ASSIGNMENT - 20 (PYTHON)

1. Set the variable test1 to the string 'This is a test of the emergency text system,' and save test1 to a file named test.txt.

```
# Set the variable test1 to the string
test1 = 'This is a test of the emergency text system.'

# Save test1 to a file named test.txt
with open('test.txt', 'w') as file:
    file.write(test1)
```

This code will create a file named test.txt and write the content of the variable test1 to it. If the file doesn't exist, it will be created. If it already exists, its contents will be overwritten.

2. Read the contents of the file test.txt into the variable test2. Is there a difference between test 1 and test 2?

```
# Read the contents of the file test.txt into the variable test2
with open('test.txt', 'r') as file:
    test2 = file.read()

# Check if there's a difference between test1 and test2
if test1 == test2:
    print("There is no difference between test1 and test2.")
else:
    print("There is a difference between test1 and test2.")
```

This code reads the contents of the file test.txt into the variable test2 and then compares it with test1. If they are the same, it prints that there is no difference. Otherwise, it prints that there is a difference.

The comparison will reveal if there are any differences between the contents of test1 and test2. If the file test.txt contains exactly the same content as the variable test1, there will be no difference. Otherwise, there will be a difference.

3. Create a CSV file called books.csv by using these lines:

```
title,author,year
The Weirdstone of Brisingamen,Alan Garner,1960
Perdido Street Station,China Miéville,2000
Thud!,Terry Pratchett,2005
The Spellman Files,Lisa Lutz,2007
Small Gods,Terry Pratchett,1992
```

```
# Lines to be written to the CSV file
lines = [
    "title,author,year\n",
    "The Weirdstone of Brisingamen,Alan Garner,1960\n",
    "Perdido Street Station,China Miéville,2000\n",
    "Thud!,Terry Pratchett,2005\n",
    "The Spellman Files,Lisa Lutz,2007\n",
    "Small Gods,Terry Pratchett,1992\n"
]

# Write the lines to the CSV file
with open('books.csv', 'w') as file:
    file.writelines(lines)
```

This code will create a file named books.csv and write the provided lines into it. Each line represents a row in the CSV file, with columns separated by commas. The first line contains the headers title, author, and year, and subsequent lines contain the corresponding data for each book.

4. Use the sqlite3 module to create a SQLite database called books.db, and a table called books with
these fields: title (text), author (text), and year (integer).

```
import sqlite3

# Connect to the SQLite database (or create it if it doesn't exist)
conn = sqlite3.connect('books.db')

# Create a cursor object to execute SQL commands
cursor = conn.cursor()

# Define the SQL query to create the 'books' table
create_table_query = '''
CREATE TABLE IF NOT EXISTS books (
    title TEXT,
    author TEXT,
    year INTEGER
)
'''

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

# Commit the changes and close the connection
```

```
conn.commit()
conn.close()
```

This code connects to the SQLite database books.db (or creates it if it doesn't exist), defines a cursor object to execute SQL commands, and then executes an SQL query to create the books table with the specified fields. Finally, it commits the changes and closes the connection to the database.

5. Read books.csv and insert its data into the book table.

```
import csv
import sqlite3

# Connect to the SQLite database
conn = sqlite3.connect('books.db')

# Create a cursor object to execute SQL commands
cursor = conn.cursor()

# Read data from the CSV file and insert it into the 'books' table
with open('books.csv', 'r') as file:
    # Create a CSV reader object
    reader = csv.reader(file)

    # Skip the header row
    next(reader)

    # Iterate over each row in the CSV file
    for row in reader:
        # Insert the row data into the 'books' table
        cursor.execute('INSERT INTO books VALUES (?, ?, ?)', row)

# Commit the changes and close the connection
conn.commit()
conn.close()
```

```
import csv
import sqlite3

# Connect to the SQLite database
conn = sqlite3.connect('books.db')
```

This code reads the data from the books.csv file using the csv.reader object, skips the header row, and then iterates over each row in the CSV file. For each row, it executes an SQL INSERT statement to insert the row data into the books table. Finally, it commits the changes and closes the connection to the SQLite database.

6. Select and print the title column from the book table in alphabetical order.

```
import sqlite3

# Connect to the SQLite database
conn = sqlite3.connect('books.db')

# Create a cursor object to execute SQL commands
cursor = conn.cursor()

# Select the title column from the 'books' table in alphabetical order
cursor.execute('SELECT title FROM books ORDER BY title ASC')

# Fetch all the rows from the result set
rows = cursor.fetchall()

# Print the title column
for row in rows:
    print(row[0])

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

This code executes an SQL SELECT statement to select the title column from the books table and orders the results in alphabetical order (ORDER BY title ASC). It then fetches all the rows from the result set and prints the title column for each row. Finally, it closes the connection to the SQLite database.

7. From the book table, select and print all columns in the order of publication.

```
import sqlite3

# Connect to the SQLite database
conn = sqlite3.connect('books.db')

# Create a cursor object to execute SQL commands
cursor = conn.cursor()
```

```
# Select all columns from the 'books' table in the order of publication
cursor.execute('SELECT * FROM books ORDER BY year ASC')

# Fetch all the rows from the result set
rows = cursor.fetchall()

# Print all columns for each row
for row in rows:
    print(row)

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

This code executes an SQL SELECT statement to select all columns (*) from the books table and orders the results by the year column (ORDER BY year ASC). It then fetches all the rows from the result set and prints all columns for each row. Finally, it closes the connection to the SQLite database.

8. Use the sqlalchemy module to connect to the sqlite3 database books.db that you just made in exercise 6.

```
from sqlalchemy import create_engine

# Create an engine to connect to the SQLite database
engine = create_engine('sqlite:///books.db')

# Establish a connection to the database
conn = engine.connect()

# Print a message to confirm the connection
print("Connection to the database established successfully.")

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

This code creates an engine using SQLAlchemy's create_engine() function to connect to the SQLite database books.db. It then establishes a connection to the database and prints a confirmation message. Finally, it closes the connection.

9. Install the Redis server and the Python redis library (pip install redis) on your computer. Create a Redis hash called test with the fields count (1) and name (‘Fester Bestertester’). Print all the fields for test.

```
import redis

# Connect to the Redis server
r = redis.Redis(host='localhost', port=6379, db=0)

# Create the Redis hash 'test' with the specified fields
r.hmset('test', {'count': 1, 'name': 'Fester Bestertester'})

# Print all the fields for 'test'
print("All fields for 'test':", r.hgetall('test'))
```

After running the code, it will connect to the Redis server, create a hash named test with the fields count and name, and set their corresponding values. Then, it will print all the fields for the hash test.

10. Increment the count field of test and print it.

```
import redis

# Connect to the Redis server
r = redis.Redis(host='localhost', port=6379, db=0)

# Increment the 'count' field of 'test' by 1
r.hincrby('test', 'count', 1)

# Print the updated value of the 'count' field
print("Updated count value:", r.hget('test', 'count').decode('utf-8'))
```

This code will connect to the Redis server, increment the value of the count field of the test hash by 1, and then print the updated value of the count field.