南华大学

计算机学院

实 验 报 告

（ 2024 ~2025学年度 第 1 学期 ）

**课程名称 Web 全栈开发**

**姓 名 黄志鹏**

**学 号 20224490129**

**专 业 医学信息工程**

# 一、实验目的

* 掌握 HTML 和 CSS 的基本使用。
* 学习 JavaScript 的交互功能。
* 运行理解附件的寻宝游戏代码
* 增加新的有趣的寻宝情节，用异步的方式实现
* 使用async/await重写寻宝过程
* 设计动画页面展示寻宝过程
* 将动画页面挂网，在此处贴出网页链接

# 二、实验环境

* 操作系统：Windows11
* 开发工具：VS Code
* 浏览器：edge
* 编程语言和框架：HTML、CSS、JavaScript

# 三、实验步骤

1. **初始化游戏环境**

* 通过 document.getElementById 获取动画容器里的图片元素（currentSceneImg）、消息容器元素（messageContainer）和虚拟键盘容器元素（virtualKeyboard），用于展示不同场景的图片、游戏提示信息和玩家交互按钮。
* 初始化一个变量 hasFailed 用于标记是否遇到失败情况。

1. **启动寻宝流程**

* 调用 animateTreasureHunt 函数启动整个寻宝游戏的模拟流程。
* 获取初始线索
* 在 animateTreasureHunt 函数中，首先将当前场景图片设置为古老图书馆内部图片（currentSceneImg.src = 'library.jpg'）。
* 调用 TreasureMap.getInitialClue 函数获取初始线索，该函数返回一个 Promise，在 1 秒后返回在古老图书馆里找到的第一个线索。
* 获取线索后，在控制台打印线索内容，并在页面上显示找到第一个线索的提示信息（通过 showMessage 函数）。

1. **解码线索获取宝藏位置**

* 使用获取到的初始线索调用 TreasureMap.decodeAncientScript 函数进行解码，该函数也是一个 Promise，如果没有线索则拒绝并返回错误信息，有线索则在 1.5 秒后返回解码成功的信息及宝藏所在位置。
* 在控制台打印解码结果，并在页面上显示解码成功及宝藏位置的提示信息。

1. **前往古老神庙**

* 调用 wait 函数等待 2 秒，模拟前往古老神庙的过程，期间显示相应提示信息，并将当前场景图片切换为古老神庙图片（currentSceneImg.src = 'temple.jpg'）。

1. **玩家选择**

* 在页面上显示玩家选择提示信息（“在神庙中，你可以选择：”），并调用 createVirtualButtons 函数创建两个虚拟按钮（“在神庙内巡视” 和 “敲开地面”）。
* 通过 getPlayerChoice 函数获取玩家选择，该函数返回一个 Promise，监听虚拟按钮点击事件来获取玩家选择的值（1 或 2）。

1. **根据玩家选择执行操作**

* 如果玩家选择 1（在神庙内巡视），则调用 handlePatrolChoice 函数，该函数内部会根据随机生成的成功率调整值（newSuccessRateModifier）显示不同提示信息，并调用 TreasureMap.searchTemple 函数在神庙中搜索宝藏相关操作，根据结果显示不同场景图片和提示信息，若遇到守卫则设置 hasFailed 为 true。
* 如果玩家选择 2（敲开地面），则调用 handleDigChoice 函数，其操作与 handlePatrolChoice 类似，但成功率调整值的范围不同。

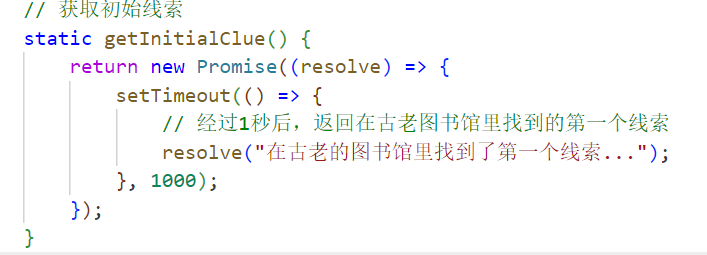
1. **打开宝藏箱（若未失败）**

* 如果在前面的步骤中没有遇到失败情况（hasFailed 为 false），则调用 TreasureMap.openTreasureBox 函数打开宝藏箱，该函数在 2 秒后返回找到传说中宝藏的信息，在控制台打印信息并在页面上显示找到宝藏的提示信息。

# 四、代码示例

1. **TreasureMap 类：**

* getInitialClue 方法：返回一个 Promise，1 秒后提供初始线索。



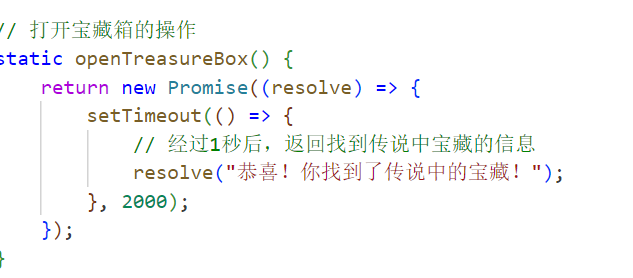
* decodeAncientScript 方法：接收线索作为参数，返回一个 Promise，根据线索情况在 1.5 秒后返回解码结果。



* searchTemple 方法：接收位置和成功率调整值作为参数，返回一个 Promise，根据随机生成的结果和调整值判断是否遇到守卫或找到神秘箱子。



* openTreasureBox 方法：返回一个 Promise，2 秒后返回找到宝藏的信息。



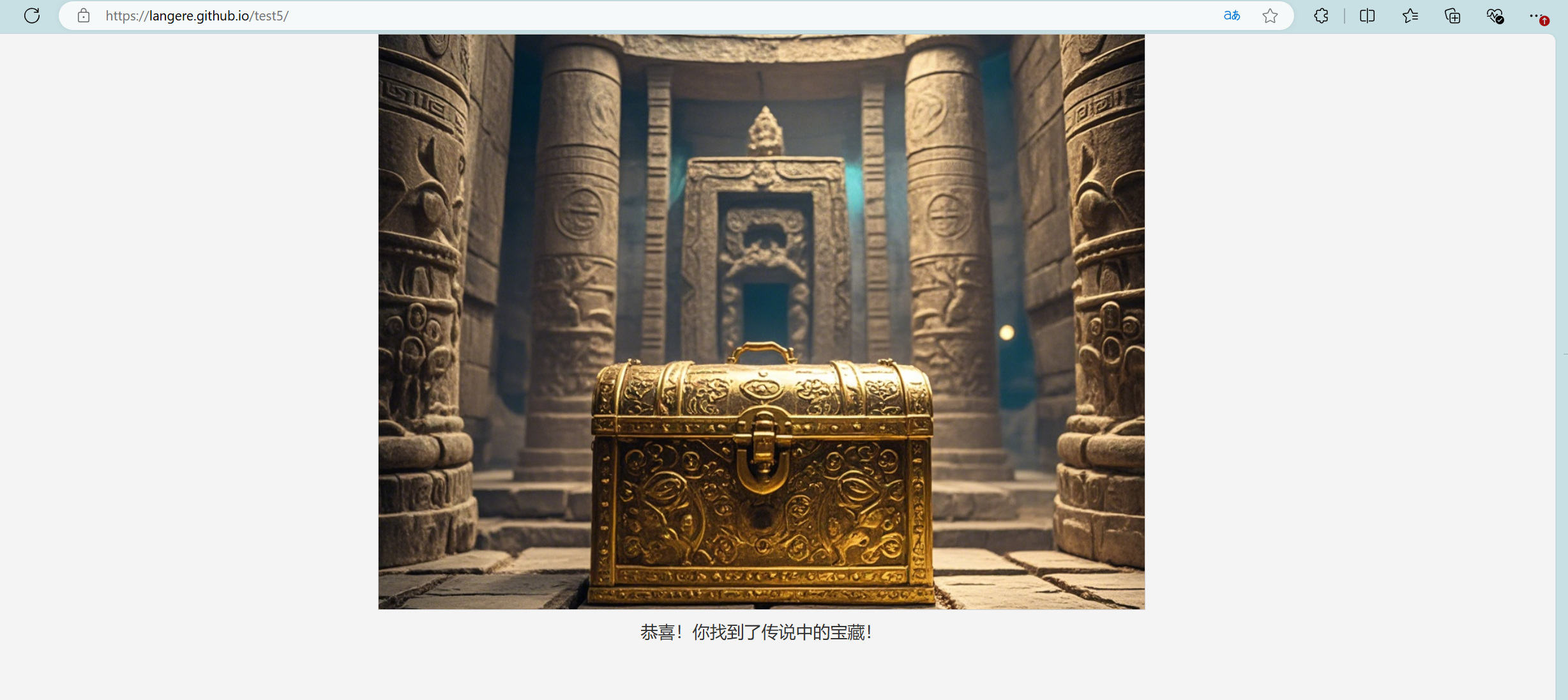
1. **animateTreasureHunt 函数**：通过异步操作和调用 TreasureMap 类的方法以及其他辅助函数来模拟整个寻宝流程，包括处理线索、玩家选择和可能的结果。



1. **其他辅助函数**：wait 函数用于等待指定毫秒数；createVirtualButtons、handlePlayerChoice、getPlayerChoice 函数用于处理玩家交互；showMessage 函数用于在页面上显示提示信息；handlePatrolChoice 和 handleDigChoice 函数分别处理玩家在神庙内巡视和敲开地面的选择逻辑。

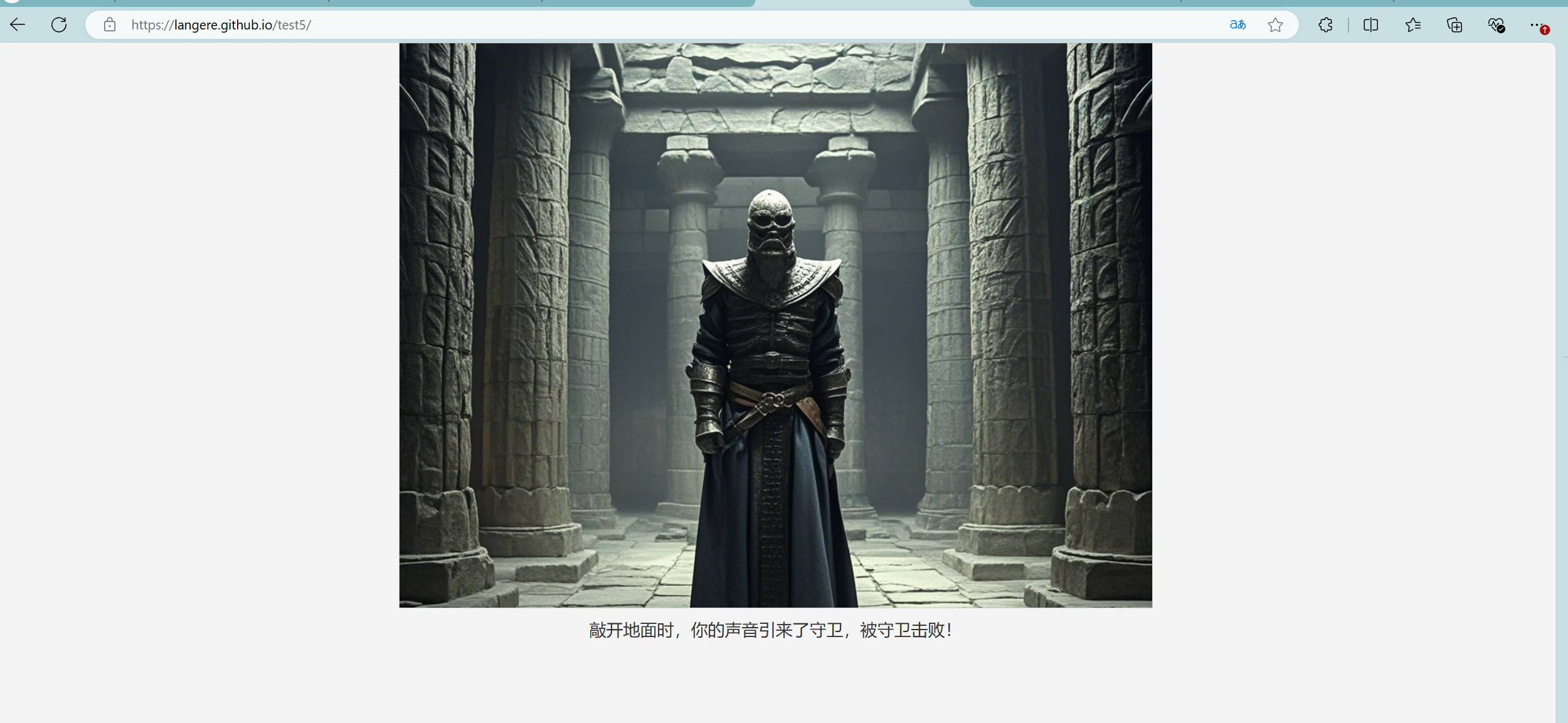
# 五、实验结果

成功结果：



失败结果：





两种选择会导致最后抽中成功结果的概率不一样，巡视选择对于找到宝藏的概率影响较小，而敲开地面选择对于找到宝藏的概率影响较大。

# 六、问题与解决

在获取玩家选择时，由于使用了异步操作和事件监听，可能会出现选择值传递不及时或不正确的情况。通过在虚拟按钮点击事件中正确设置玩家选择值，并在 getPlayerChoice 函数中监听事件，当按钮被点击时将选择值传递给 resolve 函数，确保玩家选择值能正确传递到后续的操作中。

# 七、实验总结

<https://langere.github.io/test5/>

本次实验通过 JavaScript 实现了一个简单的寻宝游戏模拟，深入展示了异步编程（Promise 和 async/await）在模拟复杂流程中的应用，以及如何通过 DOM 操作实现与用户的交互和游戏场景的展示。