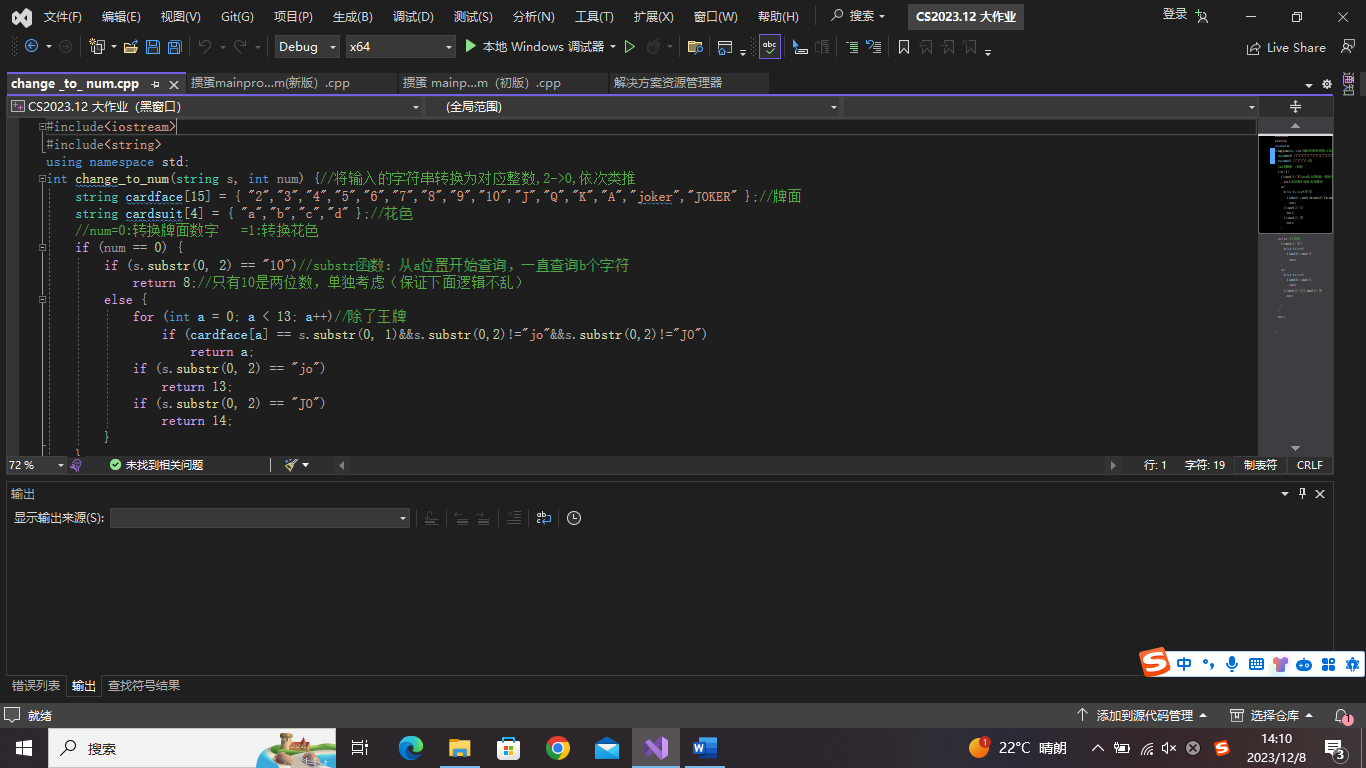
掼蛋托管机器人——书面报告

1. 概述

本机器人历经三版，通过近25个源文件和头文件进行链接，其中包含了数组，排序，选择，指针，宏命令，容器，多个库函数，easyx等多种方式进行实现，最终实现输入——打牌——输出的逻辑

二．实现

2.1从最初的main函数开始

首先，我们将main函数作为一个接口，它要做的只有输入数据的功能，而输入的牌面如何储存？在最初版（文件里有），这个工作是存入数组的，而为了让牌面大小匹配，这里我们编写了change\_to\_num函数，将牌面和花色映射为对应数字再存入数组（如图）

输入完毕后，我们最后再将数组传给judge函数，让它来判断应该如何出牌，这样初版的main函数就完成了

但是还是有一个问题，文件读取怎么办？最初尝试存入动态数组，但是由于每行的数据不同，结构体数组和链表又略显麻烦，所以最终采用了vector存入（接下来会详细阐述）

2.2 judge函数

Judge函数作为所有不同牌型判断的接口，它的作用就是判断要执行哪个函数，其中第一步就是判断牌型，这里编写了card\_sort（内含card\_compare，冒泡），函数排序（考虑级牌），接着分为两种：对手没出牌和对手出牌了（即m是否＞0），这里如果m>0,考虑到三带二的3和2的不确定性，通过冒泡统一为AABBB形式，m=0,执行m=0的三个函数；m>0,先判断对手出牌，在进行自己对应的出牌判断。

2.3 m=0

注：2.3与2.4的操作，均为先把某种类型的牌存入特定数组，待输出时再映射到hand或temp数组中传出

2.3.0 hand\_count和non\_level\_ base

hand\_count:将每种点数的牌的数量存入base数组，并考虑级牌（冒泡升序）

non\_level\_base:取消级牌设定，用于不考虑级牌的牌型

2.3.1 delete\_card函数

通过deal\_bomb找出所有炸弹，存入后暂时在hand中删除生成hand\_temp;

再利用deal\_streightfive删除所有同花顺

2.3.1.1 deal\_bomb

找位置，找数量，匹配就分别存入，有了base都好做

2.3.1.2 deal\_streight\_five

先找到顺子，再判断是不是同花顺，是就删除（由find\_streight\_five完成）

由于大于四张已经被炸弹删除，每个花色最多不会超过三张

Find\_streightfive负责做所有的排列组合存入后一个一个试是不是同花顺（card\_type函数完成），返回bool值用于确定是否删除

冒泡每个点数删除一张牌

2.3.2 find\_min函数

其他的函数，找出各自最小的点数牌型（可以重复牌，因为只会出其中一种牌型），合适确定何时只用hand\_count,何时加上non\_level\_base,基本逻辑都是从大往小遍历搜索储存，不赘述（但是是核心部分）

2.3.3run\_card函数

互相比较最小点数，最小的存入run数组，将对应的hand里的牌花色改为5，实现假删除的效果，根据不同的case执行out\_card(统一存入temp数组再传递，实现代码复用)

2.4 m>0

通过switch card\_type,不同的case使用不同的函数处理

这些函数都是遍历搜索，冒泡，bool控制，存入数组，数组传递

并且符合以下出牌原则：能出牌尽量出，在不拆炸弹和同花顺情况下只要能管上就可以拆牌

如果没有可出的，进行炸弹和同花顺的出牌判断

没牌可出，pass

不同函数的实现逻辑比较复杂，详见注释

有关函数：run\_card\_1,get\_small\_pair,get\_small\_three\_pair,get\_small\_three,get\_small\_double\_three,get\_small\_three\_two,get\_small\_five,get\_small\_bomb,pass;

2.4 pass和out\_card

Pass函数很朴素，就输出原来牌面再输出pass就行

Out\_card函数内含change\_to\_string函数（用于cout）

已经排好序了直接输出就行

2.5读取文件，main函数升级

普通数组不能满足读取文件的需要，动态数组和链表的工程量又比较大，所以升级后，vector版的新main函数出现了！

首先，为解决旧版main函数输一个数据出一个结果的方式，我们将输出数据都存入vector；

其次，将输入分为act\_input(负责直接输入) 和act\_file(负责输入文件)两个函数（如何实现详见注释）

同时用vector就自然使用了c++的string的函数和操作

这里致谢参考资料：《C++ Primer Plus》以及一系列csdn和stackoverflow的讲述文件读取，string用法的帖子

2.6图形化

参考资料：

<https://blog.csdn.net/m0_73633088/article/details/128371136>

<https://blog.csdn.net/weixin_61777209/article/details/124185685>

<https://docs.easyx.cn/zh-cn/intro>

<https://blog.csdn.net/qq_74455082/article/details/130177860>

<https://haigear.blog.csdn.net/article/details/128783027?spm=1001.2014.3001.5502>

实现效果：贴图，播放音乐，鼠标监视，设置按钮，设置文本

三．实现过程中遇到的问题

3.1出牌逻辑

3,1,1级牌可以压除大小王以外的牌，但是级牌也可以当作它的牌点使用，  
例如图 1 中，级牌为 10，手牌有 10 K Q J 9，那么这副牌可以当作 KQJ109 的顺子打出。  
那比如说这个例子，级牌为10，我能不能出10 A K Q J 或者大王?

这些问题在询问后，出牌逻辑都做了巨大改变

最初的最初版是不能拆牌的，只有牌型正好符合才出牌，好写但是逻辑奇怪

当改成可以拆牌后，一系列bug都来了

3.1.2 拆牌后，最困难的是判断同花顺，删除同花顺，和同花顺和炸弹之间的判断，中间出过×n的数组越界,vector 溢出，死循环，堆栈溢出，传形参错误，指针没复位，没初始化，边界问题没考虑，大小王忘考虑，炸弹寻找顺序有瑕疵导致bug,删多了，无法extern数组进而重写,new不停的开辟新空间（故放弃动态数组）……当能跑起来后，最初的函数并不是分开的，而是都写在judge里的，最后又将其进行拆分成15个左右的函数，而这时数组的作用域，数组是否被其他函数篡改等都需要一点点调试，但是3000＋行的代码量使调试变得十分困难

因此，好的调试能力非常重要，断点，ifdef,转到定义，跳步，全局搜索修改变量名，编译和生成在运行前的重要性……

3.1.3有记录的几个很难调试的小问题

1.有对子的时候没考虑小王对

2.没考虑拆牌！！！直接重写

3.顺子数组没传出来（变量名太像了写错了）

4.三同张打出手牌没更新（hand[][1]==5没写）

5. 级牌在顺子、三连对或二连三的牌型组合中，只能按级牌牌点大小插带。之前没考虑这个

6.run数组没考虑边界，初始化可能会没有，与其对应的k也是

7.level没做字母

8.同花顺没写hand删除

9.find\_streight\_five找到后没退出

10. get\_small\_bomb中删除同花顺语句不对，先找出最小的同花顺,然后在get\_small\_bomb中删除,是为了复用以前的找同花顺函数,但没有在找到后就退出,所以strieght\_five值为空了,所以没有删除. 如果在find\_streight\_five函数里面删除, 会影响m=0的结果.

11.deal\_bomb缺少退出条件

12.find\_bomb找四大天王需要提前

13. get\_small\_three\_two

由于没有事先排除炸弹 三带二时 二张要排除四大天王

14. get\_small\_threetwo果拆的是3张的要排除已经找好的三张

四．改进与反思

1.在写的时候，复用复制了很多代码，这些代码能否再次封装

2.并没有采取特别优质的算法，所以如果数据流切换很大会出问题

3.easyx并没有做动画与随时停止

4.代码的可拓展性需要增强，一旦修改需要修改的地方会很多

5.代码没有可移植性

6.逻辑判断的最优仍需优化

7.bgm切换，声音调整，图片和声音的路径

8.代码逻辑复杂，不易维护，小问题难以寻找

五．使用说明

本rar里包含初版rar和初版main函数，所有的新版代码以及一系列测试数据，以及图像和音乐文件，敬请斧正！