NODE JS

1- Event-Driven Architecture:
- Node.js employs an event-driven architecture where actions or events trigger callbacks.
- This enables handling multiple connections simultaneously without blocking the execution thread.

2- Asynchronous Programming:
- Node.js relies heavily on asynchronous programming to handle I/O operations efficiently.
- Asynchronous functions are non-blocking, allowing other operations to continue while waiting for I/O operations to complete.

3- npm (Node Package Manager)
- npm is the default package manager for Node.js, used for installing, sharing and managing packages or modules of reusable code.
- It provides a vast ecosystem of libraries and tools to extend Node.js functionalities.

4- Modules:
- Node.js follow the CommonJS module system, allowing modularization of code into separate files.
- Modules encapsulate code, making it reusable, maintainable and easier to manage.
- Modules are imported and exported using 'require()' and 'module.exports' respectively.

5- CommonJS Module system:
- CommonJS is a module specification for JavaScript, defining a standard way to structure and organize code into modules.
- Modules are loaded synchronously, ensuring dependencies are resolved before execution.
- Each module has its own scope, preventing global namespace pollution.

6- Callbacks:
- Callbacks are functions passed as arguments to other functions and executed later upon completion of an operation.
- In Node.js, callbacks are commonly used to handle asynchronous operations, such as reading files or making network requests.

7- Promises:
- Promises are objects representing the eventual completion or failure of an asynchronous operation.
- They provide a cleaner alternative to callbacks for handling asynchronous code, especially for avoiding callback hell.
- Promises support chaining and allow better error handling through '.then()' and '.catch()' methods.

8- Async/Await:
- Async/Await is a modern JS feature that provides a more concise and readable syntax for working with async code.
- Async functions return promises implicitly, allowing the use of 'await' keyword to pause execution until a promise is resolved.
- Async/await simplifies error handling by allowing try/catch blocks around asynchronous code.

9- Event Emitters:
- Node.js includes the EventEmitter class, which facilitates implementing custom event-driven architectures.

Date: _____ Page: _____

- Event Emitters emit named events that cause registered callbacks (listeners) to be called.
- They are commonly used for building event-driven applications, such as web servers and real-time communication systems.

10- Streams:

- Streams are objects used for reading or writing data sequentially in Node.js
- They provide an efficient way to handle large datasets or continuous data flows, such as file I/O and network comms.
- Streams can be readable, writable or duplex (both readable and writable).

11- Express.js:

- Express.js is a popular web application framework for Node.js, known for its simplicity and flexibility.
- It provides a robust set of features for building web servers and APIs, including routing, middleware support, and template engines.
- Express.js simplifies common tasks like handling HTTP requests, parsing request bodies, and setting response headers