In [2]:
```
pip install mysql-connector-python
```

```
Requirement already satisfied: mysql-connector-python in c:\users\ibmjo\anaconda3
\lib\site-packages (8.1.0)
Requirement already satisfied: protobuf<=4.21.12,>=4.21.1 in c:\users\ibmjo\anacon
da3\lib\site-packages (from mysql-connector-python) (4.21.12)
Note: you may need to restart the kernel to use updated packages.
```

In [3]:
```
pip list
```

```
Package                       Version
----------------------------- ---------------
alabaster                     0.7.12
anaconda-client               1.11.2
anaconda-navigator            2.4.0
anaconda-project              0.11.1
anyio                         3.5.0
appdirs                       1.4.4
argon2-cffi                   21.3.0
argon2-cffi-bindings          21.2.0
arrow                         1.2.3
astroid                       2.14.2
astropy                       5.1
asttokens                     2.0.5
atomicwrites                  1.4.0
attrs                         22.1.0
Automat                       20.2.0
autopep8                      1.6.0
Babel                         2.11.0
backcall                      0.2.0
backports.functools-lru-cache 1.6.4
backports.tempfile            1.0
backports.weakref             1.0.post1
bcrypt                        3.2.0
beautifulsoup4                4.11.1
binaryornot                   0.4.4
black                         22.6.0
bleach                        4.1.0
bokeh                         2.4.3
boltons                       23.0.0
Bottleneck                    1.3.5
brotlipy                      0.7.0
certifi                       2022.12.7
cffi                          1.15.1
chardet                       4.0.0
charset-normalizer            2.0.4
click                         8.0.4
cloudpickle                   2.0.0
clyent                        1.2.2
colorama                      0.4.6
colorcet                      3.0.1
comm                          0.1.2
conda                         23.3.1
conda-build                   3.24.0
conda-content-trust           0.1.3
conda-pack                    0.6.0
conda-package-handling        2.0.2
conda_package_streaming       0.7.0
conda-repo-cli                1.0.41
conda-token                   0.4.0
conda-verify                  3.4.2
constantly                    15.1.0
contourpy                     1.0.5
cookiecutter                  1.7.3
cryptography                  39.0.1
cssselect                     1.1.0
cycler                        0.11.0
cytoolz                       0.12.0
daal4py                       2023.0.2
dask                          2022.7.0
datashader                    0.14.4
datashape                     0.5.4
debugpy                       1.5.1
decorator                     5.1.1
```

```
defusedxml                0.7.1
diff-match-patch          20200713
dill                      0.3.6
distributed               2022.7.0
docstring-to-markdown     0.11
docutils                  0.18.1
entrypoints               0.4
et-xmlfile                1.1.0
executing                 0.8.3
fastjsonschema            2.16.2
filelock                  3.9.0
flake8                    6.0.0
Flask                     2.2.2
flit_core                 3.6.0
fonttools                 4.25.0
fsspec                    2022.11.0
future                    0.18.3
gensim                    4.3.0
glob2                     0.7
greenlet                  2.0.1
h5py                      3.7.0
HeapDict                  1.0.1
holoviews                 1.15.4
huggingface-hub           0.10.1
hvplot                    0.8.2
hyperlink                 21.0.0
idna                      3.4
imagecodecs               2021.8.26
imageio                   2.26.0
imagesize                 1.4.1
imbalanced-learn          0.10.1
importlib-metadata        4.11.3
incremental               21.3.0
inflection                0.5.1
iniconfig                 1.1.1
intake                    0.6.7
intervaltree              3.1.0
ipykernel                 6.19.2
ipython                   8.10.0
ipython-genutils          0.2.0
ipywidgets                7.6.5
isort                     5.9.3
itemadapter               0.3.0
itemloaders               1.0.4
itsdangerous              2.0.1
jedi                      0.18.1
jellyfish                 0.9.0
Jinja2                    3.1.2
jinja2-time               0.2.0
jmespath                  0.10.0
joblib                    1.1.1
json5                     0.9.6
jsonpatch                 1.32
jsonpointer               2.1
jsonschema                4.17.3
jupyter                   1.0.0
jupyter_client            7.3.4
jupyter-console           6.6.2
jupyter_core              5.2.0
jupyter-server            1.23.4
jupyterlab                3.5.3
jupyterlab-pygments       0.1.2
jupyterlab_server         2.19.0
jupyterlab-widgets        1.0.0
```

```
keyring                       23.4.0
kiwisolver                    1.4.4
lazy-object-proxy             1.6.0
libarchive-c                  2.9
llvmlite                      0.39.1
locket                        1.0.0
lxml                          4.9.1
lz4                           3.1.3
Markdown                      3.4.1
MarkupSafe                    2.1.1
matplotlib                    3.7.0
matplotlib-inline             0.1.6
mccabe                        0.7.0
menuinst                      1.4.19
mistune                       0.8.4
mkl-fft                       1.3.1
mkl-random                    1.2.2
mkl-service                   2.4.0
mock                          4.0.3
mpmath                        1.2.1
msgpack                       1.0.3
multipledispatch              0.6.0
munkres                       1.1.4
mypy-extensions               0.4.3
mysql-connector-python        8.1.0
navigator-updater             0.3.0
nbclassic                     0.5.2
nbclient                      0.5.13
nbconvert                     6.5.4
nbformat                      5.7.0
nest-asyncio                  1.5.6
networkx                      2.8.4
nltk                          3.7
notebook                      6.5.2
notebook_shim                 0.2.2
numba                         0.56.4
numexpr                       2.8.4
numpy                         1.23.5
numpydoc                      1.5.0
openpyxl                      3.0.10
packaging                     22.0
pandas                        1.5.3
pandocfilters                 1.5.0
panel                         0.14.3
param                         1.12.3
paramiko                      2.8.1
parsel                        1.6.0
parso                         0.8.3
partd                         1.2.0
pathlib                       1.0.1
pathspec                      0.10.3
patsy                         0.5.3
pep8                          1.7.1
pexpect                       4.8.0
pickleshare                   0.7.5
Pillow                        9.4.0
pip                           22.3.1
pkginfo                       1.9.6
platformdirs                  2.5.2
plotly                        5.9.0
pluggy                        1.0.0
ply                           3.11
pooch                         1.4.0
poyo                          0.5.0
```

```
prometheus-client            0.14.1
prompt-toolkit               3.0.36
Protego                      0.1.16
protobuf                     4.21.12
psutil                       5.9.0
ptyprocess                   0.7.0
pure-eval                    0.2.2
py                           1.11.0
pyasn1                       0.4.8
pyasn1-modules               0.2.8
pycodestyle                  2.10.0
pycosat                      0.6.4
pycparser                    2.21
pyct                         0.5.0
pycurl                       7.45.1
PyDispatcher                 2.0.5
pydocstyle                   6.3.0
pyerfa                       2.0.0
pyflakes                     3.0.1
Pygments                     2.11.2
PyHamcrest                   2.0.2
PyJWT                        2.4.0
pylint                       2.16.2
pylint-venv                  2.3.0
pyls-spyder                  0.4.0
PyNaCl                       1.5.0
pyodbc                       4.0.34
pyOpenSSL                    23.0.0
pyparsing                    3.0.9
PyQt5                        5.15.7
PyQt5-sip                    12.11.0
PyQtWebEngine                5.15.4
pyrsistent                   0.18.0
PySocks                      1.7.1
pytest                       7.1.2
python-dateutil              2.8.2
python-lsp-black             1.2.1
python-lsp-jsonrpc           1.0.0
python-lsp-server            1.7.1
python-slugify               5.0.2
python-snappy                0.6.1
pytoolconfig                 1.2.5
pytz                         2022.7
pyviz-comms                  2.0.2
PyWavelets                   1.4.1
pywin32                      305.1
pywin32-ctypes               0.2.0
pywinpty                     2.0.10
PyYAML                       6.0
pyzmq                        23.2.0
QDarkStyle                   3.0.2
qstylizer                    0.2.2
QtAwesome                    1.2.2
qtconsole                    5.4.0
QtPy                         2.2.0
queuelib                     1.5.0
regex                        2022.7.9
requests                     2.28.1
requests-file                1.5.1
requests-toolbelt            0.9.1
rope                         1.7.0
Rtree                        1.0.1
ruamel.yaml                  0.17.21
ruamel.yaml.clib             0.2.6
```

```
ruamel-yaml-conda             0.17.21
scikit-image                  0.19.3
scikit-learn                  1.2.1
scikit-learn-intelex          20230228.214818
scipy                         1.10.0
Scrapy                        2.8.0
seaborn                       0.12.2
Send2Trash                    1.8.0
service-identity              18.1.0
setuptools                    65.6.3
sip                           6.6.2
six                           1.16.0
smart-open                    5.2.1
sniffio                       1.2.0
snowballstemmer               2.2.0
sortedcontainers              2.4.0
soupsieve                     2.3.2.post1
Sphinx                        5.0.2
sphinxcontrib-applehelp       1.0.2
sphinxcontrib-devhelp         1.0.2
sphinxcontrib-htmlhelp        2.0.0
sphinxcontrib-jsmath          1.0.1
sphinxcontrib-qthelp          1.0.3
sphinxcontrib-serializinghtml 1.1.5
spyder                        5.4.1
spyder-kernels                2.4.1
SQLAlchemy                    1.4.39
stack-data                    0.2.0
statsmodels                   0.13.5
sympy                         1.11.1
tables                        3.7.0
tabulate                      0.8.10
TBB                           0.2
tblib                         1.7.0
tenacity                      8.0.1
terminado                     0.17.1
text-unidecode                1.3
textdistance                  4.2.1
threadpoolctl                 2.2.0
three-merge                   0.1.1
tifffile                      2021.7.2
tinycss2                      1.2.1
tldextract                    3.2.0
tokenizers                    0.11.4
toml                          0.10.2
tomli                         2.0.1
tomlkit                       0.11.1
toolz                         0.12.0
torch                         1.12.1
tornado                       6.1
tqdm                          4.64.1
traitlets                     5.7.1
transformers                  4.24.0
Twisted                       22.2.0
twisted-iocpsupport           1.0.2
typing_extensions             4.4.0
ujson                         5.4.0
Unidecode                     1.2.0
urllib3                       1.26.14
w3lib                         1.21.0
watchdog                      2.1.6
wcwidth                       0.2.5
webencodings                  0.5.1
websocket-client              0.58.0
```

```
Werkzeug                      2.2.2
whatthepatch                  1.0.2
wheel                         0.38.4
widgetsnbextension            3.5.2
win-inet-pton                 1.1.0
wincertstore                  0.2
wrapt                         1.14.1
xarray                        2022.11.0
xlwings                       0.29.1
yapf                          0.31.0
zict                          2.1.0
zipp                          3.11.0
zope.interface                5.4.0
zstandard                     0.19.0
Note: you may need to restart the kernel to use updated packages.
```

In [4]:
```
import mysql.connector as c
```

In [5]:
```
conn=c.connect(host="localhost",user="root",password="")
if conn.is_connected():
 print("Database connected")
else:
 print("connection failed")
```

```
Database connected
```

In [6]:
```
import mysql.connector as c
mydb = c.connect(host="localhost",user="root",password="")
mycursor = mydb.cursor()
mycursor.execute("SHOW DATABASES")
for value in mycursor:
 print(value)
```

```
('adit',)
('djangodb',)
('flaskdb',)
('information_schema',)
('mysql',)
('performance_schema',)
('phpmyadmin',)
('test',)
```

In [8]:
```
import mysql.connector as c
conn=c.connect(host="localhost",user="root",password="",database="flaskdb")
#create cursor
cur=conn.cursor()
cur.execute("show tables")
for i in cur:
 print(i)
```

```
('student',)
('user',)
```

In [10]:
```
import mysql.connector as c
mydb = c.connect(host="localhost",user="root",password="",database="flaskdb")
#create cursor
cur = mydb.cursor()
try:
 cur.execute("show databases")
except:
 mydb.rollback()
#print database
for value in cur:
```

```
 print(value)
mydb.close()
```

```
('adit',)
('djangodb',)
('flaskdb',)
('information_schema',)
('mysql',)
('performance_schema',)
('phpmyadmin',)
('test',)
```

In [11]:

```
import mysql.connector as c
mydb = c.connect(host="localhost",user="root",password="")
#create cursor
cur = mydb.cursor()
try:
 dbs=cur.execute("create database python_database")
 dbs=cur.execute("show databases")
except:
 mydb.rollback()
for value in cur:
 print(value)
```

```
('adit',)
('djangodb',)
('flaskdb',)
('information_schema',)
('mysql',)
('performance_schema',)
('phpmyadmin',)
('python_database',)
('test',)
```

In [15]:

```
mydb = c.connect(host="localhost",user="root",password="", database="flaskdb")
cur = mydb.cursor()
try:
 dbs=cur.execute("show tables")
except:
 mydb.rollback()
for value in cur:
 print(value)
```

```
('student',)
('user',)
```

In [24]:

```
# Create a connection to the MySQL server
conn = c.connect(host="localhost", user="root", password="", database="flaskdb")
cur = conn.cursor()

try:
    # Define the SQL query to create the table (with the missing closing parenthes
    create_table_query = """
    CREATE TABLE trainers (
        name VARCHAR(20) NOT NULL
    )
    """

    # Execute the SQL query to create the table
    cur.execute(create_table_query)

    # Commit the transaction to apply the changes
    conn.commit()

    print("Table 'trainers' created successfully")
```

```
except c.Error as e:
    # Rollback the transaction in case of an error
    conn.rollback()
    print(f"Error: {e}")

finally:
    # Close the cursor and the connection
    cur.close()
    conn.close()
```

```
Table 'trainers' created successfully
```

In [28]:

```
# Create a connection to the MySQL server
conn = c.connect(host="localhost", user="root", password="", database="flaskdb")
cur = conn.cursor()

try:
    # Define the SQL query to add a new column to the table
    add_column_query = """
    ALTER TABLE trainers
    ADD COLUMN age INT
    """

    # Execute the SQL query to add the new column
    cur.execute(add_column_query)

    # Commit the transaction to apply the changes
    conn.commit()

    print("Column 'age' added to the 'trainers' table successfully")

except c.Error as e:
    # Rollback the transaction in case of an error
    conn.rollback()
    print(f"Error: {e}")

finally:
    # Close the cursor and the connection
    cur.close()
    conn.close()
```

```
Column 'age' added to the 'trainers' table successfully
```

In [30]:

```
conn = c.connect(host="localhost", user="root", password="", database="flaskdb")
cur = conn.cursor()

try:
    # Insert data into the table
    insert_data_query = """
    INSERT INTO trainers (name, age)
    VALUES
    ('John', 30),
    ('Alice', 25),
    ('Bob', 28)
    """

    # Execute the SQL query to insert the data
    cur.execute(insert_data_query)

    # Commit the transaction to apply the changes
    conn.commit()

    print("Data inserted into the 'trainers' table successfully")
```

```
except c.Error as e:
    # Rollback the transaction in case of an error
    conn.rollback()
    print(f"Error: {e}")

finally:
    # Close the cursor and the connection
    cur.close()
    conn.close()
```

```
Data inserted into the 'trainers' table successfully
```

In [31]:

```
conn = c.connect(host="localhost", user="root", password="", database="flaskdb")
cur = conn.cursor()

try:
    # Define a list of tuples, each tuple contains data for a row
    data_to_insert = [
        ('John', 30),
        ('Alice', 25),
        ('Bob', 28),
        ('Eve', 32)
    ]

    # Define the SQL query with placeholders for the data
    insert_data_query = """
    INSERT INTO trainers (name, age)
    VALUES (%s, %s)
    """

    # Execute the SQL query to insert the data
    cur.executemany(insert_data_query, data_to_insert)

    # Commit the transaction to apply the changes
    conn.commit()

    print("Data inserted into the 'trainers' table successfully")

except c.Error as e:
    # Rollback the transaction in case of an error
    conn.rollback()
    print(f"Error: {e}")

finally:
    # Close the cursor and the connection
    cur.close()
    conn.close()
```

```
Data inserted into the 'trainers' table successfully
```

In [32]:

```
# Create a connection to the MySQL server
conn = c.connect(host="localhost", user="root", password="", database="flaskdb")
cur = conn.cursor()

try:
    # Define the SQL query to fetch all data from the table
    select_data_query = "SELECT * FROM trainers"

    # Execute the SQL query to fetch the data
    cur.execute(select_data_query)

    # Fetch all rows of data from the result set
    data = cur.fetchall()

    # Print the fetched data
```

```
    for row in data:
        print(f"Name: {row[0]}, Age: {row[1]}")

except c.Error as e:
    print(f"Error: {e}")

finally:
    # Close the cursor and the connection
    cur.close()
    conn.close()
```

```
Name: John, Age: 30
Name: Alice, Age: 25
Name: Bob, Age: 28
Name: John, Age: 30
Name: Alice, Age: 25
Name: Bob, Age: 28
Name: John, Age: 30
Name: Alice, Age: 25
Name: Bob, Age: 28
Name: Eve, Age: 32
```

In [33]:

```
# Create a connection to the MySQL server
conn = c.connect(host="localhost", user="root", password="", database="flaskdb")
cur = conn.cursor()

try:
    # Define the SQL query to fetch data with a specific condition
    select_data_query = "SELECT * FROM trainers WHERE age > 28"

    # Execute the SQL query to fetch the data
    cur.execute(select_data_query)

    # Fetch all rows of data that match the condition
    matching_data = cur.fetchall()

    # Print the fetched data
    for row in matching_data:
        print(f"Name: {row[0]}, Age: {row[1]}")

except c.Error as e:
    print(f"Error: {e}")

finally:
    # Close the cursor and the connection
    cur.close()
    conn.close()
```

```
Name: John, Age: 30
Name: John, Age: 30
Name: John, Age: 30
Name: Eve, Age: 32
```

In [34]:

```
# Create a connection to the MySQL server
conn = c.connect(host="localhost", user="root", password="", database="flaskdb")
cur = conn.cursor()

try:
    # Define the SQL query to update data
    update_data_query = """
    UPDATE trainers
    SET age = 29
    WHERE name = 'John'
    """
```

```
    # Execute the SQL query to update the data
    cur.execute(update_data_query)

    # Commit the transaction to apply the changes
    conn.commit()

    print("Data updated successfully")

except c.Error as e:
    # Rollback the transaction in case of an error
    conn.rollback()
    print(f"Error: {e}")

finally:
    # Close the cursor and the connection
    cur.close()
    conn.close()
```

```
Data updated successfully
```

In [35]:

```
conn = c.connect(host="localhost", user="root", password="", database="flaskdb")
cur = conn.cursor()

try:
    # Define the SQL query to delete data with a specific condition
    delete_data_query = "DELETE FROM trainers WHERE age > 30"

    # Execute the SQL query to delete the data
    cur.execute(delete_data_query)

    # Commit the transaction to apply the changes
    conn.commit()

    print("Data deleted successfully")

except c.Error as e:
    # Rollback the transaction in case of an error
    conn.rollback()
    print(f"Error: {e}")

finally:
    # Close the cursor and the connection
    cur.close()
    conn.close()
```

```
Data deleted successfully
```

In [37]:

```
# Create a connection to the MySQL server
conn = c.connect(host="localhost", user="root", password="", database="flaskdb")
cur = conn.cursor()

try:
    # Define the SQL query to drop (delete) the table
    drop_table_query = "DROP TABLE trainers"

    # Execute the SQL query to drop the table
    cur.execute(drop_table_query)

    # Commit the transaction to apply the changes
    conn.commit()

    print("Table 'trainers' deleted successfully")

except c.Error as e:
    # Rollback the transaction in case of an error
```

```
    conn.rollback()
    print(f"Error: {e}")

finally:
    # Close the cursor and the connection
    cur.close()
    conn.close()
```

```
Error: 1051 (42S02): Unknown table 'flaskdb.trainers'
```

In [38]:

```
# Create a connection to the MySQL server
conn = c.connect(host="localhost", user="root", password="")

try:
    # Define the database name to drop
    database_name = "flaskdb"

    # Define the SQL query to drop (delete) the database
    drop_database_query = f"DROP DATABASE {database_name}"

    # Create a cursor and execute the SQL query to drop the database
    cur = conn.cursor()
    cur.execute(drop_database_query)
    cur.close()

    print(f"Database '{database_name}' dropped successfully")

except c.Error as e:
    print(f"Error: {e}")

finally:
    # Close the connection
    conn.close()
```

```
Database 'flaskdb' dropped successfully
```

In [5]:

```
#ciccle and area
import math

class Circle():
    def __init__(self, radius):
        self.radius = radius

    def area(self):
        return math.pi * (self.radius ** 2)

    def perimeter(self):
        return 2 * math.pi * self.radius

# Read value of radius input from user
value_of_circle = float(input("Enter radius of circle: "))  # Convert to float for

# Create an object for the Circle class
obj = Circle(value_of_circle)

print("Area of circle:", round(obj.area(), 2))
print("Perimeter of circle:", round(obj.perimeter(), 2))
```

```
Enter radius of circle: 10
Area of circle: 314.16
Perimeter of circle: 62.83
```

In [8]:

```
#bubble sort
def bubble_sort(arr):
    n = len(arr)
```

```
    for i in range(n):
        swapped = False
        for j in range(0, n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]
                swapped = True
        if not swapped:
            break

# Example usage:
my_list = [64, 34, 25, 12, 22, 11, 90]
bubble_sort(my_list)

print("Sorted list:", my_list)
```

```
Sorted list: [11, 12, 22, 25, 34, 64, 90]
```

In [10]:

```
#palindrome
Word = str(input("Enter a number"))
if(Word==Word[::-1]):
    print("Your Word is palindrome")
else:
    print("Your Word isn't palindrome")
```

```
Enter a number1232
Your Word isn't palindrome
```

In [11]:

```
#factorial
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)

# Example usage:
number = int(input("Enter a non-negative integer: "))
if number < 0:
    print("Factorial is not defined for negative numbers.")
else:
    result = factorial(number)
    print(f"The factorial of {number} is {result}")
```

```
Enter a non-negative integer: 5
The factorial of 5 is 120
```

In [ ]: