.高级语言程序设计

实验报告

1. 题目

“BAKA9”小游戏系统

1. 开发软件

Visual Studio 2022，EasyX\_2023大暑版

1. 主要流程
2. 整体流程：
3. 启动程序，等待加载界面完成后到达主界面；
4. 通过鼠标点击对应的图标打开各种小游戏；
5. 通过小游戏内的文字提示游玩总共四个小游戏；
6. 玩后可以按照提示返回主界面，选择别的小游戏或直接关闭系统；
7. 等待一段时间后弹出一只萨卡班甲鱼；

实现思路：

打飞机游戏：创建一个自机类，一个敌机类，一个子弹类，一个继承自子弹类的地方子弹类，

红豆的挑战：创建一个自机类，一个武器类，一个敌方单位类，一个背景类

找找方块 ：创建一个用来存储棋盘的二维数组，创建一个储存坐标与颜色的方块类

麦块推箱子：创建一个存储若干地图的四维数组

1. 算法与公式：
2. 加载界面、结束界面与主界面：
3. 加载界面的实现：

不断使用cleardevice()清空屏幕并在屏幕上添加新的文字

循环更替添加loading. loading.. loading...

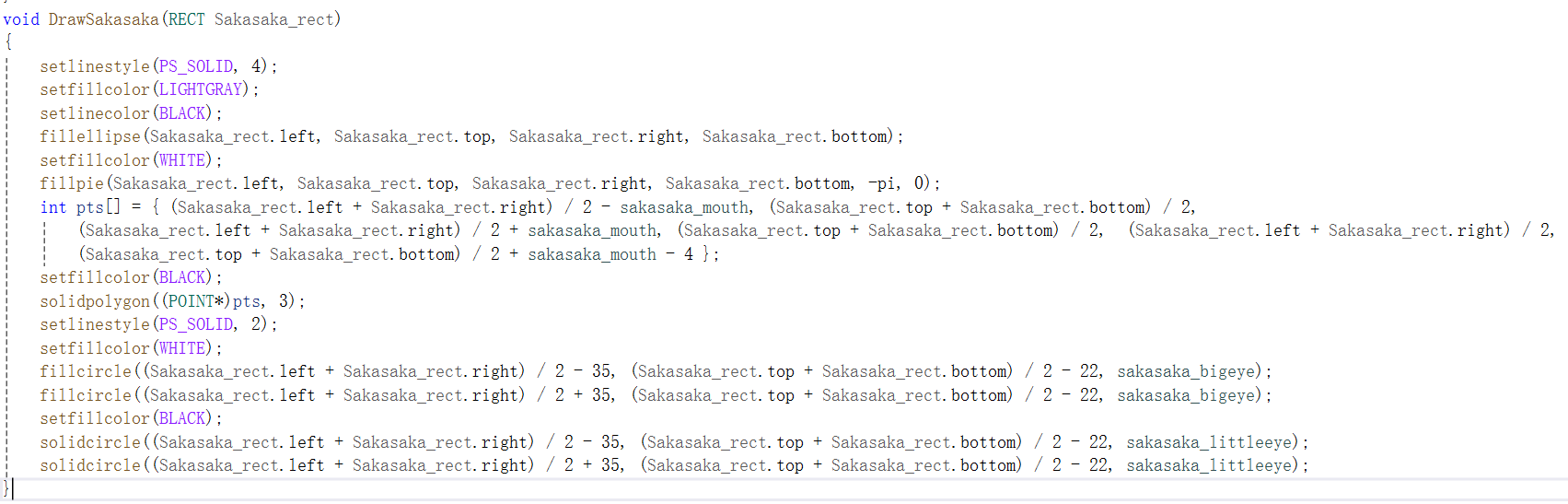
（加载界面是必要的，不然资源太多会卡）

1. 萨卡班甲鱼的绘制：

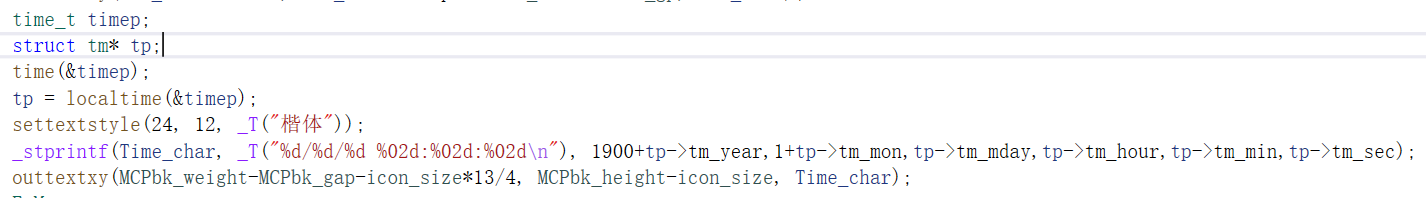
以萨卡班甲鱼外接矩形的左上角为基准点绘制一个内接椭圆，取中线，中间偏下画一个三角形。画两个眼睛涂上颜色就好了。

使用函数： rectange(),ellipse(),line(),polyline(),fillcircle();

代码如下，可以自己试试：



1. 屏幕右下角时间的显示：



Tm结构体头文件自带，\_stprintf（）是用来将int转为LCPTSTR类型的

1. 屏幕点击图标的实现；

创建一个RECT类型的变量，获取鼠标坐标，当鼠标点击时使用PTinRECT（）（头文件自带的函数）判断鼠标所在位置是否在图标RECT范围之内。

1. 打飞机游戏部分：
2. 自机的移动：

创建一个自机的类，变量如下：

IMAGE& img;//图像

RECT rect;//判定点

POINT point;//左上角坐标

POINT draw\_point;//废案

unsigned int HP;//血量

unsigned short HP\_piece = 0;//这一行为废案

创建四个布尔类型变量is\_move\_up,is\_move\_left,is\_move\_down,is\_move\_right，

当按下上下左右键时对应的布尔类型变量由零变为一，松开时由一变为零，当布尔类型变量为一时,自机的左上角坐标改变。

1. 敌机的生成与运动：

采用一个链表存储指针，每个指针指向一个 new出来的敌机变量，每隔一定的时间就会在随机位置增加若干个敌机，利用迭代器遍历所有的自机，使每个飞机前进。

1. 子弹的发射：

创建一个链表存储子弹的指针，每个指针都new了一个子弹

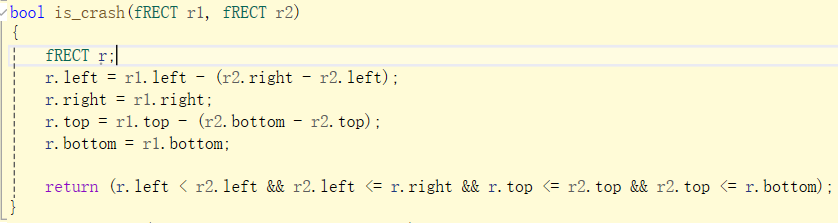
利用迭代器遍历所有飞机，每隔一段时间，在每个飞机的坐标下创建一个子弹，并用迭代器遍历子弹，控制每个子弹的前进。

1. 判定点的移动：

获取飞机实时左上角坐标，并通过计算依据坐标更改代表判定点的rect结构体变量

1. 受击的判定与效果：

遍历每个子弹，通过自己写的函数判断子弹的判定点是否与飞机的判定点重合，若重合，则先delete掉子弹，再减少一点飞机的血量（这里如果直接创建一个子弹的类erase掉会导致迭代器访问错误的地址报错）

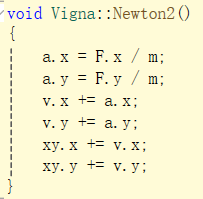
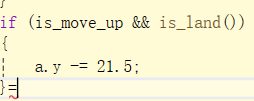


1. 背景效果的实现：

画一个比屏幕长的多背景图片，每过一段时间将图片的坐标往上移，快到底的时候将坐标重新设为0，循环往复即可。

1. “红豆的挑战”游戏部分：
2. 角色的操控与运动：

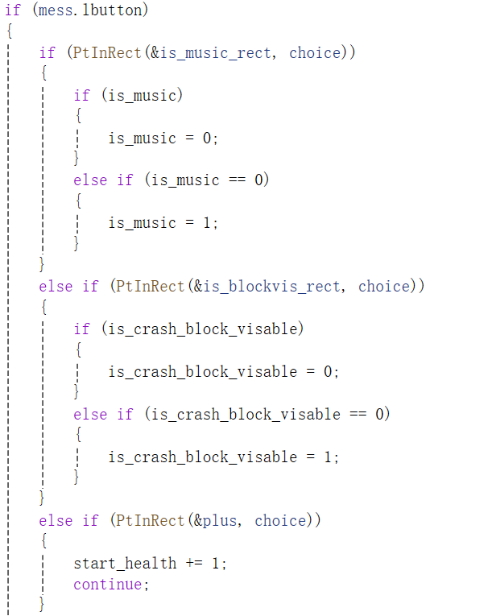
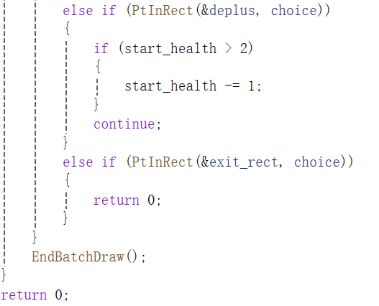
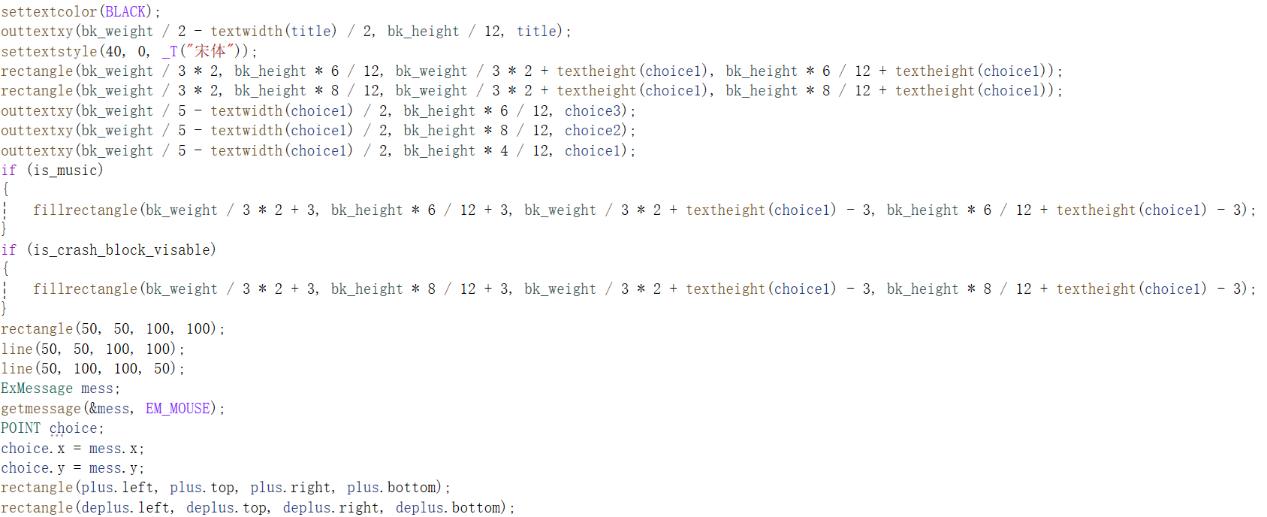
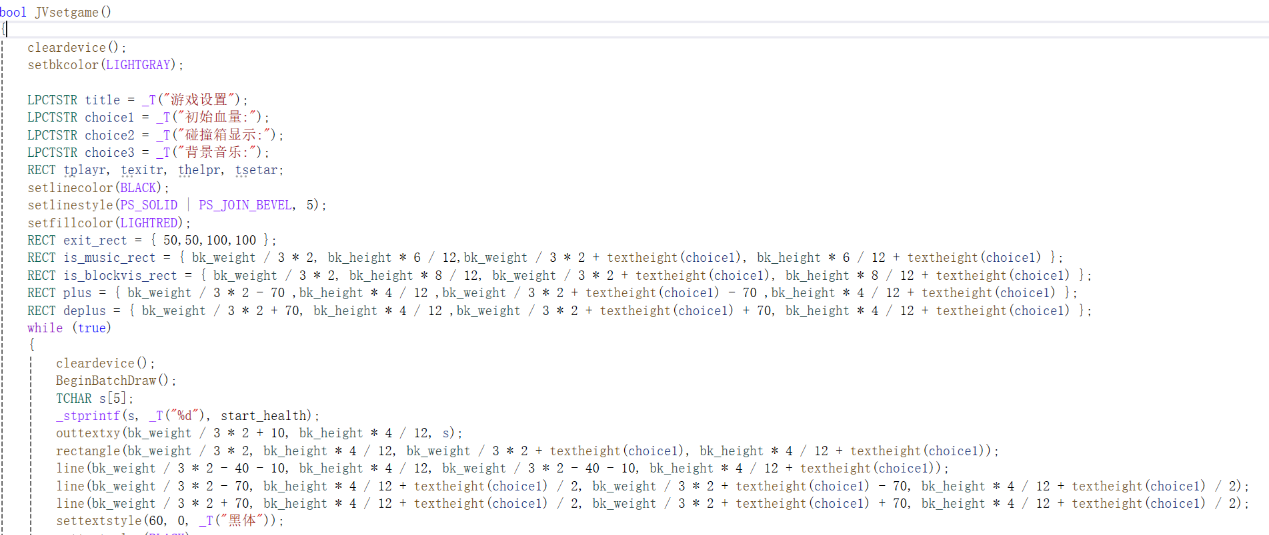
左右控制与上面那个自己的控制同理，甚至布尔类型的变量都是用的相同的。多了一个is\_attack，当按下鼠标左键的时候为1，松开为0。跳跃调用一个专门的函数计算加速度：，当落地时清空所有加速度。



1. 背景的操控：

与上面那个类似，但是改为当角色移动时背景移动，背景移动的始终比角色快，营造一种角色在移动的假象。

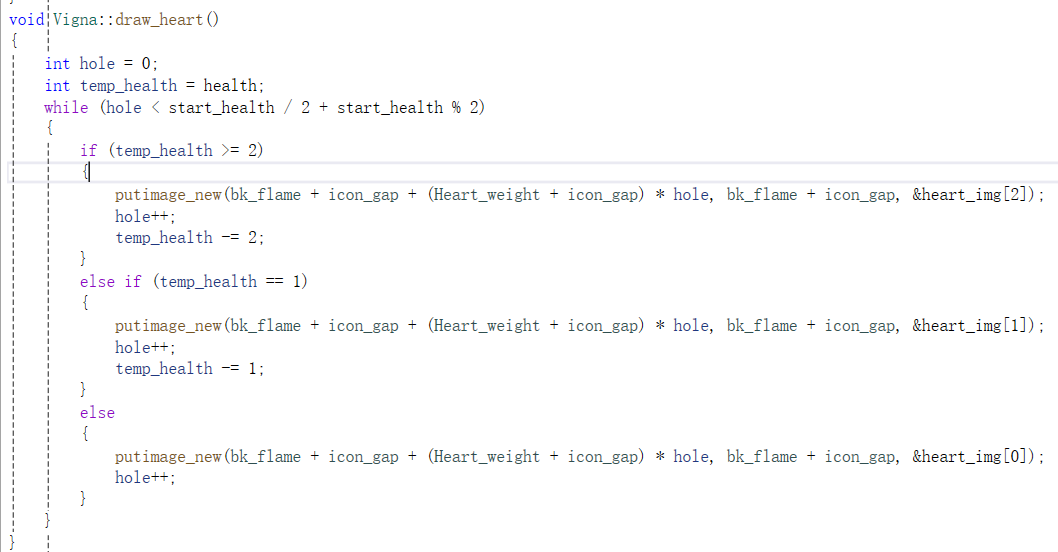
1. 设置界面：



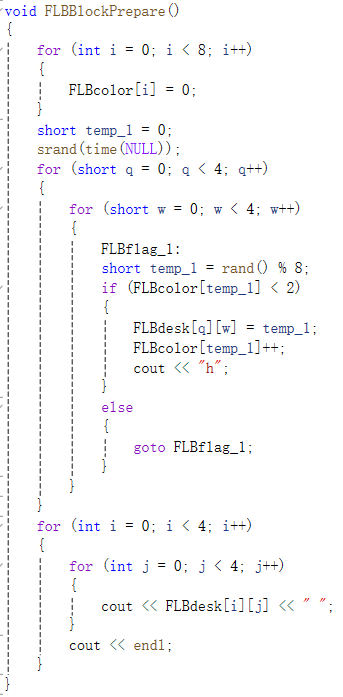
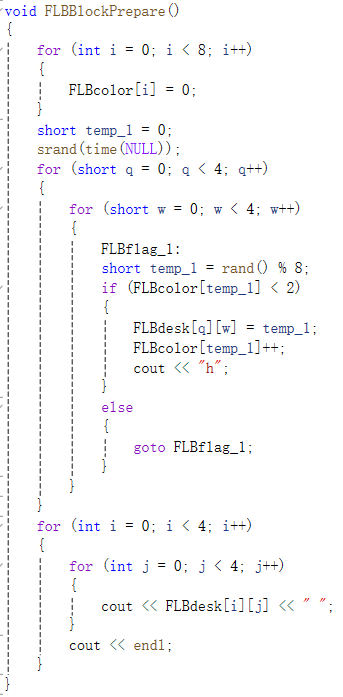
1. 攻击的判定：

与“手机的判定”类似，但是只有在is\_attack==1时执行

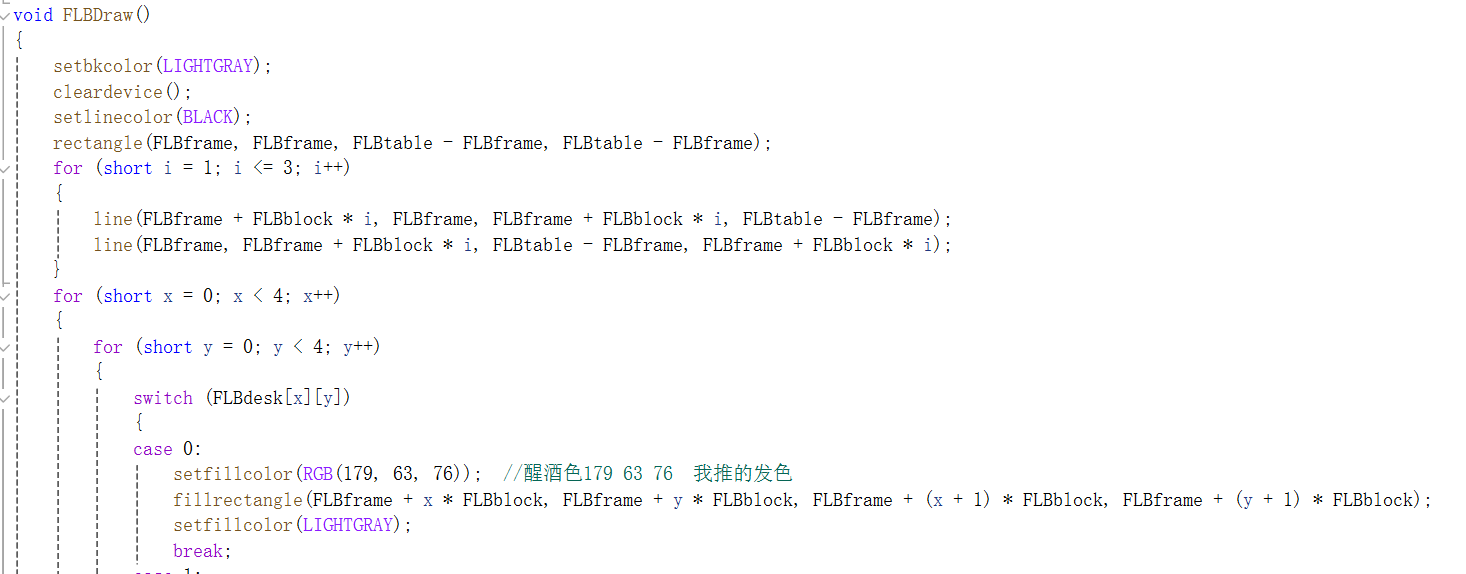
1. 血量的显示：



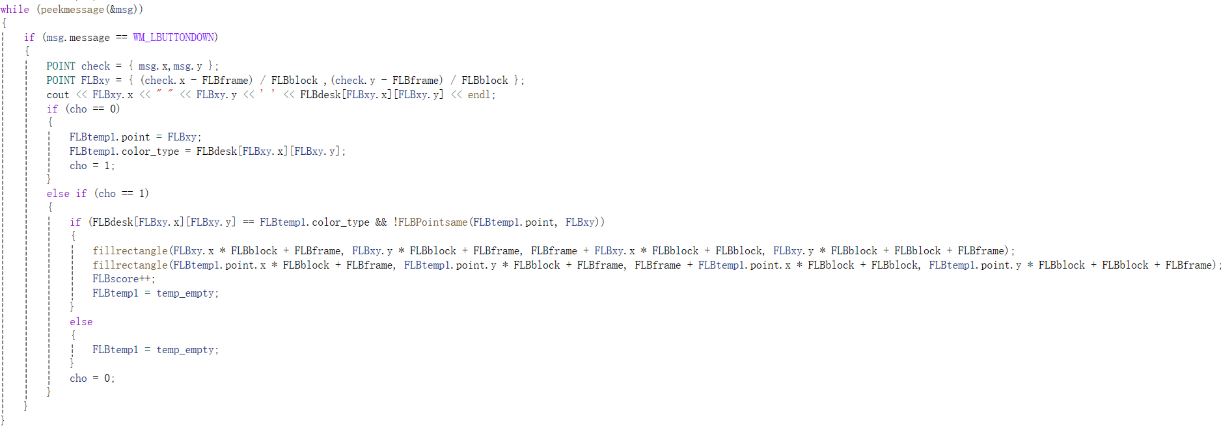
1. “找找方块”游戏部分：
2. 随机棋盘的生成：



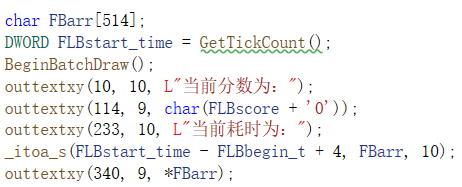
棋盘的绘制来用了switch语句判断画哪一个颜色，下面是部分节选



1. 点击的方块是否相同的判定：



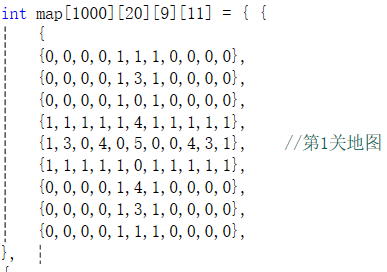
1. 时间分数的显示：



1. 推箱子部分：
2. 地图的存储：

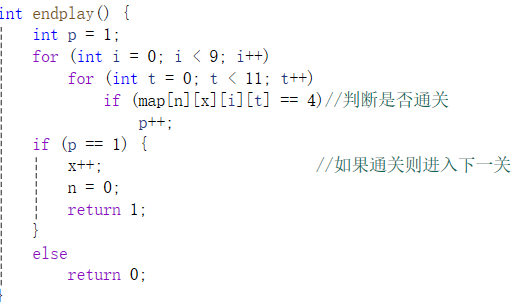
使用四维数组存储，到哪关就访问哪个数组，用不同数字代表不同物品，

这里采用四维数组是为了便于退回之前的位置，类似于时间轴，下面是代码节选



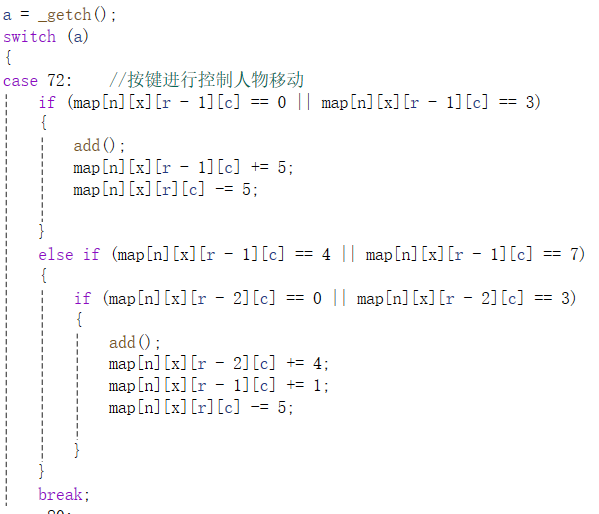
1. 胜利条件的判定：

在这里推到指定位置的箱子和未推到指定位置的箱子用的数字不同，遍历整个地图检索是否还有没有推到指定位置的箱子的编号即可，代码见下方



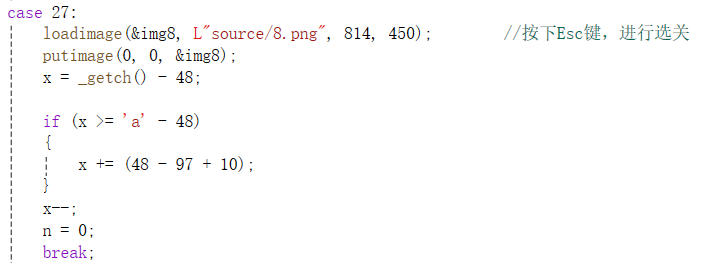
1. 角色的操控；

和前面的差不多，但是换成了\_getch（）函数，只是想试试。



1. 选关的实现；

也是通过\_getch()实现



1. 测试：

详细测试过程请见此网址： <https://www.bilibili.com/video/BV11m421T71S/?share_source=copy_web&vd_source=23b10484ee29823b6f199b5f12b650bd>

代码请见这个网站：https://github.com/VignaChu/BAKA9

1. 收获
2. 多种绘图软件的使用：
3. 构造与析构函数的使用：
4. EazyX的使用：
5. 处理多字节字符的方法
6. Windows.h头文件的使用
7. Stl容器的使用
8. 类的继承使用