NodeJS

1.Who developed NodeJS ?
Node.js was developed in 2009 by Ryan Dahl

2.What is NodeJS ?
NodeJS is a web application framework built on Google chrome's Javascript v8 engine.
NodeJS comes with runtime environment on which a Javascript based scripts can be interpreted and executed.
NodeJS provides a rich library of various Javascript modules which eases the development of a web application.
NodeJS does asynchronous I/O operations which are non-blocking (For this, NodeJS is making use of c++ libraries called libev, libuv and libeio)
NodeJS provides a way to build REST APIs easily and faster.

3.What are the advantages of NodeJS ?
NodeJS is Asynchronous and Event Driven (All APIs of NodeJS library are asynchronous that is non-blocking,it means a NodeJS based server never waits for an API to return data. NodeJS based server moves to the next API after calling an API and the response of the previous API will get received by NodeJS by a notification mechanism of Events of NodeJS)
NodeJS library is very fast in terms of code execution
NodeJS is single threaded but highly scalable (NodeJS uses a single threaded model with event looping,this event looping mechanism helps NodeJS server to respond to requests in a non-blocking way and hence makes the NodeJS server highly scalable.Traditional servers like Apache HTTP server creates a limited threads to respond to the requests.
NodeJS application never buffers any data,it simply outputs the data in chunks.

4.What is REPL in context of Node ?
REPL stands for Read Eval Print Loop
REPL represents a computer environment like a window console where a command is entered and system responds with an output.
Read - Reads user's input,parse the input into Javascript data-structure and stores in memory
Eval - Takes and evaluates the data structure
Print - Prints the result
Loop - Loops the print command until user press ctrl-c twice

5.Can we evaluate simple expression using REPL ?
yes

6.What is the difference of using var and not using var in REPL ?
If var keyword is not used then the value is stored in the variable and printed.If var keyword is used then the value is stored in the variable but not printed and to print the value use console.log( )

7.List out REPL command ?
ctrl+c - for terminating the current command
ctrl+c twice - for terminating REPL
ctrl+d - for terminating REPL
Tab keys - lists all the current commands
.break - exits from multiline expression
.save with filename - save REPL session to a file

8.What is the use of underscore variable in REPL ?
underscore variable is used to get the last result.

9.Explain NPM in NodeJS ?
NPM provides two functionalities :
online repositories for NodeJS modules
dependency management,version management,and command line utility for installing NodeJS modules

10.What is global installation of dependencies ?
Globally installed dependencies are stored in user-directory / npm directory.
Globally installed dependencies can be used in CLI (command line interface) function of any NodeJS but cannot be imported using require( ) in Node application directly.

11.What is local installation of dependencies ?
By default npm installs the dependencies locally.
Locally installed dependencies are stored in node_modules directory lying in the folder where node application is present.
Locally installed dependencies can be imported using require( ) in Node application directly.

12.How to check the already installed dependencies which are globally installed using npm ?
npm ls -g

13.What is package.JSON file ?
package.JSON file is present in the root directory of any Node application.
package.JSON file is used to define the properties of a package.

14.Name some of the attributes of package.JSON file ?
name - name of the package
version - version of the package
description - description of the package
homepage - homepage of the package
author - author of the package
contributors - name of the contributors to the package
dependencies - list of dependencies. npm automatically installs all the dependencies mentioned here in the

node_modules folder of the package
repository - repository type and url of the package
main - entry point of the package
keywords

15.How to uninstall a dependency using npm ?
npm uninstall <dependency-name>

16.How to update a dependency using npm ?
npm update <dependency-name>

17.What is a callback ?
callback is an asynchronous equivalent for a function.
A callback function is called at the completion of a given task.
Node makes heavy use of callbacks.
All APIs of node are written in such a way that they support callbacks.

18.What is a blocking code ?
If an application has to wait for some I/O operation in order to complete its execution any further, then the code responsible for waiting is called blocking code.

19.How Node prevents blocking code ?
Node prevents blocking code by providing a callback function.

20.What is an event loop ?
An event loop listens for events and then triggers a callback function when one of those events is detected.

21.What is the difference between an event and a callback function ?
A callback function is called when an asynchronous function returns its result.
when an event gets fired,its listener function starts executing.

22.What is an EventEmitter ?
EventEmitter class lies in events module.
var events = require (‘events’);
var eventEmitter = new events.EventEmitter( );
when an EventEmitter instance faces any error,it emits an ‘error’ event.
when a new listener is added, ‘newListener’ event is fired.
when a listener is removed, ‘removeListener’ event is fired.
EventEmitter provides on property and emit property.
on property is used to bind a function with the event.
emit property is used to fire an event.

23.What is a Buffer class in NodeJS ?
A buffer class is a global class which can be accessed in an application without importing a buffer module.

24.How can we convert Buffer to JSON ?
buffer.toJSON( ) is used to convert Buffer to JSON

25.How can we concatenate Buffers in NodeJS ?
Buffer.concat ([Buffer1, Buffer2]);

26.How can we compare Buffers in NodeJS ?
Buffer1.compare (Buffer2);

27.How can we copy Buffers in NodeJS ?
buffer.copy (targetBuffer [, targetStart] [, sourceStart] [, sourceEnd]);

28.Which module is used for file based operations ?
fs module is used for file based operations.
var fs = require (‘fs’);

29.Which module is used for buffer based operations ?
buffer module is used for buffer based operations.
var buffer = require (‘buffer’);

30.Which module is used for web based operations ?
http module is used for web based operations
var http = require (‘http’);

31.Does fs module provide both synchronous and asynchronous methods ?
yes

32.What is the difference between synchronous and asynchronous methods of fs module ?
Every method in fs module has both synchronous and asynchronous forms.
Asynchronous read :

```
var fs  = require (‘fs’);
fs.readFile (‘fileName’ , function (err , data) {
  if (err) {
return console.error (err);
}
console.log ( data.toString( ) );
} );
```

Asynchronous method takes the last parameter as completion function callback and first parameter of the callback function is error.It is preferred to use asynchronous methods over synchronous method as asynchronous method never blocks the program execution.

33.What is piping in Node.js ?
Piping is the mechanism of providing the output of one stream as the input of another stream.There is no limit on piping operations.

```
var fs  = require (‘fs’);

//create a readable stream
var readerStream = fs.createReadStream (‘input.txt’);

//create a writable stream
var writerStream = fs.createWriteStream (‘output.txt’);

//pipe the read and write operations
//read input.txt and write data to output.txt
readerStream.pipe(writerStream);
```

34.What is chaining in NodeJS ?
chaining is a mechanism of connecting output of one stream to another stream and create a chain of multiple stream operations.It is normally used with piping operations.

```
var fs = require (‘fs’);
var zlib = require (‘zlib’);

//compress the file input.txt to input.txt.gz
fs.createReadStream (‘input.txt’);
  .pipe ( zlib.createGzip( ) )
  .pipe ( fs.createWriteStream (‘input.txt.gz’) );
```

35.Why to use _filename in NodeJS ?
_filename is used to represent the filename of the code which is being executed.
_filename is used to resolve the absolute path of the file.

```
console.log ( _filename );
```

36.Why to use _dirname in NodeJS ?
_dirname is used to represent the name of the directory that the currently executing scripts resides in.

37.Why to use setTimeout in NodeJS ?
setTimeout is a global function and is used to run the callback after some milliseconds.

```
setTimeout (callback method, milliseconds);
```

38.Why to use clearTimeout in NodeJS ?
clearTimeout is a global function and is used to stop a timer which was created during setTimeout( )

39.What are streams ?
streams are objects that let you read data from a source or write data to a destination in continuous fashion.

40.How many types of streams are present in NodeJS ?
Node.js has four types of streams :
Readable stream - used for read operation
Writable stream - used for write operation
Duplex stream - used for both read and write operation
Transform stream - is a type of duplex stream where the output is computed based on input

41.Name some of the events fired by streams ?
A stream is an instance of an EventEmitter.
The events fired by streams are :
data event - is fired when there is data available to read
end event - is fired when no data is available to read
error event - is fired when there is any error reading or writing data
finish event - is fired when all the data has been flushed to underlying stream

42.What is the difference between tilde (~) and caret (^) ?
tilde matches the most recent minor version (the middle number)
caret matches the most recent major version (the first number)

43.What is an error-first callback ?
error-first callback is used to pass error and data.The first argument is always an error object that the programmer has to check if something went wrong.Additional arguments are used to pass data.

```
fs.readFile ( file path, function (err , data) {
if (err) {
  //handle the error
}
//use the data object
} ) ;
```

44.How to avoid callbacks hells ?
callback hells can be avoided by breaking callbacks into independent functions or using promises or using yield with Generators / promises.

45.What tools are used to ensure code consistency in styling ?
JSLint
JSHint
ESLint
JSCS

46.What is npm shrinkwrap ?
npm shrinkwrap command locks down the versions of a package's dependencies so that you can control exactly which versions of each dependency will be used when your package is installed.

47.Explain console in NodeJS ?
console is a global object and is used to print information on stdout and stderr.
console is used in synchronous manner incase of destination is either a file or terminal.
console is used in asynchronous manner incase of destination is a pipe.

48.Explain process in NodeJS ?
process is a global object and is used to get information on current process.process provides multiple events related to process activities.

49.Define OS module ?
OS module provides basic operating-system related utility functions.

50.What is the property of OS module ?
OS.EOL - is used for defining appropriate end of line marker for OS.

51.Define Path module ?
Path module provides utilities for handling and transforming file paths.

52.Define Net module ?
Net module is used for creating both clients and servers.
Net module acts as a network wrapper.

53.How to open a file ?
fs.open (path , flags , mode , callback)
path - is the string having file name including path
flags - is the behavior of the file to be opened
mode - sets the file mode but only if the file was created.The default mode is 0666,that is readable and writable
callback - has two parameters err and fd

54.How to read a file ?

fs.read (fd, buffer, offset, length, position, callback)
fd - is the file descriptor returned by fs.open( )
buffer - data is written to this buffer
offset - is the offset in the buffer to start writing at
length - specifies the number of bytes to read
position - specifies where to begin reading from in the file.If the position is null,data will be read from the current file position
callback - has three parameters err , bytesRead, buffer

55.How to write a file ?
fs.writeFile (path, data, utf-8, callback)
path - is the string having file name including path
data - is the string to be written to the file
callback - has one parameter err

56.How to close a file ?
fs.close (fd, callback)
fd - is the file descriptor returned by fs.open( )
callback - has no parameters other than a possible exception is given to the completion callback

57.How to get information about a file ?
fs.stat (path, callback)
path - is the string having file name including path
callback - has two parameters err and stats

58.How to truncate a file ?
fs.truncate (fd, len, callback)
fd - is the file descriptor returned by fs.open( )
len - is the length of the file after the file truncated
callback - has no parameters other than a possible exception is given to the completion callback

59.How to delete a file ?
fs.unlink (path, callback)
path - is the string having file name including path
callback - has no parameters other than a possible exception is given to the completion callback

60.How to create a directory ?
fs.mkdir (path, mode, callback)
path - is the string having file name including path
mode - sets the directory mode.Default is 0777
callback - has no parameters other than a possible exception is given to the completion callback

61.How to delete a directory ?
fs.rmdir (path, callback)
path - is the string having file name including path
callback - has no parameters other than a possible exception is given to the completion callback

62.How to read a directory ?
fs.readdir (path, callback)
path - is the string having file name including path
callback - has two parameters err and files

63.What is Semantic Versions (semver) ?
Semver uses three part version number like 3.9.2 and call these three numbers from left to right as the major, minor and patch numbers.
backward incompatible change increments the major number
new functionality that is backwards compatible increments the minor number
simple bug fix to existing functionality increments the patch number
For any dependency the release 1.0.0 is considered the first stable release and the semver contract does not apply to releases before it.

64.What is the difference between npm and Bower ?
The main difference between npm and Bower is the approach for installing package dependencies. npm installs dependencies for each package separately, and as a result makes a big package dependency tree (node_modules/grunt/node_modules/glob/node_modules/...), where there could be several version of the same package. For client-side JavaScript this is unacceptable: you can't add two different version for jQuery or any other library to a page. With Bower each package is installed once (jQuery will always be in the bower_components/jquery folder, regardless of how many packages depend on it) and in the case of a dependency conflict, Bower simply won't install the package incompatible with one that's already installed.
npm is most commonly used for managing Node.js modules, but it works for the front-end too when combined with Browserify and/or $ npm dedupe.
The reason many projects use both is that they use Bower for front-end packages and npm for developer tools like Yeoman, Grunt, Gulp, JSHint, CoffeeScript, etc.
In short, npm aims for stability. Bower aims for minimal resource load

65.What is package.JSON file ?
All npm packages contain a JSON file which is located at the root of the project.This JSON file is the package.JSON file.It holds various metadata (project description,project dependencies,project version,license information,etc) relevant to the project.It allows npm to identify the project as well as handle the project’s dependencies.

66.Explain NPM in NodeJS?
NPM stands for Node Package Manager (npm) and there are two functionalities which NPM takes care of mainly and they are :
Online repositories for node.js modules or packages, which can be searched on search.nodejs.org
Dependency Management, Version Management and command line utility for installing Node.js packages.

67.What is .gitignore ?
.gitignore file has a list of files you want git to ignore in your work directory.
If you're on a Mac and you have .DS_Store files in all your directories. You want git to ignore them, so you add .DS_Store as a line in .gitignore.

68.Explain .git directory ?
.git directory has
config (text file) - The configuration file for the local git repository
HEAD (text file) - The HEAD file has the value ref : refs / heads / master
objects (directory) - the files are stored as blobs and the directories are stored as git trees in this directory
refs (directory) - Everything under the refs directory is references to a commit for a branch of some type (either a local branch or a remote tracking branch)
refs / heads (directory) - Files in the refs / heads directory are branch names
refs / heads / master (text file) - is the most recent commit on this branch (master)
refs / heads / v1 (text file) - is the most recent commit on this branch (v1)
refs / remotes (directory) - Everything under the refs / remotes directory is references to a commit for a remote-tracking branch
refs / remotes / origin (directory) - The remote tracking branches for the remote repository origin are stored in this directory
refs / remotes / origin / master (text file)
refs / remotes / origin / v1 (text file)

69.List npm commands ?
npm install <module-name> - - save - to install a package and also update package.json with the installed version and package name
npm install <module-name> - - save - dev - to install a package and also update package.json with the installed version and package name,but into the devDependencies section
npm config set save true - to set save as a default for npm install
npm init - to generate a package.json file
npm config ls -l - to list all npm configuration flags
npm update npm -g - to update the global npm version
npm docs <module-name> - to display the readme.md of a module in the browser
npm uninstall <module-name> - to uninstall a package
npm edit <module-name> - to locally edit a dependency

70.what is private: true in package.JSON file ?
If we set “private: true” in package.JSON file,then npm will refuse to publish your package or repository.This a way to prevent accidental publication of private repositories.

71.what is Dev dependencies ?
The packages mentioned in Dev dependencies need to be installed while developing your module like test frameworks.

72.what is npm install - -production ?
when deploying your application,you want “npm install” to be as fast as possible,so we use npm install - -production