# 贪吃蛇介绍：

## 功能

游戏地图的绘制

贪吃蛇和食物的打印

通过方向键控制贪吃蛇行进，吃掉食物后，蛇身增长一段

蛇撞墙或者撞到自身死亡，游戏退出

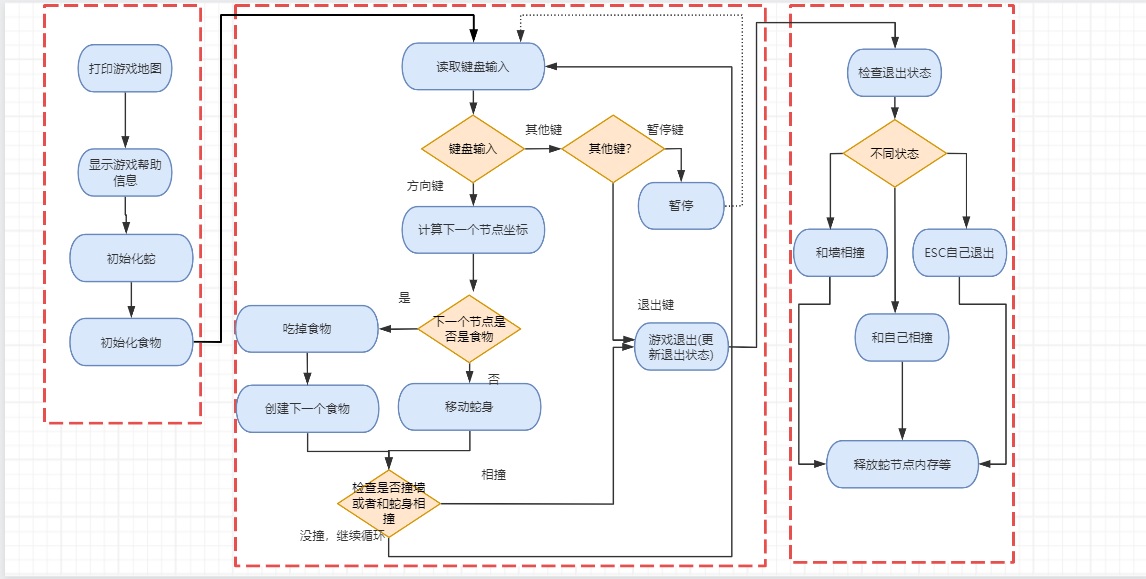
ESC键可以随时退出游戏

空格键SPACE可以随时暂停游戏

F3/F4键可以加快/减慢蛇爬行速度，增加/降低游戏难度

显示游戏当前得分状况

## 流程图



按照流程图所示，游戏可以大致分成三个大模块：

1. 游戏初始化阶段
2. 游戏进行阶段
3. 游戏结束阶段

### 游戏初始化阶段

1. 初始化游戏地图

在初始化阶段，需要在控制台程序上打印贪吃蛇游戏的围墙。

尽管Window API 提供了API函数，可以在控制台应用程序窗口中打印字符，但API在界面美化上还是有局限，因此这里主要使用EasyX。EasyX 是基于 C++ 的图形界库，可以帮助 C/C++ 初学者快速上手图形和游戏编程。EasyX的介绍参考后面章节。

1. 显示游戏的帮助信息

在游戏主界面能显示帮助信息，包括：

* 使用方向键控制蛇移动
* ESC键退出游戏
* 空格键暂停游戏
* F3加快蛇移动速度
* F4减慢蛇移动速度

1. 初始化蛇和食物

### 游戏进行阶段：

1. 读取键盘输入：

* 方向键

1. 获取下一个节点的坐标信息
2. 如果下一个节点是食物，吃掉食物并更新得分，然后创建新的食物
3. 如果下一个节点不是食物，仅仅移动蛇身。

* ESC键

1. 更新游戏状态为：ESC

* 空格键

1. 暂停挂起游戏，等待一段时间后重新运行

* F3、F4键

1. 减少蛇移动速度
2. 加大蛇移动速度
3. 判断是否撞墙或者撞到自己

如果撞墙，更新游戏状态为：KILL\_BY\_WALL

如果撞到自己，更新游戏状态为：KILL\_BY\_SELF

1. 刷新游戏当前得分情况

### 游戏结束阶段

当游戏的状态不在正常状态（NORMAL）时，游戏退出，包括：

ESC: 用户自己退出

KILL\_BY\_WALL：撞墙

KILL\_BY\_SELF：撞自己

游戏结束前，需要进行的扫尾工作：

* 释放内存，防止内存泄漏

# 二、EasyX使用

## 参考：

<https://www.dotcpp.com/course/easyx-go/>

## 下载和安装

<https://easyx.cn>

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

点击“安装”

安装的本质，是将easyX的.h头文件和.lib库文件复制到Visual C++的include和lib文件夹中。

## 第一个例子: 绘制圆

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

代码如下：

#include <iostream>

//EasyX头文件

#include <graphics.h>

#include <conio.h>

int main()

{

initgraph(640, 480); // 绘制窗口，大小640\*480像素

setfillcolor(YELLOW); // 设置填充颜色为黄色

fillcircle(200, 200, 100); // 画圆，圆心（200，200），半径100

// 按任意键继续

getchar();

closegraph();

return 0;

}

使用Easy X，首先需要进入EasyX的头文件：

//EasyX头文件

#include <graphics.h>

#include <conio.h>

运行结果：

A yellow circle on a black background

Description automatically generated

## 坐标

看到easyX绘图的效果后，普及一下EasyX会用到的几个概念：

坐标

  不仅仅画圆，以及画点、线、圆等各种图案都离不开坐标的概念，那么这个坐标是怎么计算的，那就是以窗口程序的左上角为（0，0）点，x轴和y轴分别向右和向下递增，单位为像素点

A screenshot of a computer

Description automatically generated

需要特别注意的是，一个中文字符占x=15,y=20

## 第二个例子：绘制矩形

代码：

#include <iostream>

//EasyX头文件

#include<graphics.h>

#include<conio.h>

int main()

{

// 绘制窗口，大小640\*480像素

initgraph(640, 480);

// 绘制矩形, 左上点坐标（50，50），右下坐标（200，200）

rectangle(50, 50, 200, 200);

// 按任意键继续

getchar();

closegraph();

return 0;

}

运行结果：

A black screen with a square in it

Description automatically generated

## 第三个例子：加载外部图片并显示

easyX首先通过loadimage加载图片，然后使用putimage输出到窗口指定位置.

涉及到两个函数：

**loadimage**

第一个参数是传入参数，因此需要加地址符，

第二个参数是图片的路径。 如果使用的路径是相对路径，所以用../。绝对路径当然也可以。需要注意的是，因为C语言中字符串转义字符的存在，这里需要写两个/

第三个/第四个参数：load图片的宽和高

**putimage**

在指定位置输出即可

// 加载图片

IMAGE img;

loadimage(&img, \_T("..//..//Images//Wall3.jpg"), 15, 20, true);

//向（10,10）位置开始输出此图片

putimage(0, 0, &img);

完整代码如下：

#include <iostream>

// EasyX头文件

#include <graphics.h>

#include <conio.h>

int main()

{

// 创建绘图窗口，大小为 640x480 像素

initgraph(900, 500);

// 加载图片

IMAGE img;

loadimage(&img, \_T("..//..//Images//Wall3.jpg"), 15, 20, true);

//向（10,10）位置开始输出此图片

putimage(0, 0, &img);

//任意键继续

getchar();

closegraph();

return 0;

}

输出效果：

A black screen with a white border

Description automatically generated

上面只是显示一张图片的例子，在编写贪吃蛇游戏的时候，会绘制一圈墙，下面就是程序代码：（注意一个中文字符在EasyX坐标中占x=15,y=20）

#include <iostream>

// EasyX头文件

#include <graphics.h>

#include <conio.h>

int main()

{

// 创建绘图窗口，大小为 900x500 像素（注意：900是15倍数，500是20倍数）

initgraph(900, 500);

// 加载图片

IMAGE img;

loadimage(&img, \_T("..//..//Images//Wall3.jpg"), 15, 20, true);

//上面的围墙

for (int i = 0; i<= 900; i=i+15) {

putimage(i, 0, &img);

}

//下面

for (int i = 0; i <= 900; i = i + 15) {

putimage(i, 500-20, &img);

}

//左

for (int i = 20; i < 500; i = i + 20) {

putimage(0, i, &img);

}

//右

for (int i = 20; i < 500; i = i + 20) {

putimage(900 - 15, i, &img);

}

//任意键继续

getchar();

closegraph();

return 0;

}

显示效果：

A black rectangular object with brown squares

Description automatically generated

## 第四个例子：向窗口指定位置输出文本

C语言使用printf函数输出，而且还只能顺着光标的位置打印，如果想特定位置输出还需要不停的输出换行或者回车调整光标位置。而且颜色只有白色的。而easyX可以很方便的实现这些功能，其函数如下：

**void** outtextxy(**int** x,**int** y,**LPCTSTR** str);

x,y是输出文本的坐标

str是输出文本

代码例子：

#include <iostream>

//EasyX

#include <graphics.h>

#include<conio.h>

int main()

{

initgraph(640, 480); // 创建绘图窗口，大小为 640x480 像素

outtextxy(100, 100, \_T("www.dotcpp.com"));

getchar(); // 按任意键继续

closegraph(); // 关闭绘图窗口

return 0;

}

以上代码的意思，是向窗口（100，100）位处开始输出“www.dotcpp.com”字符串，默认是白色字体，使用是不是非常方便?

运行效果：

A screen shot of a computer

Description automatically generated

## 如何擦除已经绘制的图案

对于有动态效果的程序，我们为了实现动画效果，会非常频繁的对图案有擦除的需求，对于已经绘制的图案如何擦除呢？可以调用cleardevice()函数清空屏幕，函数原型分别为：

|  |  |
| --- | --- |
| 1 | **void** cleardevice(); |

# 三、主要数据结构

# 四、代码