**1. What is the purpose of the HTTP module in Node.js?**

The http module in Node.js allows you to create web servers and handle HTTP requests and responses. It's used to build APIs, websites, and communicate over the web.

**2. How can we create an HTTP server to return multiple products in JSON format?**

```
const http = require('http');


const products = [
  { id: 1, name: 'Laptop' },
  { id: 2, name: 'Phone' }
];


const server = http.createServer((req, res) => {
  if (req.url === '/products') {
    res.writeHead(200, { 'Content-Type': 'application/json' });
    res.end(JSON.stringify(products));
  } else {
    res.writeHead(404);
    res.end('Not Found');
  }
});


server.listen(3000);
```

**3. How to use query string in URL to implement a search filter in Node.js?**

```
const http = require('http');
const url = require('url');


const products = [
  { id: 1, name: 'Laptop' },
  { id: 2, name: 'Phone' }
];


http.createServer((req, res) => {
  const parsedUrl = url.parse(req.url, true);
  if (parsedUrl.pathname === '/search') {
```

```
    const keyword = parsedUrl.query.name;
    const result = products.filter(p => p.name.toLowerCase().includes(keyword.toLowerCase()));
    res.writeHead(200, { 'Content-Type': 'application/json' });
    res.end(JSON.stringify(result));
  }
}).listen(3000);
```

**4. Explain how dynamic routing works in a Node.js HTTP server with an example.**

Dynamic routing matches URL patterns like /product/1, /product/2, etc., using string manipulation.

**Example:**

```
http.createServer((req, res) => {
  const urlParts = req.url.split('/');
  if (urlParts[1] === 'product' && urlParts[2]) {
    const id = urlParts[2];
    res.writeHead(200, { 'Content-Type': 'text/plain' });
    res.end(`Product ID: ${id}`);
  }
}).listen(3000);
```

**5. Write code to handle /product/:id route and return product details.**

```
const http = require('http');

const products = [
  { id: 1, name: 'Laptop' },
  { id: 2, name: 'Phone' }
];

http.createServer((req, res) => {
  const parts = req.url.split('/');
  if (parts[1] === 'product' && parts[2]) {
    const id = parseInt(parts[2]);
    const product = products.find(p => p.id === id);
    if (product) {
      res.writeHead(200, { 'Content-Type': 'application/json' });
      res.end(JSON.stringify(product));
```

```
  } else {
    res.writeHead(404);
    res.end('Product not found');
  }
 }
}).listen(3000);
```

**6. What is the use of fs.writeFile()? Write a sample code to create a text file.**

fs.writeFile() is used to create or overwrite a file with given content.

**Example:**

```
const fs = require('fs');

fs.writeFile('example.txt', 'Hello Node.js!', (err) => {
 if (err) throw err;
 console.log('File created');
});
```

**7. How can you read data from a file using fs.readFile()? Provide an example.**

```
const fs = require('fs');

fs.readFile('example.txt', 'utf8', (err, data) => {
 if (err) throw err;
 console.log(data);
});
```

**8. Write a Node.js code to delete a file using the fs module.**

```
const fs = require('fs');

fs.unlink('example.txt', (err) => {
 if (err) throw err;
 console.log('File deleted');
});
```

**9. How to check all network interfaces of your system using the os module?**

```
const os = require('os');

console.log(os.networkInterfaces());
```

**10. What information does os.platform() provide? Write a short explanation.**

os.platform() returns the operating system platform:

- 'win32' for Windows
- 'darwin' for macOS
- 'linux' for Linux

**Example:**

```
const os = require('os');
console.log(os.platform());  // e.g., win32
```

**11. How to parse query strings using Node.js built-in modules? Give an example.**

Use the url and querystring modules:

```
const url = require('url');
const querystring = require('querystring');

const parsedUrl = url.parse('/search?name=Laptop');
const query = querystring.parse(parsedUrl.query);

console.log(query.name);  // Output: Laptop
```

**12. Create a folder called data and save a product list JSON file using the fs module.**

```
const fs = require('fs');

const products = [
  { id: 1, name: 'Laptop' },
  { id: 2, name: 'Phone' }
];

fs.mkdir('data', { recursive: true }, (err) => {
  if (err) throw err;
  fs.writeFile('data/products.json', JSON.stringify(products), (err) => {
```

```
    if (err) throw err;
    console.log('File saved');
  });
});
```

**13. Create a Node.js HTTP server that reads product data from a file and returns it on /products.**

```
const http = require('http');
const fs = require('fs');

http.createServer((req, res) => {
  if (req.url === '/products') {
    fs.readFile('data/products.json', 'utf8', (err, data) => {
      if (err) {
        res.writeHead(500);
        res.end('Error reading file');
        return;
      }
      res.writeHead(200, { 'Content-Type': 'application/json' });
      res.end(data);
    });
  }
}).listen(3000);
```

**14. Explain how url module helps in dynamic routing and parsing query parameters.**

The url module parses URLs into components like pathname and query. It helps identify dynamic paths and extract query values.

**Example:**

```
const url = require('url');

const parsedUrl = url.parse('/product/1?color=red', true);
console.log(parsedUrl.pathname);  // '/product/1'
console.log(parsedUrl.query.color);  // 'red'
```

**15. List and explain at least 3 properties/methods from the os module used in system checks.**

1. **os.platform()** – Returns the OS platform ('win32', 'linux', etc.).
2. **os.totalmem()** – Returns total system memory in bytes.
3. **os.freemem()** – Returns available free memory in bytes.

**Example:**

```
const os = require('os');

console.log(os.platform());

console.log(os.totalmem());

console.log(os.freemem());
```