高级语言程序设计

实验报告

目录

[高级语言程序设计大作业实验报告 1](#_Toc21286)

[一. 作业题目 1](#_Toc29969)

[二. 开发软件 1](#_Toc18364)

[三. 主要流程 1](#_Toc25240)

[1． 整体流程 1](#_Toc20452)

[2． 核心算法 1](#_Toc25354)

[（1）子弹是否命中 2](#_Toc22508)

[（2） 改变植物和僵尸状态 5](#_Toc28665)

[3． 单元测试 8](#_Toc20452)

[四. 单元测试 8](#_Toc28265)

[五. 收获 8](#_Toc23116)

[1． easyx学习 8](#_Toc10799)

[2． 巩固类与对象知识 9](#_Toc16275)

[3． 问题转化和分析 9](#_Toc16275)

[4． 指针运用 9](#_Toc16275)

高级语言程序设计大作业实验报告

1. **作业题目**

制作一个简易版植物大战僵尸。

1. **开发软件**

**Visual stdio (eaxyx)**

辅助软件：PS

1. **主要流程**
   1. **整体流程**

实现思路：

1.首先，游戏开始界面通过一个按钮转换到游戏界面。

2.其次，游戏主体由plant类和zombie类组成，plant类实现各项植物功能，同时子弹可以抽象为一个bullet类；zombie类实现各项僵尸功能。

3.最后，通过判断僵尸出现轮数达到获得胜利要求，或者僵尸走到最左端达到失败要求获得胜利或者失败，游戏结束

* 1. **核心算法**

1.判断子弹和僵尸之间距离，在一定范围类时，删除子弹，播放动画，僵尸减少生命值，对三种僵尸处理类似，如果是icepea子弹还要降低僵尸速度

//检查子弹是否命中僵尸

while (pCurPlant != NULL)

{

int removePlant = 0;

int createBullet = 0;

UpdatePlant(pCurPlant, createBullet);

if (createBullet == 1)

{

if (pCurPlant->type == PLANT\_TYPE\_4 || pCurPlant->type == PLANT\_TYPE\_5)

{

int plantX = pCurPlant->x;

int plantY = pCurPlant->y;

// 保存植物类型，以便在释放后仍能使用

int plantType = pCurPlant->type;

if (gamePlant == pCurPlant)

{

gamePlant = pCurPlant->pNext;

}

Plant\* pTempPlant = pCurPlant;

pCurPlant = pCurPlant->pNext;

if (pTempPlant->pPrev != NULL)

{

pTempPlant->pPrev->pNext = pTempPlant->pNext;

}

if (pTempPlant->pNext != NULL)

{

pTempPlant->pNext->pPrev = pTempPlant->pPrev;

}

free(pTempPlant);

removePlant = 1;

Zombie\* pCurZombie = gameZombie;

while (pCurZombie != NULL)

{

int zombieX = pCurZombie->x;

int zombieY = pCurZombie->y;

// 辣椒有更大的爆炸范围

int explosionRange = 160;

if (plantType == PLANT\_TYPE\_5) {

// 辣椒的爆炸范围更大

explosionRange = 240;

}

if (HRDistance2Pos(plantX + 40, plantY + 40, zombieX + 80, zombieY + 80) <= explosionRange)

{

pCurZombie->hp = 0;

printf("zombie hp:%d\n", pCurZombie->hp);

if (pCurZombie->hp <= 0)

{

printf("zombie die!\n");

if (gameZombie == pCurZombie)

{

gameZombie = pCurZombie->pNext;

}

Zombie\* pTempZombie = pCurZombie;

pCurZombie = pCurZombie->pNext;

if (pTempZombie->pPrev != NULL)

{

pTempZombie->pPrev->pNext = pTempZombie->pNext;

}

if (pTempZombie->pNext != NULL)

{

pTempZombie->pNext->pPrev = pTempZombie->pPrev;

}

free(pTempZombie);

printf("remove zombie!\n");

gameParams.killZombieNum++;

}

}

else

{

pCurZombie = pCurZombie->pNext;

}

}

}

else if (pCurPlant->type == PLANT\_TYPE\_0)

{

Sunshine\* pNewSunshine = CreateRandSunshine();

pNewSunshine->x = HRGetRandNum(pCurPlant->x - 10, pCurPlant->x + 60);

pNewSunshine->y = pCurPlant->y;

pNewSunshine->targetY = pCurPlant->y + 50;

Sunshine\* pTemp = gameSunshine;

gameSunshine = pNewSunshine;

gameSunshine->pNext = pTemp;

if (pTemp)

pTemp->pPrev = pNewSunshine;

}

else

{

int hasZombieInRow = 0;

Zombie\* pCurZombie = gameZombie;

while (pCurZombie != NULL)

{

if (pCurZombie->row == pCurPlant->row)

{

hasZombieInRow = 1;

break;

}

pCurZombie = pCurZombie->pNext;

}

if (hasZombieInRow == 1)

{

PlantBullet\* pBullet = CreateBullet(pCurPlant->type);

pBullet->x = pCurPlant->x + 30;

pBullet->y = pCurPlant->y;

pBullet->row = pCurPlant->row;

PlantBullet\* pTemp = gameBullet;

gameBullet = pBullet;

gameBullet->pNext = pTemp;

if (pTemp)

pTemp->pPrev = gameBullet;

}

}

}

if (removePlant == 0)

pCurPlant = pCurPlant->pNext;

}

PlantBullet\* pCurBullet = gameBullet;

while (pCurBullet != NULL)

{

UpdateBullet(pCurBullet);

if (pCurBullet->x >= WINDOWS\_WIDTH)

{

if (gameBullet == pCurBullet)

{

gameBullet = pCurBullet->pNext;

}

PlantBullet\* pTempBullet = pCurBullet;

pCurBullet = pCurBullet->pNext;

if (pTempBullet->pPrev != NULL)

{

pTempBullet->pPrev->pNext = pTempBullet->pNext;

}

if (pTempBullet->pNext != NULL)

{

pTempBullet->pNext->pPrev = pTempBullet->pPrev;

}

free(pTempBullet);

}

else

{

pCurBullet = pCurBullet->pNext;

}

}

2.僵尸出现且在一定范围类时改变对应的normalpea、icepea和bigmouth状态，同样僵尸靠近各种植物，都改变状态，不移动，开始吃植物，使植物减血。

//修改植物和僵尸的状态

Plant\* pCurPlant = gamePlant;

while (pCurPlant != NULL)

{

int isPlantRemoved = 0;

int plantX = pCurPlant->x;

int plantY = pCurPlant->y;

Zombie\* pCurZombie = gameZombie;

while (pCurZombie != NULL)

{

int zombieTopLeftX = pCurZombie->x + 90;

int zombieTopLeftY = pCurZombie->y;

int zombieButtomRightX = pCurZombie->x + 160 + 90;

int zombieButtomRightY = pCurZombie->y + 160;

if (pCurZombie->row == pCurPlant->row)

{

if (HRIsRectangleCircleIntersect(zombieTopLeftX, zombieTopLeftY, zombieButtomRightX, zombieButtomRightY, plantX + 40, plantY + 40, 60))

{

pCurZombie->stop = 1;

printf("zombie stop:%d\n", pCurZombie->hp);

pCurPlant->hp -= 1;

if (pCurPlant->hp <= 0)

{

if (gamePlant == pCurPlant)

{

gamePlant = pCurPlant->pNext;

}

Plant\* pTempPlant = pCurPlant;

pCurPlant = pCurPlant->pNext;

if (pTempPlant->pPrev != NULL)

{

pTempPlant->pPrev->pNext = pTempPlant->pNext;

}

if (pTempPlant->pNext != NULL)

{

pTempPlant->pNext->pPrev = pTempPlant->pPrev;

}

free(pTempPlant);

printf("remove plant!\n");

isPlantRemoved = 1;

break;

}

}

else

{

pCurZombie->stop = 0;

}

}

pCurZombie = pCurZombie->pNext;

}

if (isPlantRemoved == 0)

{

if (pCurPlant)

pCurPlant = pCurPlant->pNext;

}

}

}

* 1. **单元测试**

每写完实现一个功能的代码，在可以运行的情况下都运行一下，以便能及时发现功能能否实现，不能也能及时发现问题所在。

1. **单元测试**

### 测试结果

其中有几次运行总会崩掉，后来发现原因是删除植物后，植物指针没有置空。

改正后功能基本都可实现。

1. **收获**

**1.eaxyx学习**

本次因为大作业第一次学习eaxyx，头次实现了代码可视化，头次见到了自己的代码能运行成什么样的东西，也提升了自己的自学能力和自学兴趣。

**2.巩固类与对象知识**

在本程序建立过程中，运用了大量类与对象的知识，恰好类与对象又是我们本学期课程的重点，对其有了进一步的理解。

**3.问题转化和分析**

将子弹打到僵尸碰撞问题，变成计算二者距离，在一定范围内后，子弹消失，播放动画，僵尸减血。

对植物和僵尸的每一项属性和行为进行分析和抽象，使得最终能用代码逐个实现，将一个大问题划分成一个个小问题之后再逐个突破。。

**4.指针运用**

对象删除后指针一定置空，不然程序极其不稳定。