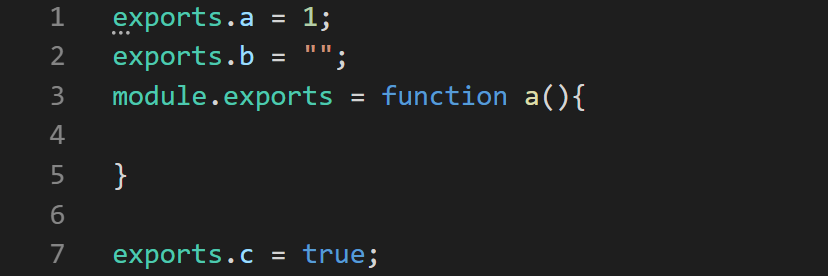
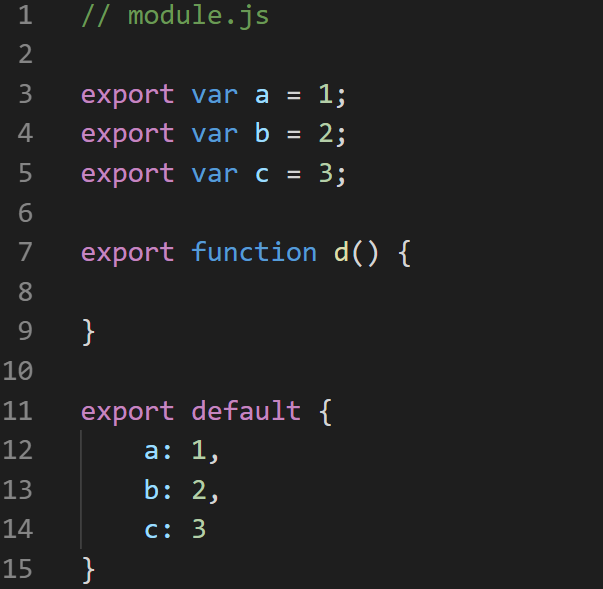
# 模块化考试题

1. **选择题（均为单选题，每小题6分，共8题， 48分）**
2. 关于模块化说法**错误**的是（ C ）  
   A. 解决了依赖关系混乱的问题  
   B. 解决了全局变量污染的问题  
   C. 解决了JS文件数量增加后导致传输缓慢的问题  
   D. 使得功能划分更加精细，工程更加便于维护
3. 常见的**模块化规范不包含**（ C ）  
   A. CommonJS  
   B. ADM  
   C. require.js  
   D. ES6 模块化
4. CommonJS的特点**不包含**（ D ）  
   A. 不污染全局变量  
   B. 使用exports导出  
   C. 可以导出任意类型的数据   
   D. 使用import导入
5. 如果使用CommonJS导入下面的模块，导入的结果类型是（ B ）  
     
   A. number  
   B. function  
   C. object  
   D. udefined
6. ES6中的基本导出和默认导出说法正确的是（ A ）  
   A. 基本导出可以有多个，默认导出只能有一个  
   B. 如果使用\*导入整个模块，只包含基本导出  
   C. 默认导出无法使用基本导出的格式  
   D. 基本导出和默认导出无法共存
7. CommonJS难以应用到浏览器端，主要的困难是（ A ）  
   A. CommonJS的导入是同步，而浏览器获取JS文件是异步的  
   B. CommonJS缺乏多种导出方式  
   C. 浏览器难以实现缓存  
   D. 浏览器无法识别入口文件
8. 从下面的ES6模块中，导入a、b和默认，下面的代码**正确**的是（ B ）  
     
   A. import a, b, default from “./module.js”  
   B. import default, {a, b} from “./module.js”  
   C. import c, {a, b} from “./module.js”  
   D. import {a, b, c as default} from “./module.js”
9. 对于 import “./a.js”，说法正确的是（ C ）  
   A. 模块a中一定没有任何导出  
   B. 该语句一定会导致模块a中的代码执行一次(纠错:因缓存,第二次导入就不执行了)  
   C. 该语句没有从a模块中导入任何内容  
   D. 该语句语法是错误的

二、**填空题（每空2分，共6空， 12分）**

CommonJS使用函数 require() 对模块进行导入。函数的参数是模块路径，在node环境中，如果要导入本地模块，路径必须以 ./ 或 ../ 开头，这一点，和ES6的标准一致。

浏览器为了实现ES6模块化，同时兼容之前的应用程序，因此选择在引入js文件的script元素中添加 type=” module ”，来代表引入的是一个模块。

在ES6模块内部，使用 export 导出 数据， export default 来实现默认导出，

使用 import 变量名 from “模块路径”来导入一个模块

**三、简答题（共2题，每题20分， 40分）**

1. 阐述模块化（非ES6模块化）应用到浏览器端需要解决哪些问题，如何解决？

问题：

1. CommonJS导入模块是同步的，导入了文件才执行下面的代码。但是浏览器导入模块，需要远程请求，同步加载会阻塞代码。
2. 模块化隐藏内部实现，要把模块代码放在立执行函数里。因为CommonJS是社区标准，浏览器厂商不愿意更改浏览器实现机制

解决方案：

1. 将加载模块变为异步加载，加载完之后调用回调函数
2. 将模块的代码放入立即执行函数中
3. ES6模块化和CommonJS模块化有哪些区别？

ES6模块化：

导入 import 导出 export

CommonJS模块化：

导入 require 导出 exports

CommonJs导入模块是同步加载，ES6导入模块是异步的

CommonJs的module.exports = {} 会覆盖export.xxx 导出的属性，

而Es6 的基本导出 export { } 和默认导出export default可以同时存在。s