Q1: What is npm? npm is the world's largest software registry, hosting over 800,000 code packages. It's the package manager for Node.js, enabling developers to share and reuse code.

Q2: How to initialize npm?

- `npm init`: Creates a `package.json` file with default settings.
- `npm init -y`: Creates a `package.json` file with default settings, skipping prompts.

Q3: What are Parcel/Webpack? Why do we need them?

Parcel and Webpack are bundlers that combine multiple JavaScript files into a single file for efficient delivery. They optimize code by minifying, removing unused code (tree shaking), and improving performance.

Q4: Key benefits of Parcel/Webpack?

Faster load times, smaller file sizes, improved code organization.

Q5: How to install Parcel?

Bash

```
npm install -D parcel
```

- `-D` flag installs Parcel as a development dependency.

Q6: Parcel commands

- `npx parcel <entry_point>`: Starts a development server.
- `npx parcel build <entry_point>`: Creates a production build.

Q7: What is the .parcel-cache folder?

The `.parcel-cache` folder stores information about your project to speed up subsequent builds. It improves build performance by caching intermediate results.

Q8: What is npx?

npx is a package runner that executes packages from the npm registry without installing them globally. It's useful for trying out tools or running scripts from packages.

Q9: Difference between dependencies and devDependencies?

- **dependencies:** Packages required by your application in production.
- **devDependencies:** Packages used for development and testing, not needed in production.

Q10: What is Tree Shaking?

Tree shaking is a process that removes unused code from your production build, resulting in smaller and faster applications.

Q11: What is Hot Module Replacement (HMR)?

HMR updates parts of an application without a full reload, improving development speed and experience.

Q12: Parcel's Superpowers

Parcel excels in:

- Development speed due to its fast build times and HMR.
- Code splitting for efficient loading of large applications.
- Image optimization for improved performance.
- Tree shaking for smaller bundle sizes.

Q13: What is .gitignore?

`.gitignore` is a file that tells Git which files or directories to ignore when committing changes. It helps prevent unnecessary files from being included in your repository.

Q14: Difference between package.json and package-lock.json?

- **package.json:** Describes your project, its dependencies, and scripts.
- **package-lock.json:** Stores exact versions of installed dependencies to ensure reproducibility.

Q15: Why avoid modifying package-lock.json?

Modifying `package-lock.json` manually can break dependency management. It's automatically generated to maintain dependency consistency.

Q16: What is node_modules?

`node_modules` is a folder containing installed packages and their dependencies. It's generally not committed to version control due to its size and the ability to recreate it from `package.json`.

Q17: What is the dist folder?

The `dist` folder contains the production-ready build of your application. It includes minified and optimized code.

Q18: What is Browserslist?

Browserslist defines the target browsers for your project, allowing build tools to optimize code accordingly.

Additional Notes:

- Consider using a linter to enforce code style and catch potential errors.
- Explore additional build tools like Webpack for more complex projects.
- Optimize images and other assets for performance.