|  |
| --- |
| 大作业报告 |
| 程序设计基础 郑皓之 黄远森 |

目录

[1.小组分工 2](#_Toc155388489)

[2.设计思路 2](#_Toc155388490)

[2.1 协作 2](#_Toc155388491)

[2.2 功能 2](#_Toc155388492)

[3.工程结构 2](#_Toc155388493)

[3.1 整体结构 2](#_Toc155388494)

[3.2 代码部分（模块介绍） 2](#_Toc155388495)

[3.3 代码部分（主程序流程及运行逻辑） 7](#_Toc155388496)

[4.整体游戏界面设计 8](#_Toc155388497)

[4.1 主菜单 8](#_Toc155388498)

[4.2关卡界面 10](#_Toc155388499)

[4.3 结算界面 12](#_Toc155388500)

[5.游戏测试 13](#_Toc155388501)

[6.自由创新关卡 18](#_Toc155388502)

[6.1 Level 4 18](#_Toc155388503)

[6.2 Level 5 18](#_Toc155388504)

[7.拓展功能/特色功能 19](#_Toc155388505)

[7.1 单步调试 19](#_Toc155388506)

[7.2 可拓展修改的关卡文件 20](#_Toc155388507)

[7.3 方向键光标输入模式 21](#_Toc155388508)

[7.4 基于多线程的“动画速度调节”和“执行中退出” 22](#_Toc155388509)

[7.5 保留用户指令的“重新开始”功能 22](#_Toc155388510)

[7.6启动动画/退出动画 23](#_Toc155388511)

[7.7 多页指令 23](#_Toc155388512)

[7.8 其他 24](#_Toc155388513)

[8.运行说明及文档格式说明 24](#_Toc155388514)

[8.1 所用的IDE /C++版本 24](#_Toc155388515)

[8.2 启动器的使用场景 24](#_Toc155388516)

[8.3 存档/配置文件的格式 24](#_Toc155388517)

[9.录屏演示链接 26](#_Toc155388518)

## 小组分工

组号：86

2023012168 郑皓之：执行模块、高级动画模块、启动器、关卡设计等

2023012174黄远森：图形界面模块、动画模块、指令读写模块等

## 2.设计思路

2.1 协作 为了进行高效的协作，我们将作业任务进行了模块化细分，以便小组成员相对独立地编写自己负责的代码部分。为实现以上效果，我们采用了如下一些策略：

1. 尽可能多地使用全局变量，尽可能使用函数将功能模块化
2. 以图形界面程序作为主框架，在图形界面中调用执行模块，再由执行模块调用动画模块（这样一来就仅仅在两次调用过程中涉及不同成员代码的对接和融合）
3. 使用VisualStudio的Live Share功能在线上对同一cpp文件进行远程编辑和调试。

2.2 功能 本小组还希望在cmd界面中尽可能地还原Human Resource Machine原游戏的功能和视觉效果，在一定程度上提高“用户友好”性。为此，我们设计了如下一些功能：

（1）依赖windows.h的光标坐标移动函数生成伪图形界面

（2）依赖键盘输入检测，实现“光标输入”功能，即，允许用户全程使用键盘方向键、enter键等按键进行游戏，而不需要手动键入各种指令

## 3.工程结构

### 3.1 整体结构

3.1.1游戏主体：本游戏所有代码在同一cpp文件中

3.1.2 启动器：一个c++编译出的exe文件，用于初始化windows terminal窗口的样式

3.1.3存档文件：本游戏分两个存档文件，分别存储闯关/成就进度以及用户偏好

3.1.4配置文件：本游戏每个关卡的数据（如：输入、期望输出、任务说明等）均以特定格式存储在一个txt文件中，从而实现了用户通过修改文件自定义关卡的功能。

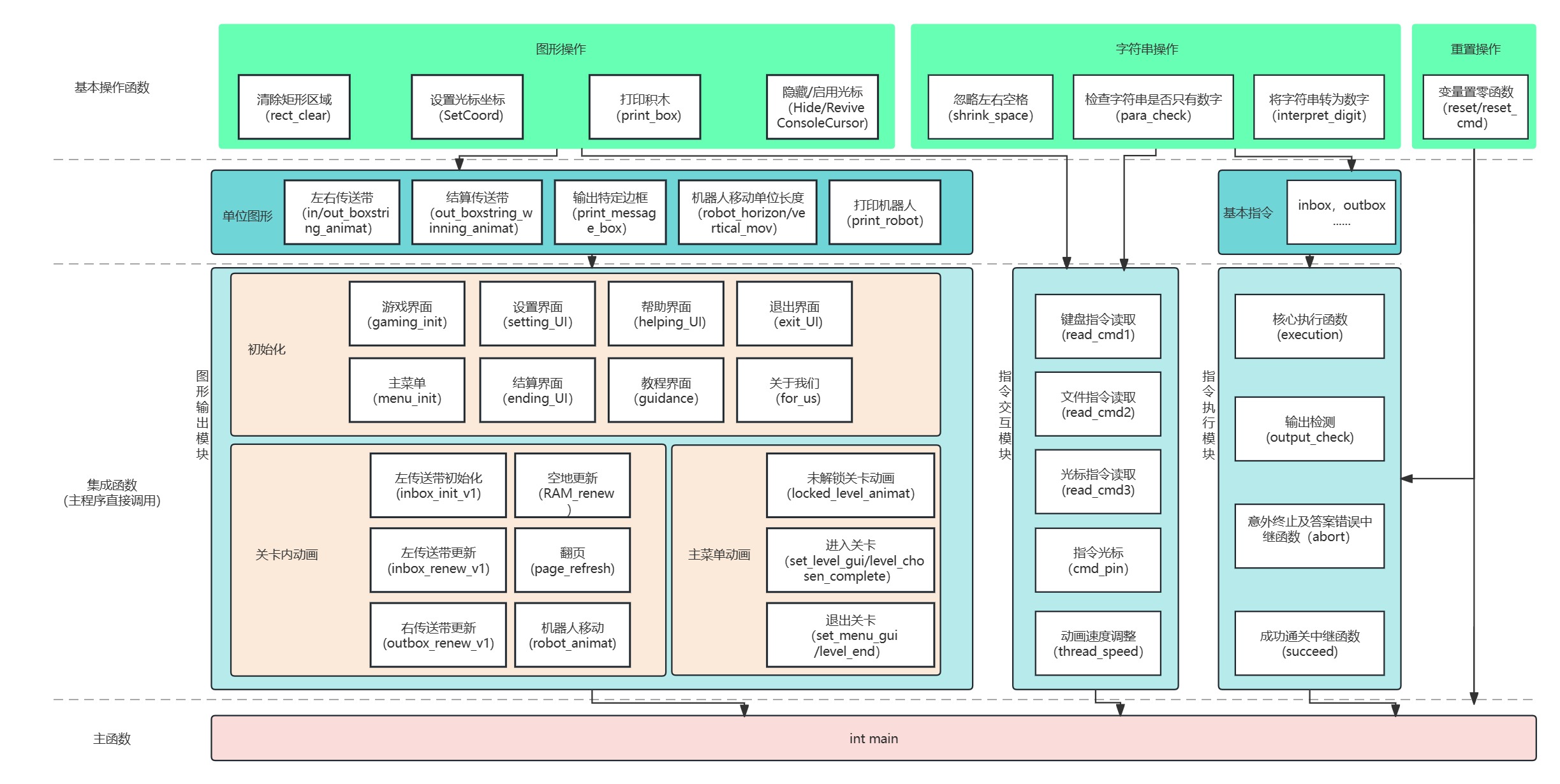
3.1.5音乐文件：两个wav背景音乐文件

3.1.6教程文件：用于存储教程页面所需的画面信息

### 3.2 代码部分（模块介绍）

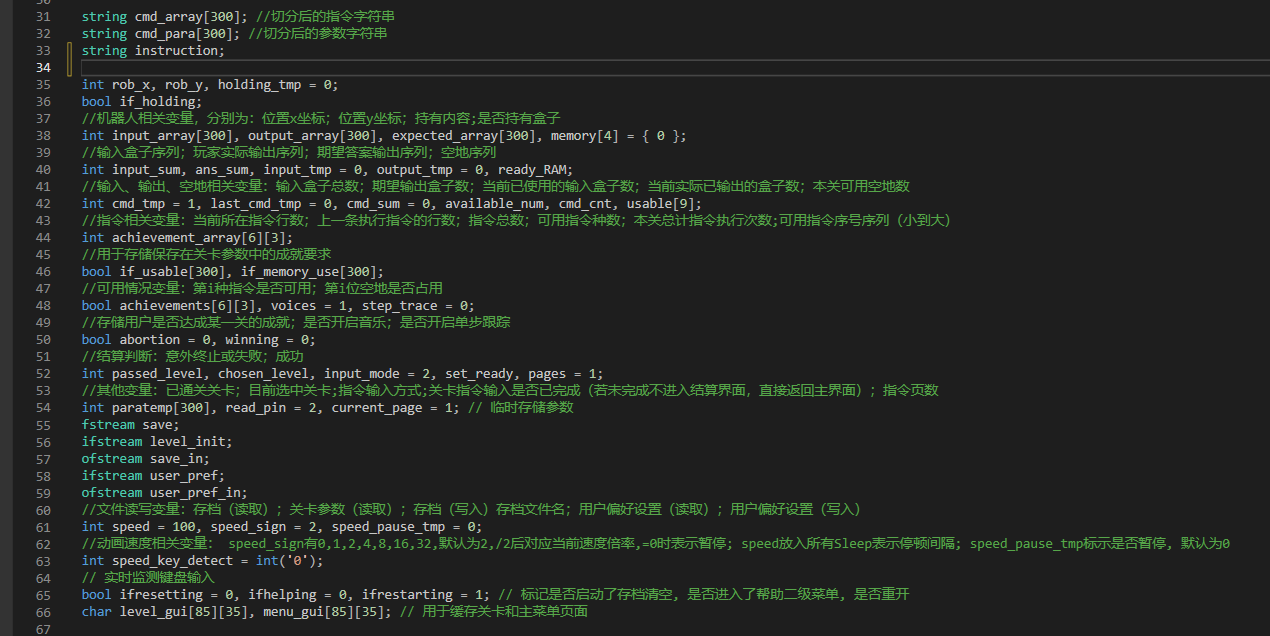
**3.2.1本游戏代码中有大量自定义函数，以下图片对这些函数进行了大致的分层：**

（该图片主要体现函数层级关系，与cpp中各函数出现顺序和所在模块不严格对应）

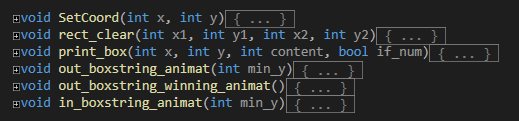


**3.2.2 以下是全局变量的设置和对各模块的的具体介绍：**

3.2.2.1 全局变量：

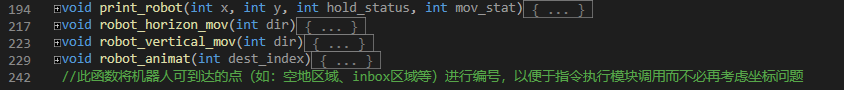


3.2.2.2 基本图形模块



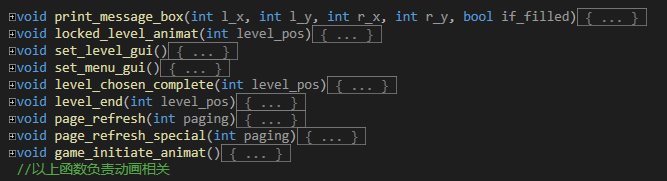
|  |  |  |
| --- | --- | --- |
| 函数名 | 参数含义 | 功能 |
| SetCoord() | (x,y) 为Cmd界面坐标 | 设置光标的坐标 |
| Rect\_clear() | (x1,y1)为左上顶点坐标，(x2,y2)为右下顶点坐标 | 清空指定矩形内所有内容 |
| Print\_box() | (x,y)为积木左上顶点坐标；content为积木内数字；if\_num判断积木内是否有数字 | 在指定位置打印一个积木，if\_num==1，则内有一数字，否则内部为空 |
| Out\_boxstring  \_animat() | min\_y代表调用函数时右传送带最上方积木上边缘的y坐标 | 使输出传送带内所有积木下移一格，（移出屏幕部分不再打印） |
| Out\_boxstring  \_winning\_animat() |  | 使输出传送带上所有积木下移直至离开屏幕（用于关卡结束时的动画） |
| in\_boxstring  \_animat() | min\_y代表调用函数时左传送带最上方积木上边缘的y坐标 | 使输入传送带内所有积木上移一格，并生成新移入屏幕的输入积木 |

3.2.2.3 机器人图形模块



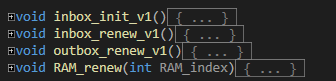
|  |  |  |
| --- | --- | --- |
| 函数名 | 参数含义 | 功能 |
| Print\_robot() | （x,y)为左上'+'符号位置; hold\_stastus记录手中是否有积木；mov\_stat记录移动状态（1代表右移，-1代表左移，0代表静止或上下移动） | 在给定位置输出一个机器人 |
| Robot\_horizon  \_mov() | Dir==1代表向右移动，-1代表向左移动 | 让机器人水平移动一格 |
| Robot\_horizon  \_mov() | Dir==1代表向下移动，-1代表向上移动 | 让机器人垂直移动一格 |
| Robot\_animat() | Dest\_index为目标位置代码（该参数从0-5分别对应：输入传送带、空地0、空地1、空地2、空地3、输出传送带） | 让机器人移动到指定位置的动画 |

3.2.2.4 高级动画模块



|  |  |  |
| --- | --- | --- |
| 函数名 | 参数含义 | 功能 |
| print\_message\_box() | (l\_x, l\_y)为左上顶点坐标；(r\_x, r\_y)为右上顶点坐标；if\_filled表示是否清空内部 | 在指定位置打印提示窗口  （主要于进入退出动画时调用） |
| locked\_level\_animat() | level\_pos表示当前光标指向的关卡 | 关卡未解锁动画 |
| set\_level\_gui() |  | 预存储关卡画面（进入动画调用） |
| set\_menu\_gui() |  | 预存储菜单画面（退出动画调用） |
| level\_chosen\_complete() | level\_pos表示当前光标指向的关卡 | 进入关卡动画 |
| level\_end() | level\_pos表示当前光标指向的关卡 | 退出关卡动画 |
| page\_refresh() | paging表示目标页面编号 | 跳转至指定代码页面 |
| page\_refresh\_special() | paging表示目标页面编号 | 跳转至指定代码页面  （显示存在区别，光标输入专用） |
| game\_initiate\_animat() |  | 游戏进入界面动画 |

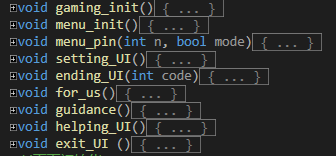
3.2.2.5 关卡内输入输出及空地动画



|  |  |  |
| --- | --- | --- |
| 函数名 | 参数含义 | 功能 |
| Inbox\_init\_v1() |  | 初始化输入传送带及其上积木 |
| Inbox\_renew\_v1() |  | 更新输入传送带内容并播放传送带移动动画 |
| outbox\_renew\_v1() |  | 更新输出传送带内容并播放传送带移动动画 |
| RAM\_renew() | RAM\_index是所更新的空地编号 | 更新指定空地内容 |

注：由于输出传送带初始没有内容，其初始化在“页面初始化”函数中实现，不再另设函数

3.2.2.6 页面初始化



|  |  |  |
| --- | --- | --- |
| 函数名 | 参数含义 | 功能 |
| Menu\_pin() | n代表所选关卡（其他特殊选项用特殊数字代表），mode代表写入方式（mode==1打印光标；mode==0删去光标） | 服务于主菜单界面，负责显示用户的光标位置 |
| Ending\_UI() | Code代表错误码（成功时参数不被使用，失败时每一个错误码代表一种出错情形） | 关卡结算界面初始化 |

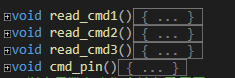
注：该模块其他函数没有参数，且对应功能已在3.2.1图片中说明，故不再赘述

3.2.2.7 基本字符串操作模块



|  |  |  |
| --- | --- | --- |
| 函数名 | 参数含义 | 功能 |
| shrink\_space() | raw\_para为待缩减字符串 | 去除输入字符串最前最后的若干空格 |
| para\_check() | para\_tmp为待检查字符串 | 判断一个字符串是否为纯(合法)数字 |
| interpret\_digit() | para\_tmp为待转换字符串 | 将纯(合法)数字字符串转为整形存储 |

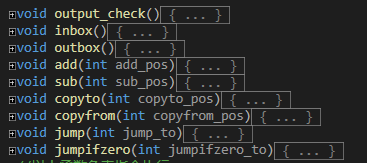
3.2.2.8 指令读取模块



|  |  |  |
| --- | --- | --- |
| 函数名 | 参数含义 | 功能 |
| cmd\_pin() |  | 在执行过程中更新指令光标位置 |

注：该模块其他函数没有参数，且对应功能已在3.2.1图片中说明，故不再赘述

3.2.2.9 指令执行模块







|  |  |  |
| --- | --- | --- |
| 函数名 | 参数含义 | 功能 |
| output\_check() |  | 检测实际输出积木与期望是否相同，并根据情况调用abort或succed函数 |
| execution() |  | 核心执行模块，在主函数中直接调用 |
| abort() | Code为要传递的错误码 | 失败时的中继函数，桥接执行函数和ending\_UI界面函数，传递错误码code |
| succeed() |  | 成功过关时的中继函数，功能类似abort，但不传递错误码 |

注：该模块其他函数均为基本指令的实现，实现功能与大作业要求文档中相同，故不再赘述

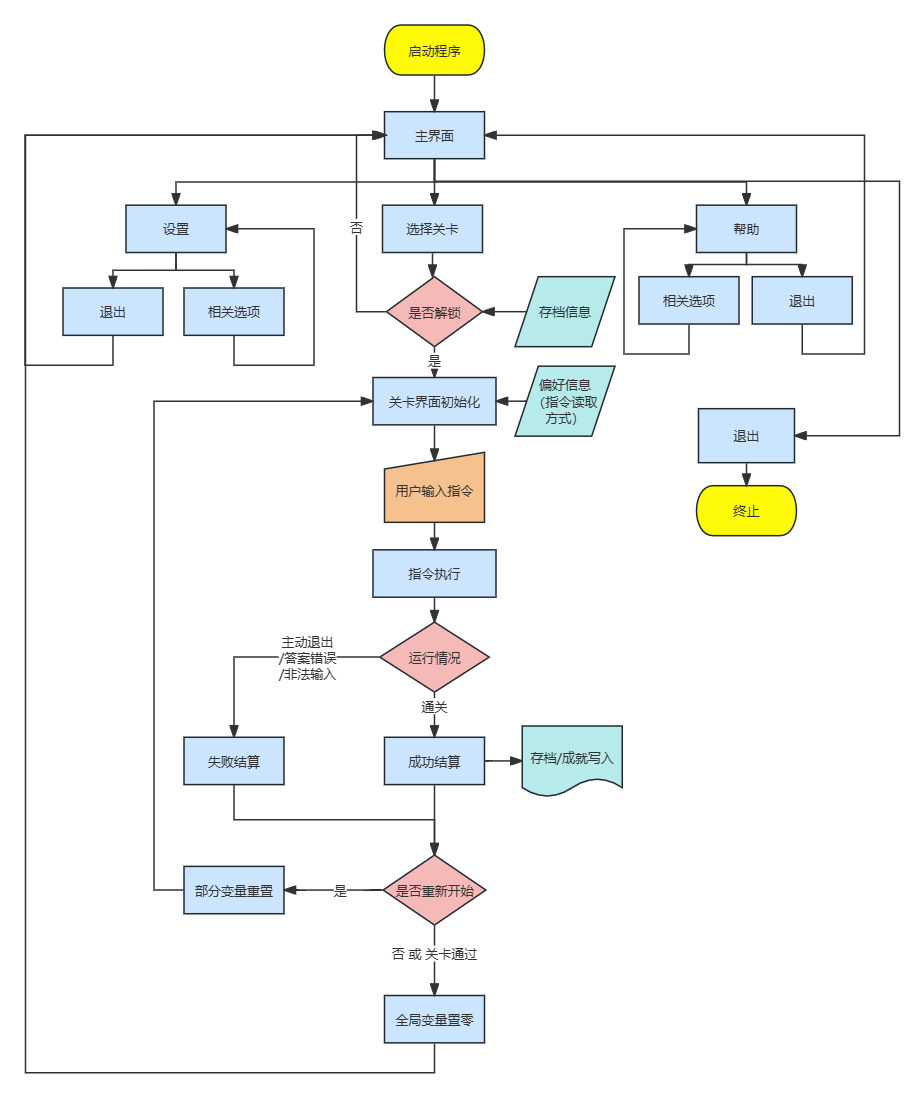
3.2.2.10 其他函数





|  |  |  |
| --- | --- | --- |
| 函数名 | 参数含义 | 功能 |
| thread\_speed() |  | 多线程监控函数，用于运行过程中终止、加速、减速的监控与调整 |
| reset() |  | 重置函数，用于结束关卡时各变量复位 |
| reset\_cmd() |  | 输入重置函数，同上 |

### 3.3 代码部分（主程序流程及运行逻辑）



## 4.整体游戏界面设计

### 4.1 主菜单

4.1.1 主界面



红色代表未解锁，白色代表已解锁，关卡下方星号代表已完成的关卡挑战（代码行数挑战和总执行次数挑战）

4.1.2 设置界面



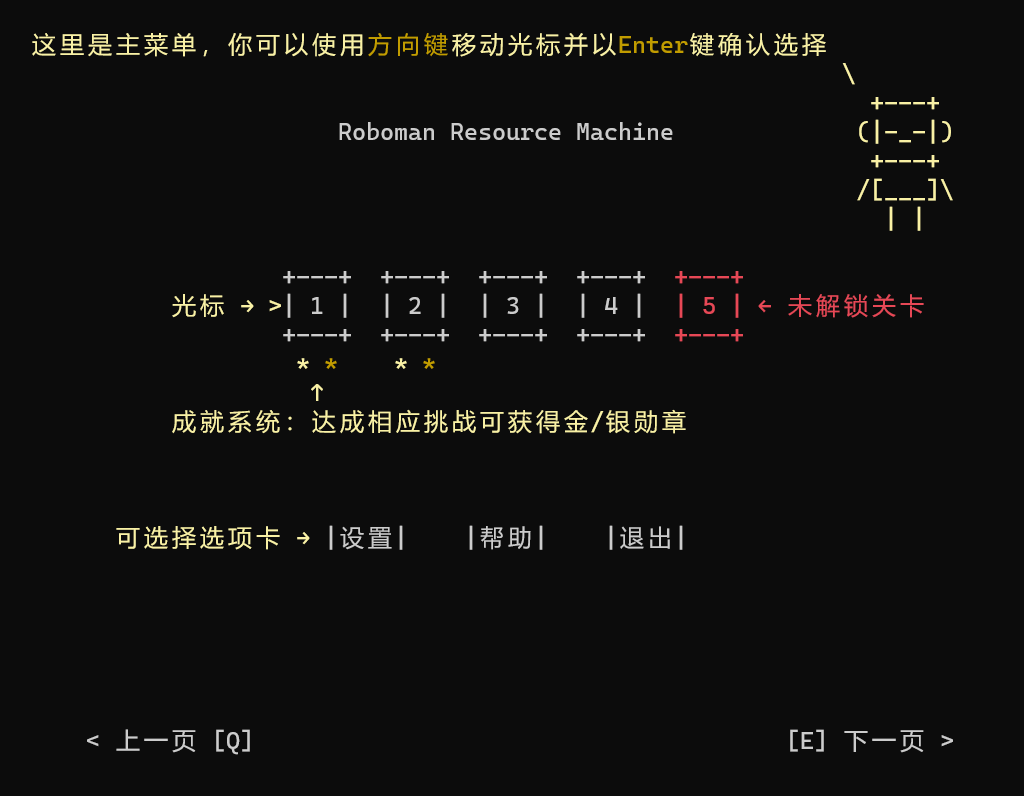
4.1.3 退出界面

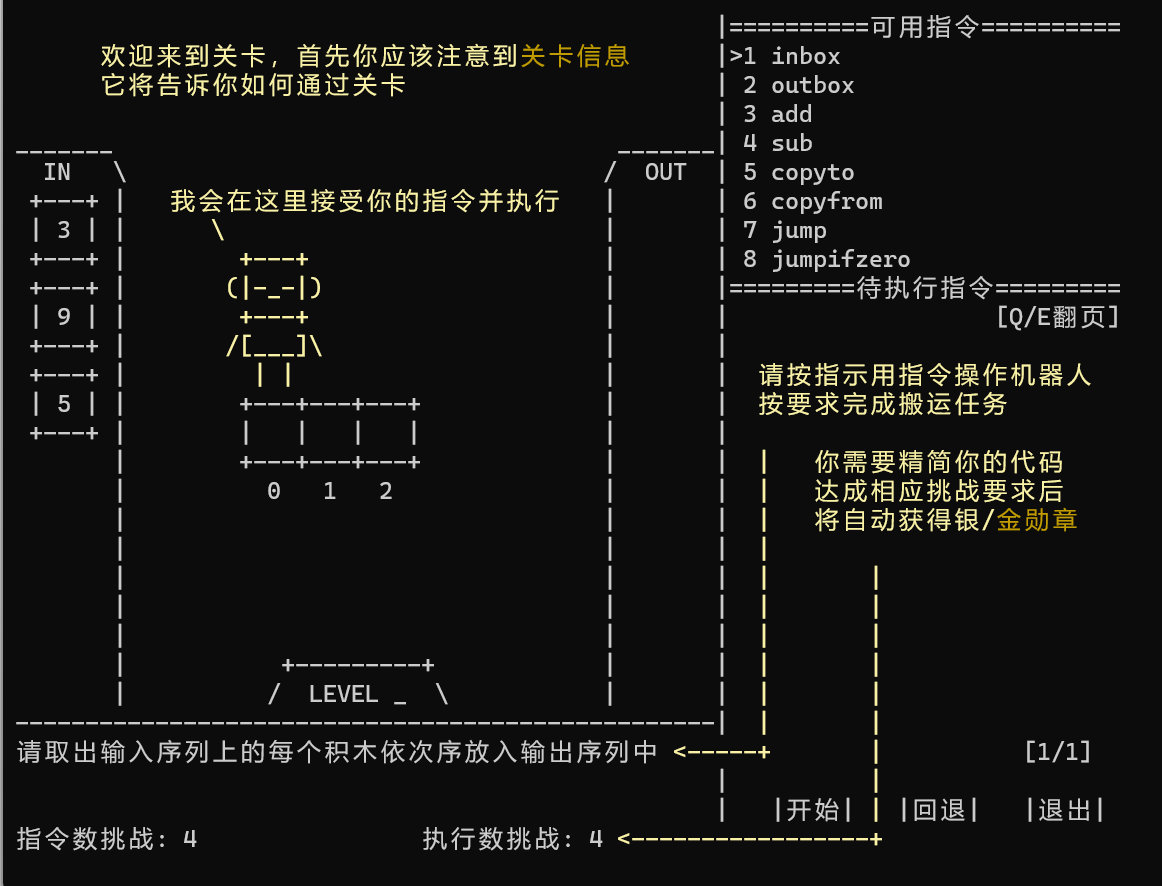
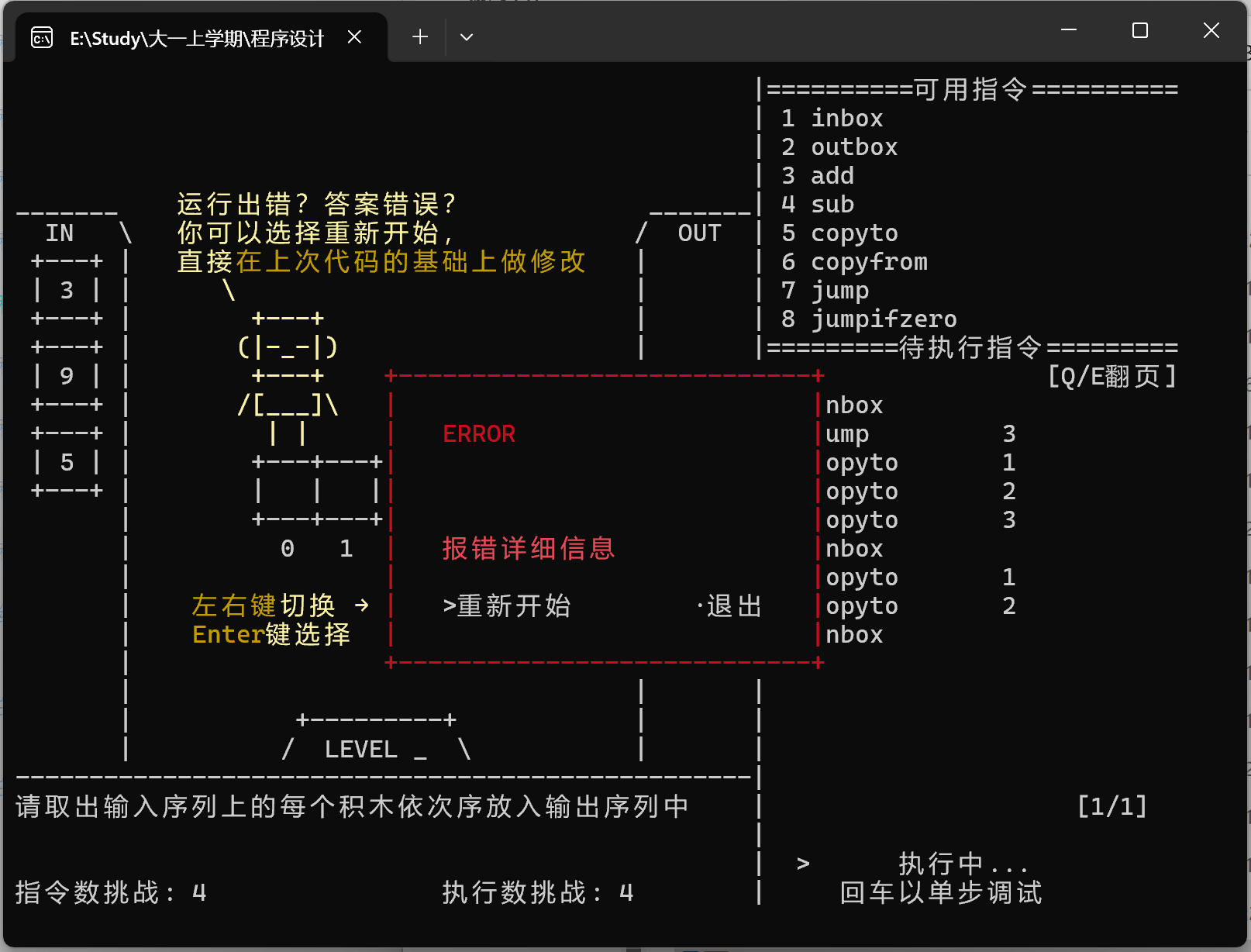


4.1.4 帮助界面

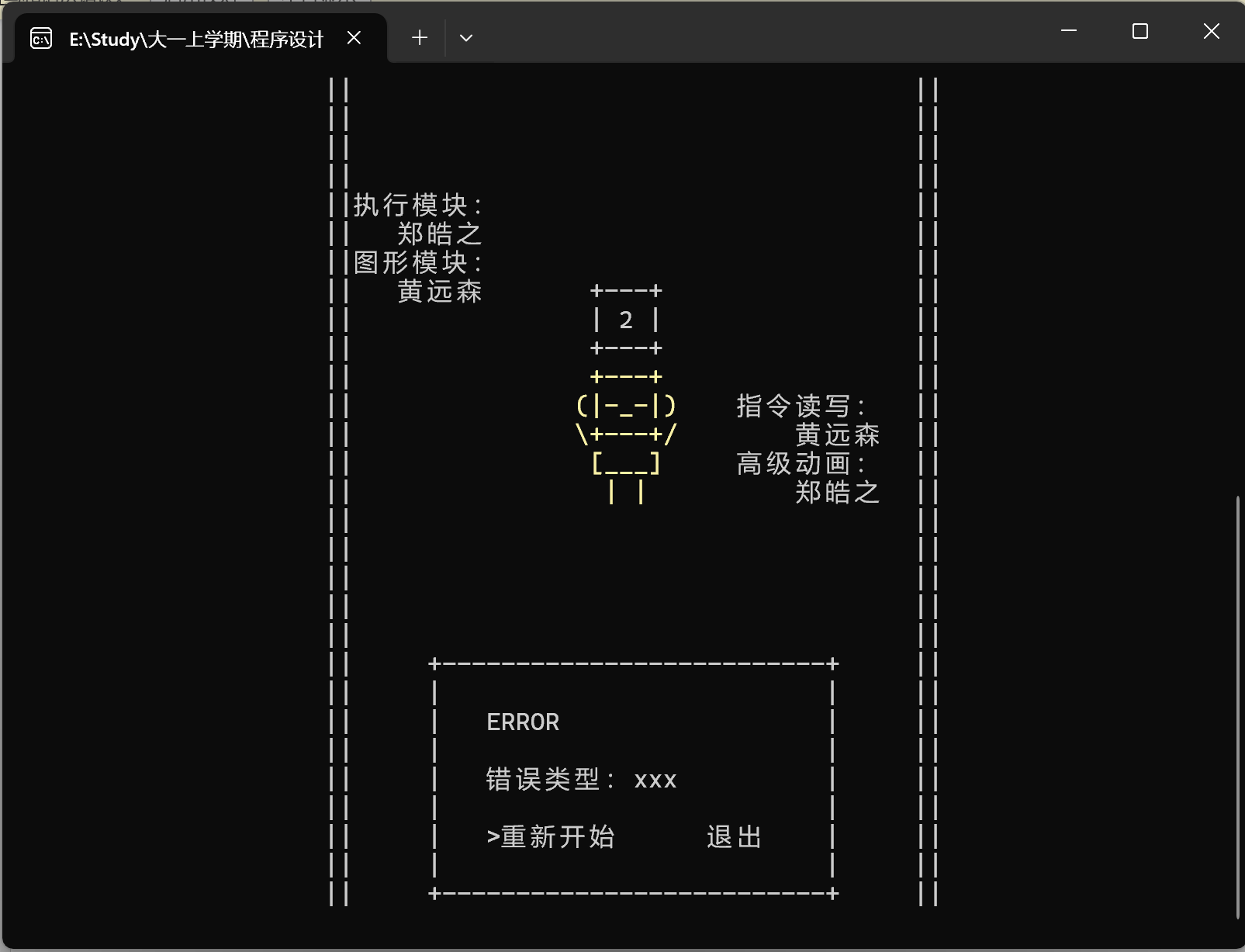
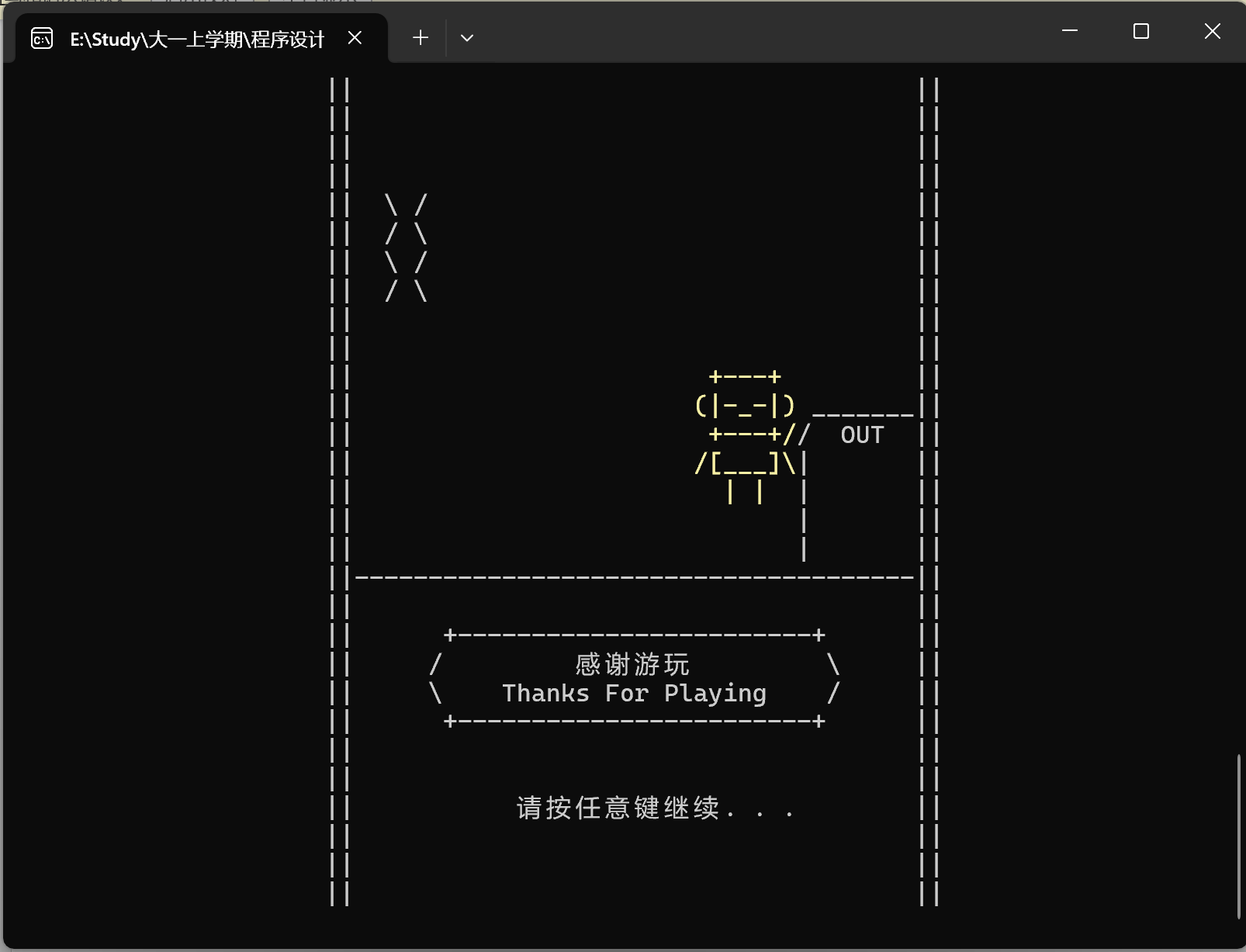


4.1.4.1 教程界面

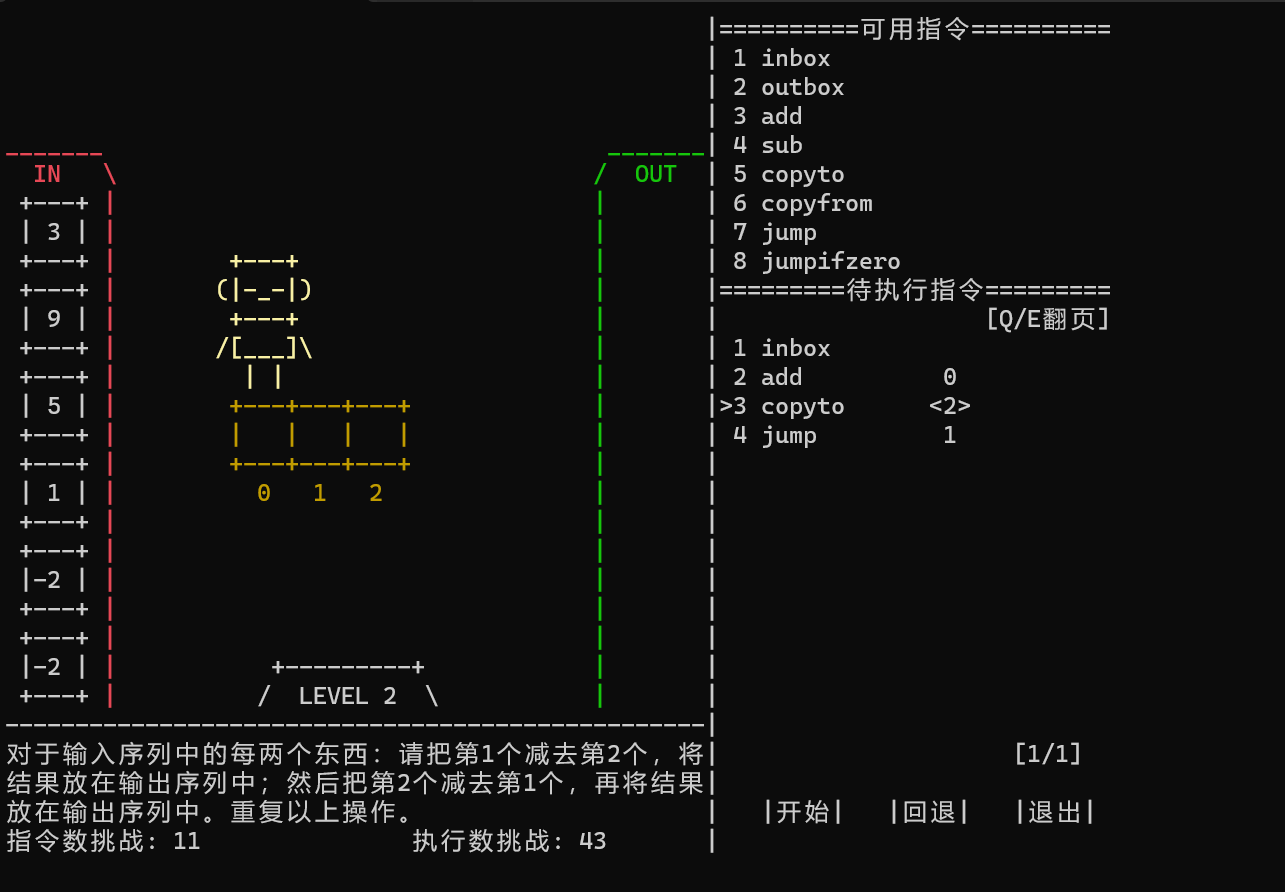
 

4.1.4.2 关于界面

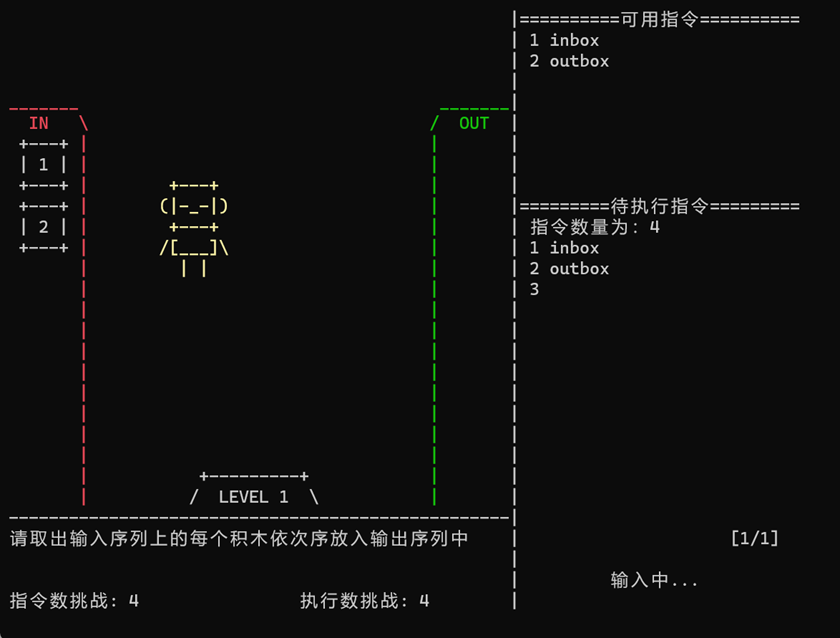
### 4.2关卡界面

4.2.1光标输入模式

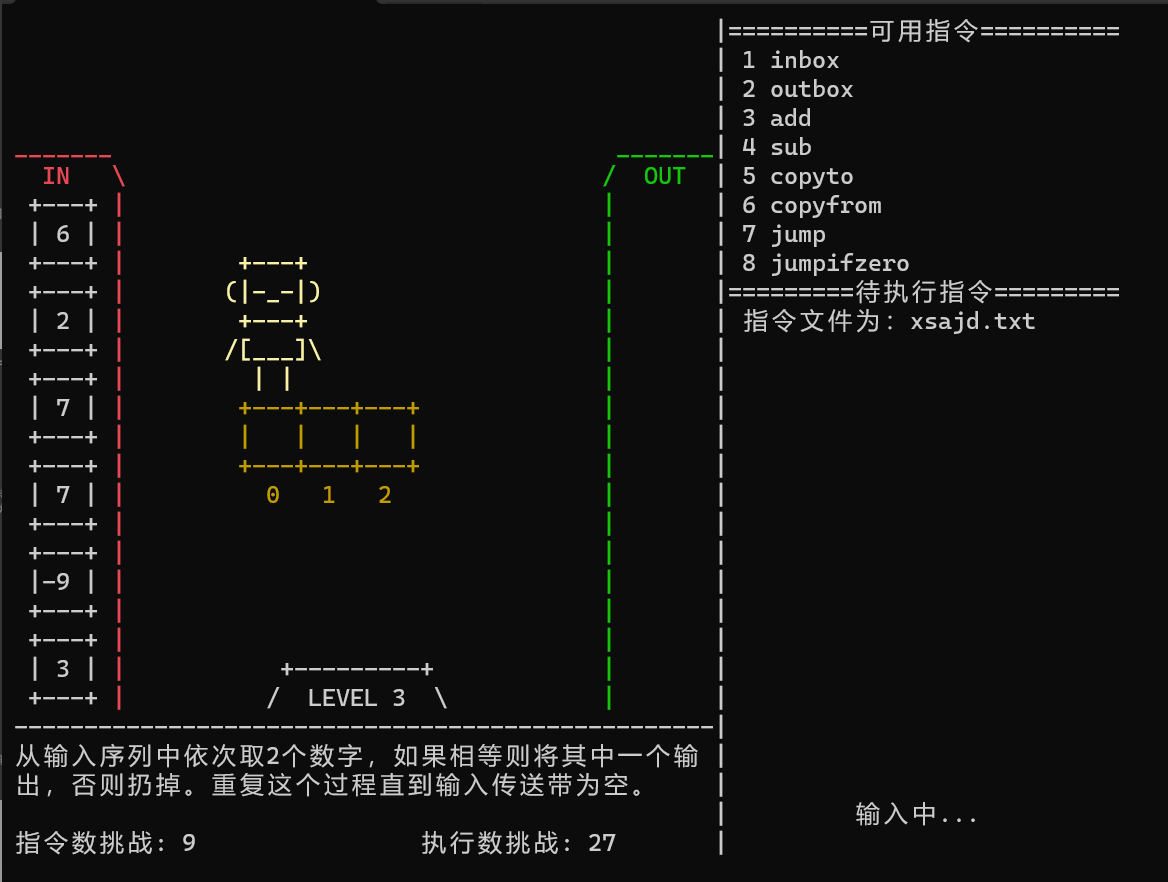


左下方为任务描述和基本关卡信息，右侧为输入区域。用户可使用Q/E在多页指令间来回切换

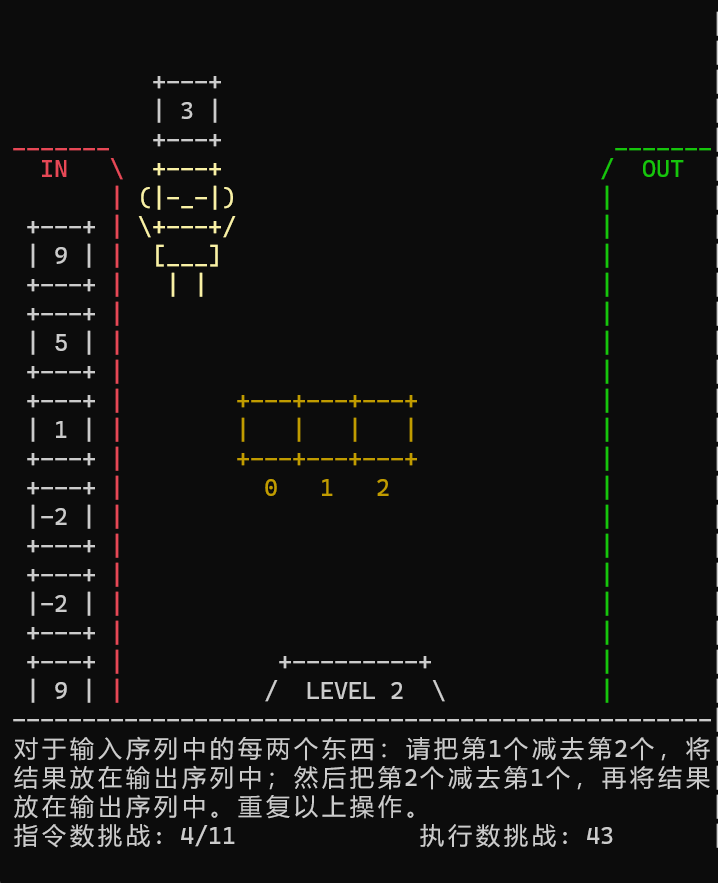
4.2.2键盘输入模式



4.2.3 文件输入模式



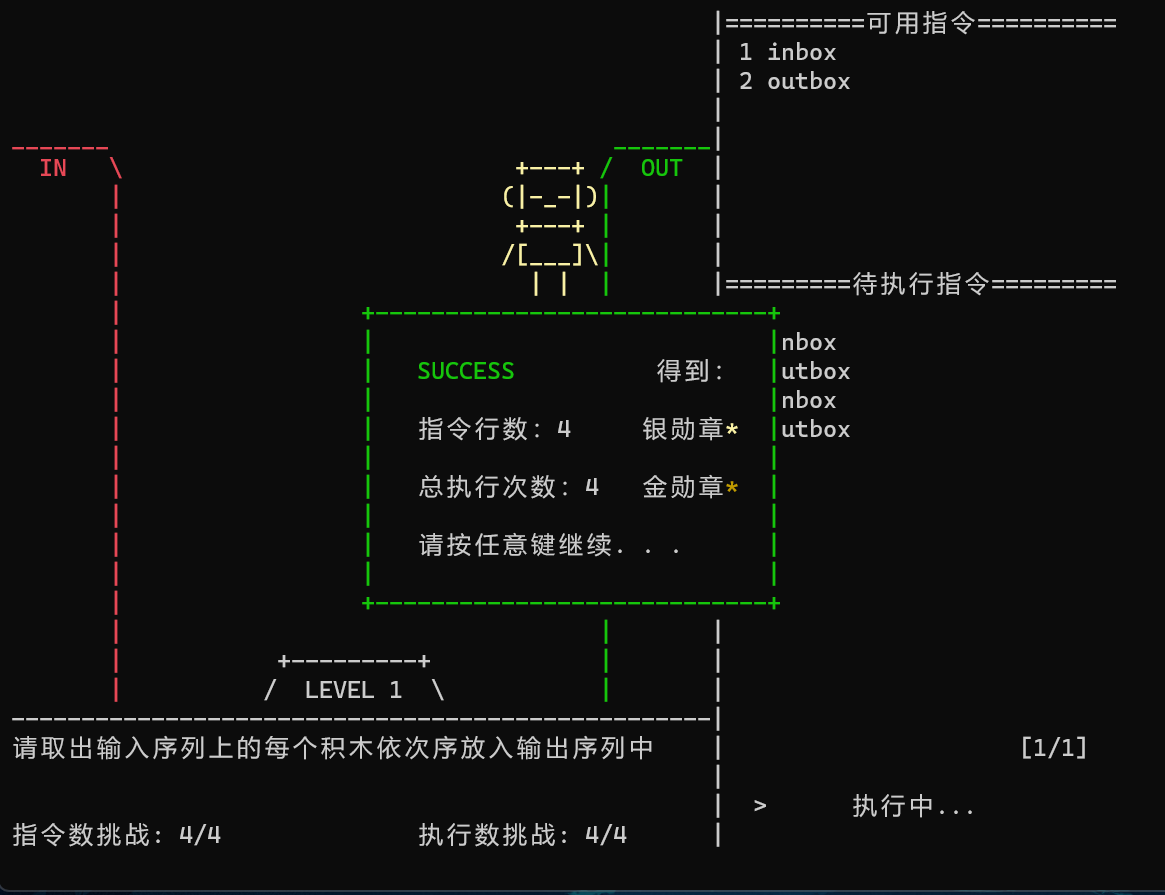
4.2.4 执行中

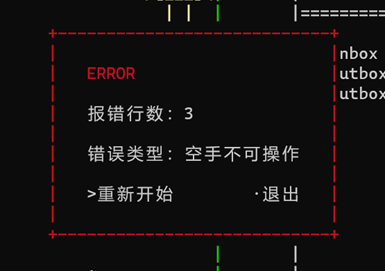
 

执行中时，左下的关卡文件信息旁会同步显示当前的实际指令执行次数和行数

### 4.3 结算界面



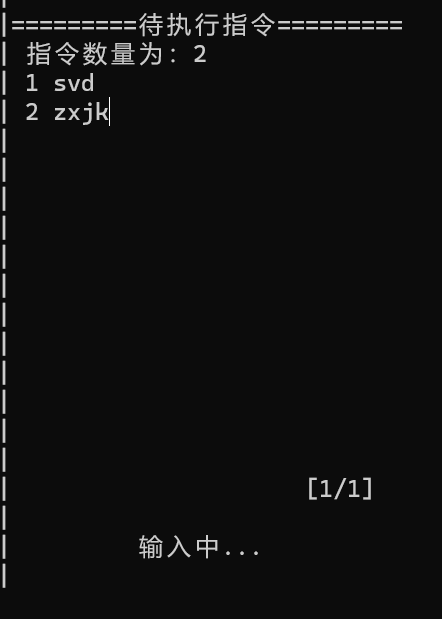
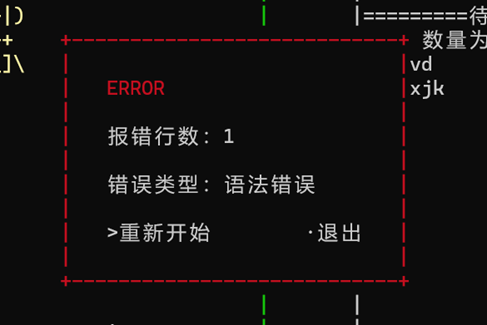
 

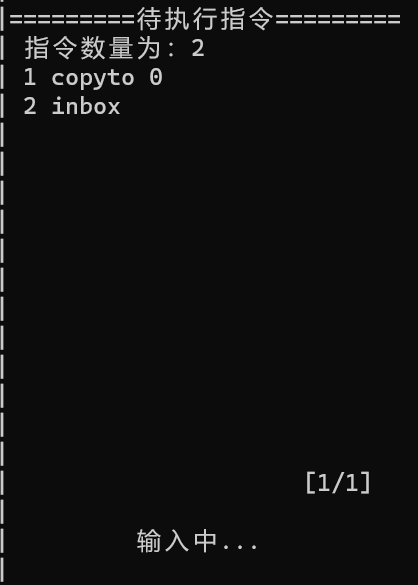
结算界面会根据情况不同显示各类信息，用户也可在此界面选择保留已输入指令重新开始关卡或直接退出到主界面

## 5.游戏测试

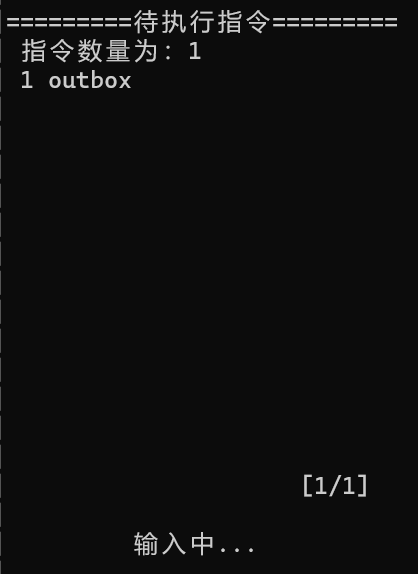
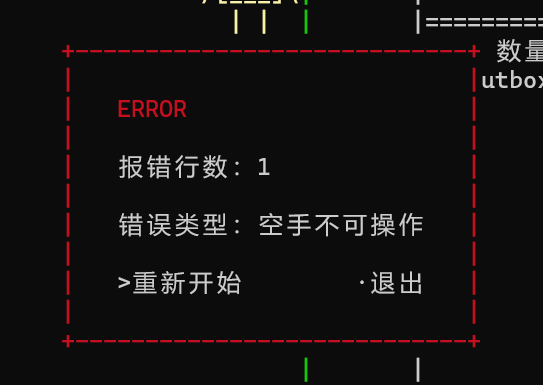
5.1 语法错误（指令不合法 或 操作数不为整数）

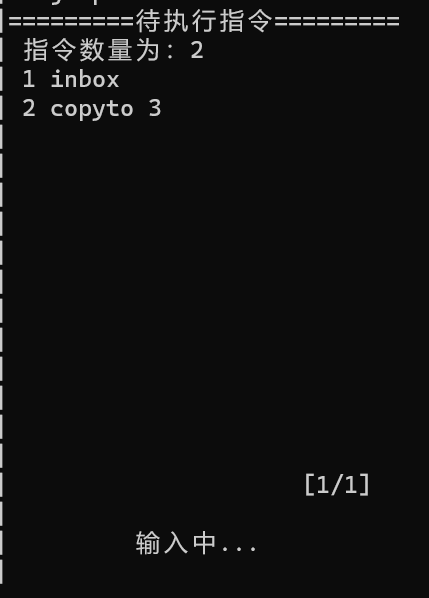
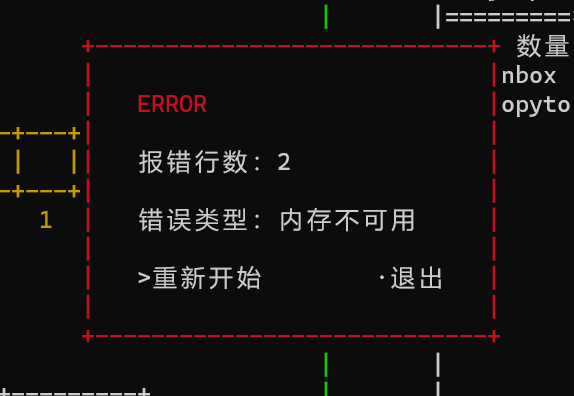
5.2 指令不可用（指令符合语法，但在本关中不允许使用该指令）

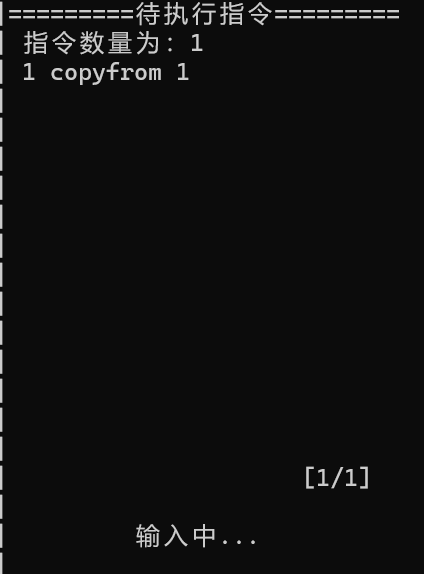
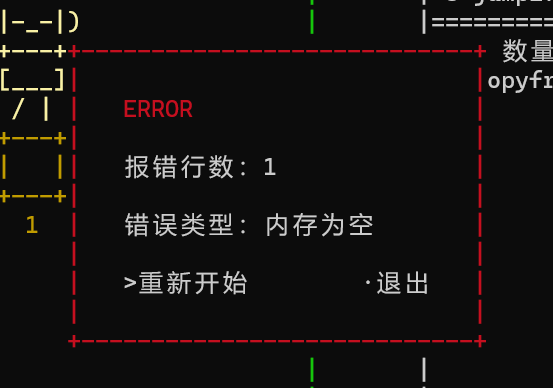
5.3 空手不可操作（在手中应持有积木的步骤中未持有积木）

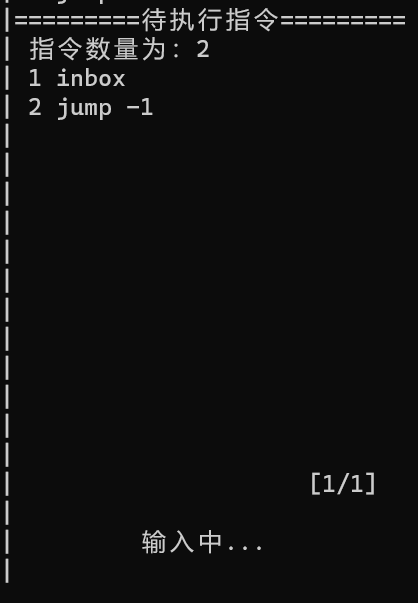
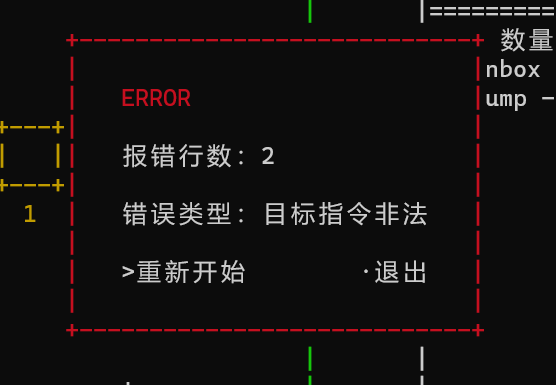
5.4 内存不可用（使用了本关中没有的空地位置编号）

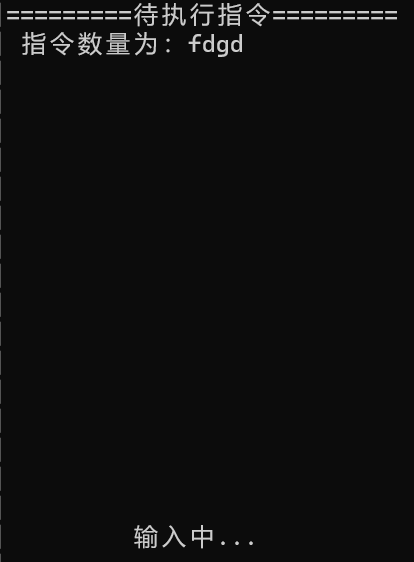
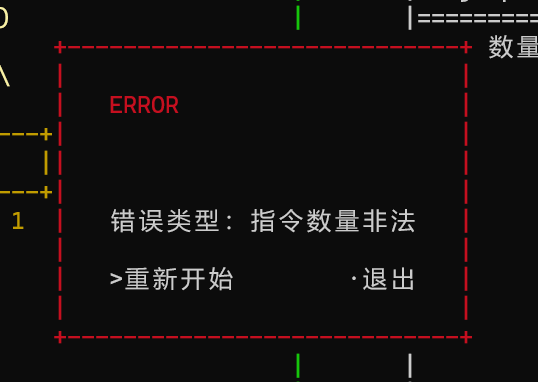
5.5 内存为空（在指定空地应有数字时，如copyfrom时，其中实际没有数字）

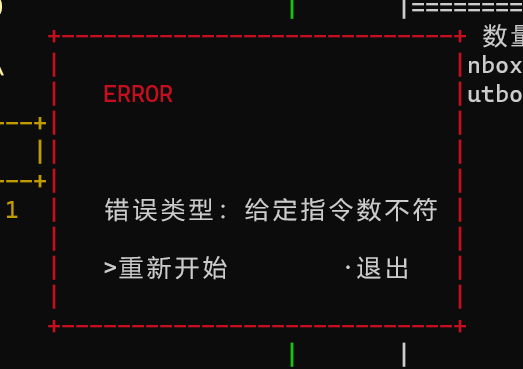
5.6 目标指令非法（jump，jumpifzero的目标指令行数不存在或为负数）

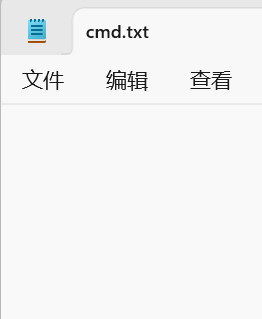
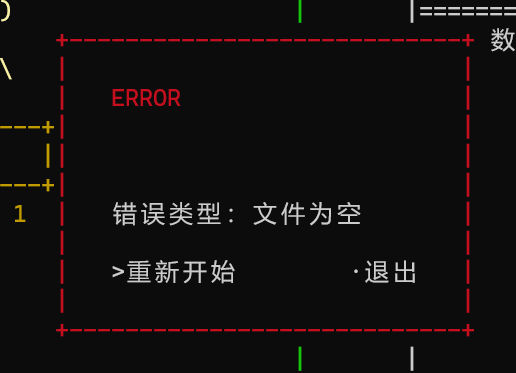
5.7 指令数量非法（指令输入不为正整数 或 输入的指令数量过多）

5.8 给定指令数不符（文件输入时，文件中的指令数量和实际指令行数不匹配）

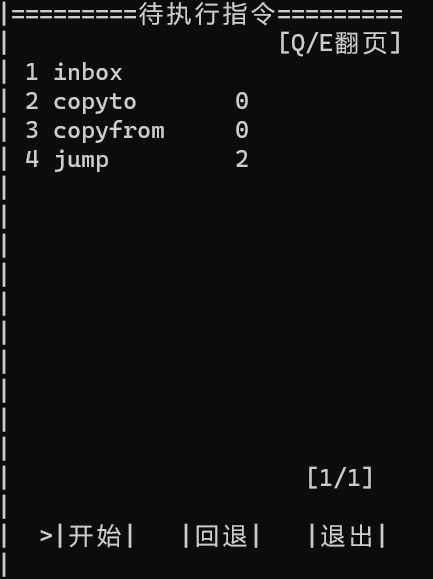
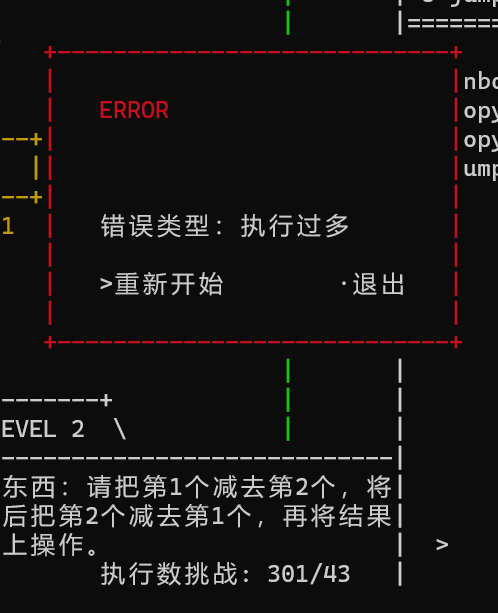
5.9 文件为空

5.10 文件不存在

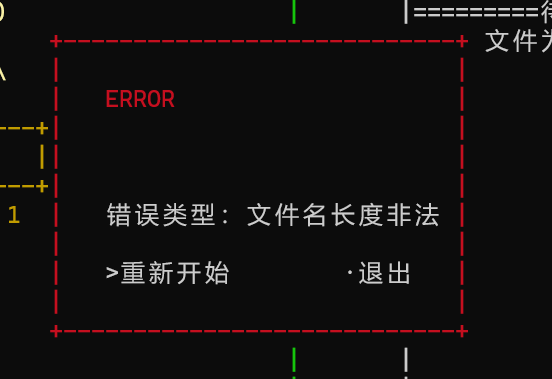
 

5.11 执行过多（为防止死循环，关卡内执行指令次数达300次时自动退出报错）

5.12 文件名长度非法（文件名长度超出界面边框 或 文件名为空）

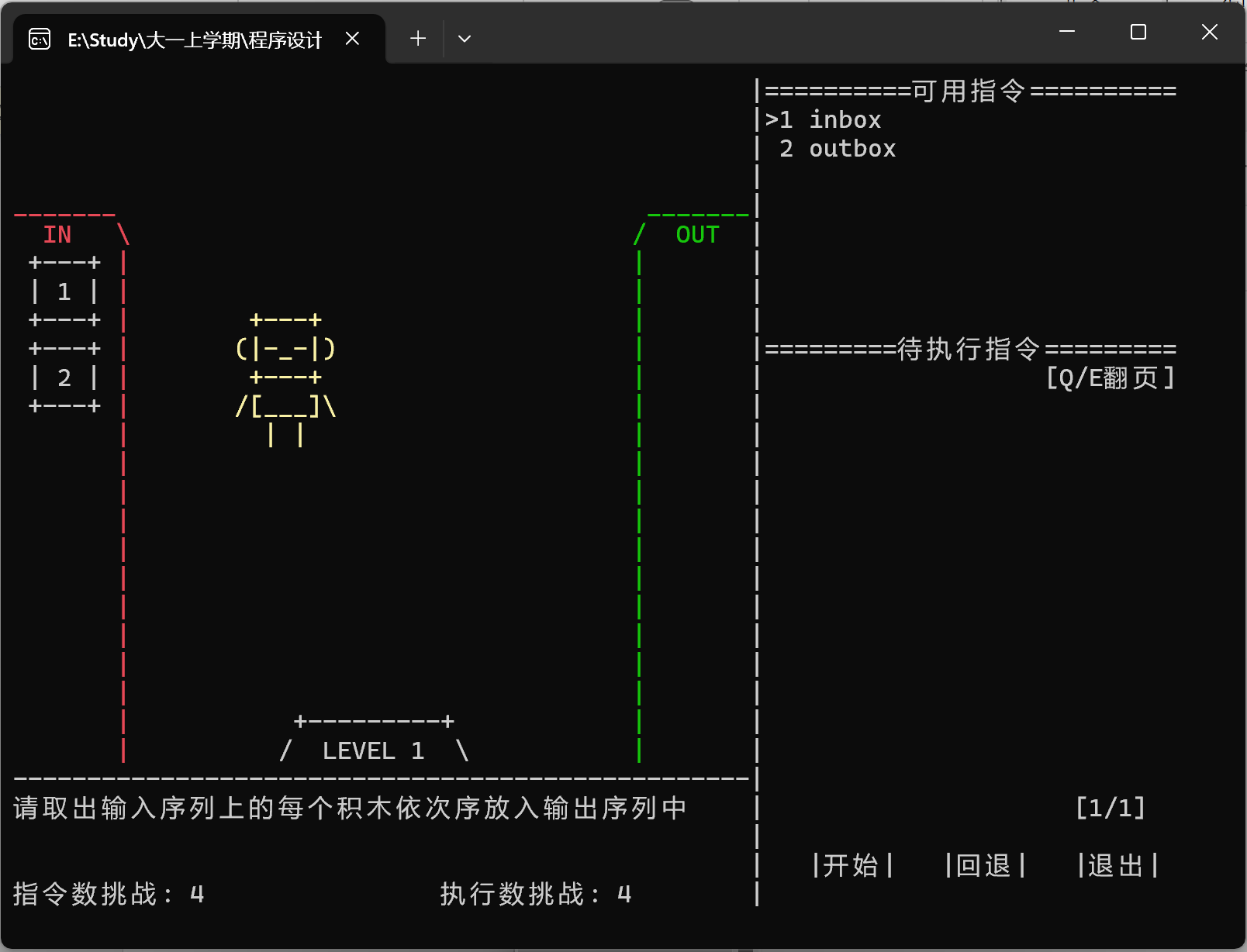


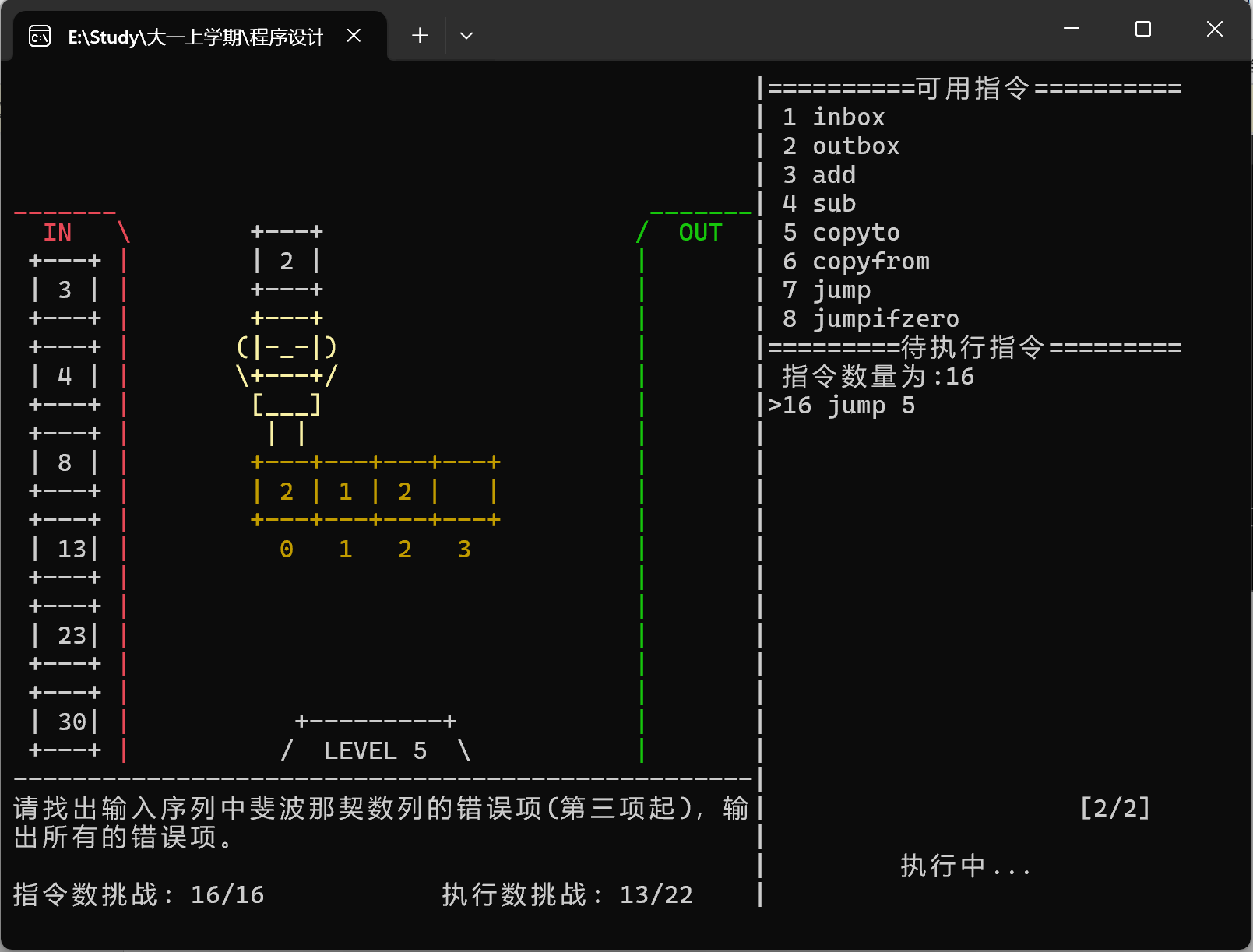
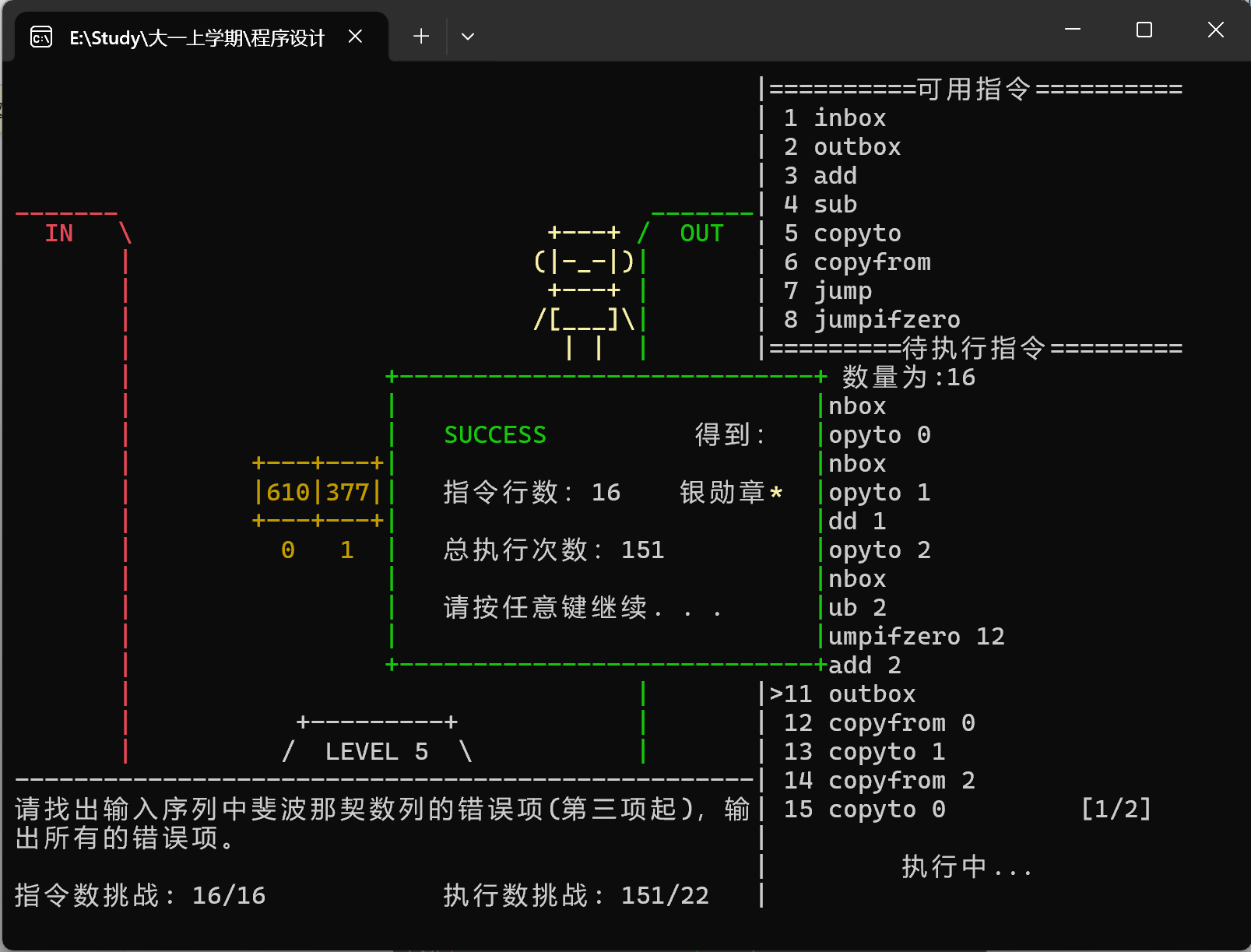


5.13 在主界面中选中未解锁关卡



5.14 游戏正常运行截图

## 6.自由创新关卡

### 6.1 Level 4

任务：不断从输入序列取积木从零开始累加，直到本次累加的积木为0时将累加结果输出。重复这一过程。

1. 关卡的输入输出

i. 输入序列：5 7 0 5 -4 8 0 0 0 5 -4 7 4 -2 6 0 1

ii. 目标输出序列：12 9 0 0 16

1. 可用空地数：3
2. 可用指令集：inbox,outbox,copyfrom,copyto,add,sub,jump,jumpifzero

标准解法：（附于游戏根目录内）

10

jump 4

copyfrom 0

outbox

inbox

jumpifzero 3

copyto 0

inbox

jumpifzero 2

add 0

jump 6

### 6.2 Level 5

任务：请找出输入序列中斐波那契数列的错误项(第三项起)，输出所有的错误项。

1. 关卡的输入输出

i. 输入序列：1 1 2 3 4 8 13 23 30 55 89 150 233 377 615 988

ii. 目标输出序列：4 23 30 150 615 988

1. 可用空地数：4
2. 可用指令集：inbox,outbox,copyfrom,copyto,add,sub,jump,jumpifzero

标准解法：（附于游戏根目录内）

（1）用于得到银勋章

16

inbox

copyto 0

inbox

copyto 1

add 1

copyto 2

inbox

sub 2

jumpifzero 12

add 2

outbox

copyfrom 0

copyto 1

copyfrom 2

copyto 0

jump 5

（2）用于得到金勋章

22

inbox

inbox

inbox

inbox

inbox

outbox

inbox

inbox

inbox

outbox

inbox

outbox

inbox

inbox

inbox

outbox

inbox

inbox

inbox

outbox

inbox

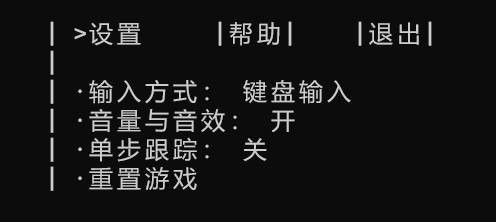
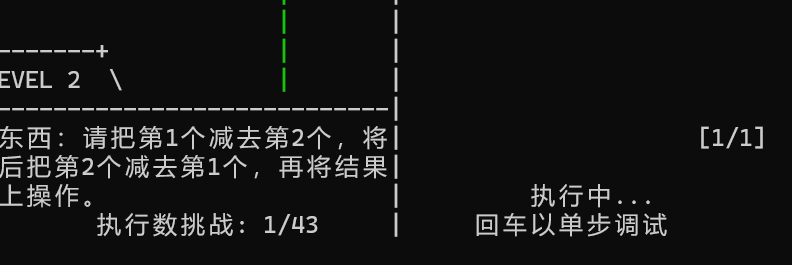
outbox

## 7.拓展功能/特色功能

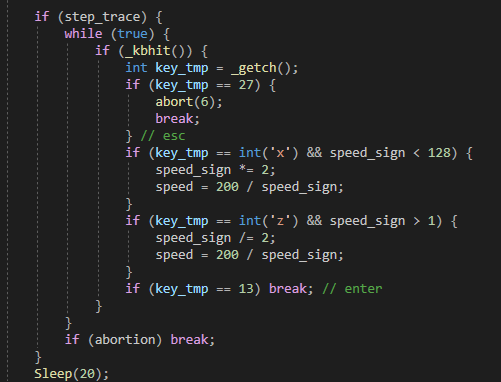
### 7.1 单步调试

7.1.1 功能描述

用户可在“主菜单->设置->单步跟踪”中启用该功能。启用后，关卡中机器人每执行完一步指令就会停下，直到用户键入enter后再开始执行下一条指令。关闭该功能能时，会连续播放机器人的执行和移动动画。

7.1.2 实现方式



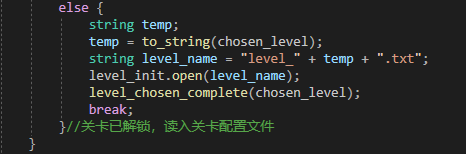
在execution()函数中判断全局变量step\_trace的值，若单步调试模式启用，execution在每次循环（也即每次执行指令）的最开始会进入上图中的while死循环，直到用户输入enter才会终止死循环，进行execution后续步骤

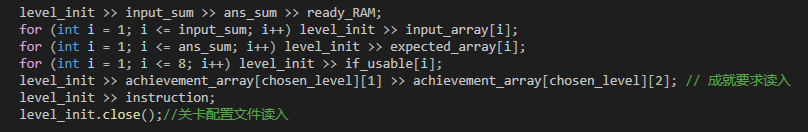
### 7.2 可拓展修改的关卡文件

7.2.1 功能描述

本游戏每个关卡配置信息都存储在一个txt文件中，用户无需修改源代码，仅需修改该文件即可拓展或修改关卡内容。该txt文件的具体格式见8.3.3

7.2.2 实现方式



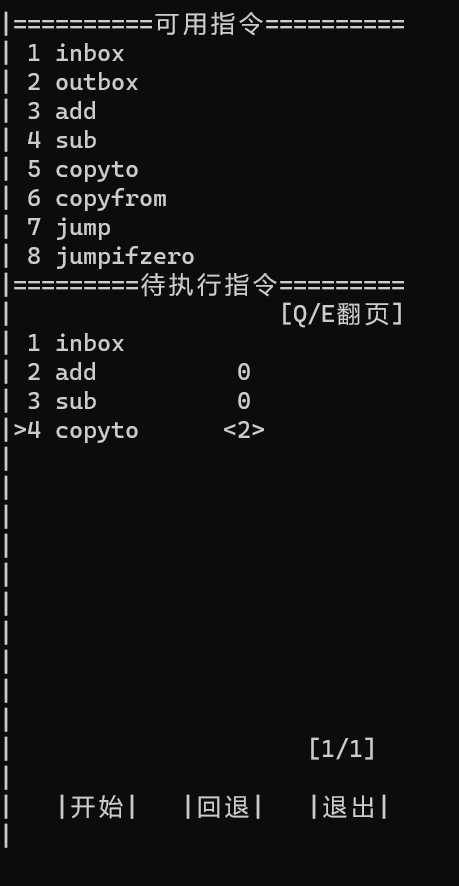


将要读取的关卡编号转成string类型，再与文件名格式用string加法拼接，从而动态改变需要读入的关卡配置文件名，最终在ifstream level\_init中打开该文件，按格式读取文件内各项数据。

### 7.3 方向键光标输入模式

7.3.1 功能描述

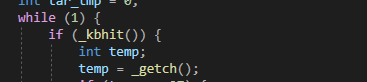
该功能允许玩家使用键盘上的方向键选择各种选项，用enter键确认，esc键退出，以此操控游戏界面上的光标

该功能在游戏中多处出现（主菜单、设置界面等）。其中最能体现该功能特点的是关卡中的“光标输入”模式。接下来将进行着重介绍。

该模式下，用户可使用方向键在可用指令列表中选择功能，按enter确认选中。若选中指令在可用指令列表中，该指令将被加入待执行列表（若为空地相关指令，操作数默认为0；jump和jumpifzero操作数默认为1）。用户也可在待执行列表中选中指令，enter键选中后可再次使用左右方向键调整操作数，再次enter选中新操作数。此外，页面下方还有三个可被选中的选项，功能分别为“开始执行”，“删去上一条输入的指令”和“退出到主界面”。

该模式下用户无需输入指令数量，无需手动键入指令名称，操作数上下限也由程序自动计算并框限（jump指令操作数上限为目前已键入的指令数，该数字会随指令的加入和删除实时更新）。因此，该模式理论上不会出现语法错误、操作数错误、指令数量错误等诸多问题，极大地提高了用户友好程度和游戏的鲁棒性。

7.3.2 实现方式



该功能主要依赖\_kbhit()和\_getch()两个函数实现

在一个死循环中不断检测键盘输入，一旦检测到输入，将键码存入temp，并进行后续操作

注：由于不同界面中各选项的位置、功能、判断条件都有较大区别，该输入功能无法集成到一个函数或一个模块中，只能在具体界面中分开编写。

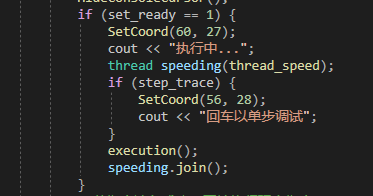
### 7.4 基于多线程的“动画速度调节”和“执行中退出”

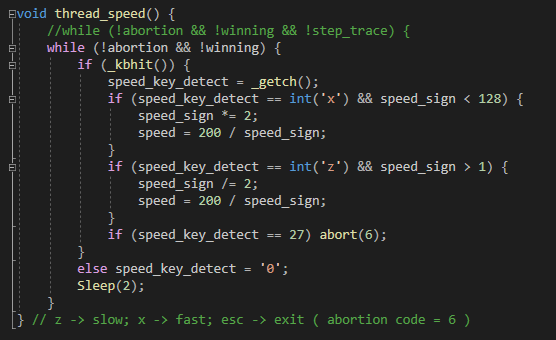
7.4.1 功能描述

该功能允许玩家在机器人执行指令的过程当中进行额外操作。玩家可按x加快动画播放速度，按z降低动画播放速度，按esc终止关卡并进入右图界面

7.4.2 实现方式

由于机器人执行指令、播放动画的过程中程序无法接受新的指令，因此该功能难以以传统方式实现。故我们使用了<thread>头文件中的多线程功能。





每一关卡开始执行时，会启用一个新的线程，并在该线程中执行监听函数thread\_speed。该函数会实时检测键盘输入，修改全局变量speed（负责加减速）或调用abort函数（终止关卡）。当所有指令执行完毕或以其他形式终止时，该线程同步终止。

### 7.5 保留用户指令的“重新开始”功能

7.5.1 功能描述

该功能允许玩家在闯关失败的结算界面中选择“重新开始“。若选择该选项，玩家将可以不回到主界面，直接重玩本关。

若玩家选择“光标输入”模式，程序还将保留玩家上一次输入的所有指令，以便进行修改而不必全部重写

7.5.2 实现方式

该功能的运行逻辑详见3.3中流程图

### 7.6启动动画/退出动画

7.6.1 功能描述

启动游戏/进入关卡/退出关卡时，会有顺滑的过场动画。（该动画效果难以以截图形式展示，请在演示视频中观看该功能）

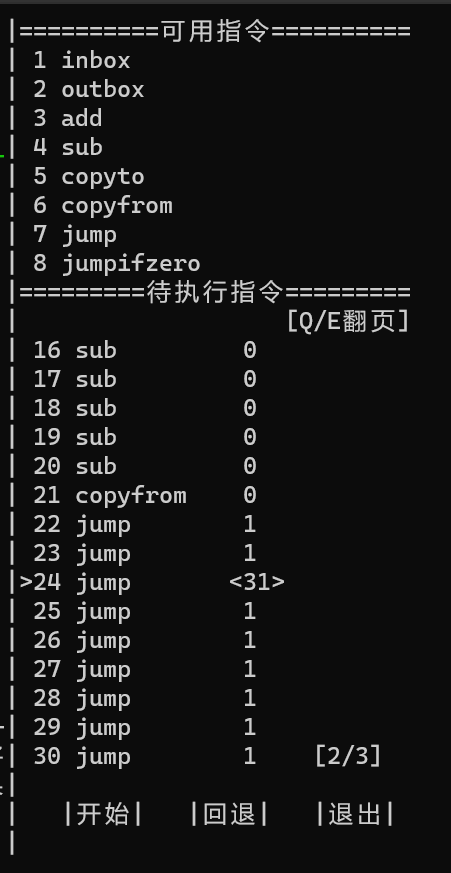
7.6.2 实现方式

动画层面主要依赖rect\_clear和print\_message\_box两个函数。

该动画涉及高频率、大规模的画面刷新。实践过程中，我们发现，为保证动画流畅性，rect\_clear函数必须使用printf而非cout。

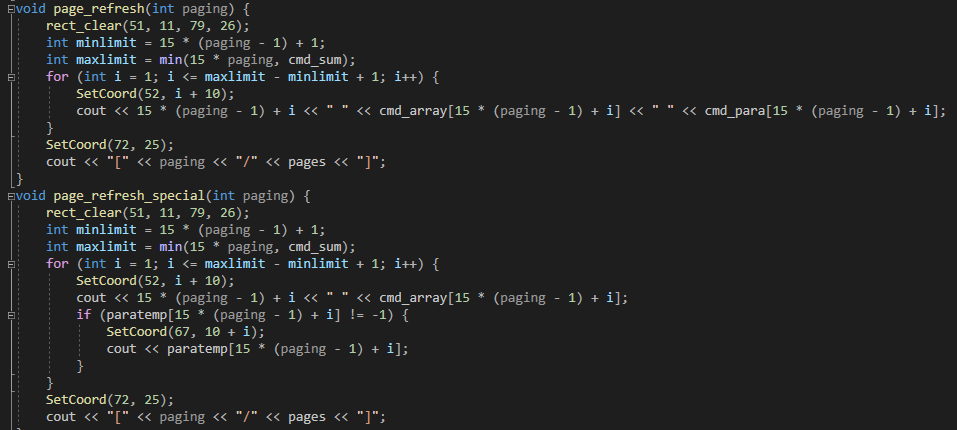
### 7.7 多页指令

7.7.1 功能描述

由于页面大小限制，单页中待执行指令列表只能输入十几行指令。该功能可以让用户在多页中输入指令（一页满后自动向后翻页），从而将指令行数上限提高到300行。同时，该功能在执行指令过程中可让箭头光标在不同页面中跳转。

该功能与三种指令输入方式均兼容。在光标输入模式下，用户还可以使用q、e键手动进行翻页。

7.7.2 实现方式



主要依赖这两个函数实现。其中，page\_refresh\_special是适配于光标输入模式的翻页函数

### 7.8 其他

7.8.1 图形化教程

7.8.2 首次进入游戏时不进入主菜单，而直接进入教程

7.8.3 背景音乐

7.8.4 用户偏好设置实时保存

7.8.5 成就系统

7.8.6 关于界面

## 8.运行说明及文档格式说明

### 8.1 所用的IDE /C++版本

我们使用visual studio进行编译和调试，<thread>头文件、to\_string函数等均要求使用C++11及以上版本

鉴于我们亲自尝试使用dev-c++等编译器打开后发生较为严重的编码和编译错误，我们强烈建议使用visual studio进行编译

若使用其他编译器出现了无法编译的情况，请使用“-static-libgcc -lwinmm”进行编译

另：由于使用了许多Windows API接口，Roboman Resource Machine只能在Windows设备上运行

### 8.2 启动器的使用场景

由于微软在win11中将旧版的windows控制台更新为了windows terminal，有诸多控制台界面相关的函数出现了不兼容的问题。为在一定程度上解决这一问题，游戏文件夹中配有一个win11专用启动器

仅Win11电脑可正常运行该启动器，该启动器将在一个大小合适的windows terminal界面中启动游戏本体程序。当然，直接运行游戏本体程序也不会有致命问题。

低版本windows电脑直接运行游戏本体就可获得大小合适的窗口。

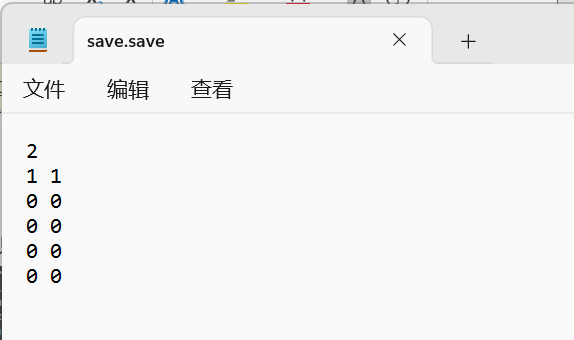
### 8.3 存档/配置文件的格式

**8.3.1 存档文件**

第一行代表已通关的最大关卡数

第2~6行对应1~5关。每行两个数，分别代表是否取得银勋章、是否取得金勋章

E.G.



本文档代表已通过第二关（解锁第三关），第一关取得两枚勋章，其他关卡未获勋章

**8.3.2 用户偏好文件**

存有三个数字，分别代表输入方式（0代表键盘输入，1代表文件输入，2代表光标输入）、音乐开关、单步跟踪开关

E.G.



输入方式：键盘输入；音乐：开；单步跟踪：开

**8.3.3 关卡配置文件**

第一行为3个整数，分别代表 输入积木数量、目标输出积木数量、可用空地数

第二行为 输入序列的内容

第三行为 目标输出序列的内容

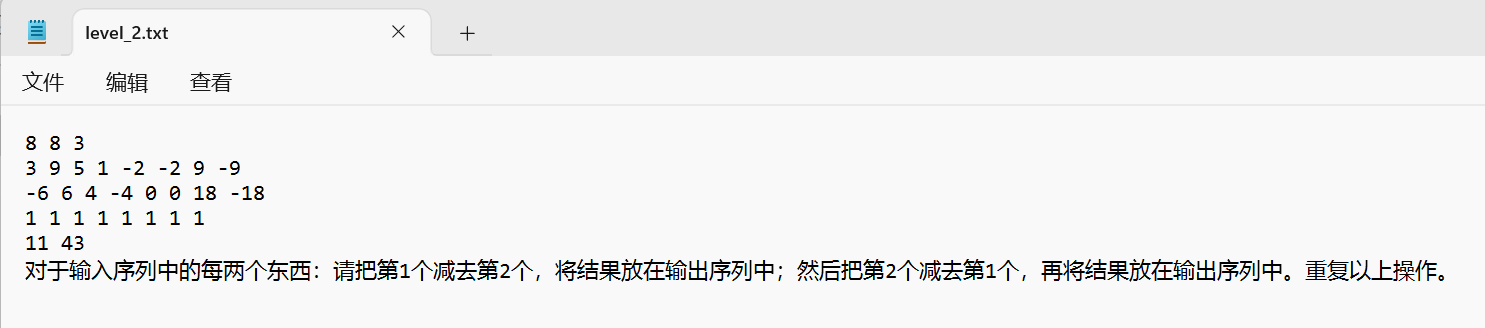
第四行为8个bool数，记录本关卡中各指令的可用情况。从左到右分别对应指令

inbox, outbox, copyfrom, copyto, add, sub, jump, jumpifzero

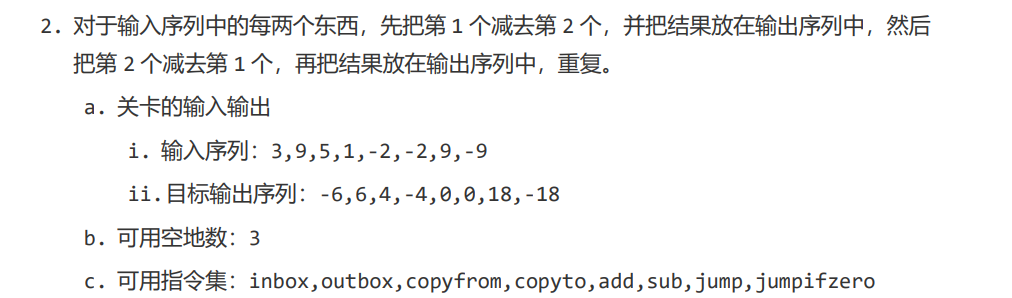
第五行为两个整数，记录勋章挑战的要求，前者为代码行数要求，后者为执行总次数要求

第六行开始是该关卡的任务说明（该说明中不可以有空格）

E.G.



该配置文件即对应第二关的作业要求（如下图）。此外，若要完成勋章挑战，代码行数需在11行及以内，总执行次数在43次及以内



**8.3.4 关于我们界面**

由于中文编码问题，请不要尝试打开Crdit.txt文件，可能会导致程序显示画面出现错误。

其内部逐行保存结算界面，并在用户进入“关于”后逐行读取打印。

## 9.录屏演示链接

清华云盘：<https://cloud.tsinghua.edu.cn/f/fbfc647fcc324ed08b87/>

百度云盘(备用)：

链接：<https://pan.baidu.com/s/1AafwO8VmLUkSbnQ08omdCg?pwd=gpga>

提取码：gpga