Slip 11

11.1

// Heart Model (Adaptee)

class HeartModel {

private boolean beating;

public void startBeating() {

System.out.println("Heart is beating");

beating = true;

}

public void stopBeating() {

System.out.println("Heart has stopped beating");

beating = false;

}

public boolean isBeating() {

return beating;

}

}

// Beat Model (Target)

interface BeatModel {

void start();

void stop();

boolean isRunning();

}

// Adapter class to adapt Heart Model to Beat Model

class HeartAdapter implements BeatModel {

private HeartModel heartModel;

public HeartAdapter(HeartModel heartModel) {

this.heartModel = heartModel;

}

@Override

public void start() {

heartModel.startBeating();

}

@Override

public void stop() {

heartModel.stopBeating();

}

@Override

public boolean isRunning() {

return heartModel.isBeating();

}

}

// Test program

public class AdapterPatternTest {

public static void main(String[] args) {

// Create an instance of Heart Model

HeartModel heartModel = new HeartModel();

// Create an adapter for the Heart Model to conform to Beat Model interface

BeatModel beatModel = new HeartAdapter(heartModel);

// Test the Beat Model

System.out.println("Start the Beat Model:");

beatModel.start();

System.out.println("Is the Beat Model running? " + beatModel.isRunning());

System.out.println("\nStop the Beat Model:");

beatModel.stop();

System.out.println("Is the Beat Model running? " + beatModel.isRunning());

}

}

11.2

#Write a python Program to find all null values in given dataset and remove them

import numpy as np

import pandas as pd

dict = {'first score':[100,90,np.nan,95], 'second score':[30,45,56,np.nan], 'third score':[np.nan,40,80,98]}

df=pd.DataFrame(dict)

print(df)

x=df.isnull()

print(x)

y=df.notnull()

print(y)

z=df.fillna(0)

print(z)

s=df.fillna(method='pad')

print(s)

a=df.fillna(method='bfill')

print(a)

b=df.replace(to\_replace=np.nan,value=-99)

print(b)

c=df.dropna()

print(c)

d=df.dropna(axis=1)

print(d)

new\_data=df.dropna(axis=0)

print(new\_data)

11.3

const mysql = require('mysql');

// MySQL connection configuration

const connection = mysql.createConnection({

host: 'localhost',

user: 'your\_username',

password: 'your\_password',

database: 'your\_database\_name',

});

// Connect to MySQL server

connection.connect((err) => {

if (err) {

console.error('Error connecting to MySQL server:', err.message);

return;

}

console.log('Connected to MySQL server');

// Select all records from the "customers" table

const selectQuery = 'SELECT \* FROM customers';

connection.query(selectQuery, (err, rows) => {

if (err) {

console.error('Error selecting records:', err.message);

connection.end();

return;

}

console.log('All records from the "customers" table:');

console.table(rows);

// Delete a specific record from the "customers" table

const customerIdToDelete = 3; // Replace with the ID of the record you want to delete

const deleteQuery = 'DELETE FROM customers WHERE id = ?';

connection.query(deleteQuery, [customerIdToDelete], (err, result) => {

if (err) {

console.error('Error deleting record:', err.message);

connection.end();

return;

}

console.log(`${result.affectedRows} record(s) deleted from the "customers" table`);

// Select all records again to verify the deletion

connection.query(selectQuery, (err, updatedRows) => {

if (err) {

console.error('Error selecting records after deletion:', err.message);

connection.end();

return;

}

console.log('All records after deletion:');

console.table(updatedRows);

// Close the MySQL connection

connection.end((err) => {

if (err) {

console.error('Error closing connection:', err.message);

} else {

console.log('Connection closed');

}

});

});

});

});

});