NodeJS Slip Programs

Unit 2 (Modules)

Note: Use command line arguments

1. Write Node.js application to demonstrate following properties/methods of **buffer** module – concat, compare, copy, equals, fill, includes, indexOf, length, slice, write
2. Write Node.js application(s)
   A. that uses user defined module that contains a function to mask the 10-digit credit card number except last 3 digits.
   B. that uses user defined module that contains a function that displays whether the entered year is a leap year or not.
3. Write Node.js application(s)
   A. that uses user defined module to accept a string and return the count of vowels in that string.
   B. that uses user defined module to check if the entered string or a number is palindrome or not.
4. Write Node.js application(s)
   A. that uses user defined module to return the factorial of a given number.
   B. that uses user defined module circle.js which exports functions area () and circumference () and displays calculated values on the console. Accept radius from the user.
5. Write Node.js application(s)
   A. that uses user defined module to find area of a rectangle and display details on console.
   B. that uses user defined module 'calculator' which has functions – add, subtract, multiply, divide. Accept 2 numbers and an operation from the user.

Unit 3 (NPM)

6. Write Node.js application from the scratch using NPM. Demonstrate the use of third-party dependencies using all NPM commands – add dependency to package.json, install package, update package, uninstall package.

Unit 4 (Web Server)

7. Create Node.js web server to demonstrate – handling HTTP requests, use of query strings and returning HTTP & JSON response.
8. Create a simple web server using Node.js that displays the college information on a web page.
9. Write Node.js application that accepts first name, last name, and date of birth (in dd/mm/yyyy format) from the user. Concatenate first name & last name and calculate the age. Greet the user with a full name along with the age, on a web page. Note: Use readline to accept input from the user.

Unit 5 (File System)

10. Write Node.js application(s)
    A. To demonstrate streams and pipes.
    B. To demonstrate all the methods of readline module.
11. Write Node.js application(s)
    A. To count the number of words in a file and display on the console.
    B. To search a particular word in file and display the count on the console. Accept the word from the user, as a command line argument.
12. Write Node.js application(s)
    A. To count the number of lines in a file and display the same on the console.
    B. To count the number of vowels in a file and display the same on the console.
13. Write separate Node.js application(s)
    A. To check if a folder exists, create a new folder, display contents of an existing folder, rename a folder and remove a folder.
    B. To check whether the given name is a directory or a file. If it is a file, truncate the content after 10 bytes. (Accept the name from the user)
14. Write Node.js application(s)
    A. To accept two file names and the contents from the user and write to two files. Then append contents of a first file to a second file. Display the contents of a second file on the console.
        i. Using streams
        ii. Using synchronous operations
        iii. Using asynchronous operations
15. Write Node.js application(s)
    A. To read contents from two files and merge the contents into a third file by converting it into uppercase. Note: Use synchronous and asynchronous operations

16. Write separate Node.js application(s)
    A. To search a particular word in a file and replace all the occurrences of that word with another word. Accept both the words from the user.
    B. To swap contents of two files. Accept files names as command line arguments.
17. Write separate Node.js application(s)
    A. To display contents of a file in the reverse order. Accept file name as command line argument.
    B. To display alternate characters from a file. Accept file name as command line argument.
18. Write Node.js application to display contents of a file in the reverse order on browser using Node.js web server.

Unit 6 (Events)

19. Write Node.js application that has an EventEmitter which will emit an event that contains information about the application's uptime, every second.
20. Write separate Node.js application(s)
    A. To raise and bind an event by returning EventEmitter object from a function.
    B. To raise and bind an event by extending the EventEmitter class.
21. Write separate Node.js application(s)
    A. To bind 2 listeners to a single event.
    B. To bind custom event 'receive_data' with 'data_receive_handler' function
22. Write separate Node.js application(s) to countdown from 10 seconds to 0. Update the user every second. Notify user when last 2 seconds are remaining and display a message “Time up!” at the end.
    A. Return EventEmitter object from a function
    B. Extend the EventEmitter class

Unit 7 (Database Connectivity)

23. Write Node.js application to demonstrate DDL and DML operations – create database, create table, drop table. Perform insert, update, delete record(s) and display the affected rows for each operation.
24. Write Node.js application to demonstrate - left, right, inner, self joins. Create and populate the necessary tables in MySQL directly.
25. Write Node.js application to create database student_db and student table with fields - rollNo, name, course, totalMarks. Insert 15 records in this table and display the same on the console from highest to lowest marks. Accept course name from the user and display average marks of all the students who have opted for that course.
26. Write Node.js application to create database customer_db and customer table with fields - custID, name, city, state, pincode. Insert 15 records in this table and display the same on the console. Accept city name from the user and delete all the customers living in that city. Display all the records after delete operation.
27. Write Node.js application to create database student_db and student table with fields - rollNo, name, subject, marks. Insert 15 records in this table and display the same on the console. Accept rollNo and subject from the user and increase the marks of that student by 5 for that subject. Display all the records after update operation.
28. Write Node.js application to create database employee_db and employee table with fields - empID, name, designation, department, salary. Insert 15 records in this table and display the same on the console. Increment salary of the employees working in a particular department by 5%. Accept department name from the user. Display all the records after salary increment.
29. Write Node.js application to create two tables product (product_code, product_name, price) and customer (cust_id, cust_name, address, product_code). Display the list of customers who purchased a particular product. Accept product name from the user.
30. Write Node.js application to display a statement of your bank account starting from the most recent transaction. Accept number of transactions to be displayed from the user. Note: Create database, table, and populate with at least 20 records.
31. Create employee and department tables in MySQL directly and populate with records. Write Node.js application to display number of employees in each department and display the highest salary in each department on a web page.
32. Write Node.js application to display your BBA CA marksheet. Accept semester name from the user and display marksheet for that semester. Note: Create the database tables and populate with data in MySQL directly.

33. Write Node.js application to create database hospital_db and tables – patient and doctor. Display the list of patients along with the details of the doctor treating them.
34. Write Node.js application to create database employee_db and employee table. Display employee details (ID, firstName, lastName, designation) as well as their manager details (ID, firstName, lastName, designation).
35. Write Node.js application that creates employee, department tables and display details of an employee with min, max, average salary for a given department. Accept department name from the user.
36. Write Node.js application to create customer table and populate it with at least 15 records. Fetch all the records from this table and write them to a file. Then read the contents of the file and display on the console.
37. Write Node.js application to create course and student tables. Populate with at least 15 records. Display course name and the number of students taking the course, only if the course has five or more students enrolled.
38. Write Node.js application to create employee, department tables. Populate with at least 15 records. Display the result to show name of the department and the age of the department's youngest employee.
39. Write Node.js application that creates employee, department tables. Populate with at least 15 records. Display the details of all the employees who joined after 1st Dec 2022.
40. Write Node.js application that creates movie table. Populate with at least 20 records. Display the details of all the movies of a particular genre. Accept genre from the user.
41. Write Node.js application that creates student table. Populate with at least 20 records. Display the details of all the students who have their personal email on Gmail.
42. Write Node.js application that creates customer and order tables. Populate with at least 15 records. Display details of the customers who have placed at least one order.
43. Write Node.js application that displays marksheet of a student on a web page after accepting the roll number from the user. Note: Create necessary tables in MySQL directly.

---------- *All The Best* ----------