**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.**

To set the variable test1 to the string 'This is a test of the emergency text system,' and save it to a file named test.txt in Python, you can use the following code:

test1 = 'This is a test of the emergency text system,'

with open('test.txt', 'w') as file:

file.write(test1)

In this code, test1 is set to the string you provided. The with statement opens the file test.txt in write mode ('w'). The file.write method writes the string stored in test1 to the file, and the file is automatically closed when the with block is exited.

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

To read the contents of the file test.txt into the variable test2 in Python, you can use the following code:

with open('test.txt', 'r') as file:

test2 = file.read()

In this code, the with statement opens the file test.txt in read mode ('r'). The file.read method reads the contents of the file and stores it in the test2 variable. The file is automatically closed when the with block is exited.

To determine if there is a difference between test1 and test2, you can use the equality operator (==) in Python, like this:

if test1 == test2:

print('test1 and test2 are equal.')

else:

print('test1 and test2 are not equal.')

This code checks if test1 and test2 are equal, and if they are, it prints 'test1 and test2 are equal.'. If they are not equal, it prints 'test1 and test2 are not equal.'.

**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**

<https://github.com/snehankitc/Python-basic-assignment-Program/blob/main/Assignment_20_Program_3>

**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).**

<https://github.com/snehankitc/Python-basic-assignment-Program/blob/main/Assignment_20_Program_4>

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

<https://github.com/snehankitc/Python-basic-assignment-Program/blob/main/Assignment_20_Program_5>

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

To select the title column from the books table in the books.db SQLite database in Python and print it in alphabetical order, you can use the sqlite3 module. Here's an example code:

import sqlite3

conn = sqlite3.connect('books.db')

cursor = conn.cursor()

cursor.execute('''

SELECT title

FROM books

ORDER BY title

''')

for row in cursor.fetchall():

print(row[0])

conn.close()

In this code, the sqlite3.connect function is used to connect to the SQLite database file books.db, and the conn.cursor method is used to get a cursor object that you can use to execute SQL commands. The cursor.execute method is used to execute an SQL command to select the title column from the books table and order the result by the title column. The cursor.fetchall method is used to get all the rows from the result set, and the for loop is used to print each title. The conn.close method is used to close the connection to the database.

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

To select all columns from the books table in the books.db SQLite database in Python and print them in the order of publication, you can use the sqlite3 module. Here's an example code:

import sqlite3

conn = sqlite3.connect('books.db')

cursor = conn.cursor()

cursor.execute('''

SELECT \*

FROM books

ORDER BY year

''')

for row in cursor.fetchall():

print(row)

conn.close()

In this code, the sqlite3.connect function is used to connect to the SQLite database file books.db, and the conn.cursor method is used to get a cursor object that you can use to execute SQL commands. The cursor.execute method is used to execute an SQL command to select all columns \* from the books table and order the result by the year column. The cursor.fetchall method is used to get all the rows from the result set, and the for loop is used to print each row. The conn.close method is used to close the connection to the database.

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

To connect to the books.db SQLite database using the sqlalchemy module in Python, you can use the following code:

from sqlalchemy import create\_engine

from sqlalchemy.orm import sessionmaker

engine = create\_engine('sqlite:///books.db')

Session = sessionmaker(bind=engine)

session = Session()

In this code, the create\_engine function from the sqlalchemy module is used to create an engine that will connect to the SQLite database books.db. The sessionmaker function is used to create a session factory, and the bind argument is used to specify the engine to use. The Session class is then used to create a session object that you can use to interact with the database.

With the session object, you can use the following code to select the title column from the books table in alphabetical order:

from sqlalchemy import text

result = session.execute(text("SELECT title FROM books ORDER BY title"))

for row in result:

print(row[0])

In this code, the text function from the sqlalchemy module is used to create a SQL text object that you can use to execute an SQL command. The session.execute method is used to execute the SQL command and get a result set. The for loop is used to print each title.

Note that you should close the session object when you are finished using it by calling session.close().

**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.**

Here is an example of how to create a Redis hash called test with the fields count (1) and name ('Fester Bestertester') using the redis library in Python:

import redis

r = redis.Redis(host='localhost', port=6379, db=0)

r.hmset('test', {'count': 1, 'name': 'Fester Bestertester'})

In this code, the redis.Redis class is used to create a connection to the Redis server running on the same machine (host='localhost') and the default Redis port (port=6379). The db argument is used to specify the database to use (db=0). The hmset method is used to set the values of multiple fields in the test hash.

To print all the fields for the test hash, you can use the following code:

result = r.hgetall('test')

print(result)

In this code, the hgetall method is used to get all the fields and values for the test hash, and the print function is used to print the result. The result will be a dictionary with the field names as keys and the field values as values.

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

Here is an example of how to increment the count field of the test hash and print its value using the redis library in Python:

import redis

r = redis.Redis(host='localhost', port=6379, db=0)

r.hincrby('test', 'count', 1)

result = r.hget('test', 'count')

print(result)

In this code, the redis.Redis class is used to create a connection to the Redis server, just as in the previous example. The hincrby method is used to increment the value of the count field by 1. The hget method is used to get the value of the count field, and the print function is used to print its value.

Note that the hincrby method returns the new value of the field, so the result of the hget method is not necessary in this case. You can simply use the following code to increment the count field and print its value:

import redis

r = redis.Redis(host='localhost', port=6379, db=0)

result = r.hincrby('test', 'count', 1)

print(result)