汉诺塔动画演示和手动练习

模块开发说明

[V1.0(版本号)]

拟 制 人\_章陆斌\_张玲珑\_章陆尉\_

审 核 人\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

批 准 人\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

[二零二零年十月二十九日]

# 模块开发说明

# 1．标题

模块名称：初始化模块

程序编制员：章陆尉

模块名称：动画演示模块，拖拽练习模块

程序编制员：章陆斌

模块名称：按钮练习模块

程序编制员：张玲珑

# 模块开发情况表

表 2-1模块开发情况表

|  |  |  |  |
| --- | --- | --- | --- |
| 功能类 别 | 序号 | 任务 | 实现度 |
| 需求说明书中所需的功能 | 1 | 汉诺塔自动演示动画 | 实100% |
| 2 | 六个按钮实现手动练习 | 实100% |
| 3 | 实现用户更改颜色属性，盘子数量以及运行速度 | 100% |
| 4 | 实现日志输出盘子移动详细步骤 | 100% |
| 5 | 实现日志输出运行时间 | 100% |
| 实现过程中添加的功能 | 6 | 拖拽来实现手动练习 | 90% |
| 7 | 拖拽和六个按钮点击的操作可以一同使用在练习中 | 100% |
| 8 | 日志框中出现的步骤拖拽和按钮点击实现同步 | 100% |

# 功能说明



图 3-1总体功能呈现

## 3.1初始化功能

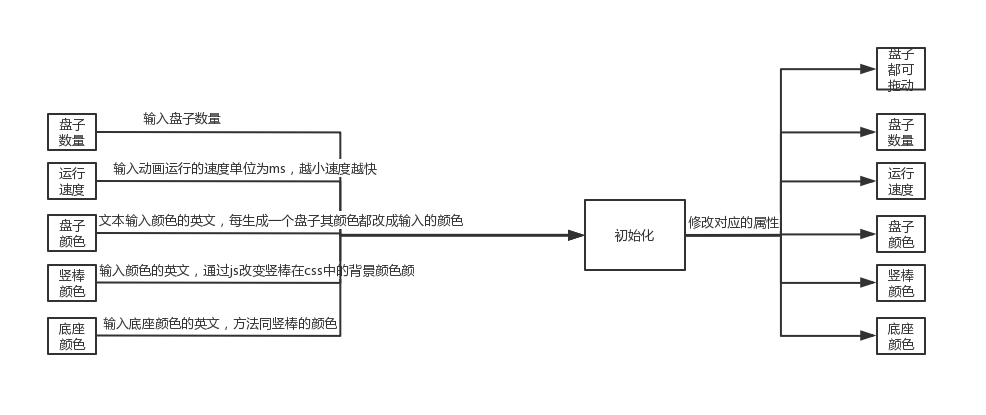


图 3-2初始化功能呈现

文本框中输入五个对应的属性可以进行修改对应值，同时所有盘子设为可拖动状态

## 3.2动画**演示**功能

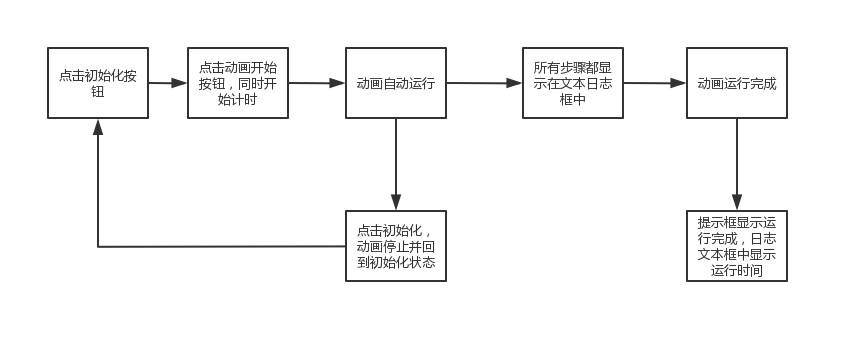


图 3-3动画演示功能呈现

先点击初始化，进行盘子的生成，然后点击动画开始，文本日志框中显示开始时间并进行计时，中途点击初始化就返回到初始化状态，每移动一步，文本框中将自动记录该步步骤，当运行完成提示框提醒移动完成，同时文本日志框中显示运行时间和结束时间。

## 3.3手动练习--按钮功能

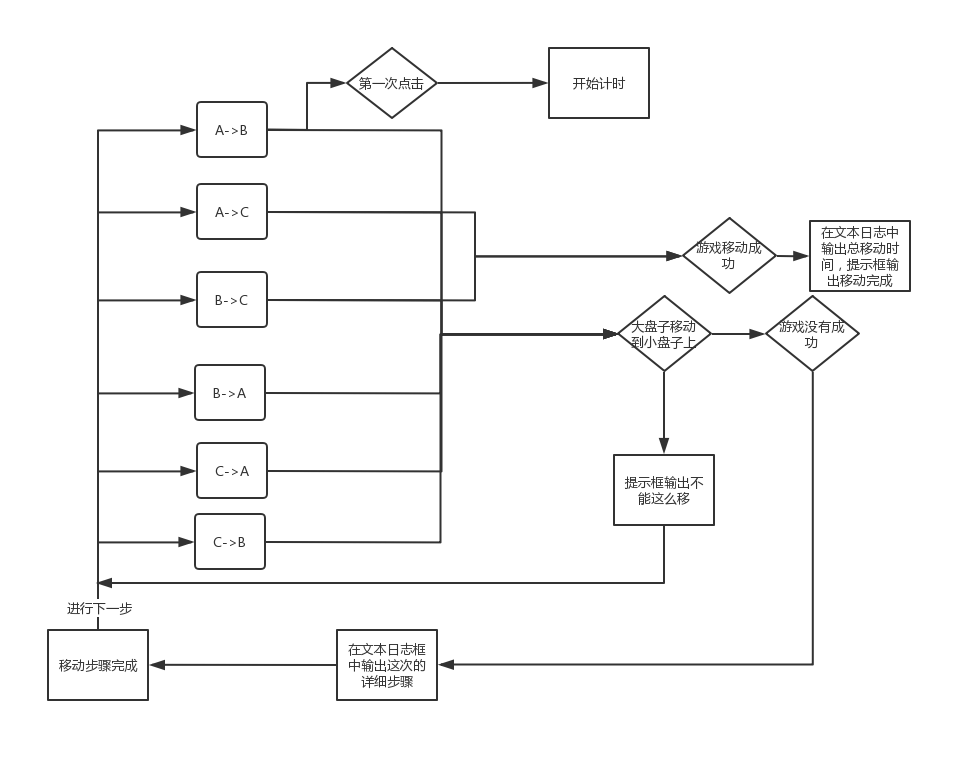


图 3-4按钮练习功能呈现

点击初始化，再点击A-B或A-C按钮，会实现自动开始计时并在文本框中显示开始时间，中途点击初始化，进入到初始状态，每按一次对应按钮就执行对应操作，如遇到大盘移动到小盘时提示框会提醒不能这么移，同时返回原先位置，当一步正常完成时，日志文本框中将会记录该步骤。当移动完成，提示框显示移动完成，同时文本日志框中显示结束时间，和移动用时。

## 3.4手动练习--拖拽功能

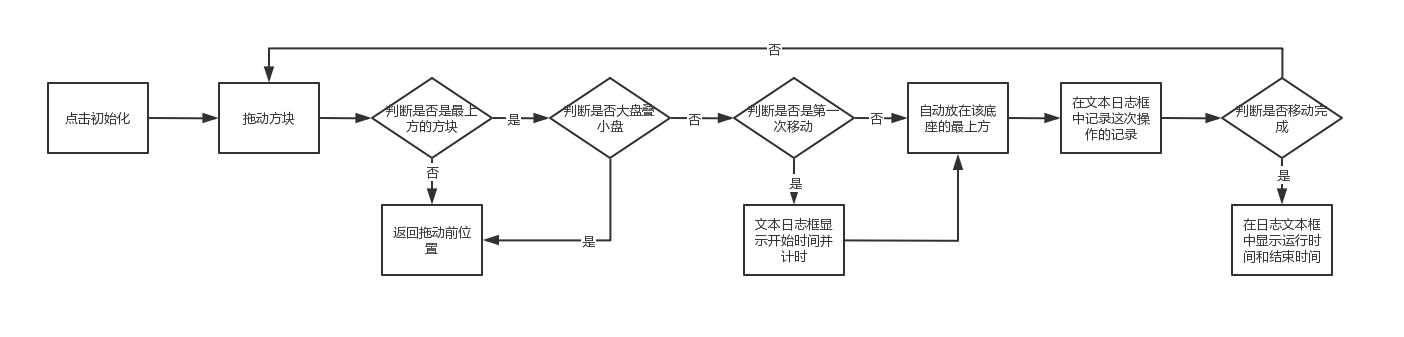


图 3-5拖拽练习功能呈现

点击i初始化，拖动最顶部盘子，当拖动的盘子不是三个底座上最上层的盘子时会返回原先位置，表示不能这么移，当拖动到A,B C底座范围时，先判断是否有大盘叠小盘的情况，没有则自动叠放在该底座最上方，同时在下面的日志文本框中记录该次操作。当是第一次成功拖动时，日志文本框显示开始时间，同时开始计时，中途点击初始化时，返回初始状态，当移动完成时，提示框显示移动完成，同时日志文本框中显示结束时间和移动用时。

拖拽功能和按钮的移动功能可以同步使用！

# 设计说明

该模块的设计，即可以让那先不懂得如何用递归求解汉诺塔问题的人，通过动画演示的方式让他们理解它的求解过程，同时在理解原理的基础上，还可以拖动鼠标或点击下方的六个操作按钮来进行手动练习。采用动画演示和手动练习的方法，使本身枯燥乏味的算法，变得有趣。

# 软件的设计结果

## 5.1初始化功能

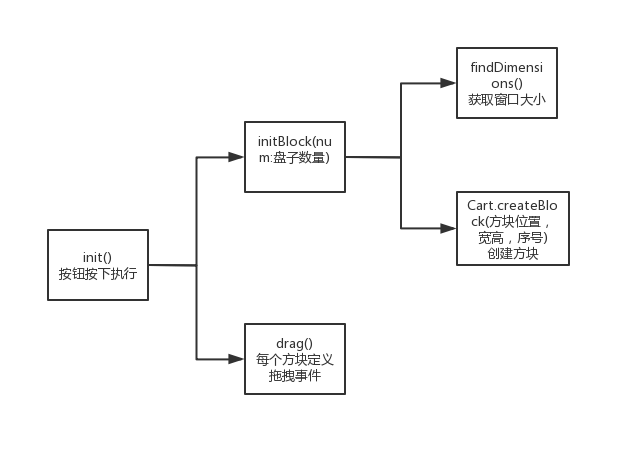
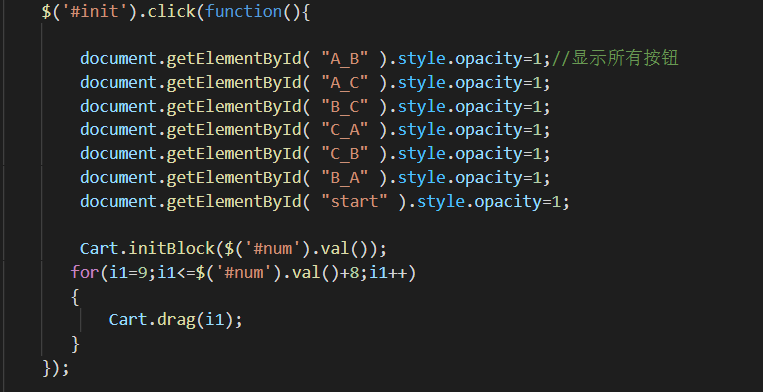


图 5-1初始化函数调用

### 5.1.1init.click()按钮事件

所有按钮都置于显示状态,由于后面动画演示时手动移动的按钮不能使用，从而每一次初始化都要让按钮全部显示,同时执行Cart.initBlock()函数和循环执行Cart.drag()函数



### 5.1.2Cart.initBlock()函数

（1）盘子数量的输入

调用设立的initBlock函数参数为盘子的数量



（2）速度的输入

调用change函数参数为输入的速度把Cart.options.speed的值赋值成了文本框输入的值，为后面动画演示时运动的速度做前提



（3）柱子和底座颜色的输入

调用函数initBlock在其中修改底座的颜色和柱子的颜色

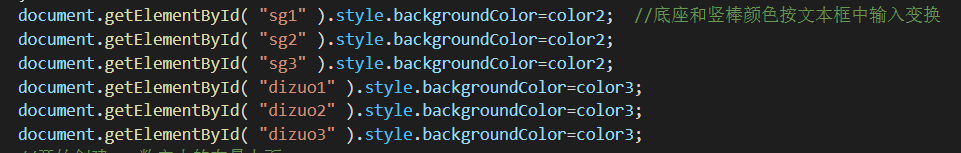
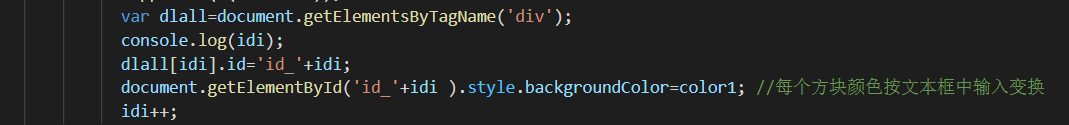


图 5-5

（4）方块颜色的输入

每创建一个方块对其创建的方块颜色进行文本框中对应颜色的修改



### 5.1.3finddimensions()函数

获取窗口的宽和高，由于要做到自适应获得当前窗口大小 用finddimensions()函数来对窗口高度winHeight和窗口宽度winWidth进行赋值。

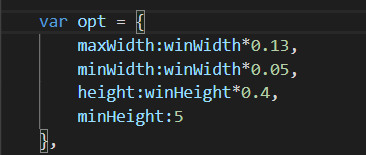


### 5.1.4生成方块堆

(1) 设定方块的宽高和位置信息

由于每个方块的高度是固定的，从而只要设立自适应的总高度除以个数-1就行了。

每个方块的宽度是由高到低递增的，从而先设定自适应的最大宽度和最小宽度。



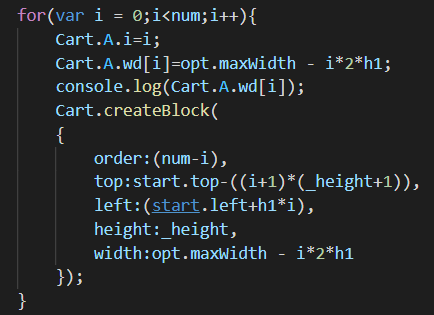


位置坐标



（2）生成每一个宽度不同的方块

用for循环生成每一个方块，每次调用createBlock函数输入位置信息(left,top),宽高，和方块序号，进过调试，（left，top，height，width，order(方块的序号)函数参数为以下值时，可以生成叠加且对称的方块堆。同时记录每个方块对应的宽度，后面在手动练习判断大小盘时有作用



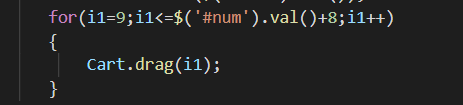
（3） createBlock(position)函数

通过传入的参数进行一个方块的生成。同时用对每个方块进行命名，在对对应id名进行方块颜色的修改。



### 5.1.5 drag()函数

用for循环给每个生成的方块定义相同的拖拽事件drag()，拖拽事件详细在拖拽模块开发中会进行详细讲解



## 5.2动画演示功能

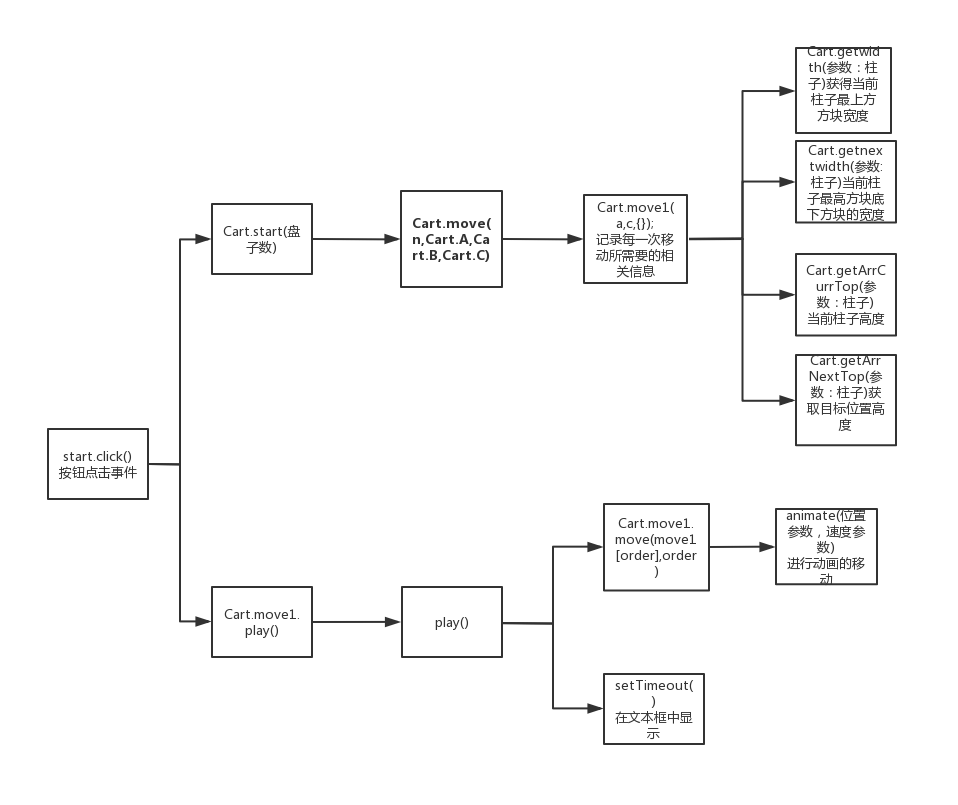


图 5-2动画演示函数调用

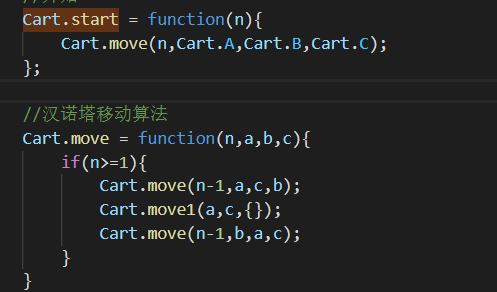
### 5.2.1start.click事件

一开始让所有手动练习的按钮设为不可以见，做到动画演示时，不能进行手动练习。执行Cart.Start()函数和Cart.move1.play()函数



### 5.2.2Cart.Start()函数

执行递归调用做到对汉诺塔问题的求解，同时执行Cart.move1(a,c,{})函数。



### 5.2.3Cart.move1(Cart.A,Cart.B,{})

（1）先判断是否有移动的盘子，是否是大盘移动到小盘上，不可以移动直接返回，并在提示框显示不能移动。



（2）对起始盘子位置高度，目的盘子位置高度，移出和移入柱子方块宽度的栈进行更改。



（3）由起始柱盘子高度，目的住盘子高度，和柱子间距离进行判断盘子该如何移动才能使移动不和另外的盘子相撞

图注：\_jg距离 \_ftop当前高度 \_ttop目标高度 \_btop b柱的高度 \_min三个柱子最低的高度

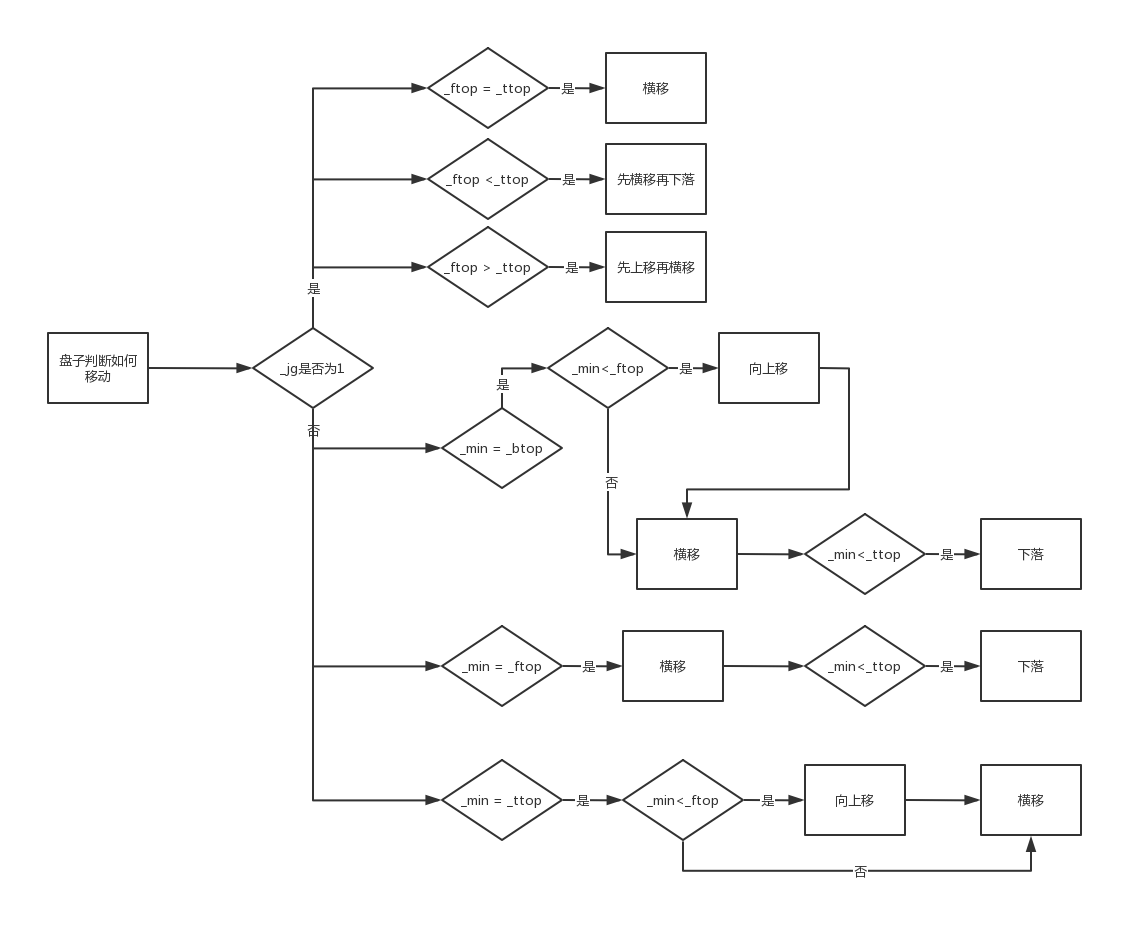
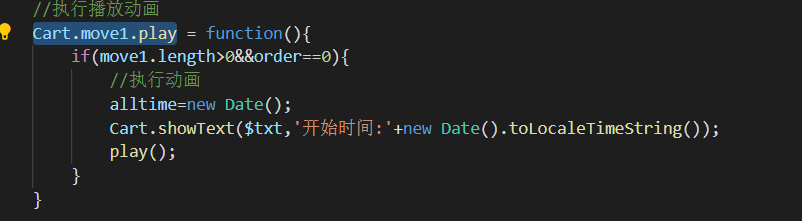


图 5-3动画移动判断思维图

代码详见index.html 277-358行

### 5.2.4Cart.move1.play（）函数

实现点击按钮就开始计时，同时执行play()函数



### 5.2.5play（）函数

该函数只要当Cart.isstart=flase时才会停止，

循环 调用Cart.move1.move(move1[order],order)函数，把上面递归求出来的每一步都进行动画演示。移动完成时在日志文本框中输出运行时间。



### 5.2.6 Cart.move1.move(move1[order],order)函数

调用jq库里的animate动画函数，放入位置参数和速度参数实现盘子移动



## 5.3手动练习-按钮功能

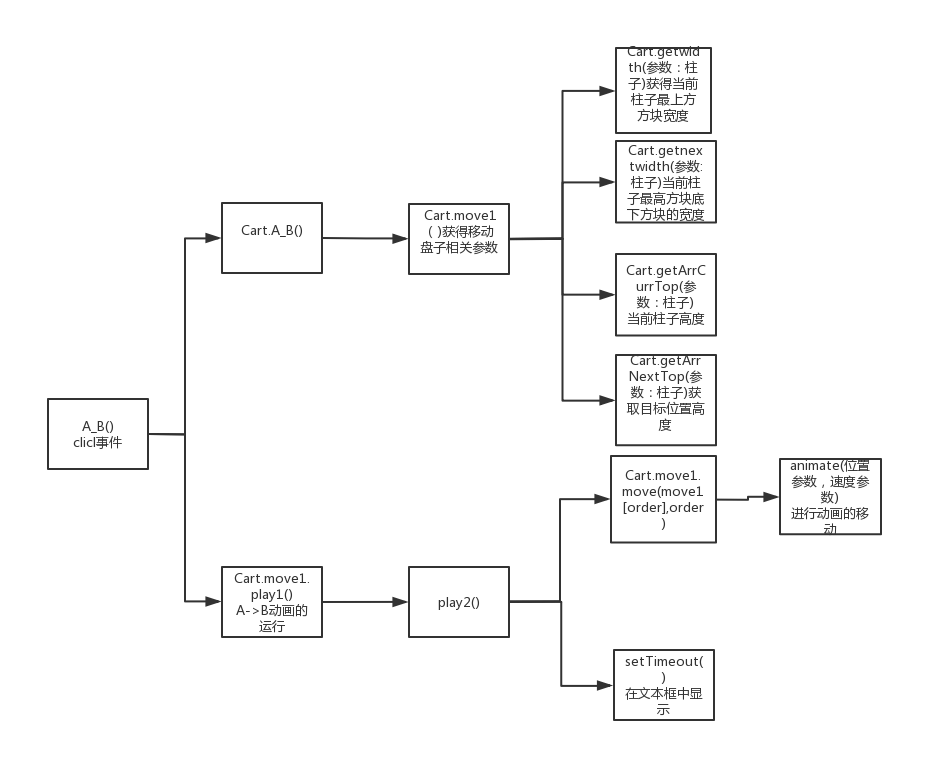
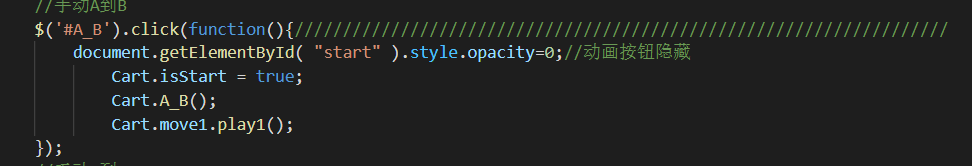


图 5-4按钮手动练习函数调用

### 5.3.1A\_B().click点击事件

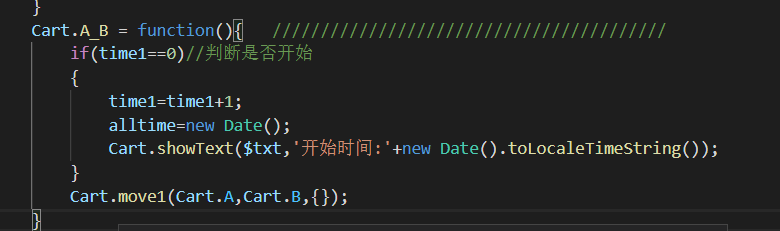
以按钮A\_B按钮为例，另外五个按钮同理

点击按钮时，让动画播放按钮消失，起到手动练习时不能使用动画播放的功能对其设为里opacity=0。同时执行Cart.A\_B()函数和Cart.move1.play1()函数



### 5.3.2Cart.A\_B()函数

在做文本日志框中显示开始时间，并执行Cart.move1(Cart.A,Cart.B,{})



### 5.3.3Cart.move1(Cart.A,Cart.B,{})

见动画演示模块2.3小节

### 5.3.4play2()函数



### 5.3.5Cart.move1.move(move1[order],order)函数

调用jq库里的animate动画函数，放入位置参数和速度参数实现盘子移动



## 5.4手动练习--拖拽功能

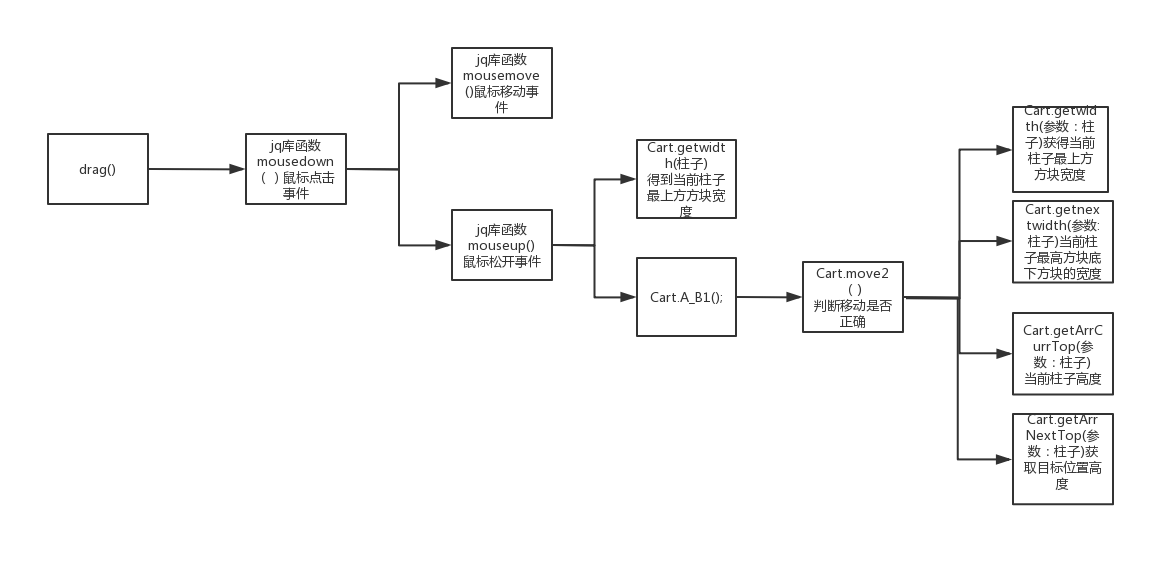


图 5-5拖拽手动练习函数

### 5.4.1 init.click()事件

拖拽功能是在初始化中对每个方块都定义了拖拽事件的函数art.drag()



### 5.4.2drag()函数

drag函数分别调用了mousedown() mouseup() mousemove()这三个jq库函数

### 5.4.3mousedown()函数

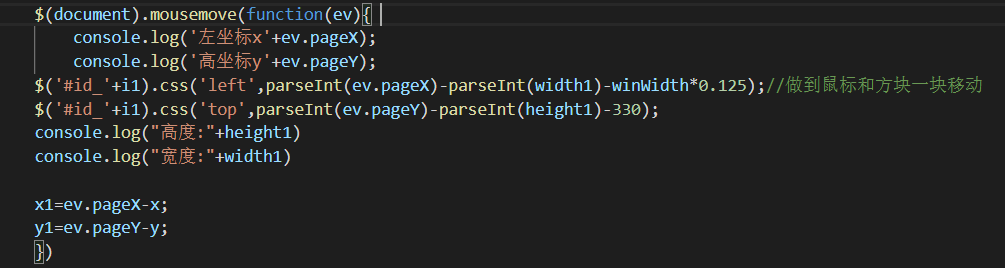
mousedown中包含了mouesup（）和mousemove（）函数

当鼠标点击该方块时就可以得到三个柱子的高度，鼠标对应的坐标，没有拖拽的位置信息和是否可以移动的bool值;



### 5.4.4mousemove()函数

这里的目的是让方块移动随着鼠标位置而移动，做到拖动的效果



### 5.4.5mouseup()函数

以B拖动到A为例子，当鼠标松开始开始判断，落到了哪个底座上，该步操作是否合理，如大盘不能放到小盘上，盘子上有盘，不能够拖动。

（1）这里判断移动的位置是否在A底座上，另外底座同理



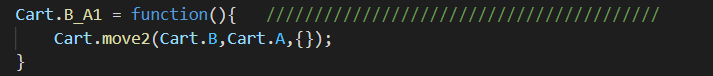
（2）这里判断盘子是从哪根柱子上移动出来的



（3）这里判断当前移动的盘子是否是柱子上最上层的盘子

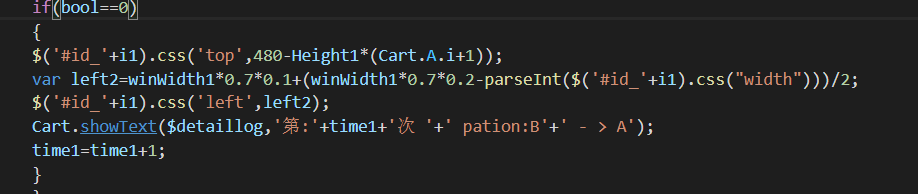


（4）这个函数用来判断大盘是否移动到小盘上方，如果有则置1，没有就是0





（5）判断如果为0则把盘子自动放入目的柱子的最上方，并在底下的文本框中显示步骤



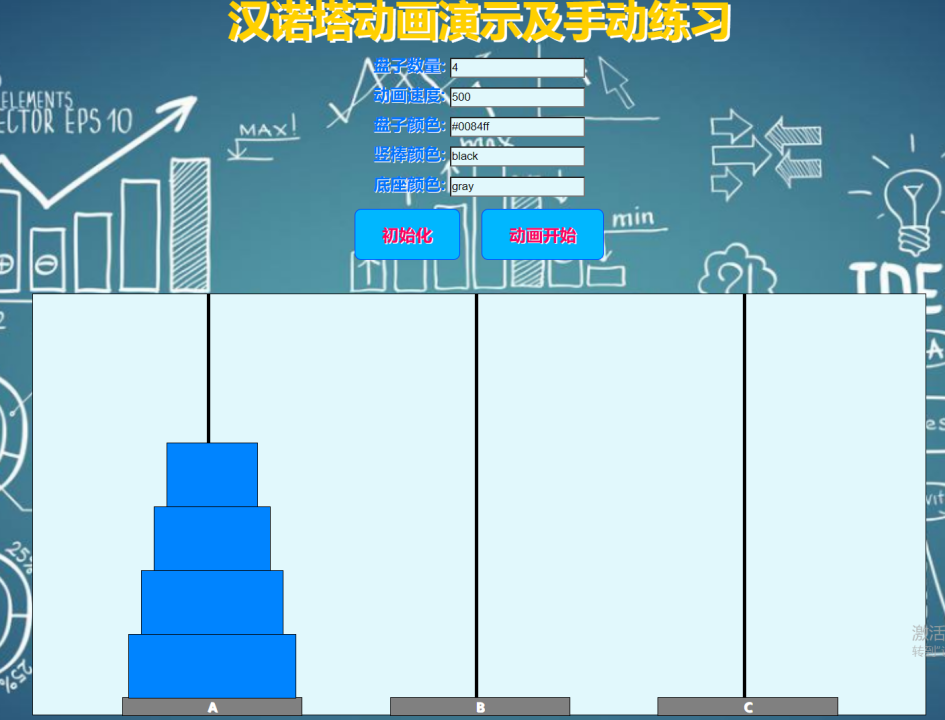
# 测试说明

说明直接要经过本模块的每一项测试，包括这些测试各自的标识符和编号、进行这些测试的目的、所用的配置和输入、预期的输出及实际的输出。

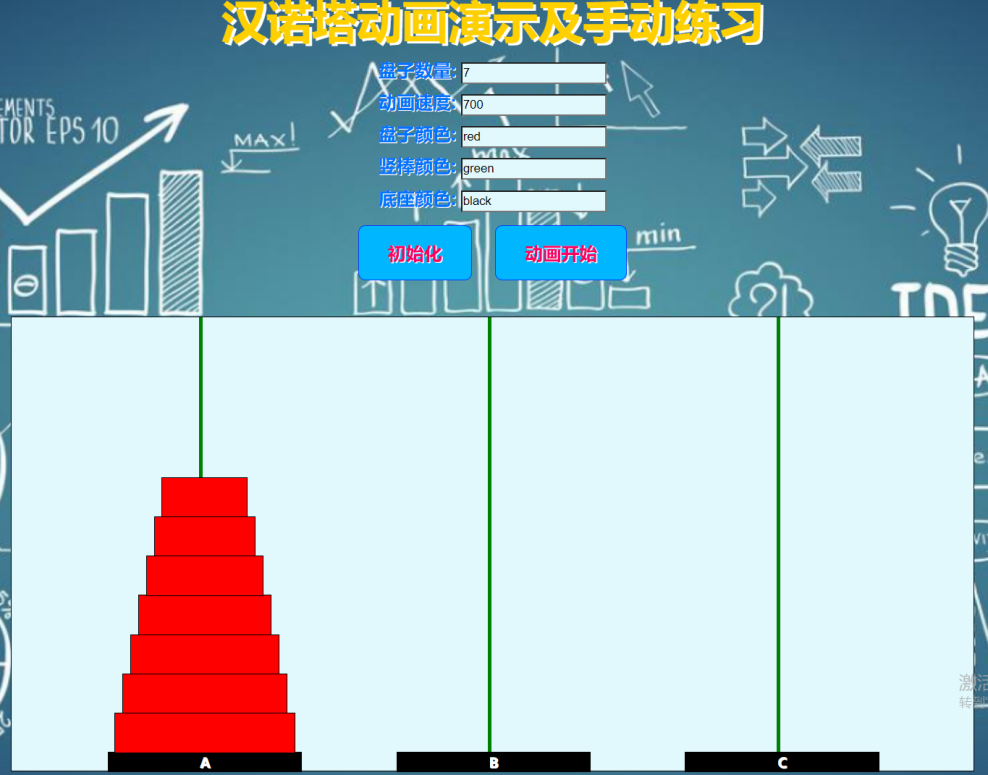
## 6.1初始化功能测试

目的：初始化功能能否正常使用，是否存在bug

1. 点击初始化按钮

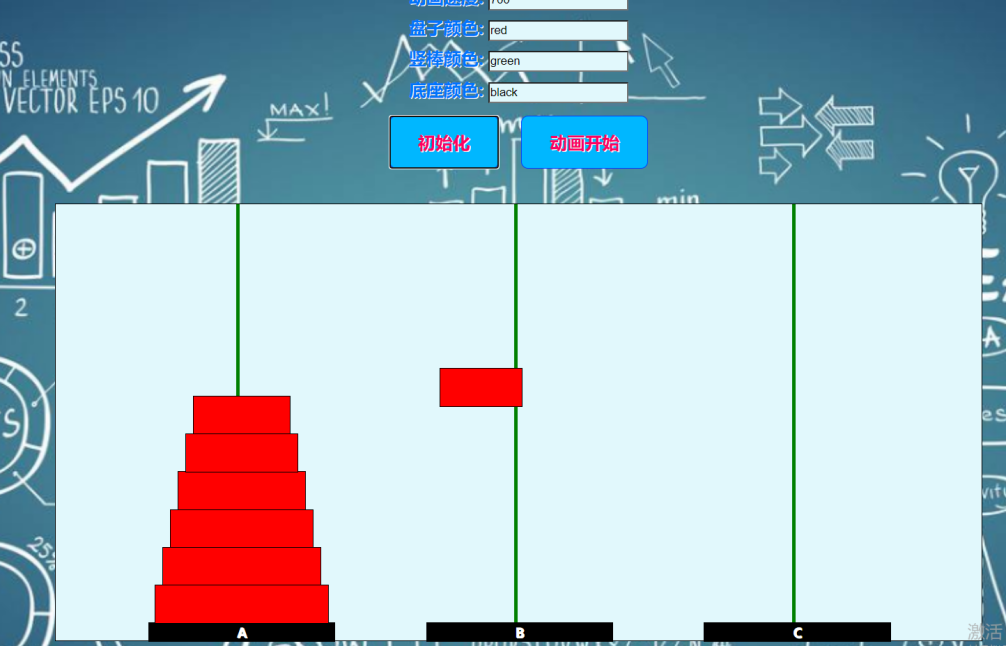


1. 更改五个文本框中的数据，盘子个数应改为7个盘子，颜色应为红色，竖棒颜色应为绿色，底座颜色应为黑色



测试结果正确

1. 拖住功能



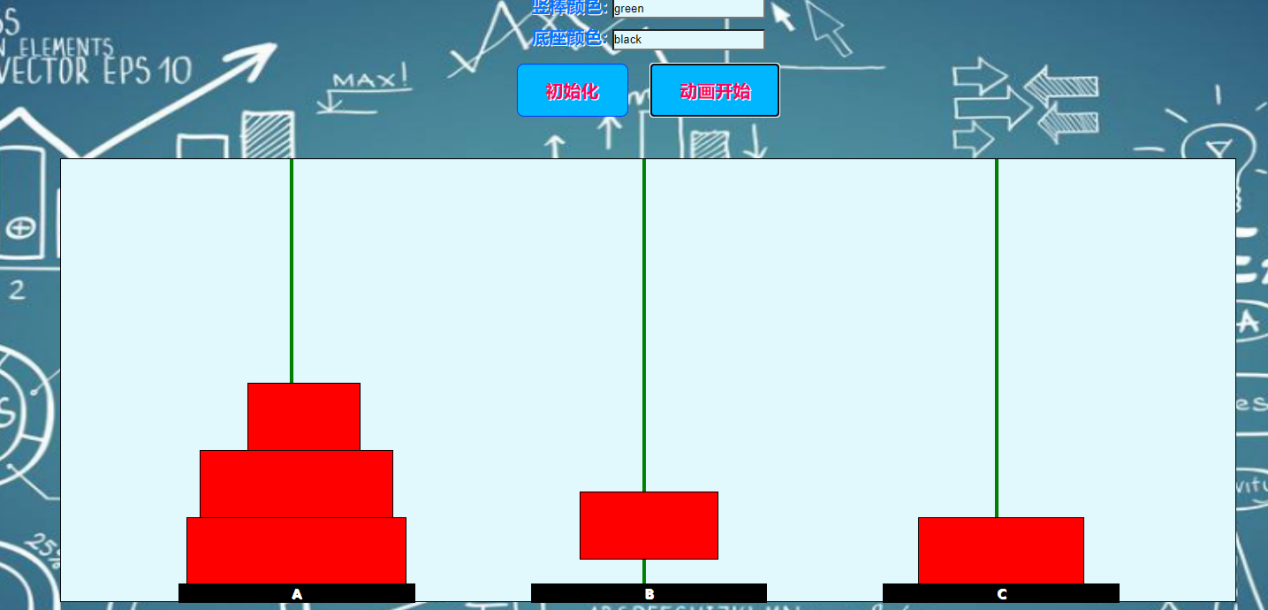
测试结果正确

总结：初始化所有功能能正常运行

## 6.2动画演示功能测试

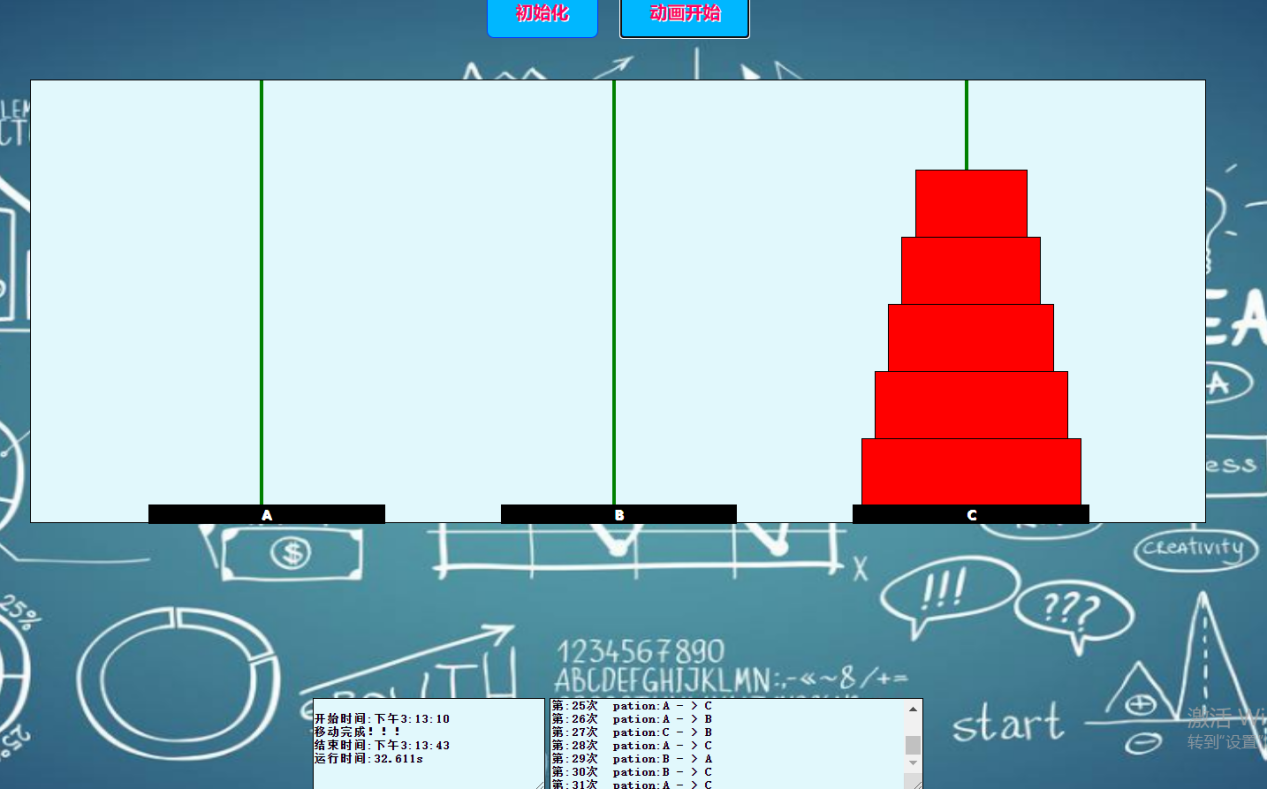
1. 点击动画开始按钮动画是否正常执行，并且底部6个按钮是否消失

目的：检测是否存在动画执行期间是否存在bug



检测结果正确

1. 是否可以完成汉诺塔求解，步骤是否为最优，底部日志框是否输出正常，运行时间是否正常。5个盘子最优步数为2^5-1=31步



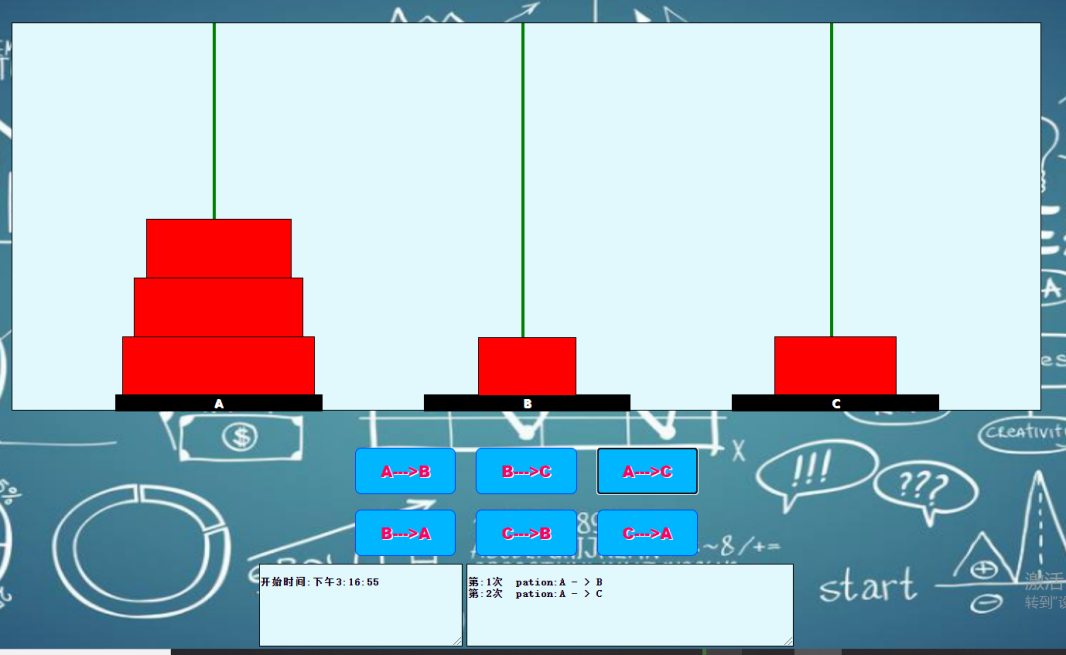
检测结果：移动成功完成，步骤数为31，运行时间与时间相符。功能正常

总结：动画演示功能检测正常

## 6.3手动练习功能检测

(1)拖拽和按钮是否能同步使用

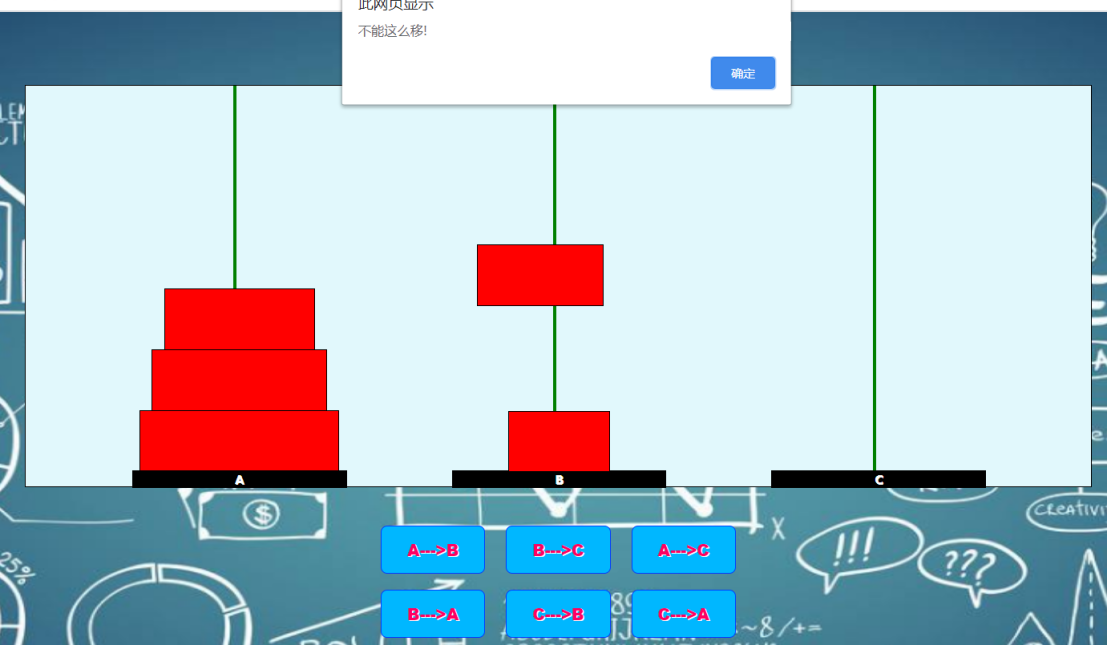
目的：检测拖拽和按钮同步使用时是否会发生bug

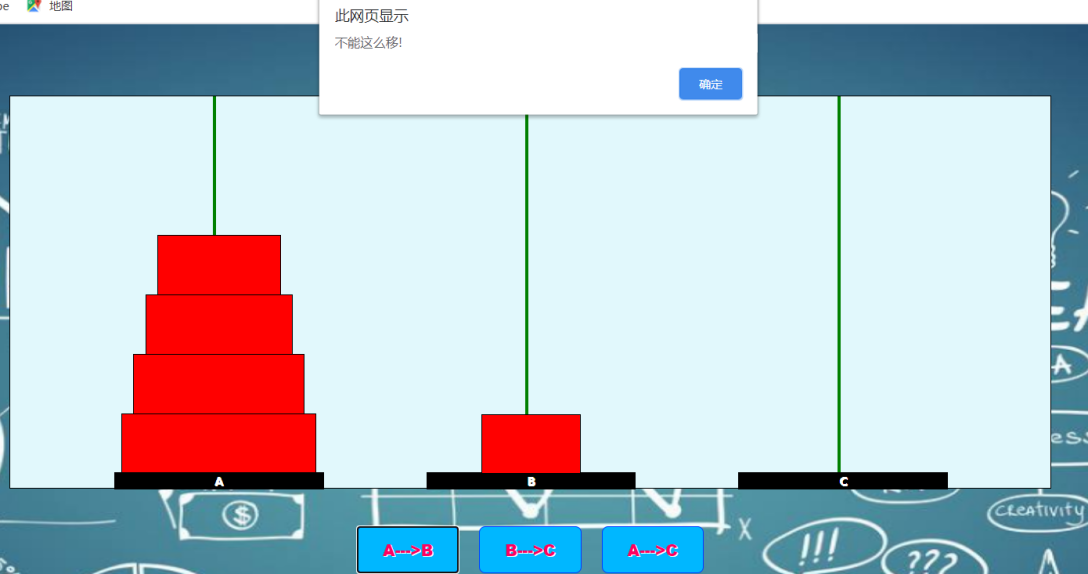


测试结果：可以同步使用

(2)按钮点解和拖拽是否能判断大盘不能叠在小盘上

目的：汉诺塔的规则中有大盘不能移动到小盘上

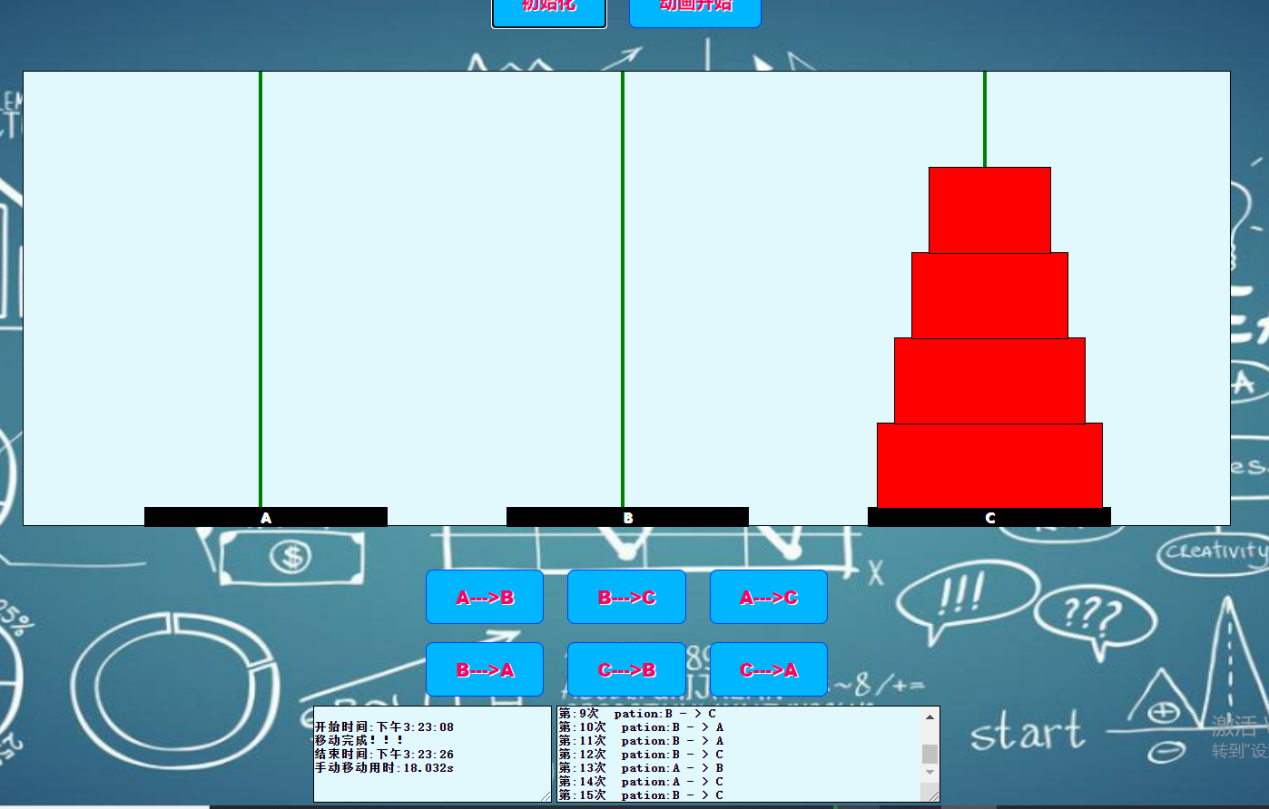




测试结果：大盘都不能移动到小盘上。并返回原来位置。测试正常

(3)能否完成汉诺塔的移动，并正确显示移动时间和移动步骤

目的：测试是否存在bug问题



测试结果：可以正确的输出移动的步骤，并显示所用的时间。

# 复审的结论

表 7-1功能实现表

|  |  |  |  |
| --- | --- | --- | --- |
| 阶段 | 序号 | 任务 | 是否实现 |
| 需求说明书中所需的功能 | 1 | 汉诺塔自动演示动画 | 实现 |
| 2 | 六个按钮实现手动练习 | 实现 |
| 3 | 实现用户更改颜色属性，盘子数量以及运行速度 | 完成 |
| 4 | 实现日志输出盘子移动详细步骤 | 完成 |
| 5 | 实现日志输出运行时间 | 完成 |
| 实现过程中添加的功能 | 6 | 拖拽来实现手动练习 | 实现 |
| 7 | 拖拽和六个按钮点击的操作可以一同使用在练习中 | 实现 |
| 8 | 日志框中出现的步骤拖拽和按钮点击实现同步 | 实现 |

从功能实现表中可以看出，我们不仅把所需的功能得以实现，同时为了让手动练习的交互体

验感更加，我们在前面6个按钮点击的基础上还添加了拖拽功能，大大提升了手动练习操作时用户的舒适感和体验感。总得来说我们不仅完成了所需功能的实现还针对用户体验进行了合理的优化和添加功能。