1. data绑定数据，methods绑定方法，方法的this指向vue实例。注意：vue实例指的是该vue对象，data是vue对象的一个属性，即使数据不写在data属性中也能被访问到，data存储数据只是习惯，通过methods的方法的this可以操作vue对象
2. @click绑定事件，可以直接用双引号写函数体，或双引号内写函数名，@click是v-on的缩写
3. Computed设置某个数据需要依赖其它数据的计算结果
4. 往模板内容中绑定数据使用插值语法{{data}}，如果往标签属性中绑定数据用v-bind。V-bind单向绑定，简写为：
5. V-bind绑定类名，常用方法：class传入一个对象，key为类名，值为布尔值，true使用该类名，false不使用该类名。也可以再通过普通定义类名的方式增加类名
6. 可以用methods的方法返回class对象，在标签的class属性中调用方法（不是传入回调）
7. 类名的数组定义方式可以定义多个类名，字符串形式是固定的，变量形式是使用data的。
8. ！！！一切data中的数据都是用变量形式引用，不加引号
9. ！！！！！methods的方法调用时，方法没有参数，回调加不加括号都可，方法有参数，回调加括号带参数正常，回调加括号无参数传入undefined，回调不加括号传入event对象。也可以用$event获取event对象
10. 单页面应用SPA其实就是单个路由管理的应用
11. 导入导出和react一样。默认导出可以重命名，自定义组件必须大写，和html标签区分开，js后缀可省略，文件夹中index作为默认导入的文件，导入时可省略
12. 导入组件使用@，@表示根目录src
13. 配置路由。<router-view></router-view>配置子路由的出口，即子路由显示在哪个地方（vuepro和reactpro入口设置不一样，reactpro入口文件是一个类，路由在里面作为一个组件，vuepro入口是一个模板，但是内容只有一个路由，即路由把所有内容都包裹了，reactpro的路由只包裹了产品内容，header没有在路由中）
14. 定义组件：上面template写html标签，下面script写脚本：（1）导入组件，component对象注册组件，设置props等对象
15. 导出组件使用name定义组件名，默认导出就不加，导入时定义（首字母大写）
16. 默认导出如果不需要任何数据和方法，可以直接导出一个对象
17. Vuepro项目views文件夹结构：index是主页面内容的容器，包含了header（独立组件）和下面的商品内容（子路由）等，login和user页面和index并列，src下的app.vue是所有路由的容器，index.vue是入口文件，创建vue实例并且使用router
18. 存在的问题：script内容代码不熟练，路由不熟练，vue3入口文件配置√，vue.use错误，页面显示不出来
19. ！！！！！易混点：router-view作为子路由显示的地方（即出口），跳转到哪个路由时，就显示哪个路由对应的组件，别的组件不显示，显示的地方就是router-view标签的位置（如果还有其它标签就一起显示，否则只显示子路由的内容，和以前react native的思路一样），reactpro没有用这种方式，直接用路由管理标签包裹，用route标签设置路由对应的组件
20. Vue public文件夹下的文件是静态资源，类似node设置的静态资源
21. Vuepro sass-loader：使用vue2，降了sass-loader版本到10解决问题
22. SEO：搜索引擎爬取页面内容（在初次访问该页面时抓取，只抓取一次），存储到服务器上，用户搜索时找到对应的标签的文档
23. Npx 可以直接加上脚手架工具并且执行脚手架工具
24. Nuxt：assert和static都是静态资源文件，static不会被webpack打包。.nuxt是源文件编译后的文件
25. Nuxt根据pages目录结构自动生成路由，所以vue项目页面结构最好也要写规范
26. Vue组件插值语法中的变量找不到的时候不会渲染组件
27. 插值语法：使用双括号进行插入变量。指令语法：v-if直接使用，v-bind:指定属性名，v-model直接使用。原理：都要使用v-bind绑定属性，只是v-model和其它直接使用的指令进行了封装，所以可以不指定属性，直接使用
28. ！！！！！生命周期：beforeCreate和created创建了data，没有el，created和beforeMount判断是否有el，没有就停止编译，判断是否有内部template，渲染的优先级顺序是render函数>内部template>外部templa，beforeMount和mount创建了组件的$el
29. Vue异步更新DOM实际上是异步更新data，类似react的异步更新state
30. Vue项目启动加载过程：1、（宏观）加载index.html文件，加载main.js文件，创建vue实例对象并把App组件的模板挂载到index.html的指定节点上。2、（微观）index.js中创建实例对象后，配置对象中没有设置el，等待$mount执行，然后通过app.$mount(‘#app’)挂载到节点，此时执行App组件的生命周期函数，加载子组件时分别执行各自的钩子函数
31. Vue3生命周期和vue2对比：销毁的钩子名变为unmount，钩子函数名前面加一个on，没有create钩子，换为了setup
32. Vue3 setup：返回对象作为vue实例的data、method等
33. Vue：computed：定义的方法名作为模板中使用的变量，方法返回值是变量值，变量是独立于data的，data中没有这个变量。Watch：方法名定义了要监听的变量，即变量是data中的，不是新定义的，方法的两个参数是新值和旧值。Filter：定义一个方法，方法接收的参数是需要处理的变量值，方法返回值是该变量的新值。Filter类似一个钩子函数，在模板展示变量之前处理一下变量值，并不是过滤作用
34. Vue3 watch：改用watch函数的方式监听，第一个参数是监听的变量，第二个参数是监听回调，第三个参数是配置对象。监听的变量可以使用回调函数返回，监听多个变量用数组
35. Vue3：computed：可以定义只读变量和可读写变量，vue2只能定义只读变量。使用方法本质上和vue2一样，只是定义的变量使用computed函数处理而不是在computed函数中定义变量的形式
36. Vue3 setup把用到的变量和方法返回即可，即只需要返回vue2的data和methods内容，watch和computed配置后即生效，不需要返回
37. **Vue内容梳理：vue2和vue3使用方法、vue3 setup（setup中定义响应式数据、定义watch、computed等），生命周期及名称，响应式原理（vue2修改对象和数组及原理），双向绑定原理（用户输入和vue数据的双向绑定不适合object.defineProperty）**
38. NextTick原理：类似JavaScript事件循环，主线程之外有一个watcher队列（可理解为JavaScript的任务队列，但是一个数据被改变多次后只按最后一次处理）和nextTick回调任务队列，主线程执行完先执行watcher队列中的任务，更新UI，再执行nextTick回调队列中的任务