Question 1:- a) Python Program to Perform All Arithmetic Operations on Two Given Numbers

CODE:-

```
a=int(input("Enter the    first number:"))
b=int(input("Enter the second number:"))
print(a+b)
print(a-b)
print(a*b)
print(a/b)
```

OUTPUT:-

```
Enter the  first number:20
Enter the second number:45
65
-25
900
0.4444444444444444
```

b) Python Program to Perform All Arithmetic Operations on user defined input numbers

CODE:-

```
n1=int(input("Enter the first    number:"))
n2=int(input("Enter the second number:"))
print("Addition of two numbers:",n1 + n2)
print("Subtraction of two numbers:",n1 - n2)
print("Multiplication of two numbers:",n1 * n2)
print("Division of two numbers:",n1 / n2)
```

OUTPUT:-

```
Enter the first  number:30
Enter the second number:40
Addition of two numbers: 70
Subtraction of two numbers: -10
Multiplication of two numbers: 1200
Division of two numbers: 0.75
```

Question 2:-

a) **Program to show how escape sequence characters works**

CODE:-

```
txt1="1. HAVE      A    NICE    DAY \a" #\a Alarm or Beep
print(txt1)
txt2="2. HELLO \b WORLD "#\b     Backspace
print(txt2)
txt3="3. PRIYAM \nJHA" #\n          New Line
print(txt3)
txt4="4. THIS    IS    \rBCIIT"#\r      Carriage Return
print(txt4)
txt5="5. PYTHON\tPROGRAMMING\tLANGUAGE" #\t  Horizontal Tab
print(txt5)
txt6="6. PYTHON \\ PRACTICAL \\    FILE" #\\     Backlash
print(txt6)
txt7="7. IT\'S    A    CASE    SENSITIVE    LANGUAGE" #\'       Single
Quote
print(txt7)
txt8= "8. \110\145\154\154\157" #A backslash followed by three integers will
result in a octal value:
print(txt8)
txt9 = "9. \x48\x65\x6c\x6c\x6f"#A backslash followed by an 'x' and a hex
number represents a hex value:
print(txt9)
```

OUTPUT:-

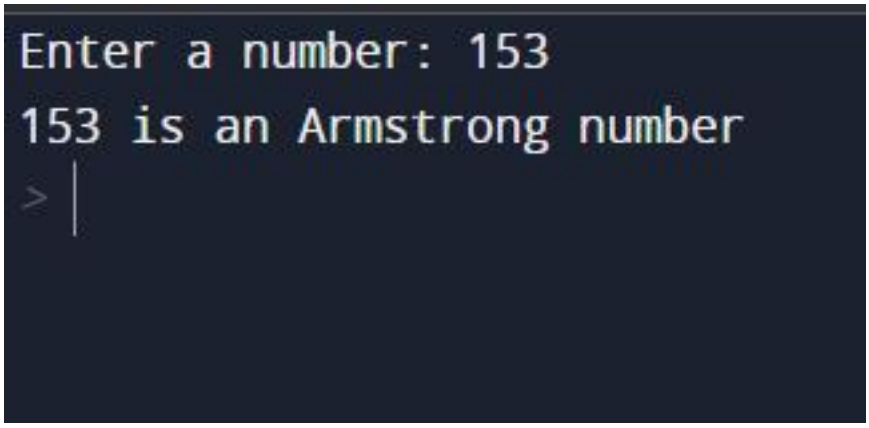
```
1. HAVE    A   NICE   DAY
2. HELLO   WORLD
3. PRIYAM
JHA
4. THIS   IS
BCIIT
5. PYTHON    PROGRAMMING LANGUAGE
6. PYTHON \ PRACTICAL \   FILE
7. IT'S   A   CASE   SENSITIVE   LANGUAGE
8. Hello
9. Hello
```

b) Python program to check if the number is an Armstrong number or not

CODE:-

```
num = int(input("Enter a number: "))
sum = 0
temp = num
while temp > 0:
    digit = temp % 10
    sum += digit ** 3
    temp //= 10
if num == sum:
    print(num,"is an Armstrong number")
else:
        print(num,"is not an Armstrong number")
```

OUTPUT:-



Question 3:-

a) Write a program to print the sum of all the primes between two ranges.

CODE:-

```
lower_value = int(input (" Enter the Lowest Range Value: "))
upper_value = int(input (" Enter the Upper Range Value: "))

print ("The Prime Numbers in the range are: ")
for number in range (lower_value, upper_value + 1):
    if number > 1:
        for i in range (2, number):
            if (number % i) == 0:
                break
        else:
                print (number)
```

OUTPUT:-

```
Enter the Lowest Range Value: 40
Enter the Upper Range Value: 50
The Prime Numbers in the range are:
41
43
47
```

b) Solve in python program: 100 + 200 / 10 - 3 * 10

CODE:-

```
Expression = 100 + 200 / 10 - 3 * 10

print(Expression)
```

OUTPUT:-

```
90.0
>
```

Question 4:-

a.) **Write a program to swap two strings.**

CODE:-

```
A = str(input("Enter the first string:"))
B = str(input("Enter the second string:"))

print("\n String before swap:")
print("A=", A)
print("B=", B)

temp=A
A = B
B = temp

print("\n String after swap:")
print("A=", A)
print("B=", B)
```

OUTPUT:-

```
Enter the first string:Priyam
Enter the second string:Jha
String before swap:
A= Priyam
B= Jha

 String after swap:
A= Jha
B= Priyam
```

b) Program to explain precedence of arithmetic operators

CODE:-

```
def precedence():
    expression1 = 5 + 3 * 2
    expression2 = (5 + 3) * 2
    expression3 = 10 / 2 + 3

    print("Expression 1: 5 + 3 * 2=",expression1)
    print("Expression 2: (5 + 3) * 2=", expression2)
    print("Expression 3: 10 / 2 + 3=", expression3)

precedence()
```

OUTPUT:-

```
Expression 1: 5 + 3 * 2= 11
Expression 2: (5 + 3) * 2= 16
Expression 3: 10 / 2 + 3= 8.0
>
```

Question 5:-

a.) Write a menu driven program to accept two strings from the user and perform the various functions using user defined functions.

CODE:-

```
def concatenate_strings(str1, str2):
    return str1 + str2

def find_length(string):
    return len(string)

def convert_to_uppercase(string):
    return string.upper()

def reverse_string(string):
    return string[::-1]

while True:
    print("Menu:")
    print("1. Concatenate Strings")
    print("2. Find Length of a String")
    print("3. Convert to Uppercase")
    print("4. Reverse String")
    print("5. Exit")

    choice = input("Enter your choice (1-5): ")

    if choice == '5':
        print("Exiting the program. Goodbye!")
        break

    if choice not in ['1', '2', '3', '4']:
        print("Invalid choice. Please enter a number between 1 and 5.")
        continue

    if choice == '1':
        str1 = input("Enter the first string: ")
        str2 = input("Enter the second string: ")
        result = concatenate_strings(str1, str2)
        print("Concatenated String:", result)
    elif choice == '2':
        string = input("Enter the string: ")
        result = find_length(string)
        print("Length of String:", result)
    elif choice == '3':
        string = input("Enter the string: ")
        result = convert_to_uppercase(string)
        print("Uppercase String:", result)
    elif choice == '4':
        string = input("Enter the string: ")
        result = reverse_string(string)
  print("Reversed String:", result)
```

OUTPUT:-

```
Menu:
1. Concatenate Strings
2. Find Length of a String
3. Convert to Uppercase
4. Reverse String
5. Exit
Enter your choice (1-5): 1
Enter the first string: priyam
Enter the second string: jha
Concatenated String: priyamjha
Menu:
1. Concatenate Strings
2. Find Length of a String
3. Convert to Uppercase
4. Reverse String
5. Exit
Enter your choice (1-5): 2
Enter the string: priyam
Length of String: 6
```

```
Menu:
1. Concatenate Strings
2. Find Length of a String
3. Convert to Uppercase
4. Reverse String
5. Exit
Enter your choice (1-5): 3
Enter the string: priyam
Uppercase String: PRIYAM
Menu:
1. Concatenate Strings
2. Find Length of a String
3. Convert to Uppercase
4. Reverse String
5. Exit
Enter your choice (1-5): 4
Enter the string: priyam
Reversed String: mayirp
Menu:
1. Concatenate Strings
2. Find Length of a String
3. Convert to Uppercase
4. Reverse String
5. Exit
Enter your choice (1-5): 5
Exiting the program. Goodbye!
```

b.) Changing the case of Python Strings.

CODE:-

```
def change_case(input_string):
    lowercase_string = input_string.lower()
    uppercase_string = input_string.upper()
    titlecase_string = input_string.title()

    print(f"Original String: {input_string}")
    print(f"Lowercase: {lowercase_string}")
    print(f"Uppercase: {uppercase_string}")
    print(f"Title Case: {titlecase_string}")

user_input = input("Enter a string: ")
change_case(user_input)
```

Output:-

```
Enter a string: pRiYaM
Original String: pRiYaM
Lowercase: priyam
Uppercase: PRIYAM
Title Case: Priyam
```

Question 6:- Write a program to find

a. Sum of Digits of a number
b. Product of digit.

CODE:-

```
def getSum(n):
    sum = 0
    for digit in str(n):
        sum += int(digit)
    return sum

def getProduct(n):
    product = 1
    for digit in str(n):
        product *= int(digit)
    return product

n = int(input("Enter the number: "))
print("Sum of digits of a number:", getSum(n))
print("Product of digits:", getProduct(n))
```

OUTPUT:-

```
Enter the number: 20
Sum of digits of a number: 2
Product of digits: 0
>
```

Question 7:- Write a program for:

A. MULTIPLICATION

CODE:-

```
def multiplication(a, b):
    result = a * b
    return result
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
result = multiplication(num1, num2)
print(f"The result of multiplication is: {result}")
```

OUTPUT:-

```
Enter the first number: 20
Enter the second number: 34
The result of multiplication is: 680.0
```

B. FACTORIAL

CODE:-

```
num = int(input("Enter a number: "))
factorial = 1
if num < 0:
    print(" Factorial does not exist for negative numbers")
elif num == 0:
    print("The factorial of 0 is 1")
else:
    for i in range(1,num + 1):
        factorial = factorial*i
    print("The factorial of",num,"is",factorial)
```

OUTPUT:-

```
Enter a number: 4
The factorial of 4 is 24
```

Question 8:- a) Write a program to find smallest and largest number in a list .

CODE:-

```
lst = []
num = int(input('Total number of elements: '))
for n in range(num):
    numbers = int(input('Enter number '))
    lst.append(numbers)
print("Largest element in the list is :", max(lst))
print("Smallest element in the list is :", min(lst))
```

OUTPUT:-

```
Total number of elements: 4
Enter number 56
Enter number 78
Enter number 90
Enter number 98
Largest element in the list is : 98
Smallest element in the list is : 56
```

b) Write a program for List slicing and correcting mistakes values in a list
CODE:-

```
def correct_mistakes(lst):
    sublist = lst[1:4]
    corrected_sublist = [value * 2 for value in sublist]
    lst[1:4] = corrected_sublist
    return lst
my_list = [1, 10, 3, 5, 7, 9]
print("Original List:", my_list)

updated_list = correct_mistakes(my_list)

print("Updated List:", updated_list)
```

OUTPUT:-

```
Original List: [1, 10, 3, 5, 7, 9]
Updated List: [1, 20, 6, 10, 7, 9]
>
```

c) Make a list with each item being increasing power of 2.

CODE:-

```
print("Enter the Total Number of Terms:")
num = int(input())

for i in range(num):
    print("2 raised to the power ", i, " is ", 2 ** i)
```

OUTPUT:-

```
Enter the Total Number of Terms:
5
2 raised to the power  0  is  1
2 raised to the power  1  is  2
2 raised to the power  2  is  4
2 raised to the power  3  is  8
2 raised to the power  4  is  16
>
```

Question 9:- Create a dictionary whose keys are month names and whose values are the number of days in the corresponding months.

- **Ask the user to enter a month name and use the dictionary to tell them how many days are in the month.**
- **Print out all keys in the alphabetically order**
- **Print out all the months with 31 days**
- **Print out the key value pairs sorted by number of days in each month**

CODE:-

```
month = { "jan" : 31 , "feb" : 28 , "march" : 31 , "april" : 30 ,
"may" : 31 , "june" : 30 , "july" : 31 , "aug" : 31 , "sept" : 30 ,
"oct" : 31 , "nov" : 30 , "dec" : 31}

mon = input("Enter the month name in short form :- ")
print("Number of days in ",mon,"=",month [ mon ])

lst = list ( month . keys() )
lst.sort()
print( lst )

print( "Month which have 31 days !!-- ")
for i in month :
    if month [ i ] == 31 :
        print( i )

print("Month according to number of days ---")
print("feb")
for i in month :
    if month [ i ] == 30 :
        print(i)
for i in month :
    if month [ i ] == 31 :
        print( i)
```

OUTPUT:-

```
Enter the month name in short form :- aug
Number of days in  aug = 31
['april', 'aug', 'dec', 'feb', 'jan', 'july', 'june', 'march', 'may', 'nov', 'oct', 'sept']
Month which have 31 days !!--
jan
march
may
july
aug
oct
dec
Month according to number of days ---
feb
april
june
sept
nov
jan
march
may
july
aug
oct
dec
```

Question 10:- Write a program that scans an email address and forms a tuple of user name and domain.

CODE:-

```
email_address = "priyam_jha@google.com"


username, domain = email_address.split("@")


print("Username:", username)

print("Domain:", domain)
```

OUTPUT:-

```
Username: priyam_jha
Domain: google.com
>
```

Question 11:- Write a program that defines a function large in a module which will be used to find larger of two values and called from code in another module .

CODE:-

```
def large(a, b):
    """Find and return the larger of two values."""
    return a if a > b else b

# Get user input or use any values of your choice
value1 = int(input("Enter the first value: "))
value2 = int(input("Enter the second value: "))

# Call the function
result = large(value1, value2)

# Display the result
print("The larger value is:", result)
```

OUTPUT:-

```
Enter the first value: 10
Enter the second value: 20
The larger value is: 20
>
```

Question 12:- Write a program:

a. To find numbers divisible by 7 and are not multiple of 5. In range of 2000-3200 inclusive.

CODE:-

```
result_list = []
for num in range(2000, 3201):
    if (num % 7 == 0 and num % 5 != 0):
        result_list.append(num)
print("Numbers divisible by 7 and not multiples of 5 in the range 2000-3200:")
print(result_list)
```

OUTPUT:-

```
Numbers divisible by 7 and not multiples of 5 in the range 2000-3200:
[2002, 2009, 2016, 2023, 2037, 2044, 2051, 2058, 2072, 2079, 2086, 2093, 2107, 2114,
    2121, 2128, 2142, 2149, 2156, 2163, 2177, 2184, 2191, 2198, 2212, 2219, 2226, 2233
    , 2247, 2254, 2261, 2268, 2282, 2289, 2296, 2303, 2317, 2324, 2331, 2338, 2352,
    2359, 2366, 2373, 2387, 2394, 2401, 2408, 2422, 2429, 2436, 2443, 2457, 2464, 2471
    , 2478, 2492, 2499, 2506, 2513, 2527, 2534, 2541, 2548, 2562, 2569, 2576, 2583,
    2597, 2604, 2611, 2618, 2632, 2639, 2646, 2653, 2667, 2674, 2681, 2688, 2702, 2709
    , 2716, 2723, 2737, 2744, 2751, 2758, 2772, 2779, 2786, 2793, 2807, 2814, 2821,
    2828, 2842, 2849, 2856, 2863, 2877, 2884, 2891, 2898, 2912, 2919, 2926, 2933, 2947
    , 2954, 2961, 2968, 2982, 2989, 2996, 3003, 3017, 3024, 3031, 3038, 3052, 3059,
    3066, 3073, 3087, 3094, 3101, 3108, 3122, 3129, 3136, 3143, 3157, 3164, 3171, 3178
    , 3192, 3199]
```

b. Which will have a list of values and then input a number to check if the value exist in the list or not.

CODE:-

```
def check_number_in_list(number, my_list):
    """Check if a number exists in the given list."""
    return number in my_list

my_list = [10, 20, 30, 40, 50]
input_number = int(input("Enter a number to check if it exists in the list: "))

if check_number_in_list(input_number, my_list):
    print(f"{input_number} exists in the list.")
else:
    print(f"{input_number} does not exist in the list.")
```

OUTPUT:-

```
Enter a number to check if it exists in the list: 30
30 exists in the list.
```

Question 13:- Write a program to:

a. Input 5 numbers in the list and print in ascending order.

b. Insert a value in the list at particular position.

c. Count elements in the list until the occurrence of the first tuple in the list.

CODE:-

```
num_list = []
for i in range(5):
    num = int(input("Enter number " + str(i + 1) + ": "))
    num_list.append(num)
print("Ascending order:", sorted(num_list))
position = int(input("Enter the position to insert the value: "))
value = int(input("Enter the value to insert: "))
num_list.insert(position, value)
print("List after insertion:", num_list)
for i, element in enumerate(num_list):
    if isinstance(element, tuple):
        print("Count of elements before first tuple:", i)
        break
```

OUTPUT:-

```
Enter number 1: 89
Enter number 2: 56
Enter number 3: 45
Enter number 4: 67
Enter number 5: 89
Ascending order: [45, 56, 67, 89, 89]
Enter the position to insert the value: 2
Enter the value to insert: 1
List after insertion: [89, 56, 1, 45, 67, 89]
>
```

Question 14:- Write a Python program to calculate the average value of the numbers in a given tuple of tuples.

CODE:-

```
def average_tuple(nums):
    result =[sum(x)/len(x)for x in zip(*nums)]
    return result
nums = ((1,2,3,3),(20,30,56,67),(90,50,40,20),(3,7,8,9))
print("Original Tuple:")
print(nums)
print("\nAverage value of the numbers of the said tuple of tuples:\n",average_tuple(nums))
nums = ((1,1,-5),(20,-14,54),(32,-21,-33),(-10,3,5))
print("\nOriginal Tuple:")
print(nums)
print("\nAverage value of the numbers of the said tuple of tuples:\n",average_tuple(nums))
```

OUTPUT:-

```
Original Tuple:
((1, 2, 3, 3), (20, 30, 56, 67), (90, 50, 40, 20), (3, 7, 8, 9))

Average value of the numbers of the said tuple of tuples:
[28.5, 22.25, 26.75, 24.75]

Original Tuple:
((1, 1, -5), (20, -14, 54), (32, -21, -33), (-10, 3, 5))

Average value of the numbers of the said tuple of tuples:
 [10.75, -7.75, 5.25]
>
```

Question 15:- Make a simple chatbot using python.

CODE:-

```
def simple_chatbot():
    print("Simple Chatbot: Hello! How can I assist you today?")
    while True:
        user_input = input("You: ").lower()
        if "hello" in user_input:
            print("Simple Chatbot: Hi there! How can I help?")
        elif "how are you" in user_input:
            print("Simple Chatbot: I'm just a computer program, but I'm doing well. Thanks for asking!")
        elif "bye" in user_input or "exit" in user_input:
            print("Simple Chatbot: Goodbye! Have a great day!")
            break
        else:
            print("Simple Chatbot: I'm sorry, I didn't understand that. Can you please rephrase or ask another question?")
if __name__ == "__main__":
  simple_chatbot()
```

OUTPUT:-

```
Simple Chatbot: Hello! How can I assist you today?
You: hello
Simple Chatbot: Hi there! How can I help?
You: how are you?
Simple Chatbot: I'm just a computer program, but I'm doing well. Thanks for asking!
You: what is python?
Simple Chatbot: I'm sorry, I didn't understand that. Can you please rephrase or ask
    another question?
You: Okay no problem
Simple Chatbot: I'm sorry, I didn't understand that. Can you please rephrase or ask
    another question?
You: bye
Simple Chatbot: Goodbye! Have a great day!
```

Question 16 :-

a) convert PDF file to Excel file using Python
b) program to reverse the content of a file and store it in another file
c) How to create a duplicate file of an existing file using Python?

CODE:-

```
import tabula


def pdf_to_excel(pdf_path, excel_path):
    # Convert PDF to DataFrame
    df = tabula.read_pdf(pdf_path, pages='all')


    # Write DataFrame to Excel
    df.to_excel(excel_path, index=False)


def reverse_file(input_file, output_file):
    with open(input_file, 'r') as f:
        content = f.read()


    reversed_content = content[::-1]


    with open(output_file, 'w') as f:
        f.write(reversed_content)


def duplicate_file(input_file, output_file):
    with open(input_file, 'rb') as source_file:
        with open(output_file, 'wb') as duplicate_file:
            duplicate_file.write(source_file.read())


# Example usage
pdf_input = 'input.pdf'
excel_output = 'output.xlsx'
reverse_input = 'input.txt'
```

```
reverse_output = 'output_reversed.txt'
duplicate_input = 'input.txt'
duplicate_output = 'output_duplicate.txt'

pdf_to_excel(pdf_input, excel_output)
reverse_file(reverse_input, reverse_output)
duplicate_file(duplicate_input, duplicate_output)

print("Tasks completed successfully!")
```

Question 17:- Create a binary file with roll number, name and marks. Input a roll number and perform the following operations:

- **update the marks.**
- **Delete the record**
- **Display the record**
- **Append the record**
- **Search the record**

CODE:-

```
import pickle

# Function to create a new file with initial student records
def create_file():
    records = {
        101: {'name': 'Alice', 'marks': 85.5},
        102: {'name': 'Bob', 'marks': 75.0},
        103: {'name': 'Charlie', 'marks': 90.2}
    }
    with open('student_records.bin', 'wb') as file:
        pickle.dump(records, file)

# Function to update marks for a given roll number
def update_marks(roll, new_marks):
    try:
        with open('student_records.bin', 'rb+') as file:
            records = pickle.load(file)
            if roll in records:
                records[roll]['marks'] = new_marks
                file.seek(0)
                pickle.dump(records, file)
                print(f"Updated marks for Roll Number {roll}")
            else:
                print(f"Roll Number {roll} not found")
    except EOFError:
```

```
        pass

# Function to delete a record based on roll number
def delete_record(roll):
    try:
        with open('student_records.bin', 'rb+') as file:
            records = pickle.load(file)
            if roll in records:
                del records[roll]
                file.seek(0)
                pickle.dump(records, file)
                print(f"Deleted record for Roll Number {roll}")
            else:
                print(f"Roll Number {roll} not found")
    except EOFError:
        pass

# Function to display all records
def display_records():
    try:
        with open('student_records.bin', 'rb') as file:
            records = pickle.load(file)
            print("Student Records:")
            for roll, details in records.items():
                print(f"Roll Number: {roll}, Name: {details['name']}, Marks:
{details['marks']}")
    except EOFError:
        pass

# Function to append a new record
def append_record(roll, name, marks):
    try:
        with open('student_records.bin', 'rb+') as file:
```

```
            records = pickle.load(file)
            if roll not in records:
                records[roll] = {'name': name, 'marks': marks}
                file.seek(0)
                pickle.dump(records, file)
                print(f"Appended record for Roll Number {roll}")
            else:
                print(f"Roll Number {roll} already exists")
    except (FileNotFoundError, EOFError):
        create_file()
        append_record(roll, name, marks)

# Function to search for a record based on roll number
def search_record(roll):
    try:
        with open('student_records.bin', 'rb') as file:
            records = pickle.load(file)
            if roll in records:
                print(f"Roll Number: {roll}, Name: {records[roll]['name']}, Marks: {records[roll]['marks']}")
            else:
                print(f"Roll Number {roll} not found")
    except EOFError:
        pass

# User input and operations
while True:
    print("\nChoose an operation:")
    print("1. Update marks")
    print("2. Delete record")
    print("3. Display all records")
    print("4. Append a new record")
    print("5. Search for a record")
```

```
print("6. Exit")

choice = input("Enter your choice (1-6): ")

if choice == '1':
    roll = int(input("Enter Roll Number to update marks: "))
    new_marks = float(input("Enter new marks: "))
    update_marks(roll, new_marks)
elif choice == '2':
    roll = int(input("Enter Roll Number to delete record: "))
    delete_record(roll)
elif choice == '3':
    display_records()
elif choice == '4':
    roll = int(input("Enter Roll Number to append: "))
    name = input("Enter Name: ")
    marks = float(input("Enter Marks: "))
    append_record(roll, name, marks)
elif choice == '5':
    roll = int(input("Enter Roll Number to search: "))
    search_record(roll)
elif choice == '6':
    break
else:
    print("Invalid choice. Please enter a number between 1 and 6.")
```

OUTPUT:-

```
Choose an operation:
1. Update marks
2. Delete record
3. Display all records
4. Append a new record
5. Search for a record
6. Exit
Enter your choice (1-6): 4
Enter Roll Number to append: 3
Enter Name: ABC
Enter Marks: 89
Appended record for Roll Number 3

Choose an operation:
1. Update marks
2. Delete record
3. Display all records
4. Append a new record
5. Search for a record
6. Exit
Enter your choice (1-6): 6
>
```

Question 18:- Write a program to Create a CSV file by entering user-id and password, read and search the password for given user id.

CODE:-

```
import csv

with open("user_info.csv","w")as obj:
    fileobj = csv.writer(obj)
    fileobj.writerow(["User ID","Password"])

    while(True):
        user_id = input("Enter ID:")
        password = input("Enter Password:")
        record = [user_id,password]
        fileobj.writerow(record)

        x = input("Press Y/y to continue and N/n to terminate the program\n")
        if x in "Nn":
            break
        elif x in "Yy":
            continue
with open("user_info.csv","r")as obj2:
    fileobj2 = csv.reader(obj2)
    given = input("Enter the user id to be searched\n")
    for i in fileobj2:
        next(fileobj2)
        if i[0]==given:
            print(i[1])
            break
```

OUTPUT:-

```
Enter ID:1
Enter Password:2345
Press Y/y to continue and N/n to terminate the program
Y/y
Enter ID:2
Enter Password:59483
Press Y/y to continue and N/n to terminate the program
N/n
Enter ID:3
Enter Password:56743
Press Y/y to continue and N/n to terminate the program
N
Enter the user id to be searched
2
59483
```

Question 19:- Write a program to show joining of NumPy arrays .
CODE:-

```
import numpy as np

array1 = np.array([[1, 2], [3, 4]])
array2 = np.array([[5, 6]])

joined_array = np.concatenate((array1, array2), axis=0)

print("Joined Array using concatenate:")
print(joined_array)
print()

vertically_stacked = np.vstack((array1, array2))

print("Vertically Stacked Array using vstack:")
print(vertically_stacked)
print()

horizontally_stacked = np.hstack((array1, array2.T))

print("Horizontally Stacked Array using hstack:")
print(horizontally_stacked)
```

OUTPUT:-

```
Joined Array using concatenate:
[[1 2]
 [3 4]
 [5 6]]

Vertically Stacked Array using vstack:
[[1 2]
 [3 4]
 [5 6]]

Horizontally Stacked Array using hstack:
[[1 2 5]
 [3 4 6]]
>
```

Question 20:- Write a program:
a. Write a program to demonstrate subplots and multiple plots in matplotlib.

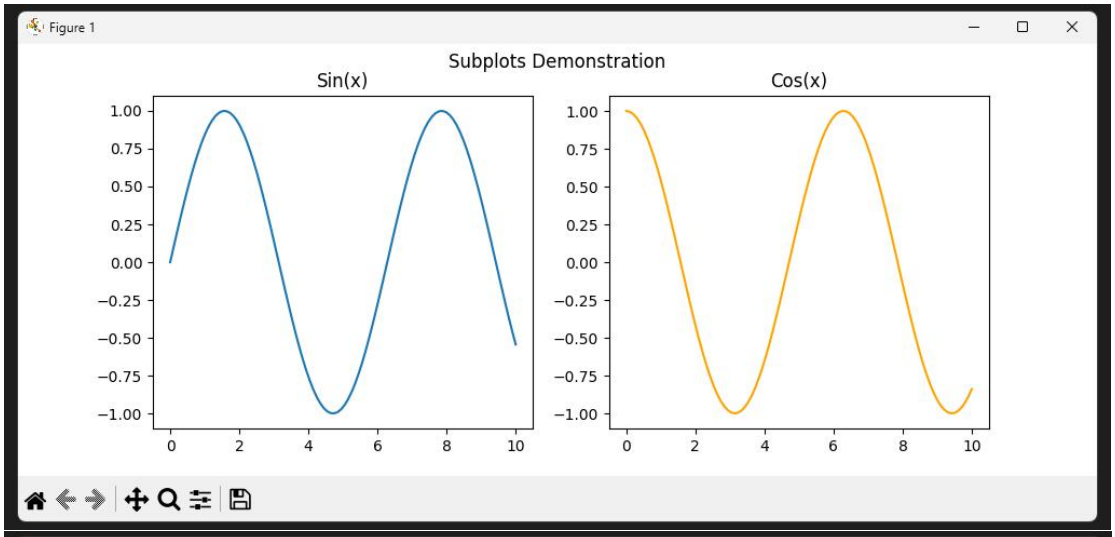
b. Demonstrate pie chart and Bar Graphs in matplotlib

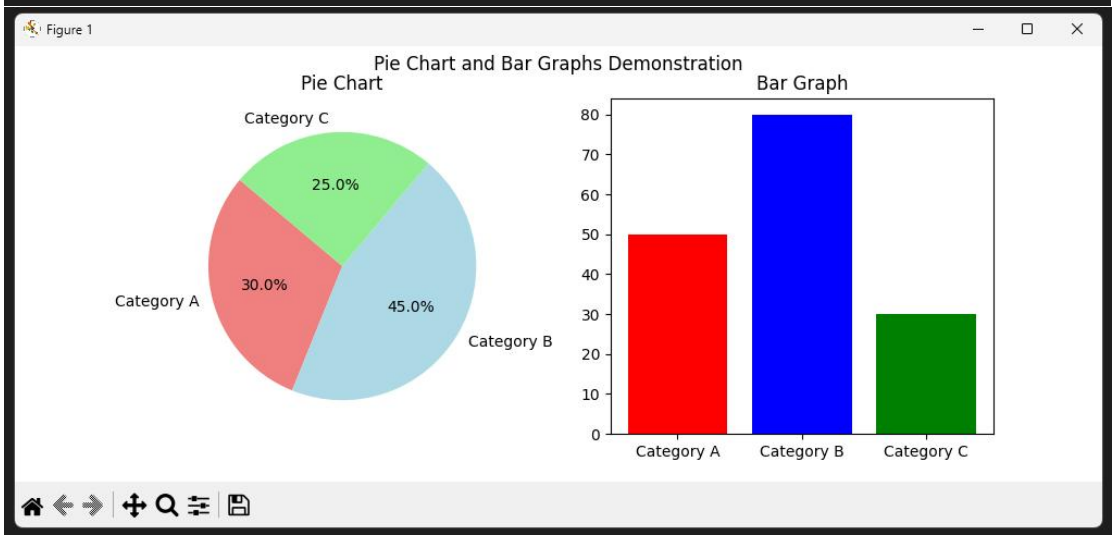
CODE:-

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

# Part a: Subplots and Multiple Plots
x = np.linspace(0, 10, 100)
y1 = np.sin(x)
y2 = np.cos(x)

# Create subplots
plt.figure(figsize=(10, 4))

# Subplot 1
plt.subplot(1, 2, 1)
plt.plot(x, y1, label='sin(x)')
plt.title('Sin(x)')

# Subplot 2
plt.subplot(1, 2, 2)
plt.plot(x, y2, label='cos(x)', color='orange')
plt.title('Cos(x)')

plt.suptitle('Subplots Demonstration')
plt.show()

# Part b: Pie Chart and Bar Graphs
labels = ['Category A', 'Category B', 'Category C']
sizes = [30, 45, 25]
colors = ['lightcoral', 'lightblue', 'lightgreen']

# Pie chart
plt.figure(figsize=(10, 4))

plt.subplot(1, 2, 1)
plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%', startangle=140)
plt.title('Pie Chart')

# Bar graph
categories = ['Category A', 'Category B', 'Category C']
values = [50, 80, 30]

plt.subplot(1, 2, 2)
plt.bar(categories, values, color=['red', 'blue', 'green'])
plt.title('Bar Graph')
```

```
plt.suptitle('Pie Chart and Bar Graphs Demonstration')
plt.show()
```

OUTPUT:-





Question 21:- Demonstrate the following functions/methods which operates on dictionary in Python with suitable examples:

i) dict( ) ii) len( ) iii) clear( ) iv) get( ) v) pop( ) vi)pop item( ) vii) keys( ) viii) values() ix) items( ) x) copy( ) xi) update( )

CODE:-

```
# Create a sample dictionary

my_dict = {'name': 'Priyam', 'age': 20, 'city': 'Delhi'}


# i) dict( )

new_dict = dict([('country', 'INDIA'), ('gender', 'Male')])

print("i) dict( ):")

print("New Dictionary:", new_dict)

print()


# ii) len( )

print("ii) len( ):")

print("Length of the dictionary:", len(my_dict))

print()


# iii) clear( )

print("iii) clear( ):")

my_dict.clear()

print("Cleared Dictionary:", my_dict)

print()


# Re-populate the dictionary for the remaining demonstrations

my_dict = {'name': 'Priyam', 'age': 20, 'city': 'Delhi'}


# iv) get( )

print("iv) get( ):")

age = my_dict.get('age')

print("Age:", age)
```

```
print()

# v) pop( )
print("v) pop( ):")
city = my_dict.pop('city')
print("Popped city:", city)
print("Updated Dictionary:", my_dict)
print()

# vi) popitem( )
print("vi) popitem( ):")
removed_item = my_dict.popitem()
print("Removed item:", removed_item)
print("Updated Dictionary:", my_dict)
print()

# vii) keys( )
print("vii) keys( ):")
keys = my_dict.keys()
print("Keys:", keys)
print()

# viii) values( )
print("viii) values( ):")
values = my_dict.values()
print("Values:", values)
print()

# ix) items( )
print("ix) items( ):")
```

```
items = my_dict.items()
print("Items:", items)
print()

# x) copy( )
print("x) copy( ):")
copy_dict = my_dict.copy()
print("Copied Dictionary:", copy_dict)
print()

# xi) update( )
print("xi) update( ):")
my_dict.update({'gender': 'Male', 'city': 'New York'})
print("Updated Dictionary:", my_dict)
```

OUTPUT:-

```
i) dict( ):
New Dictionary: {'country': 'INDIA', 'gender': 'Male'}

ii) len( ):
Length of the dictionary: 3

iii) clear( ):
Cleared Dictionary: {}

iv) get( ):
Age: 20

v) pop( ):
Popped city: Delhi
Updated Dictionary: {'name': 'Priyam', 'age': 20}

vi) popitem( ):
Removed item: ('age', 20)
Updated Dictionary: {'name': 'Priyam'}

vii) keys( ):
Keys: dict_keys(['name'])

viii) values( ):
Values: dict_values(['Priyam'])

ix) items( ):
Items: dict_items([('name', 'Priyam')])

x) copy( ):
Copied Dictionary: {'name': 'Priyam'}

xi) update( ):
Updated Dictionary: {'name': 'Priyam', 'gender': 'Male', 'city': 'New York'}
```

INDEX

S.no.	Program list	signature
1	a) Python Program to Perform All Arithmetic Operations on Two Given Numbers b) Python Program to Perform All Arithmetic Operations on user defined input numbers	
2	a) Program to show how escape sequence characters works b) Python program to check if the number is an Armstrong number or not	
3	a) Write a program to print the sum of all the primes between two ranges. b) Solve in python program: 100 + 200 / 10 - 3 * 10	
4	a) Write a program to swap two strings. b) Program to explain precedence of arithmetic operators	
5	a) Write a menu driven program to accept two strings from the user and perform the various functions using user defined functions. b) Changing the case of Python Strings.	
6	Write a program to find: a. Sum of Digits of a number b. Product of digit.	
7	Write a program for: a. Multiplication b. Factorial	
8	a) Write a program to find smallest and largest number in a list . b) Write a program for List slicing and correcting mistakes values in a list c) Make a list with each item being increasing power of 2.	
9	Create a dictionary whose keys are month names and whose values are the number of days in the corresponding months. ➢ Ask the user to enter a month name and use the dictionary to tell them how many days are in the month. ➢ Print out all keys in the alphabetically order ➢ Print out all the months with 31 days ➢ Print out the key value pairs sorted by number of days in each month	

10	Write a program that scans an email address and forms a tuple of user name and domain.	
11	Write a program that defines a function large in a module which will be used to find larger of two values and called from code in another module .	
12	Write a program: a. To find numbers divisible by 7 and are not multiple of 5. In range of 2000-3200 inclusive. b. Which will have a list of values and then input a number to check if the value exist in the list or not.	
13	Write a program to: a. Input 5 numbers in the list and print in ascending order. b. Insert a value in the list at particular position. c. Count elements in the list until the occurrence of the first tuple in the list.	
14	Write a Python program to calculate the average value of the numbers in a given tuple of tuples.	
15	Make a simple chatbot using python.	
16	a) convert PDF file to Excel file using Python b) program to reverse the content of a file and store it in another file c) How to create a duplicate file of an existing file using Python?	
17	Create a binary file with roll number, name and marks. Input a roll number and perform the following operations: ➢ update the marks. ➢ Delete the record ➢ Display the record ➢ Append the record ➢ Search the record	
18	Write a program to Create a CSV file by entering user-id and password, read and search the password for given user id.	
19	Write a program to show joining of NumPy arrays .	
20	Write a program: a. Write a program to demonstrate subplots and multiple plots in matplotlib.	

	b. Demonstrate pie chart and Bar Graphs in matplotlib	
21	Demonstrate the following functions/methods which operates on dictionary in Python with suitable examples: i) dict( ) ii) len( ) iii) clear( ) iv) get( ) v) pop( ) vi)pop item( ) vii) keys( ) viii) values() ix) items( ) x) copy( ) xi) update( )	