**第8课 吃豆精灵**

**【教学重点与难点】**

**重点：**理解面向控件的含义，学会使用按键控制角色的方法。

**难点：**运用侦测与随机函数控件创作游戏。

**【教学过程】**

**一、游戏导入**

1. 老师今天给同学们带来一个小游戏，这个游戏比较考验我们的反应速度，请同学们打开桌面上的吃豆精灵游戏试着玩一玩。 老师看到同学们玩的都非常开心，但是会玩游戏不是真正的本领，会制作游戏才是一个真正的高手，今天我们就一起来学习用scratch制作《吃豆精灵》游戏。

**二、新授知识**

1.分析游戏

用Scratch制作游戏的一般过程我们都知道：先分析角色和游戏规则，再写成程序脚本。请同学们回忆一下这个游戏有几种角色呢？（有两种，分别是吃豆精灵和豆子角色。）下面我们根据这两角色来分析一下游戏规则，了解了游戏规则也就了解了制作游戏的思路和方案。我们也可以把这个过程称之为是制定算法，根据算法我们能更有效率的搭建程序。

（吃豆精灵角色）吃豆精灵在这个游戏中有哪些功能呢？（能够左右移动，吃豆子）实现这些功能的算法我们可以这样描述

①能控制吃豆精灵左右移动；

②碰到豆子后要进行造型的切换。表示吞咽吃掉豆子。

那对应的程序该如何搭建呢？下面请同学们打开桌面上的吃豆精灵文件，根据吃豆精灵这两个功能的算法提示尝试搭建程序。

预设：老师发现有同学已经成功了，我们请这位同学上台演示一下吃豆精灵第一个功能的程序。（生演示）这位同学用了一个新的控制角色移动的方法，事件模块中的按键控制，这个控件可以设置通过左右方向键控制吃豆精灵左右移动。这位同学非常聪明当按下右移键设置移动10步，按下左移键设置移动-10步，但是这样做的话你们有没有发现什么问题呢？（吃豆精灵向左移动时面向右边）有哪位同学可以来解决这个问题呢？请你来（可以使用面向控件，选择面向90度，控件上显示的是面向右边，面向-90度，显示的是面向左边，改变方向后角色会上下翻转，因此还要将旋转方式设定成左右翻转）请你继续演示第二个功能的程序，他利用判断语句，如果碰到了绿色就进行造型的切换，张嘴后等待0.5秒再闭上嘴巴表示吃掉豆子。那能不能是碰到豆子角色呢 ？（因为豆子角色会有很多，他不是唯一的，而他们的颜色是唯一的）没有完成的同学根据正确的操作继续完成，完成的同学请跟左右同桌之间探究一下豆子有哪些功能呢？

（豆子角色）我们请这位同学来说一说，豆子角色有哪些功能呢？（豆子从上方出现，往下掉落，碰到吃豆精灵会被吃掉）实现这些功能的算法我们可以这样描述

1. 豆子随机出现在舞台的上方区域（移到控件+随机数控件）

②豆子往下掉落

③在下落的过程中碰到吃豆精灵要隐藏自己表示被吃掉。

下面请同学们好好回忆我们之前学习过的控件，根据豆子角色这三个功能的算法提示尝试搭建脚本。

预设：老师发现有些同学已经完成了，我们请这位同学上台来演示一下豆子角色的第一个功能的程序，豆子随机出现在舞台的上方区域。这里他用到“动作”模块中的移到控件以及“运算”模块中的随机函数控件。那请问随机函数的范围你是如何来选取的呢？（目测上方区域，通过鼠标指针来获取该区域最低处的y坐标）

请你继续演示豆子从舞台上方掉落的程序。这里他通过减少豆子的y坐标实现豆子垂直掉落。这里有没有同学有其他的方法？嗯，可以使用面向控件和移动控件相结合，让豆子面向下方移动，也能实现豆子垂直掉落。

请你继续演示第三个功能的程序，豆子碰到吃豆精灵会被吃掉，利用了判断语句，如果豆子碰到了吃豆精灵角色，就隐藏表示自己被吃掉了。这里我们要让他等待0.5秒，防止出现豆子消失的太快，吃豆精灵来不及切换造型的问题。当然游戏重新开始时要显示出来。

 **三、创新实践**

在完成豆子程序的搭建之后呢，我们可以复制多个豆子角色，来增加游戏的趣味性。你们觉得还可以增加哪些环节让这个游戏能够更加精彩呢？（角色大小，下落速度；添加新的角色比如一颗坏的豆子；让被吃掉的豆子以及落到地面的豆子都回到舞台上方继续落下）我们可以从角色的自身属性，角色种类的多样性或是整个游戏的故事性等方面来进行创新，下面请你们带着自己的想法来尝试一下。

预设：老师在巡视的过程发现有些同学已经完成了，老师挑选了两位同学的作品我们一起来看一看。我们先看第一位同学的作品，你来给我们说一说你进行了哪些创新呢（他让本该被吃掉消失的豆子以及落到地面y坐标<-160的豆子都重新回到了舞台上方，让游戏变得更有趣味性）

我们继续看第二位同学的作品，你来说一说你进行了哪些创新呢？（他使用随机数控件让豆子的大小以及下落的速度都各不相同，使得游戏的角色更加丰富/他添加了黑色的坏豆子，当吃到黑色的豆子游戏就会失败，让游戏变得更加具有挑战性）

**四、小结**

本节课呢我们首先分析了游戏的规则，学会了制定合适的算法，从而能够有条不紊的利用相关控件完成吃豆精灵这款游戏。希望同学们在生活中也能如此，当一个复杂的问题我们难以处理时，就要学会制定算法来进行解决。