

**AMITY INTERNATIONAL SCHOOL**  
**PRACTICAL LIST 2023-24**  
**CLASS XII – COMPUTER SCIENCE**

**File Handling, Functions and Data Structures:**

1. Write a function to create a text file containing following data:

*Neither apple nor pine are in pineapple. Boxing rings are square.  
Writers write, but fingers don't fing. Overlook and oversee are opposites.  
A house can burn up as it burns down. An alarm goes off by going on.*

- a) Read back the entire file content using read() or readlines() and print it.
- b) Append more text of your choice in the file and display the content of file with line numbers prefixed to line.
- c) Display last line of file.
- d) Display first line from 10<sup>th</sup> character onwards.
- e) Read and display a line from the file. Ask user to provide the line no. to be read.
- f) Find the frequency of words beginning with every letter i.e. (For the above example)  
Words beginning with a: 5  
Words beginning with n: 2  
Words beginning with p: 2  
Words beginning with o: 5 and so on

2. Assume that a text file named file1.txt contains some text, write a function named isvowel( ) that reads the file file1.txt and creates a new file named file2.txt, which shall contain only those words from the file file1.txt which don't start with a vowel.

For example, if the file1.txt contains:

Carry Umbrella and Overcoat When it Rains

Then the file file2.txt shall contain

Carry When Rains

3. A file containing data about a collection of students has the following format. Each line contains a first name, a second name, a registration number, no of years and a department separated by tabs.

Rajat	Sen	12345	1	CSEE
Jagat	Narain	13467	3	CSEE
Anu	Sharma	11756	2	Biology
Sumita	Trikha	23451	4	Biology
Sumder	Kumra	11234	3	MME
Kanti	Bhushan	23211	3	CSEE

- a) Write a Python program that will copy the contents of the file into a list of tuples
- b) Display full details of the student sorted by registration number

- i) The names of all students with no of year less than 3
  - ii) The number of people in each department
4. Write a program that reads myfile.txt, and builds a histogram (a dictionary having key value pair as word: occurrence) of the words in the file.
  - a) Now use histogram to print :
    - i) Total number of words
    - ii) Number of different words
    - iii) The most common words
  - b) Using above text file, myfile.txt,
    - i) Write a program that maps a list of words read from the file to an integer representing the length of the corresponding words (Use a dictionary having key value pair as length : list of word).

Now using this dictionary,

  - ii) Design a function find\_longest\_word() to display a list of longest words from file.
  - iii) Define a function filter\_long\_words(n) that takes an integer n and returns the list of words that are longer than n from file.
5. A dictionary Customer contains the following keys: roomno, name, duration. A binary file 'hotel.dat' contains details of customer checked in the hotel. Write a program to perform the following using pickle module:
  - a) Read n dictionary objects and load them into the file
  - b) Read all the dictionary objects from the file and print them
  - c) Counts the number of customers present in the hotel. (Counts the total number of customers present in the hotel.(Assume that file might have few record before adding n records in part (a))
  - d) Display those customers from the file, who have stayed more than 2 days in the hotel.
6. Sun Microsystems held a recruitment test. The file, placement.csv, contains the below format of data: The marks are from 5 different tests conducted and each col is out of 5 marks:
 

SNO	NAME	MARKS1	MARKS2	MARKS3	MARKS4	MARKS5
1	JOHN	4	3	4	2	5
2	PETER	3	4	4	3	5

  - a) Read the above file and print the data.
  - b) Write the UDF to find total no. of people who came for the placement test.
  - c) Write the UDF to find the top n Names on basis of total Marks.
7. Write a program to input a number and then call the functions
  - a) count(n) which returns the number of digits
  - b) reverse(n) which returns the reverse of a number

- c) `hasdigit(n)` which returns True if the number has a digit else False
- d) `show(n)` to show the number in its expanded form (sum of place values of the digits in n) (E.g.  $124 = 100 + 20 + 4$ )

8. A Number is a perfect number if the sum of all the factors of the number (including 1) excluding itself is equal to number. (E.g.  $6 = 1+2+3$  and  $28=1+2+4+7+14$ ). Number is a prime number if its factors are 1 and itself. Write functions:

- a) `Generatefactors()` to populate a list of factors
- b) `isPrimeNo()` to check whether the number is prime number or not
- c) `isPerfectNo()` to check whether the number is perfect number or not

Save the above as a module `perfect.py` and use in the program `main.py` as a menu driven program.

9. Roman numerals are represented by seven different symbols: I, V, X, L, C, D and M.

Roman Numeral	Number
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

For e.g., 2 is written as II in Roman numeral, just two one's added together. 12 is written as XII, which is simply X + II. The number 27 is written as XXVII, which is XX + V + II. Roman numerals are usually written largest to smallest from left to right.

However, the numeral for four is not IIII. Instead, the number four is written as IV. Because the one is before the five, we subtract it making four. The same principle applies to the number nine, which is written as IX.

There are six instances where subtraction is used:

- I can be placed before V (5) and X (10) to make 4 and 9.
- X can be placed before L (50) and C (100) to make 40 and 90.
- C can be placed before D (500) and M (1000) to make 400 and 900.

Write a UDF which takes a string (Roman Numeral) as an argument and returns the integer equivalent.

```
def romanToInt(s):
    return Ans
print(romanToInt('LVIII')) #Should print 58
print(romanToInt('MCMXCIV')) #Should print 1994
```

10. Data can be represented in memory in different ways Binary, Decimal, Octal, and Hexadecimal. Input number in decimal and desired type (Specify B for Binary, O for Octal, H for Hexadecimal) for output. Write a function to perform the conversions:

a)

SAMPLE INPUT: 12 DESIRED TYPE: B RESULT: 1100
---

b)

SAMPLE INPUT: 25 DESIRED TYPE: O RESULT: 31
---

# QUESTION 1

## • CODE:-

```
def create_file_with_content(filename):
    with open(filename, 'w') as file:
        file.write("Neither apple nor pine are in pineapple. Boxing
rings are square.\n")
        file.write("Writers write, but fingers don't fing. Overlook and
oversee are opposites.\n")
        file.write("A house can burn up as it burns down. An alarm goes
off by going on.\n")
```

### # a) Read back the entire file content using read() and display it

```
def read_file_using_read(filename):
    with open(filename, 'r') as file:
        content = file.read()
    print("a) Entire file content:")
    print(content)
```

### # b) Append more text to the file and display the content with line numbers

```
def append_and_display(filename, additional_text):
    with open(filename, 'a') as file:
        file.write(additional_text + '\n')
    with open(filename, 'r') as file:
        content = file.readlines()
    print("\nb) File content with line numbers:")
    for i, line in enumerate(content, start=1):
        print(f"{i}: {line.strip()}")
```

### # c) Display the last line of the file

```
def display_last_line(filename):
    with open(filename, 'r') as file:
        lines = file.readlines()
        last_line = lines[-1]
    print(f"\nc) Last line of the file: {last_line.strip()}")
```

### # d) Display the first line from the 10th character onwards

```
def display_first_line_from_10th_char(filename):
    with open(filename, 'r') as file:
        first_line = file.readline()
        truncated_line = first_line[9:]
    print(f"\nd) First line from the 10th character onwards:
{truncated_line.strip()}")
```

### # e) Read and display a specific line from the file based on user input

```
def read_specific_line(filename, line_num):
    with open(filename, 'r') as file:
        lines = file.readlines()
    if 1 <= line_num <= len(lines):
        print(f"\ne) Line {line_num}: {lines[line_num - 1].strip()}")
    else:
        print("\ne) Invalid line number.")
```

### # f) Find the frequency of words beginning with each letter

```

def word_frequency_by_letter(filename):
    word_counts = {}
    with open(filename, 'r') as file:
        lines = file.readlines()

    for line in lines:
        words = line.split()
        for word in words:
            # Remove punctuation and convert to lowercase
            word = word.strip('.,!?' ).lower()
            if word:
                initial_letter = word[0]
                if initial_letter in word_counts:
                    word_counts[initial_letter] += 1
                else:
                    word_counts[initial_letter] = 1

    print("\nFrequency of words beginning with each letter:")
    for letter, count in sorted(word_counts.items()):
        print(f"Words beginning with {letter}: {count}")

filename = "q1data.txt"
create_file_with_content(filename)
a = "y"
while a == "y":
    op = input("Enter the part of the question you want to do (a-f): ")
    c = op.lower()
    if c == "a":
        read_file_using_read(filename)
    elif c == "b":
        append_and_display(filename, input("Enter text to append: "))
    elif c == "c":
        display_last_line(filename)
    elif c == "d":
        display_first_line_from_10th_char(filename)
    elif c == "e":
        line_num = int(input("Enter the line number to read: "))
        read_specific_line(filename, line_num)
    elif c == "f":
        word_frequency_by_letter(filename)
    else:
        print("Invalid choice.")
    a = input("Do you want to continue? (y/n) ").lower()

```

## • OUTPUT:-

```

Enter which part of the question you want to do (a-f): a
a) Entire file content:
Neither apple nor pine are in pineapple. Boxing rings are square.
Writers write, but fingers don't fing. Overlook and oversee are
opposites.
A house can burn up as it burns down. An alarm goes off by going on.

Do you want to continue? (y/n) y
Enter which part of the question you want to do (a-f): b
Enter text to append: Maths isn't mathing.

```

b) File content with line numbers:

1: Neither apple nor pine are in pineapple. Boxing rings are square.

2: Writers write, but fingers don't fing. Overlook and oversee are opposites.

3: A house can burn up as it burns down. An alarm goes off by going on.

4: Maths isn't mathing.

Do you want to continue? (y/n) y

Enter which part of the question you want to do (a-f): c

c) Last line of the file: Maths isn't mathing.

Do you want to continue? (y/n) y

Enter which part of the question you want to do (a-f): d

d) First line from the 10th character onwards: pple nor pine are in pineapple. Boxing rings are square.

Do you want to continue? (y/n) y

Enter which part of the question you want to do (a-f): e

Enter the line number to read: 2

e) Line 2: Writers write, but fingers don't fing. Overlook and oversee are opposites.

Do you want to continue? (y/n) y

Enter which part of the question you want to do (a-f): f

f) Frequency of words beginning with each letter:

Words beginning with a: 10

Words beginning with b: 5

Words beginning with c: 1

Words beginning with d: 2

Words beginning with f: 2

Words beginning with g: 2

Words beginning with h: 1

Words beginning with i: 2

Words beginning with m: 2

Words beginning with n: 2

Words beginning with o: 5

Words beginning with p: 2

Words beginning with r: 1

Words beginning with s: 1

Words beginning with u: 1

Words beginning with w: 2

Do you want to continue? (y/n) n

q1data - Notepad

File Edit Format View Help

Neither apple nor pine are in pineapple. Boxing rings are square.

Writers write, but fingers don't fing. Overlook and oversee are opposites.

A house can burn up as it burns down. An alarm goes off by going on.

Maths isn't mathing.


# QUESTION 2

- **CODE:-**

```
def isvowel():
    f1=open('file1.txt','r')
    l=f1.readlines()
    f2=open('file2.txt','w')
    vowel='aeiou'
    for i in l:
        s=i.strip("\n")
        n=s.split()
        for j in n:
            if j[0] not in vowel and j[0] not in vowel.upper():
                f2.write(j+" ")
    f1.close()
    f2.close()
    print("Task Successfully Completed!")
isvowel()
```


- **OUTPUT:-**

Task Successfully Completed!

 file1 - Notepad

File Edit Format View Help

Carry Umbrella and Overcoat When it Rains

 file2 - Notepad

File Edit Format View Help

Carry When Rains



# QUESTION 3

## • CODE:-

```
import csv
# (A)
f=open('student.csv','r')
a=csv.reader(f)
l=[]
for i in a:
    t=tuple(i)
    l.append(t)
print("(A)")
print(l)
print()
```

## # (B) (i)

```
f=open('student.csv','r')
a=csv.reader(f)
print("(B) (i)")
for i in a:
    if i[3]<'3':
        print(i[0]+' '+ i[1])
f.close()
print()
```

## # (B) (ii)

```
f=open('student.csv','r')
a=csv.reader(f)
csee=0
bio=0
mme=0
for i in a:
    if i[4]=='CSEE':
        csee+=1
    elif i[4]=='Biology':
        bio+=1
    elif i[4]=='MME':
        mme+=1
    else:
        continue
print("(B) (ii)")
print(f'No. of people in CSEE dept = {csee}')
print(f'No. of people in Biology dept = {bio}')
print(f'No. of people in MME dept = {mme}')
f.close()
```

## • OUTPUT:-

```
(A)
[('Rajat', 'Sen', '12345', '1', 'CSEE'),
 ('Jagat', 'Narain', '13467', '3', 'CSEE'),
 ('Anu', 'Sharma', '11756', '2', 'Biology'),
 ('Sumita', 'Trikha', '23451', '4', 'Biology'),
 ('Sumder', 'Kumra', '11234', '3', 'MME'),
 ('Kanti', 'Bhushan', '23211', '3', 'CSEE')]
```

(B) (i)

Rajat Sen

Anu Sharma

(B) (ii)

No. of people in CSEE dept = 3

No. of people in Biology dept = 2

No. of people in MME dept = 1

# QUESTION 4

## • CODE:-

```
f=open('myfile.txt','r')
r=f.read()
#(A)
a=r.split()
d={}
for i in a:
    if i.lower() not in d:
        d[i.lower()]=1
    else:
        d[i.lower()]+=1
print("(A)")
print(d)
print('(i) Total no. of words is:',len(a))
print('(ii) No. of different words are:',len(d))
x=0
for i in d:
    if d[i]>x:
        x=d[i]
for i in d:
    if d[i]==x:
        print(f'(iii) The most common words in the file is: "{i}"')

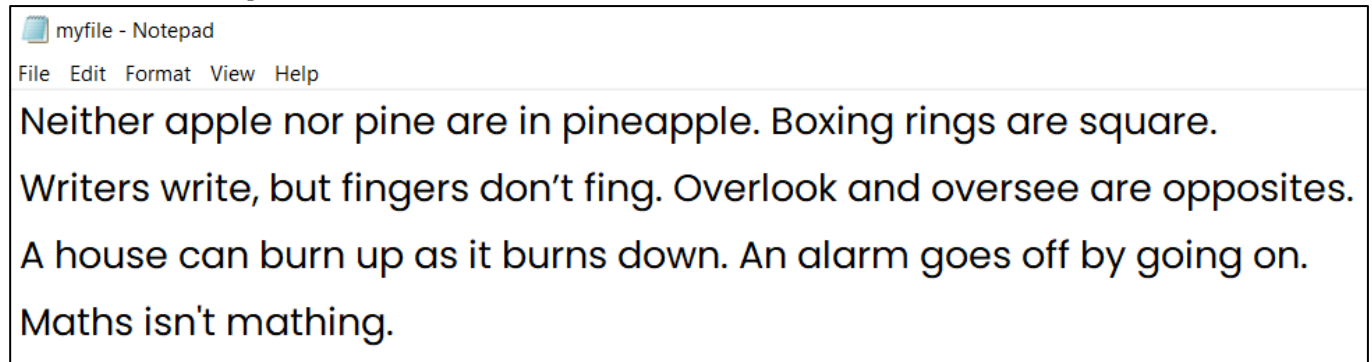
#(B) (i)
d2={}
for i in d:
    if len(i) not in d2:
        d2[len(i)]=[i]
    else:
        d2[len(i)].append(i)
print('\n\n(B) (i)',d2)

#(B) (ii)
def find_longest_word():
    n=0
    for i in d2:
        if i>n:
            n=i
    for i in d2:
        if i==n:
            print(f'(ii) The longest words in the file are {d2[i]}')

#(B) (iii)
def filter_long_words(n):
    l=[]
    for i in d2:
        if i>n:
            l.extend(d2[i])
    print(f'The list of words that are longer than {n} are:',l)

find_longest_word()
m=int(input('(iii) Enter the no. of letters to check for: '))
filter_long_words(m)
```

## • Text File (myfile.txt) :-



myfile - Notepad  
File Edit Format View Help

Neither apple nor pine are in pineapple. Boxing rings are square.  
Writers write, but fingers don't fing. Overlook and oversee are opposites.  
A house can burn up as it burns down. An alarm goes off by going on.  
Maths isn't mathing.

## • OUTPUT :-

(A)

```
{'neither': 1, 'apple': 1, 'nor': 1, 'pine': 1, 'are': 3, 'in': 1, 'pineapple.': 1, 'boxing': 1, 'rings': 1, 'square.': 1, 'writers': 1, 'write,': 1, 'but': 1, 'fingers': 1, 'don't': 1, 'fing.': 1, 'overlook': 1, 'and': 1, 'oversee': 1, 'opposites.': 1, 'a': 1, 'house': 1, 'can': 1, 'burn': 1, 'up': 1, 'as': 1, 'it': 1, 'burns': 1, 'down.': 1, 'an': 1, 'alarm': 1, 'goes': 1, 'off': 1, 'by': 1, 'going': 1, 'on.': 1, 'maths': 1, "isn't": 1, 'mathing.': 1}
```

(i) Total no. of words is: 41

(ii) No. of different words are: 39

(iii) The most common word in the file is: "are"

(B)(i) {7: ['neither', 'square.', 'writers', 'fingers', 'oversee'], 5: ['apple', 'rings', 'don't', 'fing.', 'house', 'burns', 'down.', 'alarm', 'going', 'maths', "isn't"], 3: ['nor', 'are', 'but', 'and', 'can', 'off', 'on.'], 4: ['pine', 'burn', 'goes'], 2: ['in', 'up', 'as', 'it', 'an', 'by'], 10: ['pineapple.', 'opposites.'], 6: ['boxing', 'write,'], 8: ['overlook', 'mathing.'], 1: ['a']}

(ii) The longest words in the file are ['pineapple.', 'opposites.']

(iii) Enter the no. of letters to check for: 4

The list of words that are longer than 4 are: ['neither', 'square.', 'writers', 'fingers', 'oversee', 'apple', 'rings', 'don't', 'fing.', 'house', 'burns', 'down.', 'alarm', 'going', 'maths', "isn't", 'pineapple.', 'opposites.', 'boxing', 'write,', 'overlook', 'mathing.']

# QUESTION 5

## • CODE:-

```
import pickle
# (a)
def write():
    f=open('hotel.dat','ab')
    print("\n(a) ")
    d={}
    d['roomno.']=int(input("Enter your Room Number: "))
    d['name']=input("Enter your name: ")
    d['duration']=int(input("Enter the number of days you stayed: "))
    pickle.dump(d,f)
    f.close()

# (b)
def read():
    f=open("hotel.dat","rb")
    print("\n(b) Details of customers:-")
    while True:
        try:
            d=pickle.load(f)
            print(d)
        except EOFError:
            f.close()
            break

# (c)
def count():
    c=0
    f=open("hotel.dat",'rb')
    while True:
        try:
            pickle.load(f)
            c=c+1
        except EOFError:
            f.close()
            break
    print(f"\n(c) {c}")

# (d)
def customer():
    f=open("hotel.dat","rb")
    print("\n(d) Customers who stayed for more than 2 days are: ")
    while True:
        try:
            d=pickle.load(f)
            if d['duration']>2:
                print(d)
        except EOFError:
            f.close()
            break

def menu():
    while True:
        print("\n1. Enter the details of the customer.")
        print("2. Display the details of the customer.")
        print("3. Count the number of customers.")
```

```

        print("4. Customers who have stayed in the hotel for more than
2 days.")
    print("5. Exit")
    ch=int(input("Enter your choice: "))
    if ch==1:
        write()
    elif ch==2:
        read()
    elif ch==3:
        count()
    elif ch==4:
        customer()
    elif ch==5:
        print("\nProgram Ended Successfully.")
        break
    else:
        print("Invalid Choice!")
menu()

```

## • OUTPUT :-

```

1. Enter the details of the customer.
2. Display the details of the customer.
3. Count the number of customers.
4. Customers who have stayed in the hotel for more than 2 days.
5. Exit
Enter your choice: 1

```

```

(a)
Enter your Room Number: 113
Enter your name: Avinash
Enter the number of days you stayed: 3

```

```

1. Enter the details of the customer.
2. Display the details of the customer.
3. Count the number of customers.
4. Customers who have stayed in the hotel for more than 2 days.
5. Exit
Enter your choice: 1

```

```

(a)
Enter your Room Number: 114
Enter your name: Arvind
Enter the number of days you stayed: 2

```

```

1. Enter the details of the customer.
2. Display the details of the customer.
3. Count the number of customers.
4. Customers who have stayed in the hotel for more than 2 days.
5. Exit
Enter your choice: 2

```

```

(b) Details of customers:-
{'roomno.': 101, 'name': 'Aakar', 'duration': 3}
{'roomno.': 102, 'name': 'Virat', 'duration': 2}
{'roomno.': 103, 'name': 'Cristiano', 'duration': 1}

```

```
{'roomno.': 104, 'name': 'Lionel', 'duration': 1}
{'roomno.': 105, 'name': 'Rohit', 'duration': 1}
{'roomno.': 106, 'name': 'Salman', 'duration': 1}
{'roomno.': 107, 'name': 'Shahrukh', 'duration': 3}
{'roomno.': 108, 'name': 'Georgia', 'duration': 1}
{'roomno.': 109, 'name': 'Mamta', 'duration': 1}
{'roomno.': 110, 'name': 'Neymar', 'duration': 4}
{'roomno.': 111, 'name': 'Amitabh', 'duration': 4}
{'roomno.': 112, 'name': 'Harish Chandra', 'duration': 3}
{'roomno.': 113, 'name': 'Avinash', 'duration': 3}
{'roomno.': 114, 'name': 'Arvind', 'duration': 2}
```

1. Enter the details of the customer.
2. Display the details of the customer.
3. Count the number of customers.
4. Customers who have stayed in the hotel for more than 2 days.
5. Exit

Enter your choice: 3

(c) 14

1. Enter the details of the customer.
2. Display the details of the customer.
3. Count the number of customers.
4. Customers who have stayed in the hotel for more than 2 days.
5. Exit

Enter your choice: 4

(d) Customers who stayed for more than 2 days are:

```
{'roomno.': 101, 'name': 'Aakar', 'duration': 3}
{'roomno.': 107, 'name': 'Shahrukh', 'duration': 3}
{'roomno.': 110, 'name': 'Neymar', 'duration': 4}
{'roomno.': 111, 'name': 'Amitabh', 'duration': 4}
{'roomno.': 112, 'name': 'Harish Chandra', 'duration': 3}
{'roomno.': 113, 'name': 'Avinash', 'duration': 3}
```

1. Enter the details of the customer.
2. Display the details of the customer.
3. Count the number of customers.
4. Customers who have stayed in the hotel for more than 2 days.
5. Exit

Enter your choice: 5

Program Ended Successfully.

# QUESTION 6

## • CODE:-

```
import csv
#(a)
f=open('placement.csv','r')
r=csv.reader(f)
print('(a) The placement.csv contains the following:')
for i in (r):
    if r.line_num==0:
        continue
    else:
        print(i)
#(b)
def count():
    f=open('placement.csv','r')
    r=csv.reader(f)
    for i in (r):
        r.line_num
    print(f'\n(b) Total no. of candidates that came for placement test
are {r.line_num-1}.')
#(c)
def function(k):
    return int(k[7])
def topper():
    f=open('placement.csv','r')
    r=csv.reader(f)
    l=[]
    for i in r:
        if r.line_num==1:
            continue
        else:
            s=int(i[-1])+int(i[-2])+int(i[-3])+int(i[-4])+int(i[-5])
            i.append(s)
            l.append(i)
    l.sort(key=function,reverse=True)
    n=int(input('\n(c) Enter the no. of top candidates required: '))
    print('Top',n,'candidates are: ')
    for i in range(n):
        print(f'{i+1}. {l[i]}')
count()
topper()
```

## • OUTPUT:-

```
(a) The placement.csv contains the following:
['1', 'JOHN', '4', '3', '4', '2', '5']
['2', 'PETER', '3', '2', '4', '3', '5']
['3', 'SAM', '2', '4', '3', '5', '3']
['4', 'TRACY', '4', '5', '1', '2', '4']
['5', 'ALEX', '5', '3', '2', '4', '5']
```

(b) Total no. of candidates that came for placement test are 4.

(c) Enter the no. of top candidates required: 3

Top 3 candidates are:

1. ['5', 'ALEX', '5', '3', '2', '4', '5', 19]
2. ['2', 'PETER', '3', '2', '4', '3', '5', 17]
3. ['3', 'SAM', '2', '4', '3', '5', '3', 17]



# QUESTION 7

## • CODE:-

### # (a)

```
def count(n):  
    return len(str(n))
```

### # (b)

```
def reverse(n):  
    return int(str(n)[::-1])
```

### # (c)

```
def hasdigit(n):  
    return any(c.isdigit() for c in str(n))
```

### # (d)

```
def show(n):  
    digits = [int(d) for d in str(n)]  
    expanded_form = ' + '.join([f"{digits[i]} * 10^{len(digits)-i-1}"  
    for i in range(len(digits))])  
    return expanded_form
```

```
# Main program
```

```
try:
```

```
    num = int(input("Enter a number: "))  
    print("Number of digits:", count(num))  
    print("Reverse of the number:", reverse(num))  
    print("Has digit:", hasdigit(num))  
    print("Expanded form:", show(num))
```

```
except ValueError:
```

```
    print("Invalid input. Please enter a valid number.")
```

## • OUTPUT:-

```
Enter a number: 1234
```

```
Number of digits: 4
```

```
Reverse of the number: 4321
```

```
Has digit: True
```

```
Expanded form: 1 * 10^3 + 2 * 10^2 + 3 * 10^1 + 4 * 10^0
```

# QUESTION 8

## • CODE:-

#(a)

```
def generate_factors(n):  
    factors = []  
    for i in range(1, n):  
        if n % i == 0:  
            factors.append(i)  
    return factors
```

#(b)

```
def is_prime_number(n):  
    if n <= 1:  
        return False  
    for i in range(2, int(n**0.5) + 1):  
        if n % i == 0:  
            return False  
    return True
```

#(c)

```
def is_perfect_number(n):  
    factors = generate_factors(n)  
    factor_sum = sum(factors)  
    return factor_sum == n  
# Main program  
try:  
    num = int(input("Enter a number: "))  
    factors = generate_factors(num)  
    if is_prime_number(num):  
        print(num, "is a prime number.")  
    else:  
        print(num, "is not a prime number.")  
        if is_perfect_number(num):  
            print(num, "is a perfect number.")  
        else:  
            print(num, "is not a perfect number.")  
    print("Factors:", factors)  
except ValueError:  
    print("Invalid input. Please enter a valid number.")
```

## • OUTPUT:-

```
Enter a number: 1928939  
1928939 is not a prime number.  
1928939 is not a perfect number.  
Factors: [1, 17, 113467]
```

# QUESTION 9

## • CODE:-

```
def romanToInt(s):
    roman_values = {
        'I':1,
        'V':5,
        'X':10,
        'L':50,
        'C':100,
        'D':500,
        'M':1000
    }
    total = 0
    prev_value = 0
    for char in s[::-1]:
        current_value = roman_values[char]
        if current_value < prev_value:
            total -= current_value
        else:
            total += current_value
            prev_value = current_value
    return total
s = input('Enter the roman numeral: ')
print(f'The Hindu-Arabic numeral for {s} is {romanToInt(s)}.')
```

## • OUTPUT:-

```
Enter the roman numeral: LVIII
The Hindu-Arabic numeral for LVIII is 58.
```

```
Enter the roman numeral: MCMXCIV
The Hindu-Arabic numeral for MCMXCIV is 1994.
```

# QUESTION 10

## • CODE:-

```
def binaryconversion(number):  
    binary = bin(number)  
    print(number, 'in binary system =', binary[2:])  
  
def octalconversion(number):  
    octal = oct(number)  
    print(number, 'in octal system =', octal[2:])  
  
def hexadecimalconversion(number):  
    hexadecimal = hex(number)  
    print(number, 'in hexadecimal system =', hexadecimal[2:])  
  
num = int(input('Enter a number: '))  
op = input('Enter the desired type conversion (B/O/H): ')  
if op == 'b' or op == 'B':  
    binaryconversion(num)  
elif op == 'o' or op == 'O':  
    octalconversion(num)  
elif op == 'h' or op == 'H':  
    hexadecimalconversion(num)  
else:  
    print('Invalid type conversion.')  
    exit
```

## • OUTPUT:-

```
Enter a number: 12  
Enter the desired type conversion (B/O/H): B  
12 in binary system = 1100  
  
Enter a number: 25  
Enter the desired type conversion (B/O/H): O  
25 in octal system = 31
```