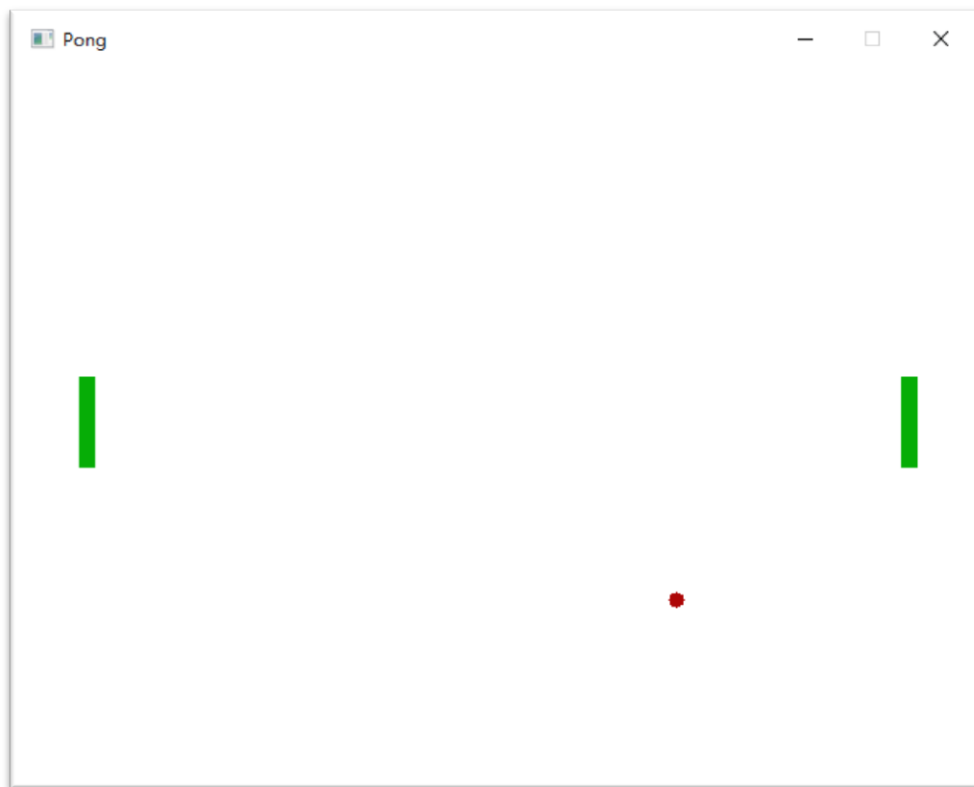


《C 语言程序设计》实习

一、实现一个小球碰撞的游戏

1. Visual Studio 2019
2. EasyX
3. 实现小球碰到窗口上下边界会反弹的功能；
4. 实现左边挡板和右边挡板上下移动的功能；
5. 实现小球碰到左边挡板或右边挡板会反弹的功能；
6. 主要界面如下图所示：



main.cpp

```
#include "pong.h"
```

```
int main()
```

```
{
```

```
    InitGame();
```

```
    GameLoop();
```

```
    QuitGame();
```

```
}
```

pong.h

```
#pragma once
```

```
#define WINWIDTH 640
```

```
#define WINHEIGHT 480
```

```
#define BALLRADIUS 5
```

```
#define PADDLEWIDTH 10
```

```
#define PADDLEHEIGHT 60
```

```
#define TIMESPAN 200
```

```
struct Ball
```

```
{
```

```
    int x;
```

```
    int y;
```

```
    int w;
```

```
        int h;

        int speedX;

        int speedY;
};

struct Paddle
{
    int x;

    int y;

    int w;

    int h;

    int speed;
};

void DrawScores();

void DrawBall();

void MoveBall();

void DrawPaddles();

// 初始化游戏

void InitGame();

void InitBall();

void InitPaddles();

// 退出游戏

void QuitGame();
```

```
// 游戏主循环函数

void GameLoop();

// 按键处理

void KeyboardControl(int key);

// 绘制游戏视图元素

void DrawView();

// 冲突检测

bool collision(Ball ball, Paddle paddle);

void MoveLeftPaddleUp();

void MoveLeftPaddleDown();

void MoveRightPaddleUp();

void MoveRightPaddleDown();

// 开始新的游戏

void RestartGame();
```

pong.cpp

```
#include "pong.h"

#include <graphics.h>

#include <conio.h>

#include <time.h>

#include <string>

#include <Windows.h>
```

```
bool isGameRunning = true;          // 游戏是否还在运行当中
```

```
Ball ball;
```

```
Paddle left_paddle;
```

```
Paddle right_paddle;
```

```
int left_score = 0;
```

```
int right_score = 0;
```

```
// 初始化游戏
```

```
void InitGame()
```

```
{  
  
    initgraph(WINWIDTH, WINHEIGHT);  
  
    setbkcolor(WHITE);  
  
    srand((unsigned)time(NULL));  
  
    HWND hWnd = GetHWND();  
  
    SetWindowText(hWnd, _T("Pong"));  
  
}
```

```
void InitBall()
```

```
{  
  
    ball.w = BALLRADIUS * 2;  
  
    ball.h = BALLRADIUS * 2;
```

```
ball.x = WINWIDTH / 2 - ball.w / 2;
```

```
ball.y = WINHEIGHT / 2 - ball.h / 2;
```

```
double rnd = rand() / (RAND_MAX + 1.0);
```

```
if (rnd < 0.5)
```

```
{
```

```
    ball.speedX = 2;
```

```
}
```

```
else
```

```
{
```

```
    ball.speedX = -2;
```

```
}
```

```
rnd = rand() / (RAND_MAX + 1.0);
```

```
if (rnd < 0.5)
```

```
{
```

```
    ball.speedY = 2;
```

```
}
```

```
else
```

```
{
```

```
    ball.speedY = -2;
```

```
}
```

```
}
```

```
void InitPaddles()
```

```
{
```

```
    left_paddle.x = 40;
```

```
    left_paddle.y = WINHEIGHT / 2 - PADDLEHEIGHT / 2;
```

```
    left_paddle.w = PADDLEWIDTH;
```

```
    left_paddle.h = PADDLEHEIGHT;
```

```
    left_paddle.speed = 5;
```

```
    right_paddle.x = WINWIDTH - (40 + PADDLEWIDTH);
```

```
    right_paddle.y = WINHEIGHT / 2 - PADDLEHEIGHT / 2;
```

```
    right_paddle.w = PADDLEWIDTH;
```

```
    right_paddle.h = PADDLEHEIGHT;
```

```
    right_paddle.speed = 5;
```

```
}
```

```
// 退出游戏
```

```
void QuitGame()
```

```
{
```

```
    closegraph();
```

```
    exit(0);
```

```
}
```

```
// 游戏主循环函数
```

```
void GameLoop()
```

```
{
```

```
    DWORD currFrameTime;
```

```
    DWORD nextFrameTime;
```

```
    DWORD delayFrameTime;
```

```
    while (isGameRunning)
```

```
    {
```

```
        InitBall();
```

```
        InitPaddles();
```

```
        while (isGameRunning)
```

```
        {
```

```
            currFrameTime = GetTickCount();
```

```
            // 判断是否有键盘按键被按下
```

```
            if (_kbhit())
```

```
            {
```

```
                int k = _getch();
```

```
        KeyboardControl(k);
    }

    MoveBall();

    BeginBatchDraw();

    DrawView();

    EndBatchDraw();

    nextFrameTime = currFrameTime + 1000 / 30;

    delayFrameTime = nextFrameTime - GetTickCount();

    if (delayFrameTime > 0)
    {
        Sleep(delayFrameTime);
    }
}

RestartGame();
}
}
```

// 按键处理

void KeyboardControl(int key)

{

 switch (key)

 {

 case 108: // 'l'

 MoveRightPaddleDown();

 break;

 case 111: // 'o'

 MoveRightPaddleUp();

 break;

 case 113: // 'q'

 isGameRunning = false;

 break;

 case 115: // 's'

 MoveLeftPaddleDown();

 break;

 case 119: // 'w'

 MoveLeftPaddleUp();

 break;

 default:

 break;

```
    }  
}  
  
// 绘制游戏视图元素
```

```
void DrawView()  
{  
    cleardevice();  
    DrawScores();  
    DrawBall();  
    DrawPaddles();  
}
```

```
void DrawScores()  
{  
  
}
```

```
void DrawBall()  
{  
    setfillcolor(RED);  
    solidcircle(ball.x, ball.y, BALLRADIUS);  
}
```

```
void DrawPaddles()

{

    setfillcolor(GREEN);

    solidrectangle(left_paddle.x, left_paddle.y, left_paddle.x + PADDLEWIDTH,
left_paddle.y + PADDLEHEIGHT);

    solidrectangle(right_paddle.x, right_paddle.y, right_paddle.x +
PADDLEWIDTH, right_paddle.y + PADDLEHEIGHT);

}
```

```
void MoveBall()

{

    ball.x += ball.speedX;

    ball.y += ball.speedY;

    if (collision(ball, left_paddle) || collision(ball, right_paddle))

    {

        ball.speedX *= -1;

    }

    if (ball.y < 0 || ball.y + ball.h > WINHEIGHT)

    {

        ball.speedY *= -1;

    }

}
```

```
// 小球移出左边界

if (ball.x < 0)

{

    ++right_score;

    InitBall();

    InitPaddles();

}


// 小球移出右边界

if (ball.x + ball.w > WINWIDTH)

{

    ++left_score;

    InitBall();

    InitPaddles();

}

}


bool collision(Ball ball, Paddle paddle)

{

    int ball_left, paddle_left;

    int ball_right, paddle_right;

    int ball_top, paddle_top;
```

```
int ball_bottom, paddle_bottom;

ball_left = ball.x;

ball_right = ball.x + ball.w;

ball_top = ball.y;

ball_bottom = ball.y + ball.h;

paddle_left = paddle.x;

paddle_right = paddle.x + paddle.w;

paddle_top = paddle.y;

paddle_bottom = paddle.y + paddle.h;

if (ball_left > paddle_right)

{

    return false;

}

if (ball_right < paddle_left)

{

    return false;

}

if (ball_top > paddle_bottom)

{

    return false;

}

if (ball_bottom < paddle_top)
```

```
{  
  
    return false;  
  
}  
  
return true;  
  
}
```

```
void MoveLeftPaddleUp()
```

```
{  
  
    left_paddle.y -= left_paddle.speed;  
  
    if (left_paddle.y < 10)  
  
    {  
  
        left_paddle.y = 10;  
  
    }  
  
}
```

```
void MoveLeftPaddleDown()
```

```
{  
  
    left_paddle.y += left_paddle.speed;  
  
    if (left_paddle.y > WINHEIGHT - PADDLEHEIGHT)  
  
    {  
  
        left_paddle.y = WINHEIGHT - PADDLEHEIGHT;  
  
    }  
  
}
```

```
}
```

```
void MoveRightPaddleUp()
```

```
{
```

```
    right_paddle.y -= right_paddle.speed;
```

```
    if (right_paddle.y < 10)
```

```
    {
```

```
        right_paddle.y = 10;
```

```
    }
```

```
}
```

```
void MoveRightPaddleDown()
```

```
{
```

```
    right_paddle.y += right_paddle.speed;
```

```
    if (right_paddle.y > WINHEIGHT - PADDLEHEIGHT)
```

```
    {
```

```
        right_paddle.y = WINHEIGHT - PADDLEHEIGHT;
```

```
    }
```

```
}
```

```
void RestartGame()
```

```
{
```

```
cleardevice();

int startX = 220;

int startY = 100;

char s[20];

setlinecolor(GREEN);

rectangle(50, 50, 590, 400);

settextcolor(GREEN);

settextstyle(50, 0, _T("宋体"));

outtextxy(startX, startY, _T("游戏结束"));

settextstyle(30, 0, _T("宋体"));

sprintf_s(s, "左边得分 : %d", left_score);

outtextxy(startX - 120, startY + 100, _T(s));

settextstyle(30, 0, _T("宋体"));

sprintf_s(s, "右边得分 : %d", right_score);

outtextxy(startX + 120, startY + 100, _T(s));

settextstyle(30, 0, _T("宋体"));

outtextxy(startX - 10, startY + 200, _T("按 r 键重新开始"));

settextstyle(30, 0, _T("宋体"));

outtextxy(startX - 10, startY + 250, _T("按 q 键退出游戏"));

while (true)

{

    char k = _getch();
```

```
    if (k == 'r')
    {
        isGameRunning = true;

        break;
    }

    if (k == 'q')
    {
        isGameRunning = false;

        break;
    }
}

}
```