**2022W11Q1ch09s6**

题量: 5 满分: 100

作答时间:*05-07 18:30*至*05-07 20:30*

[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20331660&type=2)

*96.5*分

* 批阅人
* \*\*\*
* \*\*\*
* \*\*\*
* 打分
* 97
* 96
* 未批

**一. 简答题（共5题，100分）**

**1. (简答题)**

**在IniData()函数中，所有的栅格的状态都被初始化为ncUNDOWN。但本节DrawGrid()函数在绘制栅格ncUNDOWN的状态，并不是依据栅格的mState值为ncUNDOWN。问题一：请问可能的原因是什么？问题二：既然如此，那在IniData()函数中将所有栅格的状态都初始化为ncUNDOWN的作用是什么？**

*我的答案：*

*0*分

问题一回答：因为我们在进行Mineset()函数进行布雷的行为之后，在舞台上栅格的mstate状态有部分已经依据游戏的难度设置被设置为了ncMINE，如果依据栅格的mState值进行绘制，会将没有找到的地雷显示出来，无疑失去了游戏的意义。且ispress有true和false的状态，在没有按下的情况同一画ncUNDOWN，按下按照栅格的mState的状态进行绘制。  
  
问题二回答：IniData()函数中将所有栅格的状态都初始化为ncUNDOWN的作用是将游戏开始或者是游戏重新开始的时候，对棋盘重新初始化。在一开始的时候，ncUNOWN的状态最多，后面再进行布雷以及点击插旗等相关操作。

*正确答案：*

答：①每个栅格的状态首先分pressed和unpressed，在pressed以后，才是正常显示它的mState值对应的图案。因此DrawGrid()函数在绘制栅格ncUNDOWN的状态的依据，是该栅格是否pressed。

②在IniData()函数中将所有栅格的状态都初始化为ncUNDOWN的作用是，在后面布雷的时候，用来区分该栅格是否已经布雷。当布雷结束后，所有未布雷的栅格都会被重新赋值。不信，你回去再看一遍代码。

当然，布雷方法的构建，并不是唯一的，可以有其他方案的存在。

**\*\*\*：18分**

第二个问题不是很全，没有说后面要确认是否布雷的作用

**\*\*\*：18分**

表述有点怪怪的

**2. (简答题)**

**2、在游戏绘制中，将多种素材以精灵表单的形式整合为一个纹理文件导入到纹理区，再由各个精灵对象在纹理图像的各个局部区域进行纹理提取、绘制。请问此种纹理管理方式有什么优势？**

*我的答案：*

*0*分

1、减少图片的字节  
  
2、减少了从硬盘内多次读取小文件到纹理区的可能，从而大大的提高了纹理加载的性能。当需要加载很多图片时，硬盘就会频繁的读取小文件到缓存区，造成硬盘I/O压力大，这将大大降低游戏的加载速度。  
  
3、解决了游戏开发人员在图片命名上的困扰，只需对一张集合的图片上命名就可以了，不需要对每一个小元素进行命名，从而提高了游戏的开发效率。  
  
4、更换风格方便，只需要在一张或少张图片上修改图片的颜色或样式，整个游戏的风格就可以改变。维护起来更加方便。

*正确答案：*

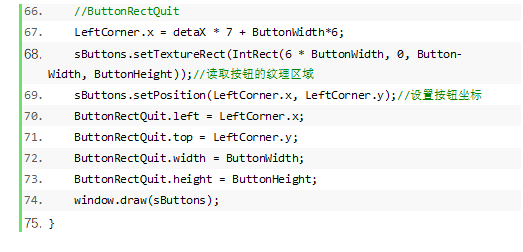
答：SFML在绘制对象的时候，纹理需通过setTexture()函数与精灵对象产生关联，但精灵对象只获取纹理的地址并没做纹理的备份。精灵对象的每次绘制通过window.draw()实现，每次精灵绘制均会访问一次纹理。window.draw()的时间开销有部分即是来自纹理数据的传输。单个大纹理区的访问，纹理数据传输的频次，要低于多个小纹理区的纹理数据传输的频次。因此，单个大纹理区的访问效率要高于多个小纹理区的访问效率，有助于渲染效率的提升。

**\*\*\*：20分**

**\*\*\*：20分**

**3. (简答题)**

**3、下方代码中第67、68、72行代码中共出现4次ButtonWidth变量。请问每次该变量值起到的作用是什么？**

****

*我的答案：*

*0*分

第67行 ButtonWidth 的作用是确定ButtonRectQuit的横坐标，因为前面已经有六个按钮和七个间隔的距离，ButtonRectQuit是第七个按钮。  
  
第68行 第一个 ButtonWidth 的作用是读取按钮的纹理区域， 6\*ButtonWidth的原因是该按钮的纹理区域在6\*ButtonnWidth开始的位置； 第二个 ButtonWidth的作用是读取按钮纹理的宽度，将从6\*ButtonWidth的位置开始读取宽度为ButtonWidth 的按钮纹理。  
  
第72行 ButtonWidth的作用是 确定ButtonRectQuit的响应区域，在鼠标响应ButtonRectQuit的功能是退出游戏，设置按钮响应的宽度。

*正确答案：*

答：第67行的ButtonWidth变量是用于辅助定位按钮的左顶点x坐标值，比如此处表明按钮的左边存在7个空隙、6个按钮宽度值，暗示该按钮的左边存在有水平排列的其他6个按钮。

第68行的2处ButtonWidth变量值，第1处表示该按钮纹理在大纹理表单上是第7个纹理图案，左边需要偏移6个按钮纹理的宽度值，左顶点坐标取（6\*ButtonWidth，0）。第2处表示该按钮纹理在大纹理表单上提取的局部纹理的宽度为ButtonWidth。

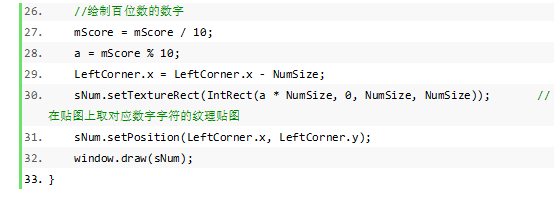
第72行的ButtonWidth变量值，表示按钮在游戏窗口上呈现的宽度尺寸为ButtonWidth个像素，用于鼠标点击的交互区设置。

**\*\*\*：20分**

**\*\*\*：20分**

**4. (简答题)**

**4、下方代码中第27、28、29行代码的作用，请分别进行描述。第30行代码中，变量a数值的对应数字贴图能够准确获取的前提条件是什么？**

****

*我的答案：*

*0*分

27：把mScore的十位数的数字去除，使百位数的数字变成mScore的个位数的数字  
  
28：对mScore取余10，把mSocre的个位数（即我们需要的百位数）提取出来  
  
29：确定数字纹理的位置，在十位数原本使用的左上角点上减去字符的大小，计算出百位的数字纹理的纹理的位置  
  
30：前提条件是精灵图已正确加载读取，精灵图内的数字是按顺序准确储存的

*正确答案：*

答：第27行代码，让mScore变量的数值以十进制的形式进行右移1位的操作；

第28行代码，提取mScore变量数值当前个位上的数值；

第29行代码，左顶点坐标左移一个数字纹理宽度的位移，用于绘制高位数字的纹理贴图。

第30行代码起作用的前提是：①所有数字纹理贴图的尺寸一致，此处为NumSizexNumSize

②纹理贴图上的数字图案排放规则是按照0123456789的顺序依次水平排列

**\*\*\*：19分**

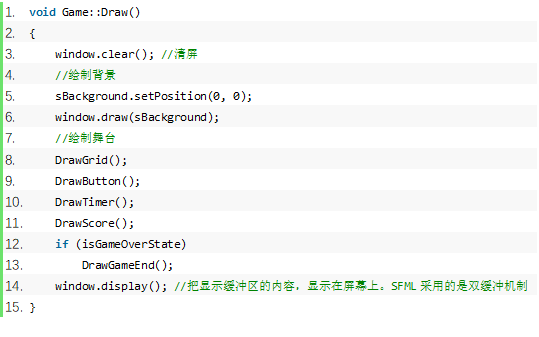
问题二，字符大小的点没有答到

**\*\*\*：18分**

前提条件不是很完整

**5. (简答题)**

**5、游戏结束画面的绘制，可通过如下方式进行实现。如第12行代码所示，当变量isGameOverState为true则对结束画面进行绘制。但如果现在游戏分3个场景关卡，关卡变量为StageLVL，请用伪代码重新实现Draw()，使得程序能对指定StageLVL的内容进行绘制。**

****

*我的答案：*

*0*分

void Game::Draw()  
  
{  
  
    window.clear();  
  
    //绘制背景  
  
    sBackground.setPosition(0, 0);  
  
    window.draw(sBackground);  
  
    switch(StageLVL)  
  
    {  
  
        case 1:  
  
            //绘制场景1  
  
            DrawGridLVL1();  
  
            ///DrawButton();  
  
            break;  
  
        case 2:  
  
            //绘制场景2  
  
            DrawGridLVL2();  
  
            //DrawButton();  
  
            break;  
  
        case 3:  
  
            //绘制场景3  
  
            DrawGridLVL3();  
  
            //DrawButton();  
  
            break;  
  
    }  
  
    //绘制舞台  
  
    //DrawGrid();  
  
    DrawButton();  
  
    DrawTimer();  
  
    DrawScore();  
  
    if (isGameOverState == ncLOSE || isGameOverState == ncWIN)  
  
    {  
  
        DrawGameEnd();  
  
    }  
  
    window.draw(mousePoint1);  
  
    window.display();  
  
}

*正确答案：*



**2022W12Q1ch09s07**

题量: 5 满分: 100

作答时间:*05-09 13:30*至*05-09 15:30*

[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20369862&type=2)

*97.5*分

* 批阅人
* \*\*\*
* \*\*\*
* 打分
* 95
* 100

**一. 简答题（共5题，100分）**

**1. (简答题)**

**游戏过程中，如果游戏窗口发生尺寸变化，可能导致获取到的鼠标点击坐标出现偏差。请列举两种禁止游戏窗口大小缩放的方式。**

*我的答案：*

*0*分

方法1：  
  
// create own viewsf::View view = window.getDefaultView();while (window.isOpen()){  
    sf::Event event;  
    while (window.pollEvent(event))  
    {  
        if (event.type == sf::Event::Closed)  
            window.close();  
  
        if (event.type == sf::Event::Resized) {  
            // resize my view  
            view.setSize({  
                    static\_cast<float>(event.size.width),  
                    static\_cast<float>(event.size.height)  
            });  
            window.setView(view);  
            // and align shape  
        }  
    }  
  
限制窗口的大小  
  
方法2：  
  
使用sf::Style::Close  
  
窗口就无法调整大小了

*正确答案：*

①通过sf::Event::Resized事件响应，将游戏窗口修改为原来尺寸；

②修改游戏窗口属性，禁用Resize设定

**\*\*\*：20分**

**\*\*\*：20分**

**2. (简答题)**

**游戏过程中，如果游戏窗口发生尺寸变化，可能导致获取到的鼠标点击坐标出现偏差。请设计一种解决方案使得游戏窗口发生缩放的时候，鼠标点击能够得到正常响应。**

*我的答案：*

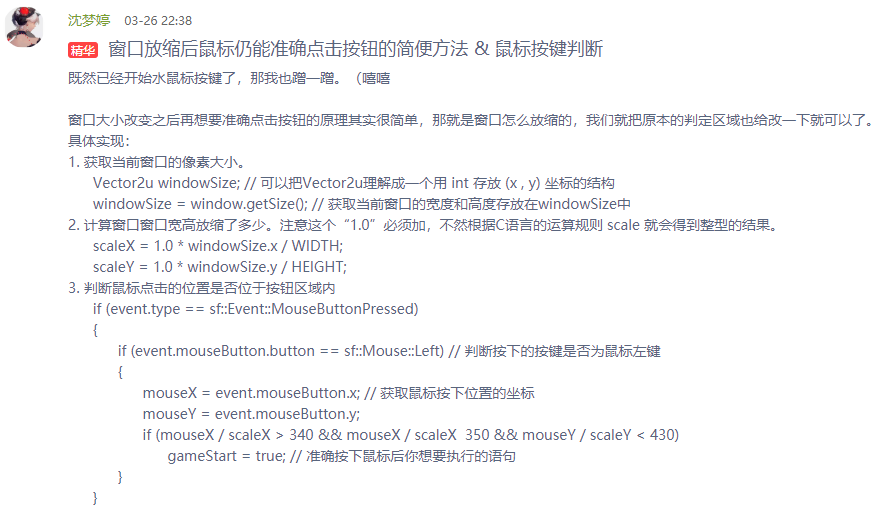
*0*分

在所有的涉及到游戏窗口缩放的例子，考虑游戏窗口以及响应区域的大小之间的协同。这里需要考虑的是将两个不一定一致的游戏区域进行协同需要用到的函数时mapPixelToCoords()通过这类函数能够响应区域实时对游戏的窗口进行成比例的变化类似于计算机图形学当中的世界坐标系与局部坐标系直接的协同。

|  |
| --- |
|  |
| mousePoint = (Vector2i)window.mapPixelToCoords(Mouse::getPosition(window));     double x = mousePoint.x;      double y = mousePoint.y;  用以下代码可以实现无视缩放，100%判定 |

|  |
| --- |
|  |
|  |

*正确答案：*

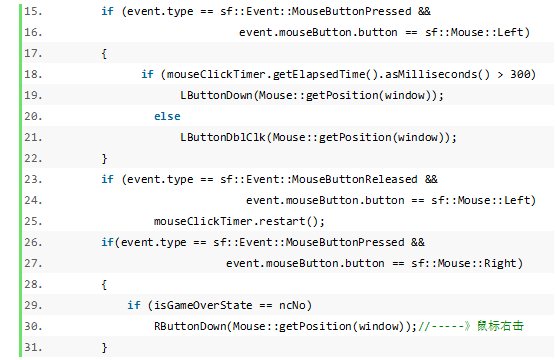


**\*\*\*：20分**

**\*\*\*：20分**

**3. (简答题)**

**下方代码，当快速在两处不同位置进行单击操作，会被误判为双击操作。请给出一种解决方案，以规避该种误判。**

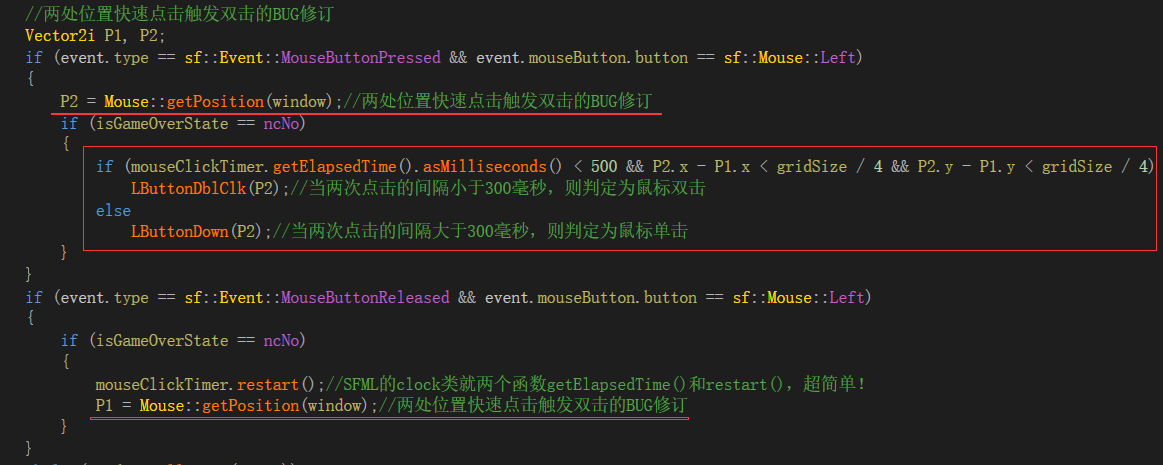
****

*我的答案：*

*0*分

首先，我们要明确如何区分快速在两处不同位置进行单击操作和双击的区别Windows根据两个条件来做这个区分：（1）双击的时间间隔两次单击会产生四个鼠标点击消息，如果第三个消息（第二次按下）和第二个消息（第一次弹起）间隔短于指定值，则把第三个消息为鼠标双击消息；第四个消息为鼠标按键弹起。这个指定的时间间隔，在Windows XP SP2上缺省是0.5秒，其他操作系统可能相同。在系统中，这个间隔值是可以通过API函数读取和重新设置的。例如通过::GetDoubleClickTime调用可以得到这个值。通过::SetDoubleClickTime进行设置。（2）两次鼠标击点的空间距离在第一次点击时，Windows以击点为中心，检测一个矩形区域，如果第二次点击不落在这个区域内，那就不把第三个消息算作WM\_LBUTTONDBLCLK消息。这个矩形区域的缺省大小，在Windows XP SP2上缺省是4pt×4pt。可以以SM\_CXDOUBLECLK或SM\_CYDOUBLECLK为参数调用::GetSystemMetrics得到。这个值也是可以设置的。设置的方法是通过SPI\_SETDOUBLECLKWIDTH或SPI\_SETDOUBLECLKHEIGHT为第一个参数来调用::SystemParametersInfo。设置的结果对系统中其他的应用程序也起作用

*正确答案：*



解决方案的核心是对于双击的条件进行更加严格的限定。

**\*\*\*：20分**

**\*\*\*：20分**

**4. (简答题)**

**在《扫雷》游戏中，如果第一下鼠标点击为右键单击，请问游戏能否正常进行？如果能请说明理由。如果不能请尝试给出修正方案。**

*我的答案：*

*0*分

按照书本的案例，第一下鼠标点击为右键单击，游戏不能正常进行。  
  
给出修正的方案在右键单击的函数中，如果游戏没有开始的情况下并没有将布雷的函数包含进去。所以在进行第一次鼠标点击为右键单击的时候，由于雷的函数没有布下，导致游戏信息缺失游戏进程中断。解决方案在进行第一次鼠标单击的情况下，进行布雷函数的包含

*正确答案：*

答：如果第一下鼠标点击为右键单击，则出现的情况是先插旗后布雷。因此在后续布雷的时候要注意到在布雷之前已经显示为旗的格子的状态。两个原则：①布雷时，不能改变已经插旗格子的当前显示状态；②插旗是玩家的一种判断，布雷的时候，要对当前格子的真实状态进行正常赋值，即旗的显示，不能影响当前格子真实状态产生，不然游戏会出BUG。

因此，需要三个状态变量，分别记录当前格的：①真实值；②显示值；③点击状态。②和③显示值和点击状态可以进行合并。

**\*\*\*：15分**

如果右键单击就布雷，在鼠标第一次左键单击时，可能会踩雷，有违微软规则。

**\*\*\*：20分**

**5. (简答题)**

**经典扫雷游戏中，关于鼠标的双击方式有两种定义。一是鼠标左键快速点击两次，另一种是鼠标左右键同时按下。基于前面的内容，请思考下：鼠标左右键同时按下的双击，应该如何实现？**

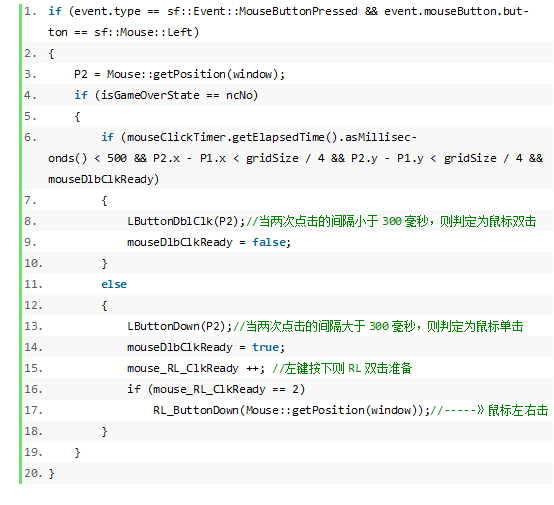
*我的答案：*

*0*分

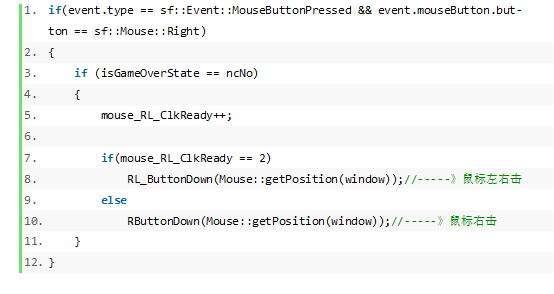
// 左右键同时按下的双击函数为 RL\_ButtonDown()  
  
// 针对鼠标左右键双击的优先级响应高于单击优先级响应的问题，如若双键已经按下，则优先执行左右双键的响应，不执行单键的响应。同时，对鼠标左键双击的计时步骤做了调整，将左键双击的响应放在 MouseButtonReleased 条件判定中。  
  
void Game::Input()  
  
{  
  
    Event event;  
  
    while (window.pollEvent(event))  
  
    {  
  
        /////////////////////////////////////////////////////////////////////////////////////////////////  
  
        //关闭退出  
  
        if (event.type == sf::Event::Closed)  
  
        {  
  
            window.close();  
  
            gameQuit = true;  
  
        }  
  
        if (event.type == Event::EventType::KeyReleased && event.key.code == Keyboard::Escape)  
  
        {  
  
            window.close();  
  
            gameQuit = true;  
  
        }  
  
        //关闭退出  
  
        //左键点击响应  
  
        if (event.type == sf::Event::MouseButtonPressed && event.mouseButton.button == sf::Mouse::Left)  
  
        {  
  
            P2 = Mouse::getPosition(window);  
  
            if (isGameOverState == ncNo)  
  
            {  
  
                if (mouseClickTimer.getElapsedTime().asMilliseconds() < 500 && P2.x - P1.x < gridSize / 4 && P2.y - P1.y < gridSize / 4 && mouseDlbClkReady)  
  
                {  
  
                    LButtonDblClk(P2);  
  
                    mouseDlbClkReady = false;  
  
                }  
  
                else  
  
                {  
  
                    LButtonDown(P2);  
  
                    mouseDlbClkReady = true;  
  
                    mouse\_RL\_ClkReady++;  
  
                    if (mouse\_RL\_ClkReady == 2)  
  
                    {  
  
                        RL\_ButtonDown(Mouse::getPosition(window));  
  
                    }  
  
                }  
  
            }  
  
        }  
  
        //左键点击响应  
  
        ////////////////////////////////////////////////////////////////////////////////////////////////  
  
        //右键点击响应  
  
        if (event.type == sf::Event::MouseButtonPressed && event.mouseButton.button == sf::Mouse::Right)  
  
        {  
  
            if (isGameOverState == ncNo)  
  
            {  
  
                mouse\_RL\_ClkReady++;  
  
  
  
                if (mouse\_RL\_ClkReady == 2)  
  
                {  
  
                    RL\_ButtonDown(Mouse::getPosition(window));  
  
                }  
  
                else  
  
                {  
  
                    RButtonDown(Mouse::getPosition(window));  
  
                }  
  
            }  
  
        }  
  
        //左键释放响应  
  
        if (event.type == sf::Event::MouseButtonReleased && event.mouseButton.button == sf::Mouse::Left)  
  
        {  
  
            mouse\_RL\_ClkReady = 0; //状态清除  
  
            if (isGameOverState == ncNo)  
  
            {  
  
                mouseClickTimer.restart();  
  
                mouseAction = LButtonDownFunc;  
  
                mousePoint = Mouse::getPosition(window);  
  
                P1 = Mouse::getPosition(window);  
  
                if (isGameBegin == false) //设置游戏难度系数  
  
                {  
  
                    if (ButtonRectEasy.contains(event.mouseButton.x, event.mouseButton.y))  
  
                    {  
  
                        gamelvl = 1;  
  
                    }  
  
                    if (ButtonRectNormal.contains(event.mouseButton.x, event.mouseButton.y))  
  
                    {  
  
                        gamelvl = 2;  
  
                    }  
  
                    if (ButtonRectHard.contains(event.mouseButton.x, event.mouseButton.y))  
  
                    {  
  
                        gamelvl = 3;  
  
                    }  
  
                    Initial();  
  
                }  
  
            }  
  
            if (ButtonRectBG.contains(event.mouseButton.x, event.mouseButton.y)) //更换背景  
  
            {  
  
                imgBGNo++;  
  
                if (imgBGNo > 7)  
  
                {  
  
                    imgBGNo = 1;  
  
                }  
  
                LoadMediaData();  
  
            }  
  
            if (ButtonRectSkin.contains(event.mouseButton.x, event.mouseButton.y)) //更换皮肤  
  
            {  
  
                imgSkinNo++;  
  
                if (imgSkinNo > 6)  
  
                {  
  
                    imgSkinNo = 1;  
  
                }  
  
                LoadMediaData();  
  
            }  
  
            if (ButtonRectRestart.contains(event.mouseButton.x, event.mouseButton.y)) //重来  
  
            {  
  
                Initial();  
  
            }  
  
            if (ButtonRectQuit.contains(event.mouseButton.x, event.mouseButton.y)) //离开  
  
            {  
  
                window.close();  
  
                gameQuit = true;  
  
            }  
  
        }  
  
        if (event.type == sf::Event::MouseButtonReleased && event.mouseButton.button == sf::Mouse::Right)  
  
        {  
  
            mouse\_RL\_ClkReady = 0;  
  
        }  
  
    }  
  
}

*正确答案：*

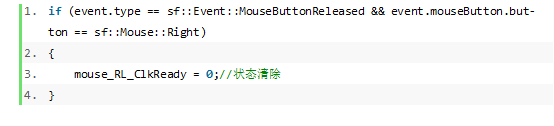
双键点击的状态管理，由于不确定左右两键哪个键会被先按下，所以它的管理比左键单键的双击管理稍微复杂一点。解决方案是将mouse\_RL\_ClkReady变量设置为int型，左右键的点击均会使得该变量+1，当该变量值为2则表示左右双键均以按下，可调用RL\_ButtonDown()函数进行双键响应操作。左键按下的RL双击准备及响应调用的示例如下所示



右键按下的RL双击准备及响应调用的示例如下所示。



通过前面两段代码，在Input函数中，不管左键先按，还是右键先按，只要按键累积次数达到2次（即mouse\_RL\_ClkReady=2），则调用RL\_ButtonDown()函数。当左右双键的任意键松开，则mouse\_RL\_ClkReady = 0，进行归零。下方代码为右键松开的代码示例。



**2022W12Q2ch09s08**

题量: 4 满分: 100

作答时间:*05-16 13:30*至*05-16 15:30*

[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20553341&type=2)

*98*分

* 批阅人
* \*\*\*
* \*\*\*
* 打分
* 100
* 96

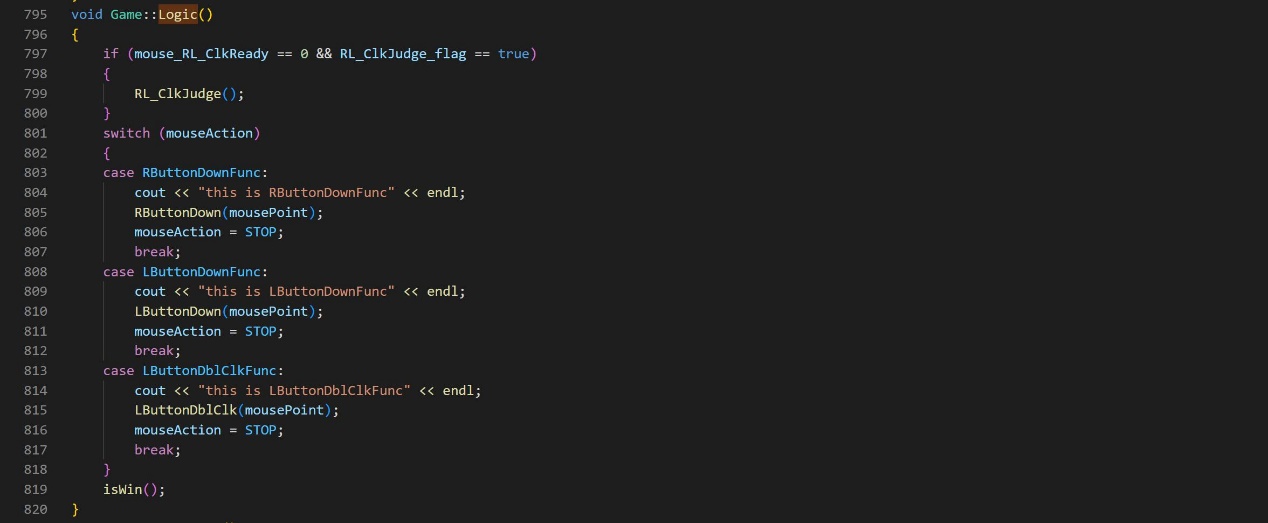
**一. 简答题（共4题，100分）**

**1. (简答题)**

**1、如果要将void RButtonDown(Vector2i mPoint); void LButtonDown(Vector2i mPoint); void LButtonDblClk(Vector2i mPoint)；3个函数放置在Logic()函数中进行响应，程序的框架结构应该进行如何改动？**

*我的答案：*

*0*分

通过switch的方式在Logic()函数中加入三个函数的响应，并且在Logic()中每次点击之后都进行了一次判断胜利的函数。

*正确答案：*

答：①首先创建变量mouseAction在Logic模块中管理当前要激活哪个鼠标响应函数，同时记录鼠标点位置；

②创建mouseFunction枚举类型

typedef enum MOUSEFUNCTION

{

RButtonDownFunc;

LButtonDownFunc;

LButtonDblClkFunc

};

③将原来Input模块中，调用鼠标响应函数的地方，全部改用mouseAction先记录需要响应的鼠标函数；例如用调用RButtonDown()的地方，用mouseAction=RButtonDownFunc;进行替代。

④在Logic中，构建鼠标响应的代码段，或进行函数封装。主体内容用switch(mouseAction)进行分类

switch(mouseAction)

{

case RButtonDownFunc:

RButtonDown(Vector2i mPoint);

break;

……

}

**\*\*\*：25分**

**\*\*\*：25分**

**2. (简答题)**

**2、为提升玩家的游戏体验，《扫雷》游戏采用了哪些提升游戏节奏的功能设置？**

*我的答案：*

*0*分

1、NullClick()函数，将周围是已经标记正确或者是没有雷的情况下，翻开空白的格，并继续遍历直到结束  
  
2、在剩下一或者多个未揭开的格子，在当前插的旗子都是对的，游戏会进行判断如果正确，会直接游戏胜利  
  
3、左键双击或者左右键一起响应，会帮助将周围部分格翻开。

*正确答案：*

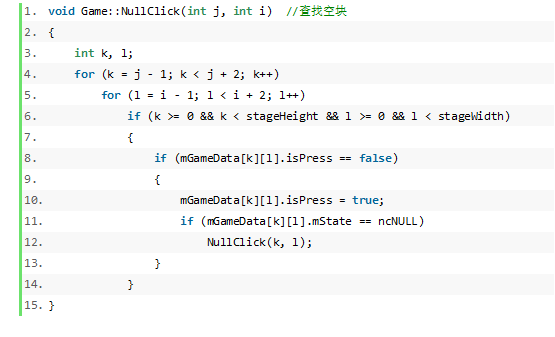
答：鼠标双击响应，快速揭开周边8领域中未揭开的格子；击空NullClick响应，通过递归调用，快速揭开一片非雷的格子；左右键同时按下，允许玩家进行安全的双击测试。

**\*\*\*：25分**

**\*\*\*：25分**

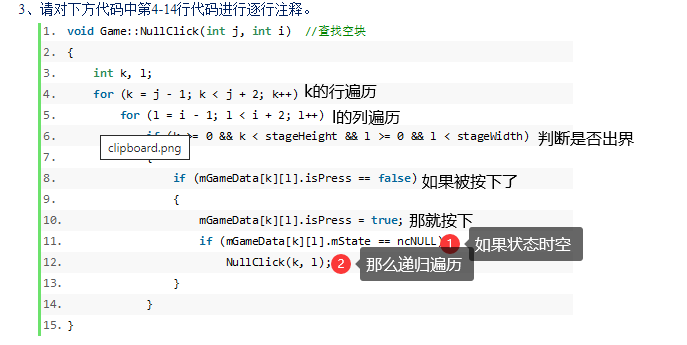
**3. (简答题)**

**3、请对下方代码中第4-14行代码进行逐行注释。**

****

*我的答案：*

*0*分



*正确答案：*

答：

第4-5行：对鼠标点击位置(j,i)的周边8领域进行遍历

第6行：判段(j,i)周边8领域是否都在舞台区域，只对舞台区域的格子进行响应

第8行：位置(k,l)的格子是否未揭开

第10行：将位置(k,l)的格子标记为揭开状态

第11行：位置(k,l)格子的状态是否为“空”

第12行：以位置(k,l)格子为中心，调用NullClick()函数。

**\*\*\*：25分**

**\*\*\*：21分**

**4. (简答题)**

**4、isWin()函数第21-22行代码的if判定条件有否存在漏洞，它能否判断剩下未点击的块是不是都是雷？请给出你的理由。**

****

*我的答案：*

*0*分

可以。逻辑有两条

第一条：当你插的旗子数量等于雷的数量的时候，遍历所有的方格，判断旗子对应的是否为雷，然后正确计数++，在第二个if里面判断正确数量等于雷，判断结束

第二条：你的旗子较少，雷较多，但是剩下的旗子数量就等于没有掀开的格子数量，此时已经掀开了所有非雷，也应当判断结束

*正确答案：*

答：可以的。在游戏未结束的情况下，只有雷全被找出，才会被判定游戏胜利。

采用反证法。设：

总雷数mMineNum

旗子数mFlagCalc

未揭开格子unPress

假设前面有旗子误标，也就是标为“旗”的格子实际不是“雷”，那该“雷”必然在剩下未揭开的格子里。那剩下未揭开的格子数必然大于总雷数减“旗”子数。

未揭开格子unPress>总雷数mMineNum - 旗子数mFlagCalc

统计的c值 = 未揭开格子unPress + 旗子数mFlagCalc > 旗子数mFlagCalc

由于C值必然大于总雷数，第26行代码条件无法满足，游戏无法结束。

因此，只有在Flag全部判定正确的情况下，才会可能使得第26行代码条件被满足。当然，这个时候需要非雷的格子全部被揭开。

综上所述，该判定能够判断剩下未点击的块是不是都是雷

**2022W13Q1ch09s09**

题量: 4 满分: 100

作答时间:*05-16 13:30*至*05-16 15:30*

[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20553208&type=2)

*97.5*分

* 批阅人
* \*\*\*
* \*\*\*
* \*\*\*
* 打分
* 95
* 100
* 未批

**一. 简答题（共4题，100分）**

**1. (简答题)**

**1、微软《扫雷》游戏的左右键双击操作是否允许点击在游戏舞台中未揭开的格子上？如果可以，请说明会发生的结果。若不可以，请说明理由。**

*我的答案：*

*0*分

可以的，在未揭开的格子上左右键什么也不会发生，他会显示一个覆盖层，但是不会产生任何效果或作用。

*正确答案：*

答：真的可以的。左右键同时按下后，点击格子的周围8领域中，未揭开的格子处于“待揭开状态”，当任意键松开之后，状态还原。

**\*\*\*：25分**

**\*\*\*：25分**

**2. (简答题)**

**2、阅读下方代码，并回答以下问题：**

**①请解释下第15行代码的判断条件是什么意思。**

**②微软《扫雷》游戏的左右键双击操作的规则是：如果左右双击点错了格子会将待揭开的格子状态还原，如果点对了则直接将对应格子揭开。第17-21行代码直接将相应格子状态还原，请问为什么？**

**③请解释下第24行代码的判断条件是什么意思。**

**④第24-25行代码直接调用鼠标左键的双击响应函数，请问会不会出现触雷的情况？**

****

*我的答案：*

*0*分

①15行代码的判定条件是是否这个格子是否已经被标了旗帜或者是否是雷  
  
②17～22行将备用状态的方格还原，即取消该方格的备用状态  
  
③第24行代码判断8邻域内的旗帜的数量和雷的数量是否正确  
  
④不会的，第24行代码判定了周围8邻域内的旗数判定正确，然后再执行鼠标的双击响应

*正确答案：*

答：①格子被标记为旗的同时，它本身是“雷”，同时满足条件的时候，才表明雷被找出来了。

②错了，会还原；对了则直接打开，那打开的时候，前面是否还原，其实无所谓。与其在后面再进行一次判断，不如这里顺手直接还原。让代码更精简一点。至少能省下2行for循环的代码。

此处，是为了引导大家思考一个问题？代码量优先？还是运算量优先？编译器里面其实是有对应参数设置，只是课上我们没说。

③在扫雷游戏案例格子值的枚举设置中数字1-8对应编号是3-10。也就是说枚举的值与实际真实数值存在偏差2.

④不会触雷。第24行代码的意思是，如果8领域中所有的雷都有找出来，那旗数flagNum+2 ，会与格子的真实值相等，此时调用双击函数快速打开未揭开的格子。雷没找对的情况下，是不会触发左键双击函数。

**\*\*\*：25分**

**\*\*\*：25分**

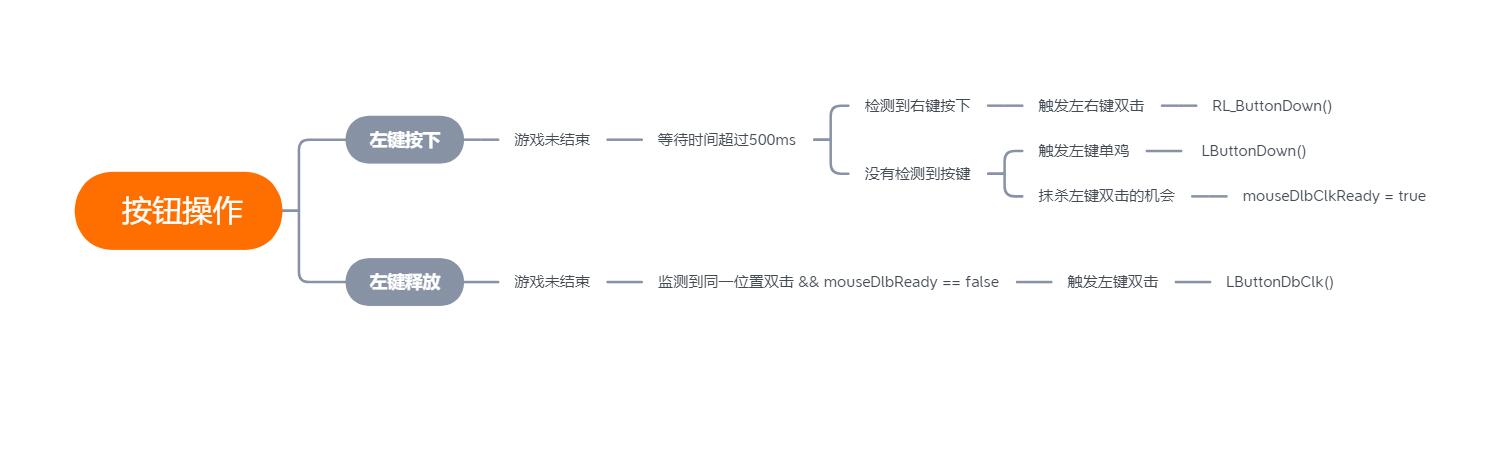
**3. (简答题)**

**3、鼠标左右键双击操作是由左键、右键的单击所构成。但微软《扫雷》游戏中鼠标左右键双击操作的优先级高于单击操作的优先级。即当判定为左右双键操作的时候，则不执行单键的响应。请结合下方代码，对左右键的按键设定进行梳理，以脑图的形式绘制左右键双击和各自单击的关系。（提示，要把按键的“按下”和“释放”两种状态纳入做统一考虑）**

****

*我的答案：*

*0*分



*正确答案：*

答案：略，

左键的按下和释放基本就参照题中代码的逻辑执行。右键的响应请放在按键释放时候执行，左右双击的响应发生在“按下”状态，如果已发生，其实已经不存在右键点击的前提。

另外，再好好思考一下，左右键双击的执行逻辑，尤其其中的逻辑锁设定

**\*\*\*：25分**

**\*\*\*：25分**

**4. (简答题)**

**4、案例中undownOpen()函数的功能设定是什么？它与unCover()函数的功能有没重叠？**

*我的答案：*

*0*分

没有重叠。  
  
uncover()函数是将剩余未被找出的雷的位置标注出来，将所有未被点击且是雷的方格全部标记出来显示  
  
undownopen()函数的作用是在游戏结束时，将所有未揭开的格子揭开

*正确答案：*

答案：undownOpen()函数：游戏胜利后，将所有未揭开格子进行揭开

unCover()函数：游戏失败后，将所有未揭开的雷显示出来

**2022W13Q2ch10s01**

题量: 8 满分: 100

作答时间:*05-18 18:30*至*05-18 20:30*

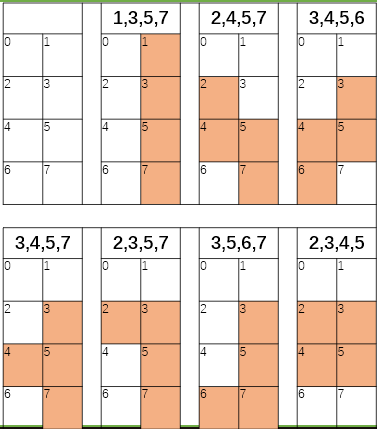
[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20603502&type=2)

*95.5*分

* 批阅人
* \*\*\*
* \*\*\*
* 打分
* 97
* 94

**一. 简答题（共8题，100分）**

**1. (简答题)1、俄罗斯方块的数字描绘方式并不是唯一的。那如下图所示的表示中，其中图形T的编码（3,4,5,7）能否改为（1,2,3,5）？可能会有什么影响？**

****

*我的答案：*

*0*分

可以的。俄罗斯方块中的图形块的数字描绘方式并不是唯一的，通常通过一个2\*4的矩阵区域来进行表示。目前给出编码方式仅仅是一个参考。同样的“L”的编码为（2，4，6，7），“T”的编码为（2，4，5，6），等等。由此可见，数字描绘图形块的编码并不是唯一的。  
  
但是，旋转的时候可能会涉及到一个旋转中心点的问题。不过这些问题也可以通过算法去解决。目前的图图案的旋转中心为5。

*正确答案：*

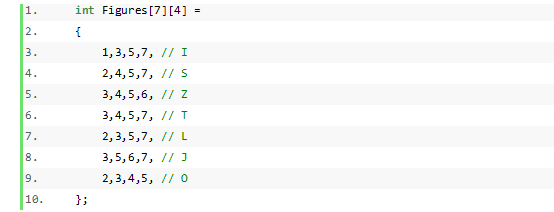
答案：图形的编码方式可以改。但是建议各图形的中心点能够尽量保持一致。不然在图形做旋转操作的时候，需要增加其他的规则。

**\*\*\*：12.5分**

**\*\*\*：12.5分**

**2. (简答题)**

**2、用二维数组对俄罗斯方块的7种图形进行存储，如下所示。在绘制图形时候，需要知道图形的显示坐标。那要对图形T进行绘制，需如何提取图形T的4个方格的坐标？**

****

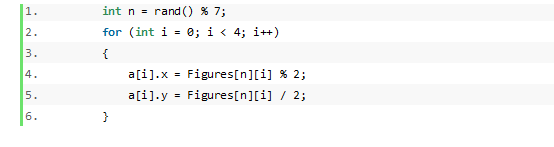
*我的答案：*

*0*分

在绘制图形的时候，需要知道图形的显示坐标。对图标T进行绘制，获取2\*4矩阵区域上的坐标，可以采用以下代码：  
  
int n=rand()%7;  
  
for(int i=0;i<4;i++)  
  
{  
  
    a[i].x=Figures[n][i]%2;  
  
    a[i].y=Figures[n][i]/2;  
  
}

*正确答案：*

答案：通过下方代码获得俄罗斯方块相应的图形。基于图形的二维数组设置，当n=3时，数组a保存图形T的4个标准方格的坐标



**\*\*\*：12.5分**

**\*\*\*：12.5分**

**3. (简答题)3、俄罗斯方块有7种基本图形，常见颜色7种。请问在准备游戏素材时候，需要准备7\*7一共49种的方块图案么？请给出你的解决思路。**

*我的答案：*

*0*分

不需要。  
  
提供以下三种解决思路：  
  
拼图法  
  
准备7个不同颜色的小方块，在使用时按照需要拼成7种基本图形  
  
上色法  
  
准备7种基本图形，但颜色都是灰色的，在使用时按照需要着色成7种颜色  
  
上色拼图法  
  
准备1个灰色方块，在使用时，按照需要组合成7种基本图形并进行着色成7种颜色

*正确答案：*

答案：不需要。为节省游戏导入素材的内存存储空间，通常是设置7种颜色的标准方格，然后根据图形的编码及其在舞台上的坐标，由标准方格进行俄罗斯各方块的绘制。即用标准方格，根据需要组拼各方块图形。

**\*\*\*：12.5分**

**\*\*\*：12.5分**

**4. (简答题)4、俄罗斯方块有7种基本图形，每个图形均有4个标准方块构成，但标准方块的坐标编码均不相同。请问如何控制它们在舞台上移动运动？请给出俄罗斯方块图形在游戏中的移动思路，最好以伪代码的形式表达。**

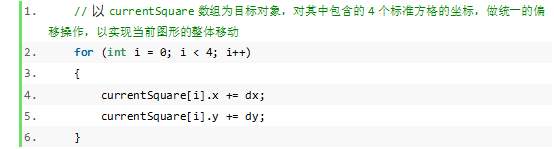
*我的答案：*

*0*分

图像块的水平移动逻辑代码如下：  
  
void Tetris::Logic()  
  
{  
  
    //水平移动  
  
    for(int i = 0;i<4;i++)  
  
    {  
  
        tempSquare[i]=currentSquare[i];  
  
        currentSquare[i].x+=dx;  
  
    }  
  
    if(!hitTest()) //如果撞上了  
  
        for(int i=0;i<4;i++)  
  
            currentSquare[i]=tempSquare[i]];//到左右边界，不能移出边界  
  
}  
  
其中，currentSquare记录当前活动图形块的4个组成小方块的坐标信息;tempSquare为临时的数组变量，用于保存当前活动图形的位置信息;hitTest()函数为碰撞检测函数。hitTest()函数的具体代码如下：  
  
bool hitTest()  
  
{  
  
    for(int i=0;i<4;i++)  
  
        if(currentSquare[i].x<0||currentSquare[i].x>=STAGE\_WIDTH||currentSquare[i].y >= STAGE\_HEIGHT)  
  
            return false;  
  
        else if(Field[currentSquare[i].y][currentSquare[i].x])  
  
            return false;  
  
    return true;  
  
}

*正确答案：*

答案：如下图，以currentSquare数组为目标对象，对其中包含的4个标准方格的坐标，做统一的偏移操作，以实现当前图形的整体移动



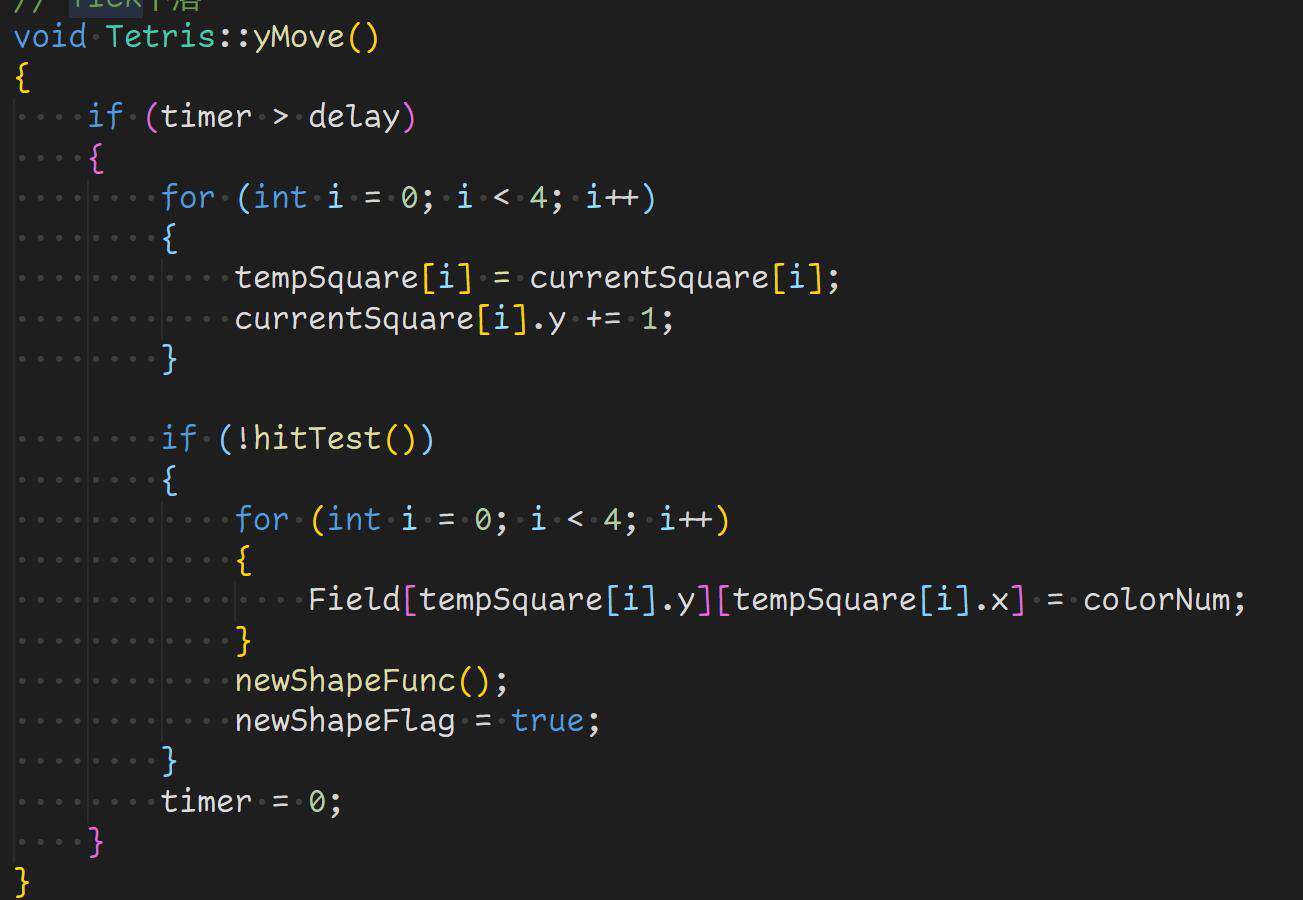
**\*\*\*：12.5分**

**\*\*\*：12.5分**

**5. (简答题)5、俄罗斯方块有7种基本图形，每个图形均有4个标准方块构成，但标准方块的坐标编码均不相同。当它们在舞台上运动时候，比如竖直掉落，7种图形需要有7种运动轨迹计算方式么？请给出7种图形在游戏中做竖直掉落的代码设计。**

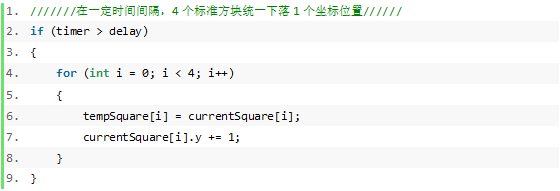
*我的答案：*

*0*分



*正确答案：*

答案：每个图形中的方块均有自己的坐标参数，在游戏逻辑中，以标准方块为单位进行处理，并不区分具体图形。对于同一图形中的4个方块，只要确保它们具有统一的坐标偏移，在游戏舞台上这4个标准方块，则能构成准确的俄罗斯方块图形。方块竖直掉落的代码如下所示：

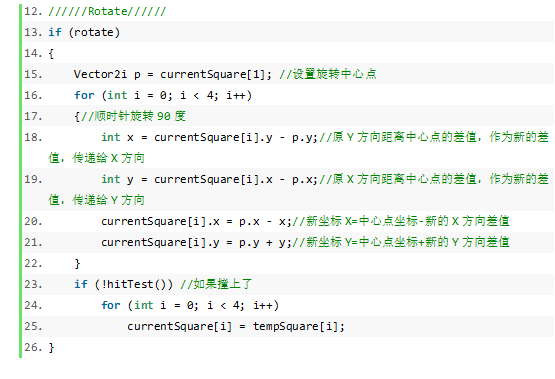


**\*\*\*：12.5分**

**\*\*\*：12.5分**

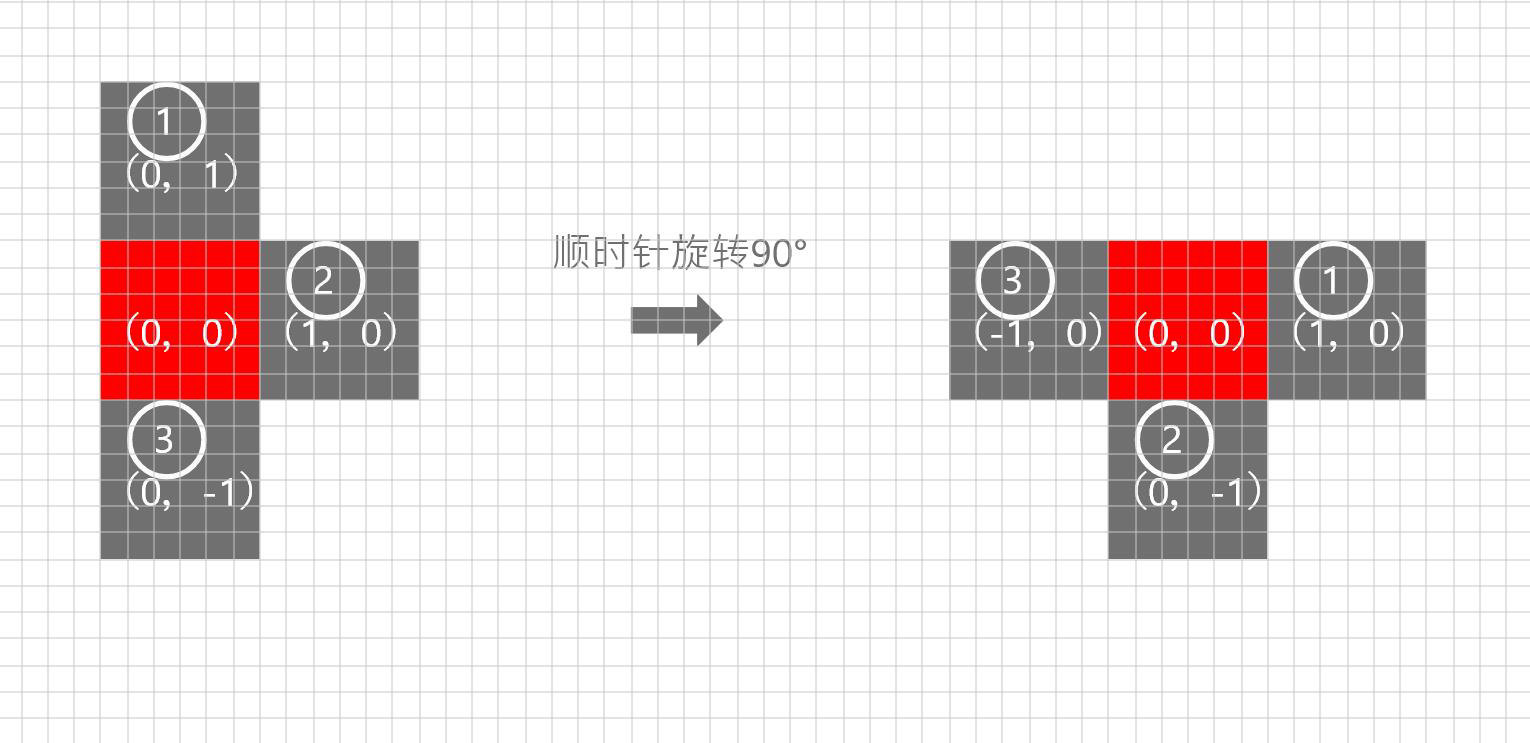
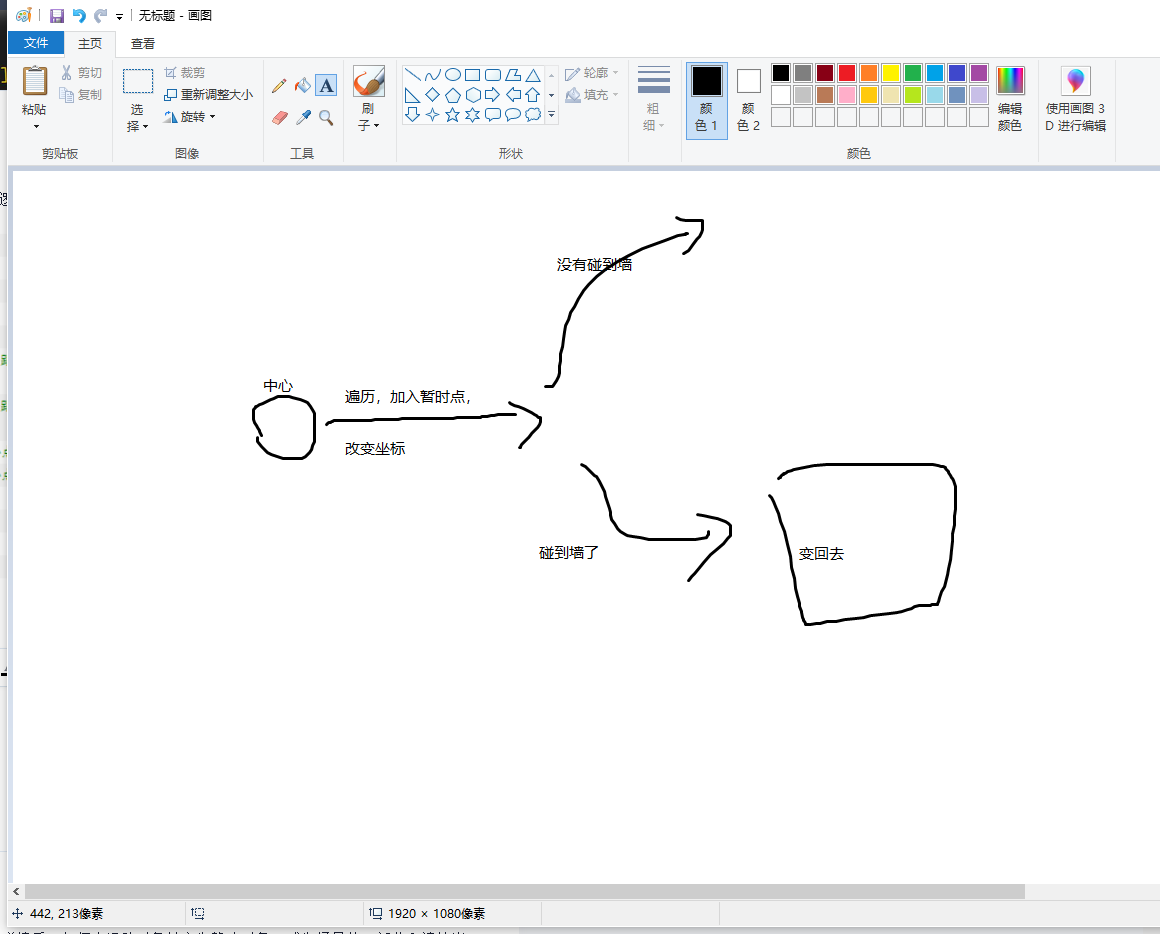
**6. (简答题)**

**6、请绘制一个示意图，图解下方的俄罗斯方块旋转代码的算法逻辑?**

****

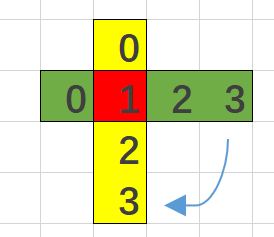
*我的答案：*

*0*分



*正确答案：*

因为每次都是顺时针旋转，所以将x方向相对于旋转中心1的偏移量转换为y方向相对于旋转中心1的偏移量；y方向同理



**\*\*\*：10分**

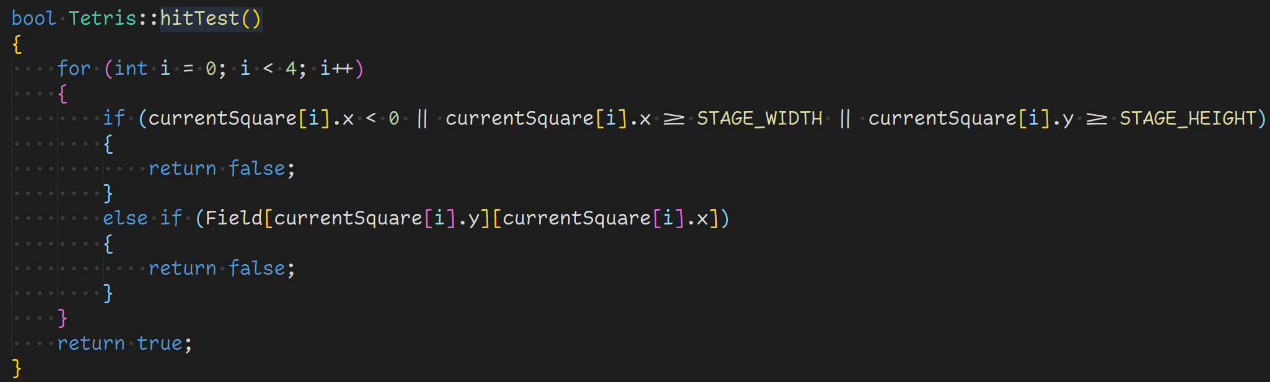
**\*\*\*：11分**

**7. (简答题)7、俄罗斯方块游戏中，运动的方块图形与场景发生碰撞后，如何由运动对象转变为静止对象，成为场景的一部分？请给出方案设计和伪代码示意。**

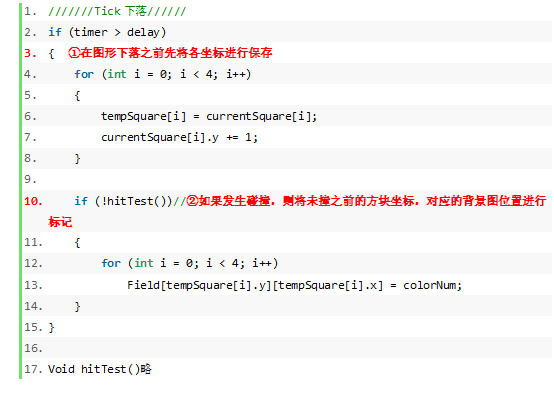
*我的答案：*

*0*分

方案设计：进行碰撞检测，如果运动的图形触底后，则转变为背景并且生成新图形块儿。



*正确答案：*



**\*\*\*：12.5分**

**\*\*\*：12.5分**

**8. (简答题)**

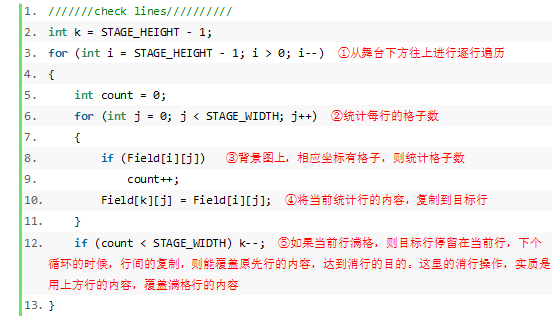
**8、俄罗斯方块游戏中，当出现消行的时候，如何让消除行的上方的方块降下来？请给出相应代码，并用文字说明其中的算法逻辑。**

*我的答案：*

*0*分

对游戏界面进行自左向右、自下而上的遍历。将位于上一层的方块儿置为背景色，并将该方块儿正下方的方块置为该方块的颜色。

*正确答案：*



**2022W14Q1ch10S02**

题量: 6 满分: 100

作答时间:*05-23 13:30*至*05-23 15:30*

[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20708396&type=2)

*92.2*分

* 批阅人
* \*\*\*
* \*\*\*
* 打分
* 89.4
* 95

**一. 简答题（共6题，100分）**

**1. (简答题)1、游戏平台类和游戏类各自实现的功能是什么？**

*我的答案：*

*0*分

游戏平台类：作为游戏Tetris运行的平台，它需要负责游戏资源的加载、管理，游戏初始化、输入、逻辑、绘制的调度工作。

游戏类Tetris：是游戏内容实现的主体，它服从平台对它的调度和管理。在Tetris头文件中增加一个玩家的角色变量，用于玩家角色的标记。

*正确答案：*

答：游戏平台类负责游戏资源的加载、管理，游戏初始化、输入、逻辑、绘制的调度工作。

游戏类是游戏内容实现的主体，它服从平台对它的调度和管理。

**\*\*\*：16分**

**\*\*\*：16.6分**

**2. (简答题)2、如何让游戏类的实例对象知道自己在游戏平台上是哪个对象，以便正确接收平台传递的输入消息，并作出正确的响应。**

*我的答案：*

*0*分

在游戏类中定义一个ID号，每个实例对象构建的时候为他们赋予独有的一个ID号用于判断；在消息传递的过程中，实例对象可以通过这个属性来判断消息是否是传递给他的

*正确答案：*

答：在游戏类中增加一个玩家的角色变量，当实例化游戏类对象的时候，同时对角色变量进行赋值。这样实例对象就能知道自己在游戏平台上的标识。

**\*\*\*：16.6分**

**\*\*\*：16.6分**

**3. (简答题)3、游戏Tetris类的实例对象，如何知道平台转过来的某一个按键输入是发给自己的？**

*我的答案：*

*0*分

会传入一个ID

if (role == rolePLAYER1)  
mCornPoint = { P1\_STAGE\_CORNER\_X，P1\_STAGE\_CORNER\_Y}

 if (role mm rolePLAYER2)  
mCornPoint = { P2\_STAGE\_CORNER\_X,P2\_STAGE\_CORNER\_Y }

*正确答案：*

答，在Tetris::Input(sf::Event event)中，首先判定实例对象的角色。不同的角色，只响应自己所被设定的输入消息。以此避免信息的混淆。

**\*\*\*：15分**

输入消息

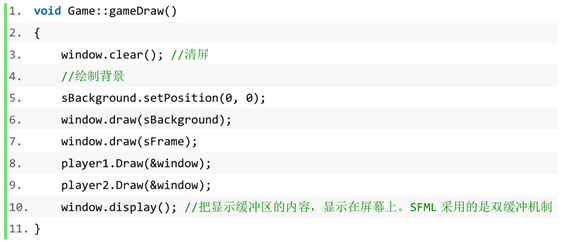
**\*\*\*：16.6分**

**4. (简答题)4、Game类绘制背景，Tetris实例对象绘制各自的游戏画面。如何确保它们的内容是绘制在同一个window窗口上。**

*我的答案：*

*0*分

Game类作为调度平台，要负责游戏画面的绘制。下方代码中，第3-7行代码，实现游戏背景的绘制。游戏主体对象的绘制，则交由实例对象的Draw()进行完成，如第8-9行代码所示。为让实例对象能明确知道在哪个window进行绘制，在第8-9行代码中，将window的指针地址作为形参传送给具体实例对象。这里需要注意的是，在gameDraw函数中，当所有内容绘制完毕后，需要调用display()函数进行显示操作。如第10行代码所示



Tetris::Draw(sf::RenderWindow\* window)函数

需要注意的是，窗口显示函数display()的调用，在一次游戏循环中只调用一次。因此统一在gameDraw函数中调用。Tetris对象要服从Game平台对它的调度和管理，不要在Tetris::Draw(sf::RenderWindow\* window)函数中再次调用window类的display()函数。通过形参获得指定的游戏绘制窗口后，则只需进行常规绘制就行。

*正确答案：*

答：将Game类中的window的指针地址作为形参传送给具体实例对象，实例对象再在指定的window上进行绘制。

**\*\*\*：16.6分**

**\*\*\*：16.6分**

**5. (简答题)**

**5、在Input函数中调用window.setKeyRepeatEnabled(false);的作用是什么？该函数需要每个游戏循环都执行一次么？**

*我的答案：*

*0*分

window. setKeyRepeatEnabled(false);//按键按下只响应一次

*正确答案：*

答：让按键按下时候只响应一次，避免多玩家之间的按键冲突。不需要，其实在窗口创建之后，执行一次就可以。

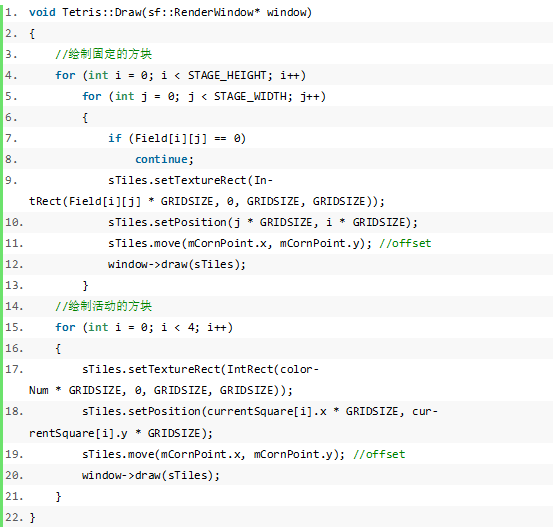
**\*\*\*：8.3分**

**\*\*\*：13.6分**

第二个问题没回答

**6. (简答题)**

**6、下方Draw()函数绘制时候，先绘制固定方块，后绘制活动方块的顺序能不能前后互换？为什么？**

****

*我的答案：*

*0*分

目前是可以的。因为目前不涉及方块的消除和快速下落。如果到了后期，就无法互换，因为活动的方块是建立在固定的方块的之后的才进行的

*正确答案：*

答：当碰撞检测被严格执行的时候，可以互换绘制的顺序。但通常不这么建议。通常的绘制顺序是①UI界面②舞台背景③静止的对象④活动对象。根本的原因是希望能避免错误的遮挡。原理同之前的图层遮挡。

**2022W14Q2ch10S03**

题量: 4 满分: 100

作答时间:*05-25 18:30*至*05-25 20:30*

[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20762584&type=2)

*95*分

* 批阅人
* \*\*\*
* \*\*\*
* \*\*\*
* 打分
* 100
* 100
* 85

**一. 单选题（共1题，25分）**

**1. (单选题)**

**1、双人版的Tetris游戏，每个玩家都应该有自己的Next区显示下个方块图形。但本节中怎么只设置一个Next显示区域左顶点坐标变量？最可能的原因是什么？**

* A. A.讲义上漏定义了一个顶点坐标变量；
* B. B.两玩家共用一个Next区，一个顶点坐标变量够用了；
* C. C.两个玩家是Tetris类的两个对象，Tetris类中定义一个顶点坐标变量，不妨碍它的实例对象拥有自己的顶点坐标；
* D. D.图形的尺寸是事先定义的，在玩家1的Next图形左顶点坐标的基础上做适当偏移，就能得到玩家1的Next图形左顶点坐标，一个顶点坐标变量够用了。

*我的答案:* C *正确答案:* C

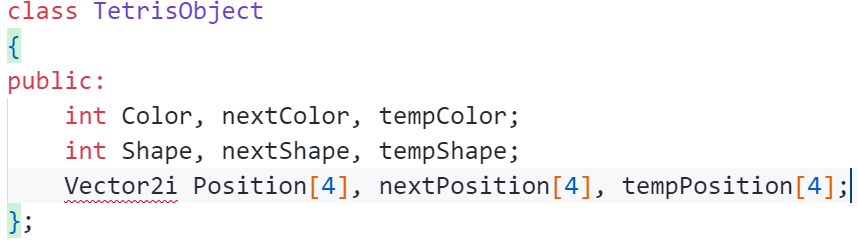
*25*分

**二. 简答题（共3题，75分）**

**2. (简答题)2、俄罗斯方块的方块对象通常具有一些属性变量，例如颜色Color、图形Shape、4个方块坐标SquarePosition。请尝试构建一个类TetrisObject，将它们封装起来。**

*我的答案：*

*0*分



*正确答案：*

 Class TetrisObject

{

Public:

    int Color;

    Int shape;

    Vector2i SquarePosition[4];

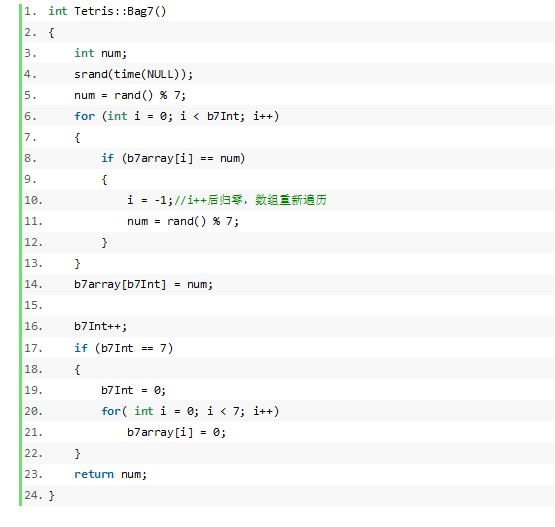
}

**\*\*\*：25分**

**\*\*\*：25分**

**\*\*\*：20分**

**3. (简答题)Bag7()函数通常会先对b7array[]数组进行初始化。下方代码是对b7array[]数组赋初值为0。请问，当第5行代码，随机到0号图形时，会不会由于第8行的条件判断，导致0号图像被重置为下一个随机图形？理由是什么？如果会被重置，有没什么代码改进的建议。**

****

*我的答案：*

*0*分

不会的。  
因为在第 6 ~ 13 行中，是对此时随机到的方块儿前面的方块儿进行遍历，检测该方块儿与之前的是否重复，所以当随机到0号图形时，只要前面没有出现0号图形就不会重置。符号是＜，所以循环进不去。

*正确答案：*

不会。因为其中第6行for循环的表达式2，已做了数组比对范围限制。当前的num值，只会与之前的取值进行比对。数组的0号元素，没有前值，所以不会进入循环体。

如若当心，可以更改b7array[]数组的初始值，比如初始化为比6大的整数。

**\*\*\*：25分**

**\*\*\*：25分**

**\*\*\*：20分**

**4. (简答题)请以静态分配的方式实现Bag7算法。请给出b7array[]数组的赋值方案，以及相应的使用方案？**

*我的答案：*

*0*分

生成链表0，1，2，3，4，5，6

int i=rand()%(链表的长度)

b7arrary[7-链表长度]=i；

--链表长度；

以上循环7次

*正确答案：*

Bag7()

{

      ①按序对b7array[]数组赋初值，b7array[i]=i；

      ②通过随机交换的原则，对b7array[]数组中元素进行两两交换；

}

③设全局变量b7Int=0；并调用Bag7()完成b7array[]数组初始化

④使用代码示例：

nextShapeNum = b7array[b7Int];

b7Int++;

if(b7Int>6)

{

   b7Int = 0;

   Bag7();

}

**2022w14Q3Ch10S04**

题量: 3 满分: 100

作答时间:*05-25 18:30*至*05-25 20:30*

[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20762644&type=2)

*95*分

* 批阅人
* \*\*\*
* \*\*\*
* \*\*\*
* 打分
* 100
* 95
* 90

**一. 简答题（共3题，100分）**

**1. (简答题)1、在Tetris游戏一开始，Hold区是空的。此时应该如何避免Hold区出现图形绘制情况。请列举2种以上方案。**

*我的答案：*

*0*分

1. 设置一个flag，当空时，不绘制
2. 默认初始化一个图形，当作奖励

*正确答案：*

参考答案：①类似讲义的方案，不画；②Hold区图形的属性做相应设置，比如把4方块的坐标设置在舞台窗口之外，图形的alpha值改为0等等。

**\*\*\*：33.3分**

**\*\*\*：31.3分**

**\*\*\*：30分**

**2. (简答题)2、Tetris游戏中，进行Hold区方块的交换后，交换前后图形在舞台上的坐标，应该如何传递？请在本讲义方案的基础上，给出不同的方案。**

*我的答案：*

*0*分

可以在传递位置时，获取选转的角度，和中心坐标点，然后以中心点为交换点进行互换，这样就不会产生问题，当发现非法交换时就拒绝交换

*正确答案：*

参考答案：反正后面有碰撞检测，那就直接传递前后方块的重心坐标。

**\*\*\*：33.3分**

**\*\*\*：30.3分**

**\*\*\*：30分**

**3. (简答题)**

**3、多人游戏之间，如想交换游戏物品或道具，通常采用怎样的数值传递方式？请列举两种**

*我的答案：*

*0*分

1. 设置全局性的变量或对象
2. 采用static 的静态变量

*正确答案：*

参考答案：用于物品交换的静态成员变量、用于物品交换的全局变量、在平台类中设置用于物品交换的成员变量。总之，要有一个可被访问，生命周期足够长的，用于物品交换的变量

**2022w15Q1Ch10S05**

题量: 3 满分: 100

作答时间:*05-30 13:30*至*05-30 15:30*

[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20859984&type=2)

*100*分

* 批阅人
* \*\*\*
* \*\*\*
* \*\*\*
* 打分
* 100
* 100
* 未批

**一. 单选题（共2题，66.6分）**

**1. (单选题)**

**1、在2\*4的矩阵中，我们标记第1、3、5、7编号的方格构成的图形为Tetris中的图形I。为便于图形做踢墙旋转时候，对旋转中心点进行变量。请问图形I的编码值取哪个比较合适？**

* A. （1,3,5,7）
* B. （7,1,3,5）
* C. （3,1,5,7）
* D. （5,3,1,7）

*我的答案:* D *正确答案:* D

*33.3*分

**2. (单选题)图形S在左边靠墙时候（如下图所示），踢墙旋转的可能结果会是哪个？**

****

****

* A. 图中A选项
* B. 图中B选项
* C. 图中C选项
* D. 图中D选项

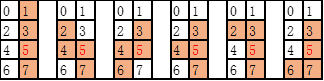
*我的答案:* A *正确答案:* A

*33.3*分

**二. 简答题（共1题，33.4分）**

**3. (简答题)**

**下方6种图形均以5号位为图形的旋转中心。图形S、Z、T、L、J的其他3方格均处于5号位的周边8领域内，图形旋转时候，它们能够依然处于5号位的周边8领域内。图形旋转后的位置偏移量较小。但图形I并不具备该特性。当图形I进行两次旋转时候，会给人一种图形在快速下降的赶紧。请尝试给出一种方案，解决图形I旋转时候，位置偏移量过大的问题。**

****

*我的答案：*

*0*分

1. 在变换I图形的时候，把选转中心暂时的改变，由此一来，，就可以达到在多次旋转后，位置不改变的目的
2. 在变换i图形的时候，对y值进行补偿，用+1/-1来调整位置

*正确答案：*

检测3、5号位在舞台上的高度差，以此对中心5号位及其他方格的高度差进行修正

**2022w15Q2Ch10S06**

题量: 4 满分: 100

作答时间:*05-30 13:30*至*05-30 15:30*

[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20860009&type=2)

*84.5*分

* 批阅人
* \*\*\*
* \*\*\*
* \*\*\*
* 打分
* 99
* 70
* 未批

**一. 单选题（共1题，25分）**

**1. (单选题)**

**1、在俄罗斯方块游戏中，为避免方块直接撞击底部，通常会在Logic函数中设置底部缓冲模块slowLoading。请问在Logic函数中，关于slowLoading模块的论述正确的是**

* A.

A、底部缓冲模块主要是预测下一步是否撞击底部，影响的是下一个周期的移动，应处于下降模块yMove的后面；

* B.

B、底部缓冲模块会决定方块后续的下降响应时间，应处于下降模块yMove的前面；

* C.

C、底部缓冲模块是独立存在，与其他模块不存在必然的先后关系，处于任意位置均可；

* D.

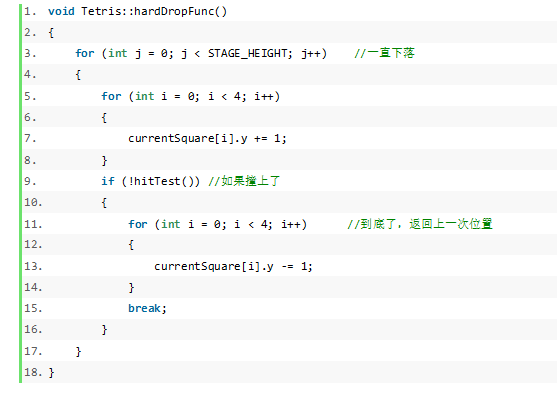
D、以上都不对

*我的答案:* B *正确答案:* C

*0*分

**二. 简答题（共3题，75分）**

**2. (简答题)2、Hard Drop通常叫速降、硬降或瞬降，是省略方块的下降过程，将方块直接放置在下方会发生碰撞接触的地方。在对方块行进路径进行判定的过程中，例如下方代码所示，第3行代码将遍历的范围设置为0至STAGE\_HEIGHT即舞台的高度。请问原因是什么？**

****

*我的答案：*

*0*分

一开始让其正常下落，9-16行寻找最终停留的位置，将高度变为舞台的全部高度就是，然后遍历所有的格子，让其一直下落

*正确答案：*

答案：因为方块图形的高度不确定，舞台上底部的堆积情况不确定，因此将遍历的范围设置为舞台的整个高度，以避免遗漏。由于高度最低的图形是2方格高的图形O，遍历的范围也可以缩减为STAGE\_HEIGHT-2

**\*\*\*：24分**

**\*\*\*：10分**

**3. (简答题)3、在俄罗斯方块游戏中，为避免方块直接撞击底部，通常会在Logic函数中设置底部缓冲模块slowLoading。请简述slowLoading模块是怎样实现底部缓冲功能的。**

*我的答案：*

*0*分

void Tetris : :slowLoading()//底部缓冲{  
for (int i = e; i < 4; i++){

    tempSquare[i] = currentsquare[i];currentsquare[i].y += 1;  
    if (!hitTest())//如果撞上了  
        delay = DALAYVALUE\*2;for (int i = e; i < 4; i++)  
    currentsquare[i] = tempSquare[i];  
}

一开始给出以上代码

底部缓冲模块的目的是为避免方块图形触底碰撞之后，直接变成背景方块。希望在碰撞的瞬间，给予玩家一点额外的反应时间作缓冲。因此需要在正常的方块下落过程中，不断地去确认当前位置是否触底。是否触底的判断条件在于再下一步是否会碰撞。如下方代码第3-9行则是检测当前位置再下移一格是否发生碰撞。若下一步会触底碰撞，则表示当前位置已经与底接触，则需要给出底部缓冲时间。这里如第9行代码所示，我们将缓冲时间设置为正常下落时间的两倍间隔。  
  
第3-9行代码毕竟是属于预判断，因此需要在判断结束后将方块的坐标进行还原。如第10-11行代码所示。  
当slowLoading()函数对变量delay的数值进行变更后，要记得用某种机制再将delay变量的值改回来。不然游戏会一直处于底部缓冲模式。本着模块化管理的思想，我们选择在slowLoading ()函数中进行更改。具体如下所示:  
void Tetris : : slowLoading(）{  
    for (int i = e; i < 4; i++){  
        tempSquare[i] = currentSquare[i];currentSquare[i].y += 1;  
    }  
    if ( !hitTest())//如果撞上了  
        delay = DELAYVALUE\* 2;//触底时候，时间延长，形成缓冲

    else  
        delay = DELAYVALUE;//非触底时候，则正常下落时间

    for (int i = 0; i <4; i++)  
    currentsquare[i] = tempsquare[i];

}

*正确答案：*

答案：①判断下一步是否触底撞击；②当预测到下一步会发生碰撞，则对方块的下落时间间隔进行加大；③时间间隔加大后，原来单个循环周期的计时累计就不会触发方块的下落事件，此时的表现是该循环周期内方块停留在指定高度不会下落；④当经过若干循环周期，计时累计值超过之前变更后的时间间隔，则方块继续下落。⑤前后方块驻留及最终下落之间存在的时间间隔会大于正常的方块下落时间间隔，就此形成缓冲。

**\*\*\*：25分**

**\*\*\*：25分**

**4. (简答题)4、在下方俄罗斯方块的投影shadow()函数中，第7-21行代码，为什么不直接用shadowSquare数组存储投影的最终位置，而是通过currentSquare参与遍历计算，并由第22-27行代码实现数值的中转？**

****

*我的答案：*

*0*分

方块投影通常需要知道投影的距离。

因此在下方代码的第3-4行代码中，先获取当前方块各方格的坐标位置。

第7-21行代码通过遍历的方式去寻找投影距离，以此确定方块投影最终会停留的位置。

第22-27行代码赋予currentSquare和shadowSquare数组各自正确的数值。

*正确答案：*

答案：因为代码中的hitTest函数里面，是采用currentSquare进行碰撞检测。被绑架了。然后老师还很懒，没有重新书写hitTest函数的意愿。

**2022w15Q2Ch10S07**

题量: 3 满分: 100

作答时间:*06-01 18:30*至*06-01 20:30*

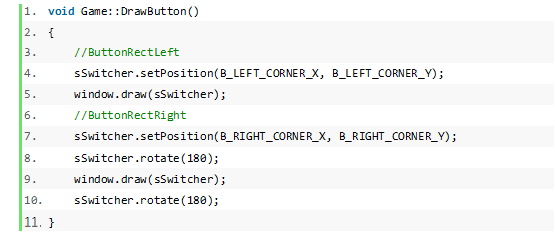
[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20901189&type=2)

*95.8*分

* 批阅人
* \*\*\*
* \*\*\*
* 打分
* 91.6
* 100

**一. 简答题（共3题，100分）**

**1. (简答题)1、游戏界面中绘制箭头的时候，通常是对单个箭头素材进行复用。比如在绘制左右箭头时候。通常是绘制好一个箭头后，将素材旋转180度，接着绘制另一个箭头。请问下方代码中。为什么出现两次rotate(180)的操作？第8、10行代码的作用分别是什么？**

****

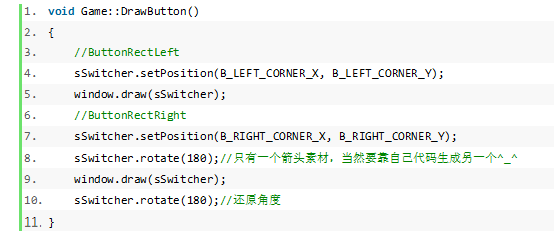
*我的答案：*

*0*分

第八行：只有一个箭头素材，需要使用选转代码，然后生成另一个向右的按钮

第十行：换原角度

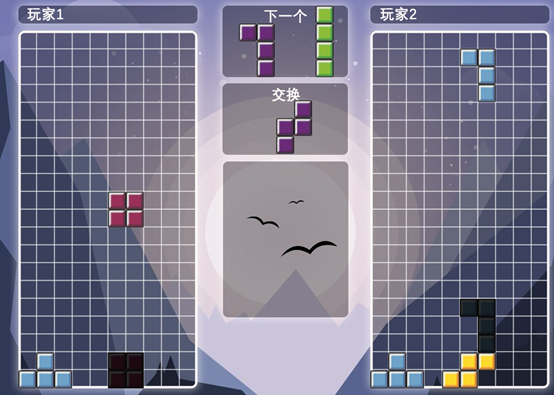
*正确答案：*



**\*\*\*：33.3分**

**\*\*\*：33.3分**

**2. (简答题)2、如下图所示的俄罗斯方块游戏中，游戏舞台边缘处的方块跑出原游戏舞台边框。请设计一个方案，解决该问题。**

****

*我的答案：*

*0*分

显然是下落的时候，方块覆盖了背景图片，所以我们可以把背景拆成两部分，分别是虚线网格和白色的边框。

然后再Draw（）中采用  虚线网格->阴影->方块->外白色边框  的顺序完成绘制

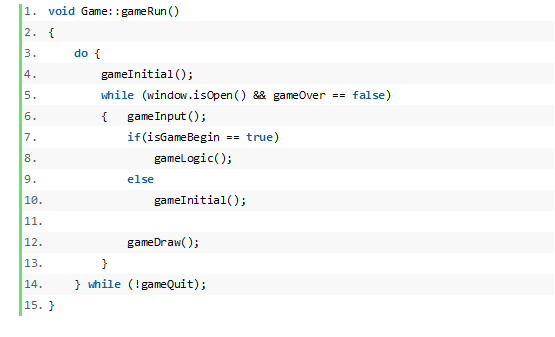
*正确答案：*

答案：额外再制作一个游戏舞台边框的素材，在游戏绘制函数的尾端，单独再绘制一次游戏舞台边框，以覆盖原先超出界线部分的方块内容。

**\*\*\*：33.3分**

**\*\*\*：33.3分**

**3. (简答题)3、游戏的状态通常有3种，分别为游戏初始状态、游戏运行状态和游戏结束状态。下方的游戏循环代码中，当isGameBegin为true的时候，则游戏循环执行gameLogic()函数，游戏进行常规运行状态；当isGameBegin为false的时候，则游戏循环执行gameInitial()函数，让游戏回到初始状态。请问，游戏的结束状态应该如何设置？**

****

*我的答案：*

*0*分

isGameBegin的具体取值由“开始/结束”按钮的鼠标点击响应控制。当游戏正常结束或玩家点击结束后，游戏结束，程序进行重新初始化。此时玩家再次点击开始按钮，游戏重新开始。

*正确答案：*

答案：本题是个开放性的题目。通常的操作有两种，①gameLogic()函数涵盖了游戏结束状态，

gameDraw()函数能够根据gameLogic()函数传递的结果，直接绘制游戏结束状态；②在游戏循环中再增加一种游戏结束状态的判定。

**2022w15Q2Ch10S08**

题量: 3 满分: 100

作答时间:*06-01 18:30*至*06-01 20:30*

[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20901243&type=2)

*100*分

* 批阅人
* \*\*\*
* \*\*\*
* 打分
* 100
* 100

**一. 单选题（共1题，33.3分）**

**1. (单选题)1、俄罗斯方块游戏中，以下Draw()函数的绘制顺序合理的是哪个（）？**

* A.

A. 活动方块、阴影、背景方块、消行动画、Next区方块；

* B.

B. Next区方块、阴影、活动方块、背景方块、消行动画；

* C.

C. Next区方块、背景方块、消行动画、活动方块、阴影；

* D.

D. Next区方块、背景方块、活动方块、阴影、消行动画；

*我的答案:* B *正确答案:* B

*33.3*分

**二. 简答题（共2题，66.7分）**

**2. (简答题)2、在俄罗斯方块游戏中，基于7种基本图形的特征，消行的时候一次最多可以消除4行（例如图形I）。但图形I不一定能够确保每次都消除4行，在一些特殊的情形下，甚至会出现只消除目标4行其中的个别行数。此时如果要做消行动画，请问如何避过不做消行的方格行。**

*我的答案：*

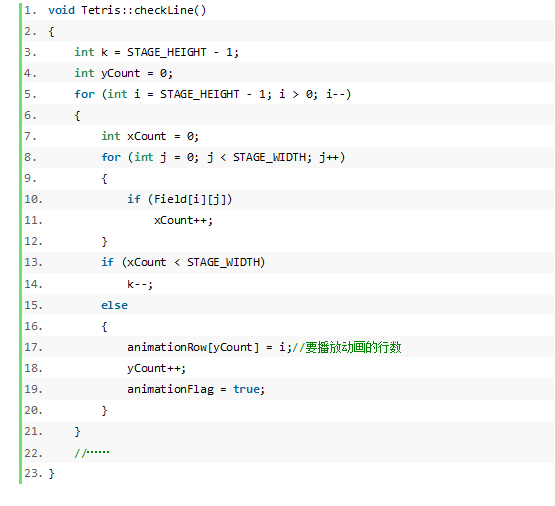
*0*分

理论上图形I可以最多消除4行，但一些特殊情况下，4行中间的某几行可能是不满足消行的要求。其他的图形方块也有类似的情况。因此，我们定义animationRow[]数组用于标记具体发生动画特效的行号。  
在Tetris::Initial(Texture\*tex)函数中进行变量初始化，具体如下所示。其中，第10行代码将animationRow[]数组的初值赋值为异常值-1。  
  
animationFlag = true;l//动画开启,由游戏是否开始决定animationctrlValue = 1.0;  
// ....  
for (int i = e; i < 4; i++){  
currentSquare[i].x = Figures[currentShapeNum][i] % 2 + STAGE\_WIDTH /2;

currentsquare[i].y = Figures[currentShapeNum][i]/2；  
nextSquare[i].x = Figures[nextShapeNum][i] % 2;  
nextSquare[i].y = Figures[nextShapeNum][i]/ 2;  
animationRow[i] = -1;//要播放动画的行数，-1为异常值，表待定的意思  
for (int i = e; i <STAGE\_HEIGHT; i++)  
{  
    for (int j = 0; j <STAGE\_WIDTH;j++)  
        Field[i][j] = 0;  
)

*正确答案：*

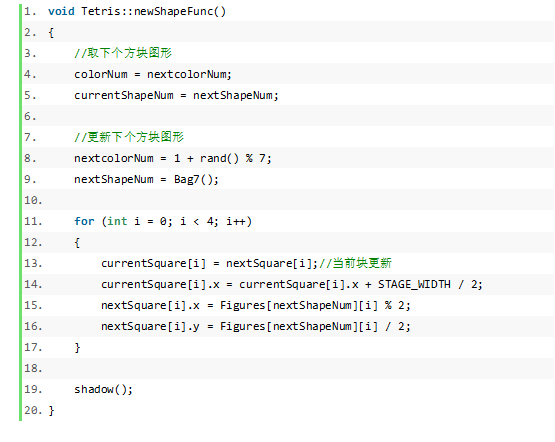
答案：如下所示，将需要消行的行号存储下来。



**\*\*\*：33.3分**

**\*\*\*：33.3分**

**3. (简答题)3、俄罗斯方块游戏中活动方块的投影，通常通过在Logic()函数中调用shadow()函数进行实现。下方代码为俄罗斯方块游戏中的新方块图形生成函数。请问为何在第19行代码中调用shadow()函数？是否多余了？**

****

*我的答案：*

*0*分

不多于

如果不加入，那对于新生的方块而言，是不会产生阴影的，只会在下一个循环通过主函数shadow（）生成阴影，从而产生不良影响

*正确答案：*

答案：第19行代码中调用shadow()函数的目的在于对shadowSquare[]数组的坐标进行更新，使之与currentSquare[]的更新保持同步。不然，当前shadowSquare[]数组中保存的还是currentSquare[]的上一轮数据。

**2022w16Q1Ch10S09**

题量: 3 满分: 100

作答时间:*06-06 13:30*至*06-06 15:30*

[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20967408&type=2)

*99.5*分

* 批阅人
* \*\*\*
* \*\*\*
* 打分
* 99
* 100

**一. 简答题（共3题，100分）**

**1. (简答题)1、讲义中的俄罗斯方块游戏，点击界面上的开始按钮之前，游戏舞台上是空的。点击开始按钮之后，程序直接进入游戏循环模块。那初始的游戏方块图形是怎么生成的？为什么在游戏开始之前，游戏舞台是空的？**

*我的答案：*

*0*分

游戏一开始的时候，绘制了一个最基本的函数和界面，图形已经生成，但是依旧没有被放在绘制图形里面；当你点击的时候，跳出本次循环，进入下一个游戏循环，正式开始游戏和绘制，因此舞台是空的

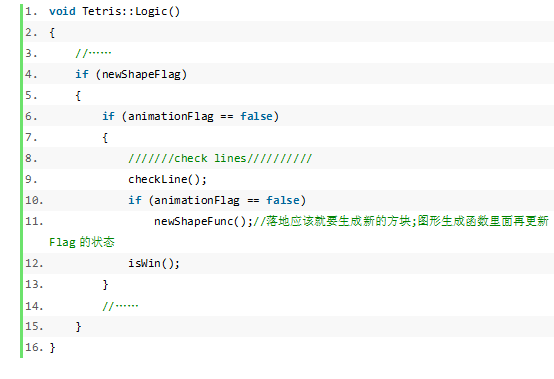
*正确答案：*

答案：初始的游戏方块图形是由初始化函数生成的，点击开始游戏之前，之所以在游戏舞台上没看到游戏方块图形，是由于绘图函数Draw()里面做了设置，如果游戏没开始则不绘制方块图形。

**\*\*\*：32.3分**

**\*\*\*：33.3分**

**2. (简答题)2、通常来说，Tetris游戏的胜负判定规则是如果新生成的方块图形与背景方块相撞则游戏判负，即胜负判定函数isWin()函数是在新图形生成函数newShapeFunc()函数之后跟着执行。即每次新生成方块的时候，都做一次isWin()胜负判定。请结合本章的内容，试分析下方的代码为什么这样调用isWin()函数？会不会影响游戏结果的判定？**

****

*我的答案：*

*0*分

不会，实际上胜负的判断输赢是独立于游戏现有逻辑的，只要判断每一次改变，也就是每一次逻辑发生的时候产生isWin()函数就可以了，所以isWin()函数可以放在任何能被运行的地方

*正确答案：*

答案：在Logic()函数中，有两个地方要调用newShapeFunc()函数，严格来说那就要进行两次调用isWin()函数。我太懒了，所以我只调用了一次。哝，就讲义里面你们看到的。这种做法是不可取的，要被唾弃的。大家引以为鉴。但是呢，newShapeFunc()函数在上方代码中执行的概率，要比在动画结束后被调用的概率高很多。再退一步说。当消行动画结束后（大佬，都发生消行了，应该不会马上就死吧？我也不知道不要问我），第一时间没调用isWin()函数，那下一个游戏循环是不是一定会调用isWin()函数?也就是说，在时间判定上相差一个游戏循环周期。哪位大佬帮我算下，一个游戏周期的时间间隔是多少？

**\*\*\*：33.3分**

**\*\*\*：33.3分**

**3. (简答题)3、双人版Tetris游戏里有个问题挺有意思的。当出现胜负判定时候，游戏是否要停止/结束？如果游戏就此结束，那当游戏只有一位玩家的时候，游戏如何进行？如果游戏不就此结束，那游戏总会有个尽头。当两个玩家都Lost的时候，如何设定游戏的ending？**

*我的答案：*

*0*分

当双人版Tetris游戏出现胜负判定时候，游戏可以选择结束也可以选择继续进行。这是一个开放性的问题，如果游戏就此结束，那么只有一位玩家的时候，玩家可以选择继续冲刺更高的分数也可以选择停止结束游戏，不过接下来将会是单人版的继续充分；游戏不就此结束，游戏总有个尽头，两个玩家都lost的时候，我们可以通过分数的比较判别游戏的胜负，最后玩家可以选择继续或者是结束游戏。

*正确答案：*

答案：两种方案，①设置一个单人模式，实例化一个Tetris对象就好；②等两个玩家都GameOver后，比较两人的得分情况，再出一个GameOver的界面，宣布最终结果。

**2022w16Q2Ch11S01**

题量: 2 满分: 100

作答时间:*06-06 13:30*至*06-06 15:30*

[智能分析](https://stat2-ans.chaoxing.com/study-knowledge/ans?courseid=95301317&cpi=149874575&clazzid=51189127&ut=s&relationId=20967436&type=2)

*100*分

**一. 单选题（共2题，100分）**

**1. (单选题)Box2D引擎最早是用什么语言开发的？**

* A.

A. Flash ActionScript

* B. C++
* C. Java
* D. C#

*我的答案:* B *正确答案:* B

*50*分

**2. (单选题)以下哪种表述是错误的？**

* A. Box2D引擎可以来模拟2D刚体物体的物理运动和碰撞。
* B. Box2D引擎的各种语言版本，均是官方正式发行的
* C. Box2D引擎不适合用于模拟柔性物体的物理运动和碰撞。
* D. Box2D引擎各种语言版本的API并不是完全一致的

*我的答案:* B *正确答案:* B