ST. JOSEPH'S COLLEGE PILATHARA

(Affiliated to Kannur University)

Pilathara (P.O.), Kannur (Dt.), Kerala – 670504

www.stjosephscollege.ac.in

Department of Computer Application

BCA Practical Record

Reg. No. : _____

Name : _____

Semester : SIX

Subject : 6B22BCA Lab VI: Python Programming

ST. JOSEPH'S COLLEGE PILATHARA

Department of Computer Application

BCA Practical Record

CERTIFICATE

This is to certify that this is a bonafide record of practical work done by Mr./Ms. _____, Reg. No. _____of 6th Semester BCA in "Python Programming" as prescribed by Kannur University during the period _____to _____

Examiner 1 **Lecturer in Charge**

Examiner 2 **Head of the Department**

Place :

Date :

Program 1

Write a program to find the largest from a list of numbers.

```
nums = map(int, input("Enter the integers: ").split())
ans = max(nums, default=None)
print("Largest number in the list =", ans)
```

OUTPUT

Sample 1

```
Enter the integers:
Largest number in the list = None
```

Sample 2

```
Enter the integers: 12 13 -22 0 456 -123456789
Largest number in the list = 456
```

Program 2

Write a program to generate first n perfect numbers.

```
def is_perfect_number(num):
    if num < 2:
        return False
    return sum(i + num // i for i in range(2, int(num**0.5) +
1) if num % i == 0) + 1 == num

def generate_perfect_numbers(limit):
    num, count = 1, 0
    while count < limit:
        num += 1
        if is_perfect_number(num):
            yield num
            count += 1

n = int(input("Enter the limit: "))
if n < 1:
    print("INVALID limit...")
else:
    print(f"Generating the First {n} Perfect Numbers...")
    for c, x in enumerate(generate_perfect_numbers(n),
start=1):
        print(f"{c}. {x}")
    print("DONE")
```

OUTPUT

Sample 1

```
Enter the limit: 0
INVALID limit...
```

Sample 2

```
Enter the limit: 4
Generating the First 4 Perfect Numbers...
1. 6
2. 28
3. 496
4. 8128
DONE
```

Program 3

Write a program to perform the binary search.

```
def binary_search(arr, key):
    l, u = 0, len(arr) - 1

    while l <= u:
        mid = (l + u) // 2
        if arr[mid] == key:
            return mid
        elif arr[mid] > key:
            u = mid - 1
        else:
            l = mid + 1

    return -1

arr = input("Enter the sorted array elements: ").split()
key = input("Enter the element to search: ")

print(f"...Searching for '{key}' in {arr}")

idx = binary_search(arr, key)

if idx == -1:
    print("Element NOT FOUND...")
else:
    print(f"{key} FOUND at position {idx + 1}")
```

OUTPUT

Sample 1

```
Enter the array elements:
Binary Search:-
Enter the element to search:
...searching for '' in []
Element NOT FOUND...
```

Sample 2

```
Enter the array elements: 4 7 1 8 9 3 1
Binary Search:-
Enter the element to search: 6
...searching for '6' in ['1', '1', '3', '4', '7', '8', '9']
Element NOT FOUND...
```

Sample 3

```
Enter the array elements: 123 x abc 98.765 0 -43
Binary Search:-
Enter the element to search: abc
...searching for 'abc' in ['-43', '0', '123', '98.765', 'abc', 'x']
abc FOUND at position 5
```

Program 4

Write a program to find the square root of a number using bisection search method.

```
def bisection_method(num, tol=0.00001):
    x, y = 0, num

    while True:
        sqrt = (x + y) / 2.0
        if abs(sqrt**2 - num) <= tol:
            return sqrt
        elif sqrt**2 < num:
            x = sqrt
        else:
            y = sqrt

n = float(input("Enter the number: "))

if n < 0:
    print("Complex roots...")
else:
    print("Bisection Method:-")
    ans = bisection_method(n)
    print(f"Square Root of {n} = {ans:.5f}")
```

OUTPUT

Sample 1

```
Enter the number: 0
Bisection Method:-
Square Root of 0.0 = 0.0
```

Sample 2

```
Enter the number: -1
complex roots...
```

Sample 3

```
Enter the number: 64
Bisection Method:-
Square Root of 64.0 = 8.0
```

Sample 4

```
Enter the number: 12345.6789
Bisection Method:-
Square Root of 12345.6789 = 111.11
```

Sample 5

```
Enter the number: 6784131315486756 13.49853231002154034
Bisection Method:-
Square Root of 6.784131315486756e+17 = 8.2366e+08
```

Program 5

Write a program to generate Fibonacci series using recursion.

```
def fibonacci_number(n):
    if n <= 1:
        return n
    return fibonacci_number(n - 1) + fibonacci_number(n - 2)



def generate_fibonacci_series(limit):
    for i in range(n):
        yield fibonacci_number(i)



n = int(input("Enter the limit: "))
if n < 1:
    print("INVALID limit...")
else:
    print(f"Generating the First {n} Fibonacci Numbers using
recursion...")
    for c, x in enumerate(generate_fibonacci_series(n),
start=1):
        print(f"{c}. {x}")
    print("DONE")
```

OUTPUT

Sample 1

```
Enter the limit: 0
INVALID limit...
```

Sample 2

```
Enter the limit: 15
Generating the First 15 Fibonacci Numbers using recursion...
1. 0
2. 1
3. 1
4. 2
5. 3
6. 5
7. 8
8. 13
9. 21
10. 34
11. 55
12. 89
13. 144
14. 233
15. 377
DONE
```

Program 6

Write a program to find the LCM and GCD of 2 numbers.

```
import math


a, b = map(int, input("Enter two numbers: ").split())
GCD = math.gcd(a, b)
LCM = int(a * b / GCD) if GCD != 0 else 0
print("LCM =", LCM)
print("GCD =", GCD)
```

OUTPUT

Sample 1

```
Enter two numbers: 0 0
LCM = 0
GCD = 0
```

Sample 2

```
Enter two numbers: 0 5
LCM = 0
GCD = 5
```

Program 7

Write a program to perform merge sort.

```
def merge(left, right):
    result = []
    i = j = 0

    while i < len(left) and j < len(right):
        if left[i] < right[j]:
            result.append(left[i])
            i += 1
        else:
            result.append(right[j])
            j += 1

    result.extend(left[i:])
    result.extend(right[j:])
    return result

def merge_sort(arr):
    if len(arr) < 2:
        return arr

    mid = len(arr) // 2
    left = merge_sort(arr[:mid])
    right = merge_sort(arr[mid:])
    return merge(left, right)

arr = list(map(int, input("Enter a list of integers:
").split()))
```

```
print("Sorted Array =", merge_sort(arr))
```

OUTPUT

Sample 1

```
Enter a list of integers:
...performing Merge Sort on []
Sorted Array = []
```

Sample 2

```
Enter a list of integers: 99
...performing Merge Sort on [99]
Sorted Array = [99]
```

Sample 3

```
Enter a list of integers: 12 13 -22 0 12345 -98765
...performing Merge Sort on [12, 13, -22, 0, 12345, -98765]
Sorted Array = [-98765, -22, 0, 12, 13, 12345]
```

Program 8

Write a program which reads the contents of a file and copy the contents to another file after changing all the letter to upper case. Exceptions should be handled.

```
import sys


error = None
try:
    filename = input("Enter the filename: ")
    with open(filename, mode="rt") as infile:
        text = infile.read()
        with open("output_file.txt", mode="wt") as outfile:
            uppercase_text = text.upper()
            print("...conversion DONE")
            outfile.write(uppercase_text)
            print("<output_file.txt> created SUCCESSFULLY")
except EOFError as e:
    error = e
    sys.stderr.write("! INVALID filename...")
except OSError as e:
    error = e
    sys.stderr.write("! CANNOT open file...")
except BaseException:
    print(sys.exc_info())
finally:
    if error:
        print(repr(error))
```

OUTPUT

Sample 1

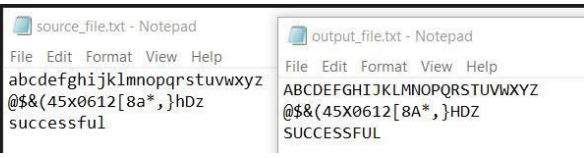
```
Enter the filename: ^Z
EOFError()
! INVALID filename...
```

Sample 2

```
Enter the filename: abc
FileNotFoundError(2, 'No such file or directory')
! CANNOT open file...
```

Sample 3

```
Enter the filename: source_file.txt
...conversion DONE
<output_file.txt> created SUCCESSFULLY
```



Program 9

Write a program to find the prime numbers in a list of numbers.

```
def is_prime(n):
    if n <= 1:
        return False
    i = 2
    while i*i <= n:
        if n % i == 0:
            return False
        i += 1
    return True


nums = map(int, input("Enter a list of numbers: ").split())
primes_found = False
print("Prime Numbers in this list:")
for num in filter(is_prime, nums):
    print(num)
    primes_found = True
if not primes_found:
    print(None)
```

OUTPUT

Sample 1

```
Enter a list of numbers:
Prime Numbers in this list:
None
```

Sample 2

```
Enter a list of numbers: 4 6 10 21 100
Prime Numbers in this list:
None
```

Sample 3

```
Enter a list of numbers: 2 4 6 8
Prime Numbers in this list:
2
```

Sample 4

```
Enter a list of numbers: -3 0 4 1 12 2 3 39 41 87 89
Prime Numbers in this list:
2
3
41
89
```

Program 10

Write a python program to perform the following

a) Create table students with fields name,sex,rollno,marks

b) Insert some rows into the table

c) Update the marks of all students by adding 2 marks

d) Delete a student with a given rollno

e) Display the details of a student with a given rollno

```
import pymysql

connection = pymysql.connect(
    host="localhost",
    user="student",
    password="123",
    autocommit=True
)
cursor = connection.cursor()

try:
    cursor.execute("CREATE DATABASE IF NOT EXISTS
bca_practical_pymysql")
    print("Database created/exists.")
except Exception as e:
    print("Error creating database:", repr(e))

cursor.execute("USE bca_practical_pymysql")

try:
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS students (
            name VARCHAR(255) NOT NULL,
            sex ENUM('M', 'F', 'U') NOT NULL,
            rollno INT NOT NULL PRIMARY KEY,
            marks DECIMAL(5,2) NOT NULL CHECK(marks >= 0)
        )
```

```
    ''')
    print("Table 'students' created/exists.")
except Exception as e:
    print("Error creating table:", repr(e))

cursor.execute("DELETE FROM students")

dummy_data = [
    ("Alice", "F", 101, 85.00),
    ("Bob", "M", 102, 78.00),
    ("Charlie", "M", 103, 90.00)
]

try:
    for i, record in enumerate(dummy_data, start=1):
        sql = "INSERT INTO students VALUES (%s, %s, %s, %s)"
        cursor.execute(sql, record)
        print(f"Record {i} inserted successfully.")
except Exception as e:
    print("Error during insertion:", repr(e))

try:
    cursor.execute("UPDATE students SET marks = marks + 2")
    if cursor.rowcount == 0:
        raise RuntimeError("No records to update.")
    print("Updated marks for all students.")
except Exception as e:
    print("Error during update:", repr(e))

rollno_to_delete = 102

try:
    cursor.execute("DELETE FROM students WHERE rollno = %s",
(rollno_to_delete,))
    if cursor.rowcount == 0:
        raise KeyError(f"No student with
rollno={rollno_to_delete}")
    print("Student deleted successfully.")
except Exception as e:
    print("Error during deletion:", repr(e))

rollno_to_display = 103

try:
```

```
    cursor.execute('''
        SELECT CAST(name AS CHAR) AS Name,
               CAST(sex AS CHAR) AS Sex,
               CAST(rollno AS CHAR) AS Rollno,
               CAST(marks AS CHAR) AS Marks
        FROM students
        WHERE rollno = %s
    ''', (rollno_to_display,))
    result = cursor.fetchone()
    if result is None:
        raise KeyError(f"No student with
rollno={rollno_to_display}")
    print("Student details:")
    for col, val in zip(("Name", "Sex", "Rollno", "Marks"),
result):
        print(f"{col:<8}: {val}")
except Exception as e:
    print("Error during display:", repr(e))

connection.close()
print("Program Execution SUCCESSFUL")
```

OUTPUT

Segment 1

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                                   1: python, mysql

Microsoft Windows [Version 10.0.19043.1110]
(c) Microsoft Corporation. All rights reserved.

C:\Users\eindr\Home\School\3. BCA\Course\V Semester\Lab\Python>cd "10. Database Management"

C:\Users\eindr\Home\School\3. BCA\Course\V Semester\Lab\Python\10. Database Management>python python_and_mysql.py

Creating Database <bca_practical_pymysql>...
# DATABASE Creation Successful

Creating Table <students>...
# CREATION successful

INSERTION:-
Enter the number of records: 
```

```
Microsoft Windows [Version 10.0.19043.1110]
(c) Microsoft Corporation. All rights reserved.

C:\Users\eindr\Home\School\3. BCA\Course\V Semester\Lab\Python>mysql -u root -p
Enter password: *******
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 17
Server version: 8.0.25 MySQL Community Server - GPL

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> USE bca_practical_pymysql;
Database changed
mysql> SHOW tables;
+---------------------------------+
| Tables_in_bca_practical_pymysql |
+---------------------------------+
| students                        |
+---------------------------------+
1 row in set (0.00 sec)

mysql> DESC students;
+--------+------------------+------+-----+---------+-------+
| Field  | Type             | Null | Key | Default | Extra |
+--------+------------------+------+-----+---------+-------+
| name   | varchar(255)     | NO   |     | NULL    |       |
| sex    | enum('M','F','U') | NO   |     | NULL    |       |
| rollno | int              | NO   | PRI | NULL    |       |
| marks  | decimal(5,2)     | NO   |     | NULL    |       |
+--------+------------------+------+-----+---------+-------+
4 rows in set (0.01 sec)

mysql> SELECT * FROM students;
Empty set (0.00 sec)

mysql> 
```

Segment 2

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                                   1: python, mysql

Creating Database <bca_practical_pymysql>...
# DATABASE Creation Successful

Creating Table <students>...
# CREATION successful

INSERTION:-
Enter the number of records: 5
-> Record 1 -
        Enter the name: Pqr
        Enter the sex: F
        Enter the roll number: 101
        Enter the marks: 85.13
# Record 1 inserted successfully
-> Record 2 -
        Enter the name: Mno
        Enter the sex: !@#
        Enter the roll number: 102
        Enter the marks: 45.67
DataError(1265, "Data truncated for column 'sex' at row 1")
-> Record 3 -
        Enter the name: Efg
        Enter the sex: M
        Enter the roll number: 101
        Enter the marks: 100
IntegrityError(1062, "Duplicate entry '101' for key 'students.PRIMARY'")
-> Record 4 -
        Enter the name: Def Lmn
        Enter the sex: U
        Enter the roll number: 102
        Enter the marks: 1234
DataError(1264, "Out of range value for column 'marks' at row 1")
-> Record 5 -
        Enter the name: Abc Xyz
        Enter the sex: M
        Enter the roll number: 102
        Enter the marks: 95.67
# Record 5 inserted successfully

Adding 2 marks for all students...
# UPDATION successful

Deleting a student with the given rollno...
Enter the rollno: 
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> USE bca_practical_pymysql;
Database changed
mysql> SHOW tables;
+---------------------------------+
| Tables_in_bca_practical_pymysql |
+---------------------------------+
| students                        |
+---------------------------------+
1 row in set (0.00 sec)

mysql> DESC students;
+--------+------------------+------+-----+---------+-------+
| Field  | Type             | Null | Key | Default | Extra |
+--------+------------------+------+-----+---------+-------+
| name   | varchar(255)     | NO   |     | NULL    |       |
| sex    | enum('M','F','U') | NO   |     | NULL    |       |
| rollno | int              | NO   | PRI | NULL    |       |
| marks  | decimal(5,2)     | NO   |     | NULL    |       |
+--------+------------------+------+-----+---------+-------+
4 rows in set (0.01 sec)

mysql> SELECT * FROM students;
Empty set (0.00 sec)

mysql> SELECT * FROM students;
+------+----+--------+-------+
| name | sex | rollno | marks |
+------+----+--------+-------+
| Pqr  | F  |    101 | 85.13 |
+------+----+--------+-------+
1 row in set (0.00 sec)

mysql> SELECT * FROM students;
+---------+----+--------+-------+
| name    | sex | rollno | marks |
+---------+----+--------+-------+
| Pqr     | F  |    101 | 87.13 |
| Abc Xyz | M  |    102 | 97.67 |
+---------+----+--------+-------+
2 rows in set (0.00 sec)

mysql> 
```

Program 11

Create a simple Login window using Tkinter.

```
import tkinter as tk
import tkinter.font as tk_font

root = tk.Tk()
root.title("Login Window using Tkinter")
root.geometry('600x300')

text_font = tk_font.Font(family="Courier New", size=12,
weight="bold")
input_font = tk_font.Font(family="Lucida Console", size=10)

username_label = tk.Label(root, text="Username", font=text_font)
password_label = tk.Label(root, text="Password", font=text_font)
username_entry = tk.Entry(root, width=50, font=input_font)
password_entry = tk.Entry(root, show="*", width=50, font=input_font)
login_button = tk.Button(root, text="LOGIN", font=text_font)

username_label.grid(row=0, column=0, padx=20, pady=10, ipadx=5,
ipady=20)
password_label.grid(row=1, column=0, padx=20, pady=10, ipadx=5,
ipady=20)
username_entry.grid(row=0, column=1, padx=20, pady=10, ipadx=5,
ipady=10)
password_entry.grid(row=1, column=1, padx=20, pady=10, ipadx=5,
ipady=10)
login_button.grid(row=2, column=1, padx=20, pady=20, ipadx=100,
ipady=10)
```

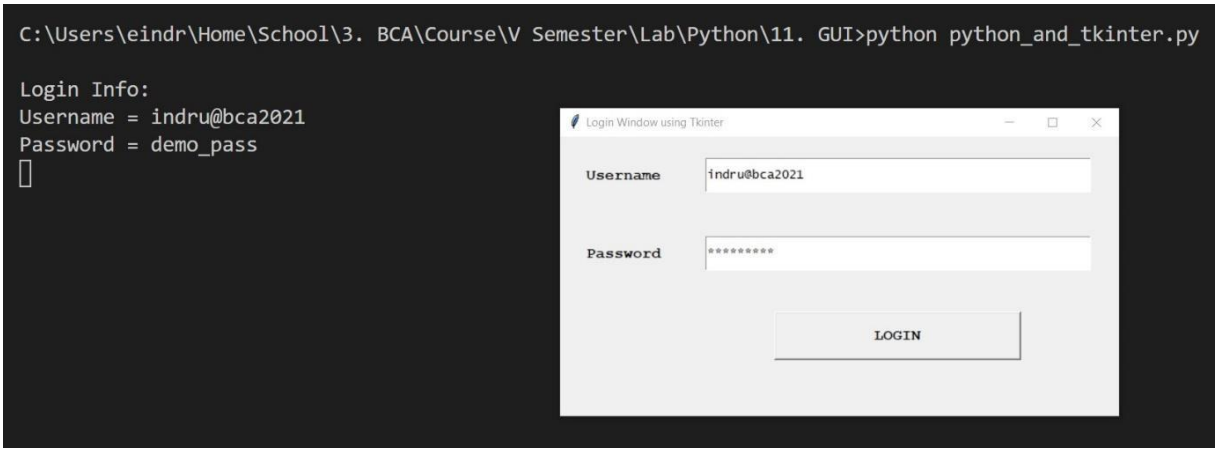
```
def login():
    print("\nLogin Info:")
    print("Username =", username_entry.get())
    print("Password =", password_entry.get())



login_button.config(command=login)


root.mainloop()
```

OUTPUT



Program 12

Create a plot for the mathematical function x^2. The title of the plot and the axes should be labelled.

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

l = int(input("Enter the inclusive lower bound: "))
u = int(input("Enter the inclusive upper bound: "))

print("Plotting...")
x = np.arange(l, u, 0.1)
y = x**2

plt.title("Quadratic Function")
plt.xlabel("x")
plt.ylabel("x²")
plt.plot(x, y)
plt.show()
```

OUTPUT

```
C:\Users\eindr\Home\School\3. BCA\Course\V Semester\Lab\Python>cd "12. Data Visualization"

C:\Users\eindr\Home\School\3. BCA\Course\V Semester\Lab\Python\12. Data Visualization>conda activate

(base) C:\Users\eindr\Home\School\3. BCA\Course\V Semester\Lab\Python\12. Data Visualization>python pyplot_quadratic.py
Enter the inclusive lower bound: -100
Enter the inclusive upper bound: 100
Plotting...
```