI Flow Control: https://www.youtube.com/watch?v=PqFKRqpHrjwFor loop: https://www.youtube.com/watch?v=0ZvaDa8eT5s While loop: https://www.youtube.com/watch?v=HZARImviDxg Exceptions: https://www.youtube.com/watch?v=6SPDvPK38tw
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. b) Develop a python program to convert binary to decimal, octal to hexadecimal using functions. Functions: https://www.youtube.com/watch?v=BVfCWuca9nw Arguments: https://www.youtube.com/watch?v=ijXMGpoMkhQ Return value: https://www.youtube.com/watch?v=nuNXiEDnM44

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.
	b) Write a Python program to find the string similarity between two given strings **Sample Output: Sample Output:** **Original string: Original string:** **Python Exercises Python Exercises** **Python Exercises Python Exercise** **Similarity between two said strings: Similarity between two said strings: 1.0** **0.967741935483871** **Strings: https://www.youtube.com/watch?v=lSItwlnF0eU** **String functions: https://www.youtube.com/watch?v=9a3CxJyTq00**
4	**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.** **Lists: https://www.youtube.com/watch?v=Eaz5e6M8tL4** **List methods: https://www.youtube.com/watch?v=8-RDVWGktuI Tuples: https://www.youtube.com/watch?v=bdS4dHIJGBc** **Tuple operations: https://www.youtube.com/watch?v=TItKabcTTQ4 Dictionary: https://www.youtube.com/watch?v=4Q0pW8XBOkc Dictionary methods: https://www.youtube.com/watch?v=oLeNHuORpNY**
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.** **b) Develop a python program that could search the text in a file for phone numbers (+919900889977) and email addresses (sample@gmail.com)** **Regular expressions: https://www.youtube.com/watch?v=LnzFnZfHLS4**
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.** **Files: https://www.youtube.com/watch?v=vuyb7CxZgbU** **https://www.youtube.com/watch?v=FqcjKewJTQ0** **File organization: https://www.youtube.com/watch?v=MRuq3SRXses**

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.** **OOP's concepts: https://www.youtube.com/watch?v=qiSCMNBIP2g Inheritance: https://www.youtube.com/watch?v=Cn7AkDb4pIU**
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.** **Overriding: https://www.youtube.com/watch?v=CcTzTuIsoFk**
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** **Web scraping: https://www.youtube.com/watch?v=ng2o98k983k Excel: https://www.youtube.com/watch?v=nsKNPHJ9iPc**
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** **PDFs: https://www.youtube.com/watch?v=q70xzDG6nls** **https://www.youtube.com/watch?v=JhQVD7Y1bsA** **https://www.youtube.com/watch?v=FcrW-ESdY-A** **Word files: https://www.youtube.com/watch?v=ZU3cSl51jWE JSON files: https://www.youtube.com/watch?v=9N6a-VLBa2I**
Python (Full Course): https://www.youtube.com/watch?v=_uQrJ0TkZlc	
Pedagogy	For the above experiments the following pedagogy can be considered. Problem based learning, Active learning, MOOC, Chalk &Talk

PART B – Practical Based Learning

A problem statement for each batch is to be generated in consultation with the co-examiner and student should develop an algorithm, program and execute the program for the given problem with appropriate outputs.

Course Outcomes:

CO 1. Demonstrate proficiency in handling of loops and creation of functions.
CO 2. Identify the methods to create and manipulate lists, tuples and dictionaries.
CO 3. Discover the commonly used operations involving regular expressions and file system. CO 4. Interpret the concepts of Object-Oriented Programming as used in Python.
CO 5. Determine the need for scraping websites and working with PDF, JSON and other file formats.

Assessment Details (both CIE and SEE)
The weightage of Continuous Internal Evaluation (CIE) is 50% and for Semester End Exam (SEE) is 50%. The minimum passing mark for the CIE is 40% of the maximum marks (20 marks). A student shall be deemed to have satisfied the academic requirements and earned the credits allotted to each course. The student has to secure not less than 35% (18 Marks out of 50) in the semester-end examination (SEE). The student has to secure 40% of sum of the maximum marks of CIE and SEE to qualify in the course.

Continuous Internal Evaluation (CIE):

CIE marks for the practical course is 50 Marks.

The split-up of CIE marks for record/ journal and test are in the ratio 60:40.

- **Each experiment to be evaluated for conduction with observation sheet and record write-up. Rubrics for the evaluation of the journal/write-up for hardware/software experiments designed by the faculty who is handling the laboratory session and is made known to students at the beginning of the practical session.**
- **Record should contain all the specified experiments in the syllabus and each experiment write-up will be evaluated for 10 marks.**
- **Total marks scored by the students are scaled downed to 30 marks (60% of maximum marks).**
- **Weightage to be given for neatness and submission of record/write-up on time.**
- **Department shall conduct 02 tests for 100 marks, the first test shall be conducted after the 8th week of the semester and the second test shall be conducted after the 14th week of the semester.**
- **In each test, test write-up, conduction of experiment, acceptable result, and procedural knowledge will carry a weightage of 60% and the rest 40% for viva-voce.**
- **The suitable rubrics can be designed to evaluate each student's performance and learning ability. Rubrics suggested in Annexure-II of Regulation book**
- **The average of 02 tests is scaled down to 20 marks (40% of the maximum marks).**

The Sum of scaled-down marks scored in the report write-up/journal and average marks of two tests is the total CIE marks scored by the student.

Semester End Evaluation (SEE):

- **SEE marks for the practical course is 50 Marks.**
- **SEE shall be conducted jointly by the two examiners of the same institute, examiners are appointed by the University**
- **All laboratory experiments are to be included for practical examination.**
- **(Rubrics) Breakup of marks and the instructions printed on the cover page of the answer script to be strictly adhered to by the examiners. OR based on the course requirement evaluation rubrics shall be decided jointly by examiners.**
- **Students can pick one question (experiment) from the questions lot prepared by the internal /external examiners jointly.**
- **Evaluation of test write-up/ conduction procedure and result/viva will be conducted jointly by examiners.**
- **General rubrics suggested for SEE are mentioned here, writeup-20%, Conduction procedure and result in -60%, Viva-voce 20% of maximum marks. SEE for practical shall be evaluated for 100 marks and scored marks shall be scaled down to 50 marks (however, based on course type, rubrics shall be decided by the examiners)**
- **Students can pick one experiment from the questions lot of PART A with equal choice to all the students in a batch. For PART B examiners should frame a question for each batch, student should**

develop an algorithm, program, execute and demonstrate the results with appropriate output for the given problem.

- **Weightage of marks for PART A is 80% and for PART B is 20%. General rubrics suggested to be followed for part A and part B.**
- **Change of experiment is allowed only once and Marks allotted to the procedure part to be made zero (Not allowed for Part B).**
- **The duration of SEE is 03 hours**

Rubrics suggested in Annexure-II of Regulation book

Textbooks:

1. **Al Sweigart, "Automate the Boring Stuff with Python",1stEdition, No Starch Press, 2015. (Available under CC-BY-NC-SA license at https://automatetheboringstuff.com/)**
2. **Reema Thareja "Python Programming Using Problem Solving Approach" Oxford University Press.**
3. **Allen B. Downey, "Think Python: How to Think Like a Computer Scientist",**

2nd Edition, Green Tea Press, 2015. (Available under CC-BY-NC license at http://greenteapress.com/thinkpython2/thinkpython2.pdf)

Course Outcomes:

At the end of the Course, the Student will be able to:

CO1	Understand the concepts of python programming language.
CO2	Apply the concepts of python programming language in solving real world problems
CO3	Analyse the concepts of python programming language in solving real world problems.
CO4	Design applications built in and user define data types of python programming language
CO5	Demonstrate an application using python programming language with a modern tool.

1a) Write a python program to find the best of two test average marks out of three test's marks accepted from the user.

```
m1 = int (input("Enter the marks in the first test: "))
m2 = int (input("Enter the marks in second test: "))
m3 = int (input("Enter the marks in third test: "))

if (m1 > m2):
          if (m2 > m3):
                  total = m1 + m2
        else:
                  total = m1 + m3
elif (m1 > m3):
                  total = m1 + m2
        else:
                  total = m2 + m3

Avg = total / 2
print ("The average of the best two test marks is: ",Avg)
```

OUTPUT

```
Enter the marks in the first test: 45
Enter the marks in second test: 78
Enter the marks in third test: 23
The average of the best two test marks is:  61.5
```

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.

```
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 1

```
Enter a value : 1222221
Palindrome
1 appears 2 times
2 appears 5 times
```

OUTPUT 2

```
Enter a value : 12223
Not Palindrome
1 appears 1 times
2 appears 3 times
3    appears 1 times
```

2 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.

```
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 : 0

Error in input



Enter a number : 7

fn(7) = 8
```

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

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

OUTPUT:

```
Enter a binary number : 0001
1
Enter a octal number : 675
1BD
```

3a. Write a Python program that accepts a sentence and find the number of words, digits, uppercase letters and lowercase letters.

```
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 :  cse department at dsatm

This sentence has 4 words

This sentence has 0 digits 0 upper case letters 20 lower case letters

Enter a sentence : Cse department at DSATM 1234

This sentence has 5 words

This sentence has 4 digits 6 upper case letters 14 lower case letters
```

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

```
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
dsatmcse
Enter String 2
dsatmcse
Similarity between two said strings:
1.0
Enter String 1
cse department at dsatm
Enter String 2
ise department at dsatm
Similarity between two said strings:
0.9565217391304348
```

4a. Write a python program to implement insertion sort and merge sort using lists.

```
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

[873, 781, 899, 241, 248, 452, 800, 716, 752, 583]

Sorting using Insertion Sort

[241, 248, 452, 583, 716, 752, 781, 800, 873, 899]


Unsorted List

[625, 71, 883, 877, 263, 872, 298, 781, 537, 97]

Sorting using Merge Sort

[71, 97, 263, 298, 537, 625, 781, 872, 877, 883]
```

4b. Write a program to convert roman numbers in to integer values using dictionaries.

```
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 : X

10
```

5 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.

```
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 : 123-456-2345
Without using Regular Expression
Valid phone number
Using Regular Expression
Valid phone number

Enter a phone number : 123-fgt-7866
Without using Regular Expression
Invalid phone number
Using Regular Expression
Invalid phone number
```

5b. Develop a python program that could search the text in a file for phone numbers (+919900889977) and email addresses (sample@gmail.com)

```
                    import re
                    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('example.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)
```

```
INPUT                                                        OUTPUT

EXAMPLE.TXT

cse     cse@gmail.com         +919876543210      cse@gmail.com        +919876543210

ise     ise@dsatm.edu.in      76545664           ise@dsatm.edu.in

ece     ece@dsatm.edu.in      12345678           ece@dsatm.edu.in

eee     eee@abc.efg.hij.com   12sd345g           eee@abc.efg
```

6. 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.

```
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(20):
    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 : example.txt
1 : this is phone number +918151894220
2 : no phone number here
3 : here we have one +829392938876
```

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

```
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 : cdp
Archive myZip.zip created successfully
```

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

```
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")
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.

```
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()
Enter Employee name : Sameer
```

OUTPUT

```
Enter Employee ID : A123

Enter Employee Dept : CSE

Enter Employee Salary : 85750


Employee Details

Name :  Sameer

ID :  A123

Dept :  CSE

Salary :  85750


Enter new Salary : 88800

Updated Salary 88800
```

8. **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**.

```
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 : csedsatm
Given string is not a Palindrome
Enter a integer : 123321
Given integer is a Palindrome
```

9 a) Write a python program to download the all XKCD comics.

```
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')
# 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.')
```

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...
```

9b. Demonstrate python program to read the data from the spreadsheet and write the data in to the Spreadsheet.

```
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"
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()
int('All comics downloaded.')
```

OUTPUT

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

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

```
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

This program allows you to extract specific pages from two PDF files, “birds.pdf” and “birdspic.pdf,” by entering the page numbers as user input. Once you input the desired page numbers, the program fetches those pages from both PDF files and combines them into a new file called “output.pdf.”

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

```
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}")
```

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

OUTPUT

```
Current temperature: 15.45°C
Humidity: 64%
Weather description: clear sky
```

PART- B QUESTIONS

1. Write a Python program to find the biggest of three numbers
   i)without taking input from the user
   ii)taking input from the user
2. Write a python program using functions to add two numbers
3. Write a python program
   i)to compare two strings
   ii)to join two strings
4. Write a python program using list to show the following slicing operation
   a)i)items from index 2 to index 4
   ii)items from index 5 to end
   iii)items beginning to end
   b)Write a program to show empty tuple,tuple with integers,tuple with different data typesanf nested tuple
5. Write a program to read the first line using readline()
6. Write a program in python to copy all the contents of one file to another file in upper case
7. Write a program in Python to demonstrate inheritance property for the Student class
8. Write a program in python to demonstrate polymorphism
9. Write a program to show the exception handling in Python
10. Show the working of Nested for loop in lists

1.Write a Python program to find the biggest of three numbers

i)without taking input from the user

ii)taking input from the user

```
# Python program to find the largest number among the three input numbers


# change the values of num1, num2 and num3
# for a different result
num1 = 10
num2 = 14
num3 = 12


# uncomment following lines to take three numbers from user
#num1 = float(input("Enter first number: "))
#num2 = float(input("Enter second number: "))
#num3 = float(input("Enter third number: "))


if (num1 >= num2) and (num1 >= num3):
   largest = num1
elif (num2 >= num1) and (num2 >= num3):
   largest = num2
else:
   largest = num3


print("The largest number is", largest)
```

Result: The largest number is 14

2.Write a python program using functions to add two numbers

```
# function with two arguments

def add_numbers(num1, num2):

   sum = num1 + num2

   print("Sum: ",sum)


# function call with two values

add_numbers(5, 4)
```

Output: Sum: 9

3.Write a python program i)to compare two strings ii)to join two strings

```
i) str1 = "Hello, world!"

str2 = "I love Python."

str3 = "Hello, world!"

# compare str1 and str2

print(str1 == str2)

# compare str1 and str3

print(str1 == str3)
```

output False

True

```
ii) greet = "Hello, "
```

```
 name = "Jack"
# using + operator
result = greet + name
print(result)
```

Output: Hello, Jack

4.Write a python program using list to show the following slicing operation

a)i)items from index 2 to index 4

ii)items from index 5 to end

iii)items beginning to end

```
# List slicing in Python
my_list = ['p','r','o','g','r','a','m','i','z']
# items from index 2 to index 4
print(my_list[2:5])
# items from index 5 to end
print(my_list[5:])
# items beginning to end
print(my_list[:])
```

b)Write a program to show empty tuple,tuple with integers,tuple with different data typesanf nested tuple

```
 # Different types of tuples
# Empty tuple
```

```
my_tuple = ()

print(my_tuple)

# Tuple having integers

my_tuple = (1, 2, 3)

print(my_tuple)

# tuple with mixed datatypes

my_tuple = (1, "Hello", 3.4)

print(my_tuple)

# nested tuple

my_tuple = ("mouse", [8, 4, 6], (1, 2, 3))

print(my_tuple)
```

c)Write a python program to implement dictionary to print States and their capitals

5.Write a program to read the first line using readline()

```
myfile = open("demo.txt", "r")
myline = myfile.readline()
print(myline)
myfile.close()
```

demo.txt

```
Testing - FirstLine
Testing - SecondLine
Testing - Third Line
Testing - Fourth Line
Testing - Fifth Line
```

Output:

Testing – FirstLine

6 .Write a program in python to copy all the contents of one file to another file in upper case

```
To open the first file in read mode
f1 = open("sample file 1.txt", "r")
```

```
# To open the second file in append mode
f2 = open("sample file 2.txt", "a")

# For loop to traverse through the file
for line in f1:

    # Writing the content of the first
    # file to the second file

    # Using upper() function
    # to capitalize the letters
    f2.write(line.upper())
```

7 a)Write a proram in Python to demonstrate inheritance property for the Student class

```
class Person:
  def __init__(self, fname, lname):
    self.firstname = fname
    self.lastname = lname

  def printname(self):
    print(self.firstname, self.lastname)

class Student(Person):
  def __init__(self, fname, lname, year):
    super().__init__(fname, lname)
    self.graduationyear = year

x = Student("Mike", "Olsen", 2019)
print(x.graduationyear)
```

8.Write a program in python to demonstrate polymorphism

```
# A simple Python function to demonstrate
# Polymorphism

def add(x, y, z = 0):
            return x + y+z

# Driver code
print(add(2, 3))
print(add(2, 3, 4))
```

9.Write aprogram to show the exception handling in Python

```
# Program to handle multiple errors with one
# except statement
# Python 3
def fun(a):

        if a < 4:

                # throws ZeroDivisionError for a = 3
                b = a/(a-3)

        # throws NameError if a >= 4
        print("Value of b = ", b)

try:
        fun(3)
        fun(5)

# note that braces () are necessary here for
# multiple exceptions
except ZeroDivisionError:
        print("ZeroDivisionError Occurred and Handled")
except NameError:
        print("NameError Occurred and Handled")
```

output:

```
ZeroDivisionError Occurred and Handled
```

10.Show the working of Nested for loop in lists

```
adj = ["red", "big", "tasty"]
fruits = ["apple", "banana", "cherry"]

for x in adj:
  for y in fruits:
    print(x, y)
```

output:

```
red apple
red banana
red cherry
big apple
big banana
big cherry
tasty apple
tasty banana
tasty cherry
```

Viva questions

1. Why python is called Object oriented language
2. What are the characteristics of Python?
3. Who is the founder of Python?
4. Why List is called Sequential and mutable data type?
5. Give examples of immutable data types
6. Give the differences between Python and C languages
7. Give the differences between Dictonary and Tuple.
8. Explain Escape characters in Python.
9. How to give multiline comment in python?
10. What are the supported data types in Python?
11. What is the output of print str[0] if str = 'Hello World!'?
12. What is the output of print str[2:5] if str = 'Hello World!'?
13. What is the output of print str * 2 if str = 'Hello World!'?
14. How will you convert a string to an int in python?
15. What is the purpose of // operator?
16. Explain polymormishm and Encapsulation in Python
17. Give the difference between shallow () and deep()
18. Discuss Inheritance
19. Explain class concepts in Python
20. Find the output for the following code

    a)while count < nterms:

    ```
        print(n1)
        nth = n1 + n2
        # update values
        n1 = n2
        n2 = nth
        count += 1
    ```