项目一

HELLO WORLD

发现新世界

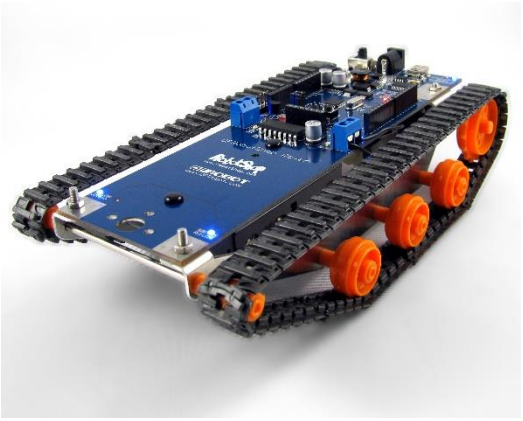
一天早上，你从梦中醒来，转头看了一眼床头的闹钟。七点三十四分！说好的七点钟的闹铃呢！上学要迟到了！你从床上弹起来，抓起昨晚换下来的外衣匆忙穿上，跑进卫生间洗脸刷牙，头发还乱糟糟的，但是没时间管它了，你跑进厨房，打开冰箱，发现里面什么也没有，昨天晚上你已经把吃的都吃掉了，看来你只能饿着肚子上课了。你把桌上的书本一股脑塞进书包，抓起钥匙跑出家门，眼前一片阴郁的灰色，空气中弥漫着雾霾的味道……

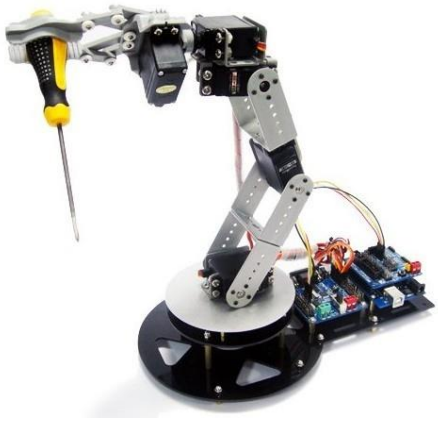
你是否遇到过这样的窘境呢？每个人都可能会遇到这样倒霉的时候吧。

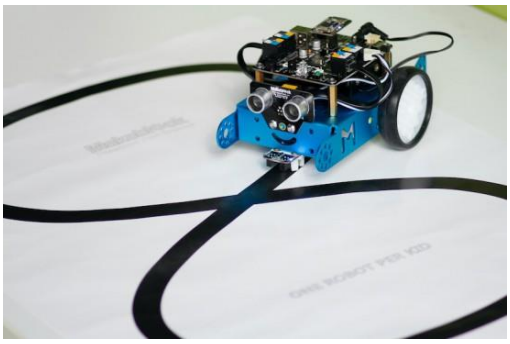
设想另一种情况，早上七点，阳光透过窗帘，照进你的房间。你的闹钟感受到了阳光，开始大叫，在房间里四处乱跑，等你抓到它，它才会安静。这时，你也就清醒了。在你去洗漱的同时，厨房里的机械手臂夹起烤好的面包，还有你喜欢吃的果酱，一杯热牛奶，放在一辆小餐车的盘子上。你洗漱完毕，走进厨房。小餐车停在你的面前，托盘抬高，让你舒服地享用早餐，为一天的学习、工作做好准备。安装在室外的湿度检测器、温度检测器、PM2.5 检测器为你实时测量温度并提供预报，将数据传送到衣柜，衣柜则为你挑选好今天适合穿的衣服和鞋子，并为你决定今天是否需要戴口罩出门。你无需为每一件小事作出选择，可以节省精力专注于你的学习或事业。

这样的生活看上去很不错嘛！而且，我有一个好消息要告诉你，这样的生活，离我们已经不远了！这些为你的生活提供着各种便利的机器人，已经很容易被制造出来。它们有着共同的“大脑”，或者说“灵魂”——Arduino 控制板。通过 Arduino，人们可以发明出很多有意思的、有用处的机器，为生活提供便利。还是那句老话，只有想不到，没有做不到。

下面是几样用 Adruino 主控板以及各种零件做出的作品。







Arduino 入门

Arduino 开源系统简介

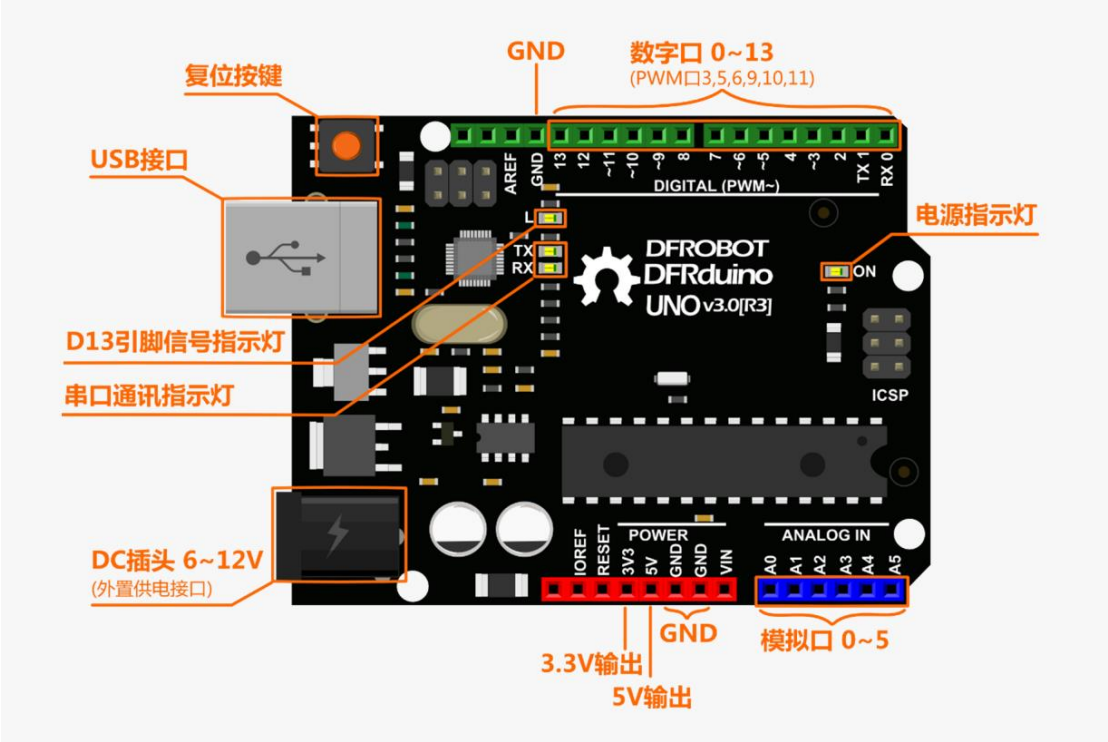
- Arduino 是一个开放源码电子原型平台，拥有灵活、易用的硬件和软件(板子及在此之上的软件)。
- Arduino 可以接收来自各种传感器的输入信号从而做到监测环境的效果，并通过控制光源，电机以及其他执行器来影响其周围环境(类似于人类的工作)。
- 硬件低廉，软件免费。
- 可以完成的例子：
  - 当咖啡煮好时，咖啡壶就发出“吱吱”声提醒
  - 当邮箱有新邮件时，电话就会发出铃声通知
  - 自制一个心率监测器，将每次骑脚踏车的记录存进存储卡
  - 复制一张门禁卡、饭卡

- .....

Mixly 创意电子标准套件介绍

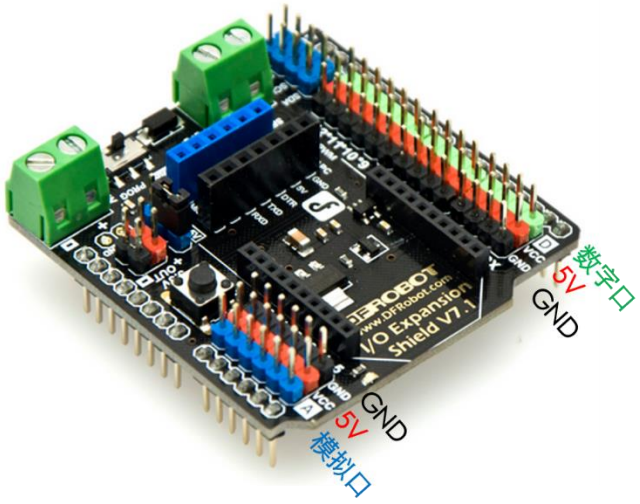
- 3-80cm 红外接近开关
- 模拟角度电位器
- 数字蜂鸣器模块
- 模拟环境光线传感器
- 模拟声音传感器
- 数字大按钮模块
- 数字 LED(白/红/蓝)
- TowerPro SG90 舵机
- 红外接收模块+遥控器
- 超声波传感器
- LCD 液晶屏
- 传感器 IO 扩展板 V7
- DFRduino UNO R3
- USB 电缆

Arduino UNO 介绍



Arduino UNO 扩展板介绍

注意：说明文字的颜色对应原件导线的颜色。



Mixly 安装使用

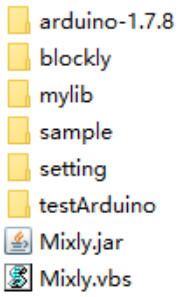
1. 下载 Mixly（下载地址：http://maker.bnu.edu.cn/）

Mixly 是北师大教育学部创客教育实验室提供的免费工具。

- Mixly for Arduino---Mixly **系统下载**

- **跳转至百度云下载：**

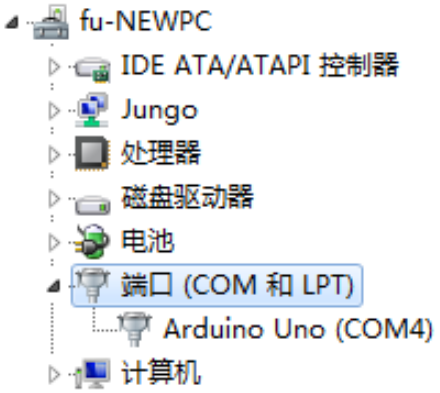
- **下载完成后解压缩：**



2. 连接 Arduino 并安装驱动

- **右键单击“我的电脑”---属性---设备管理器---更新驱动程序软件**

- **驱动程序目录：**mixly\arduino-1.7.8\drivers
- **更新完成后显示串口号（图中为** COM4 **）**

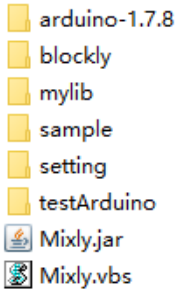


本节任务

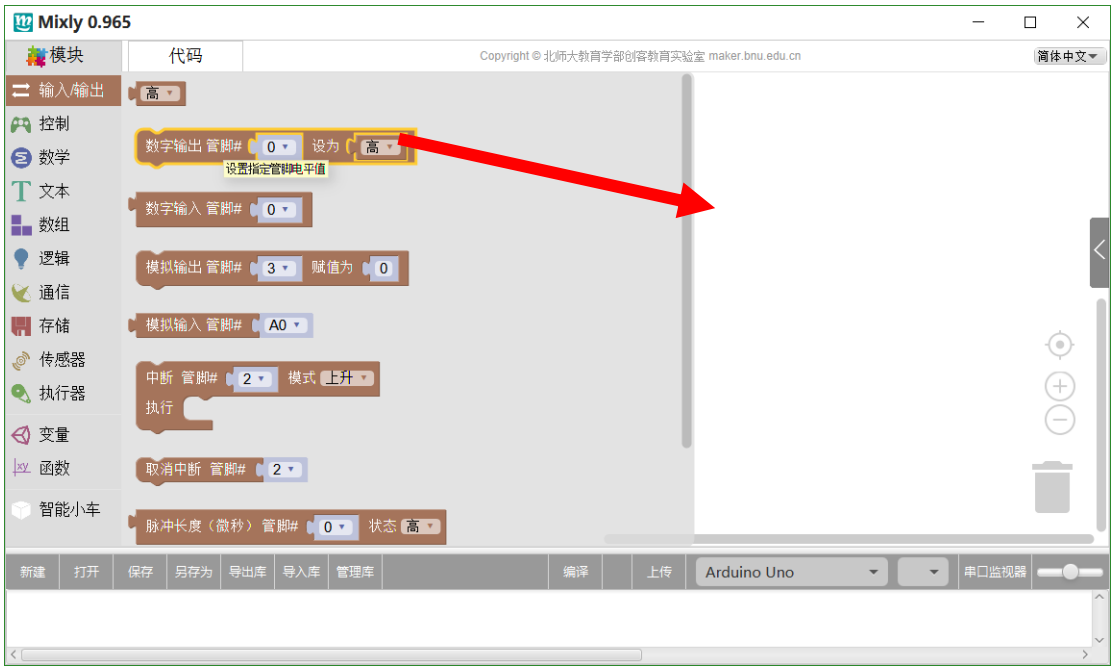
任务 1——点亮板载 LED 灯

1. 程序编写

双击右图中的 Mixly.vbs 文件，即可打开 Mixly 软件：

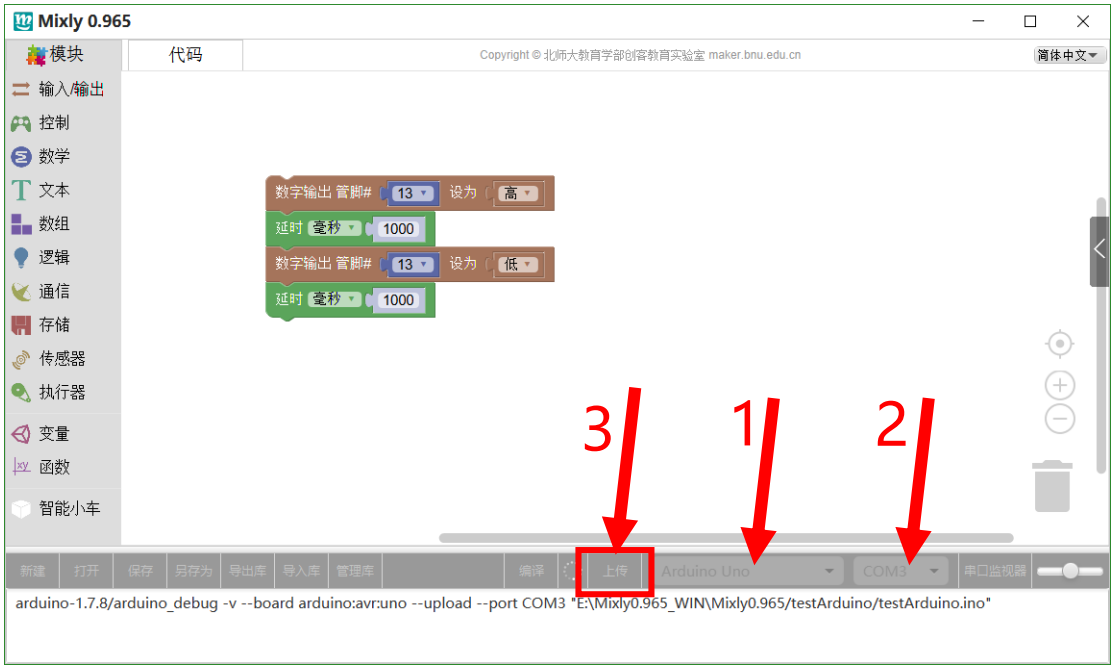


在“输入输出”菜单中找到“数字输出”模块，点击并拖动至空白处。



在“控制”菜单中找到“延时模块”，拖动至空白处并与“数字输出”模块拼接。

编写好程序后，单击下图中的上传按钮，将程序上传到 Arduino 主控板上。（注意，在上传程序之前，要设置好主控板型号和 COM 接口号，点击上传按钮右侧的下拉菜单即可设置）



上传之后，所有的按钮都变为灰色，无法点击，以保证上传过程不被干扰。

这里，我们见到了两个非常常用的模块——**数字输出**和**延时**。

数字输出是 Arduino 主控板对原件的控制方式之一。它向输出的电路传送数字信号——0 和 1。0 意味着输出低电平，电路不会接通；1 则是输出指高电平，电路接通。

上面的程序中，将 13 号管脚的数字输出设为高，与其连接的板载 LED 灯便会被点亮。经过 1 秒钟的延时（延时过程中，硬件保持延时开始时的状态，直到设定的时间结束），数字输出变为低，灯就会熄灭，之后保持熄灭状态 1 秒钟。

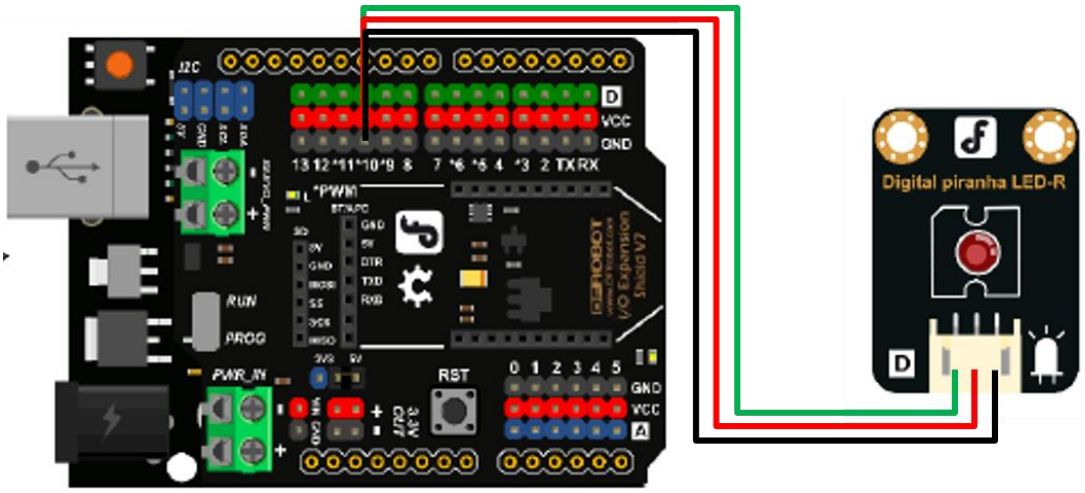
可以看到，板载 LED 灯在熄灭 1 秒后又重新亮了起来，1 秒钟后又熄灭，如此重复下去。这是因为，Mixly 和 Arduino 默认这段程序是重复执行的。如果没有其他干预，程序便会一直重复执行。

并且，灯总是亮 1 秒，灭 1 秒，这个重复不会发生变化。这是因为，这些程序的模块是按它排列的顺序执行的，主控板不会先执行第一个模块，然后跳过延时的模块，直接去执行第三个模块，或者是按任何与程序不一样的顺序执行。

任务 2——点亮真实的 LED 灯

1. 硬件连接

将 LED 灯与 10 号管脚相连。注意插线时的颜色对应。



2. 程序编写

我们只需将管脚号从 13 换成 10 即可。

任务 3——让 LED 越闪越快

1. 程序编写

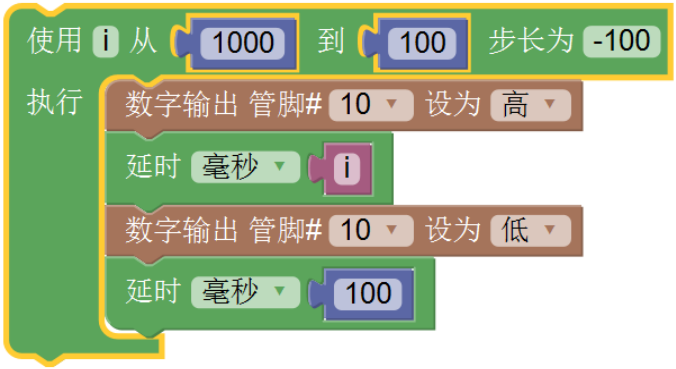
首先来看这样一段程序（右图）：

不难看出，这段程序是由其中的一小段不断重复而成的：

-

按照之前讲过的顺序执行原则，这一组模块就会一直重复下去。如果我们想让它执行很多次（比如 100 次），一种办法就是，把 100 个这样的模块组前后连接在一起。

可想而知，那样接起来的程序会非常长。并且，如果想知道它被重复执行了多少次，数起来也非常麻烦。那么，有没有一种模块，可以自动实现这个重复的过程，并且让人不怎么费力就知道重复执行了多少次呢？有的。



2. 代码讲解：

上面这段程序，实现了“重复”这个想法。它就是程序中常用的循环结构。如何理解这个

循环结构呢？“使用 i 从 1000 到 100 步长为 -100”这句话是什么意思呢？

循环的次数通过变量 i 的大小来控制。所谓**变量**，就是在程序运行过程中大小发生变化的量。循环执行过程中，变量 i 从 1000 开始，每次减小 100，直到减小到 100，然后循环结束。每次循环，都会执行一次包在其中的程序（这里就是控制灯亮灭的程序）。

知识点小结

元件

1. LED

Mixly 程序模块

1. 数字输出
2. 延时
3. 循环

程序设计

1. 顺序结构
2. 循环结构
3. 变量

项目二

S.O.S.

什么是 S.O.S.?

S.O.S.可以被展开为：

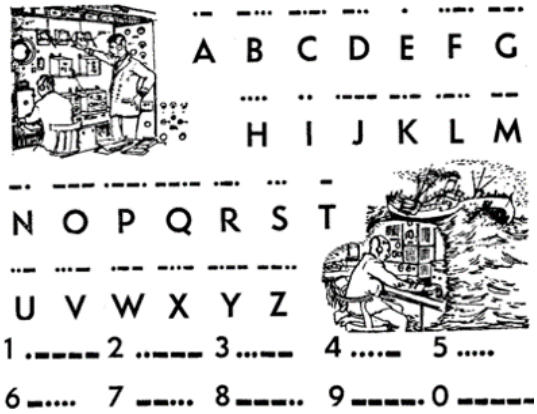
- Save Our Souls （拯救我们的灵魂）
- Save Our Ship （拯救我们的船）
- Send Our Succour （速来援助）
- Saving Of Soul （救命）

这是全球通用的求救信号。当你身在异乡，突遇险情，SOS 求救信号或许可以救你一命。

S.O.S.如何表示？

摩尔斯电码中，用...---...（**三短三长三短**）来表示 S.O.S.求救信号。

摩尔斯电码由美国人萨缪尔·摩尔斯于 1844 年发明。它是一种时通时断的信号代码，通过不同的排列顺序来表达不同的英文字母、数字和标点符号，如右图所示。

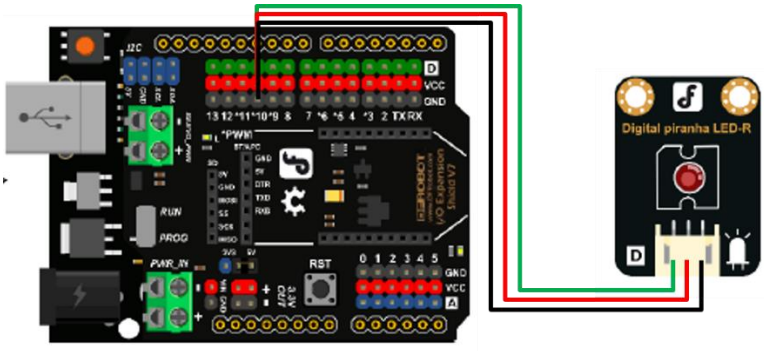


本节任务

任务 1——用 LED 发出摩尔斯电码中的求救信号。

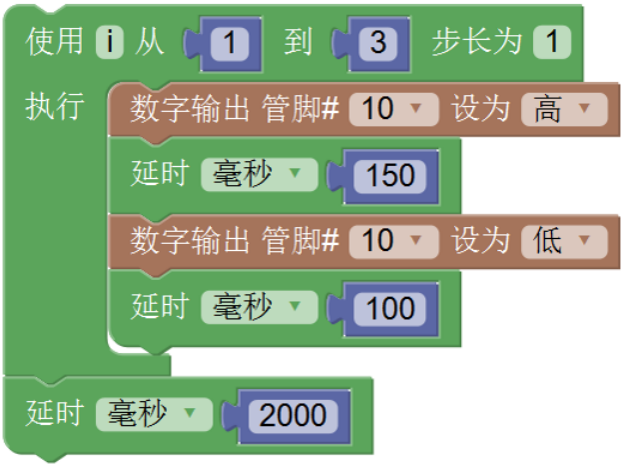
1. 硬件连接：

将 LED 与主控板 10 号管脚相连。注意插线时的颜色对应。



2. 程序编写

观察摩尔斯电码 S.O.S 的表示形式（“...---...”），其中长短依次重复出现，可以采用上一任务中的**循环结构**：



这一段程序不是完整的，还不能实现任务要求。它的作用只是表示第一个字母 S。请你在此基础上完成程序，实现任务要求。

知识点小结

元件

1. LED 模块

Mixly 程序模块

1. 数字输出
2. 延时
3. 循环

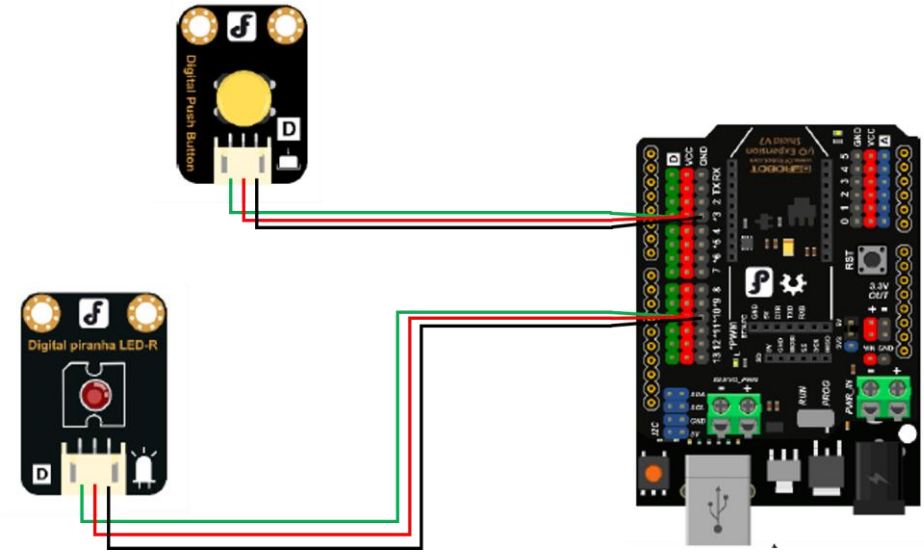
项目三
神奇的按钮

按钮是我们在这套元件中接触的第一个**输入设备**，具有**按下**（高）和**抬起**（低）两种状态。默认状态为**抬起**。生活中的按钮可以说无处不在，遥控器、计算器、手机、电脑等各种电子设备上的按键，都是按钮。

本节任务

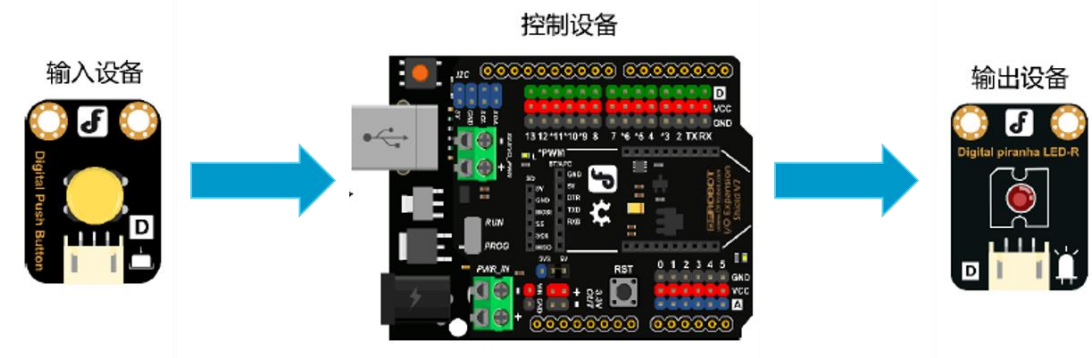
任务 1——让按钮简单工作

1. 实现的功能：按下按钮灯亮，松开按钮灯灭。
2. 硬件连接：按钮→3；LED→10。注意插线时的颜色对应。



3. 程序编写

首先是实现功能的逻辑，如下图：



输入设备向控制设备发送信号，控制设备对其进行处理，并控制输出设备进行相应的输出工作。

当按钮被**按下**时，向 2 号管脚**输入高电平**，此时 10 号管脚应**输出高电平**；
当按钮被**抬起**时，向 2 号管脚**输入低电平**，此时 10 号管脚应**输出低电平**。

程序如下图所示：

数字输入：有两种状态，即“高”和“低”。按钮按下为高，抬起为低。支持数字输入的管脚为：2~13，A0~A5（其实 0 号和 1 号管脚也支持数字输入，但因为负责串口通信功能，一般情况下不使用）。

任务 2——简易延时灯

1. 实现的功能：按下按钮灯亮，3 秒钟后灯灭。
2. 程序编写：

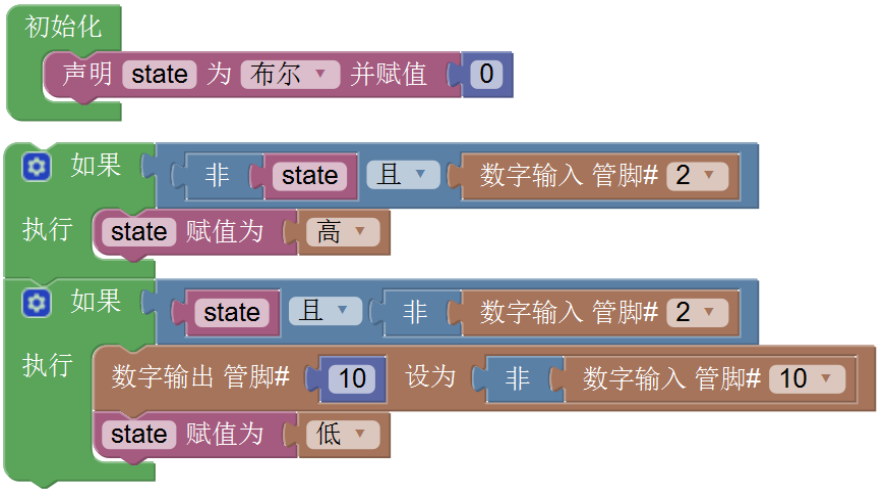
这里我们看到了一个新的结构——**条件结构**。它是如何应用的呢？请看下图：

如果当前状态符合判断条件的要求，判断模块内（红色边框内的）的程序将会被执行。如果不符合，则那部分代码将会被跳过，直接执行接下来黑色框中的程序。

任务 3——使用按钮模拟开关（1）

1. 实现的功能：每次按下，LED 灯切换亮灭。

2. 程序编写：

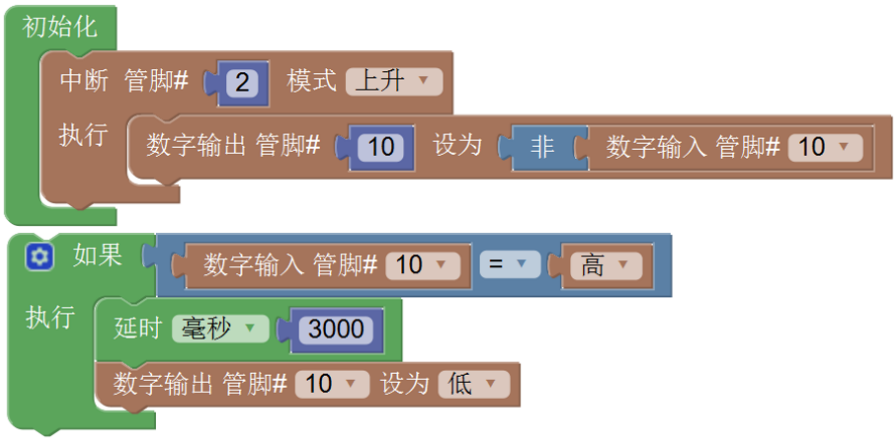


这段程序中，我们使用了“初始化”模块。相比于它下面的程序，初始化模块在整个程序运行过程中只执行一次。在初始化模块中，我们使用了定义变量的模块。变量的名称是 state，它是一个布尔变量，即它的值有“高”和“低”两种。以后我们还会见到在初始化没模块中添加其他模块的例子。

任务 4——使用按钮模拟开关（2）

1. 实现的功能：按下开关灯长亮，再按一下 3 秒后灯灭。

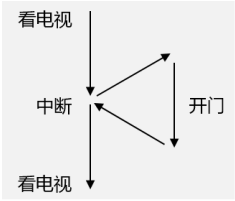
2. 程序编写：



这段程序中，我们使用了一个新的模块——中断。

中断过程是指，在程序顺序执行的过程中，当主控板收到规定的某些数字输入信号时，会立即暂停当前执行的程序，转而去执行中断部分的程序，当中断部分执行完毕，再回到刚才暂停的地方继续执行原来的程序。注意：主控板上只有 2 号和 3 号管脚支持中断功能。

举一个简单的例子：比如你在家看电视，突然门铃响了（家人回来了），那么你不得不停下看电视先去开门，之后你又可以继续看电视啦！在整个过程中接电话就是一个中断过程，门铃响就是中断的标志，即触发中断的条件。



上面的程序中，当人按下按钮时，便会暂停当前程序进入中断，执行中断中的模块。

知识点小结

元件

1. 按钮

Mixly 块

1. 数字输入
2. 条件判断
3. 中断

项目四

简易入侵检测仪

元件介绍——红外接近开关

红外接近开关是一种**数字输入设备**。具有**未入侵（高）**和**入侵（低）**两种状态。默认状态为**未入侵（高）**。

红外接近开关：

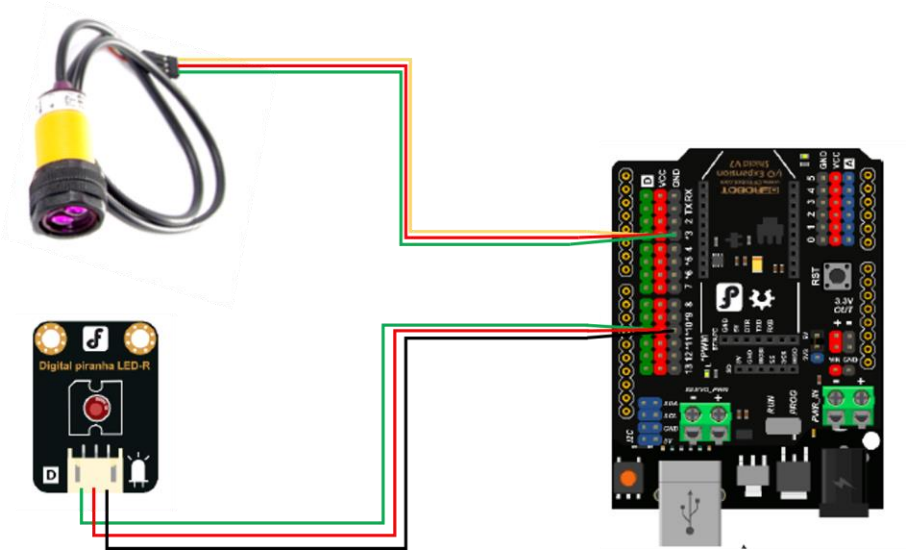
- **红外线属于一种电磁射线。可见光波长是 380nm-780nm，红外线波长为 780nm-1mm，而红外接近开关使用的是接近可见光波长的近红外线。**
- **利用被检测物体对红外光束的遮光（位于红外发射器另一侧的接收器接收不到红外线）或反射（位于红外发射器同侧的接收器接收到物体反射回来的红外线），检测物体的有无。**
- **红外接近开关对所有能反射光线的物体均可检测。**
- 使用数字输入管脚，读取输入口数值 1/0 代表有无接收红外信号。

本节任务

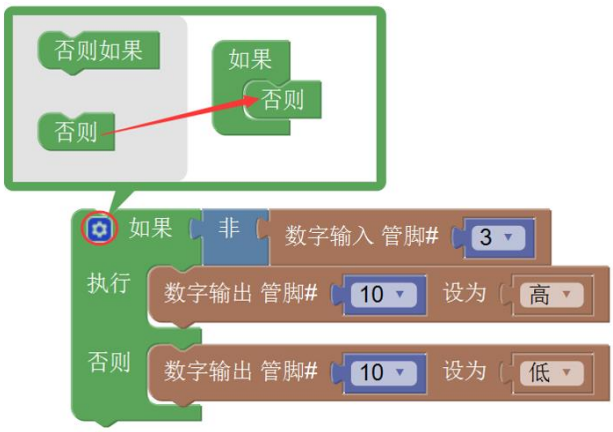
任务 1——简单入侵检测仪

1. 实现的功能：人近灯亮，人走灯灭

2. 硬件连接：红外入侵检测仪→3；LED→10。注意红外接近开关的插线：黄接绿，红接红，绿接黑



3. 程序编写



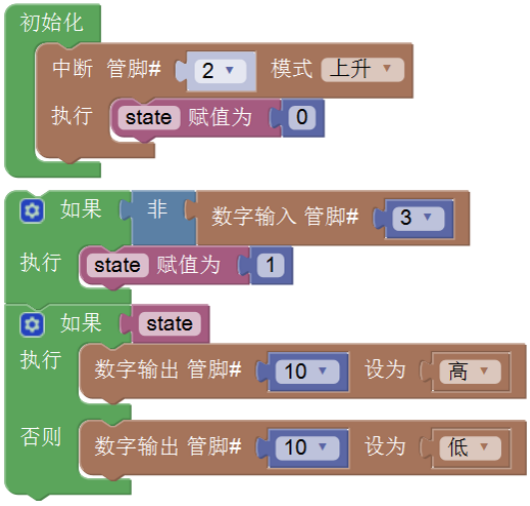
点击图中的蓝色的设置按钮，将里面左侧的模块拖动至右侧，就可看到下方模块的结构发生变化。再次点击蓝色按钮关闭设置。

任务 2——记录入侵

1. 实现的功能：

2. 硬件连接：红外入侵检测仪→3；LED→10；按钮→2。注意红外接近开关的插线：黄接绿，红接红，绿接黑

3. 程序编写



知识点小结

元件

1. 红外接近开关

Mixly 模块

1. 数字输入
2. 条件判断
3. 中断

项目五

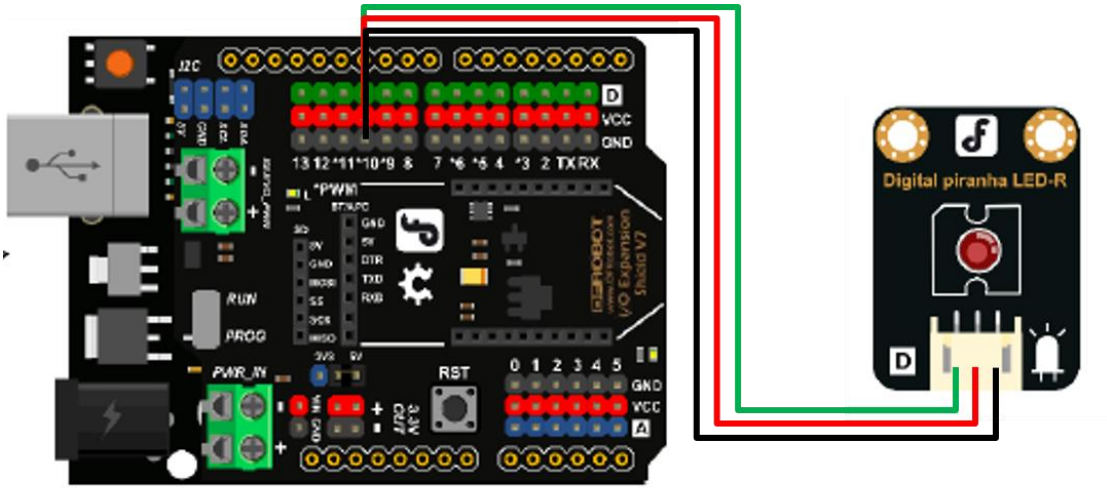
可调灯

灯光在微电脑控制之下完成由暗到亮再由亮到暗的逐渐变化，感觉像是在呼吸，所以称为**呼吸灯**。呼吸灯广泛应用于手机、无线路由器之上，如你的手机里面有未处理的通知，比如说未接来电，未查收的短信等等，呼吸灯就会由暗到亮的变化，像呼吸一样那么有节奏，起到一个通知提醒的作用。

本节任务

任务 1——简易呼吸灯

1. 实现的功能：灯的亮度逐渐变化。
2. 硬件连接：LED→10。注意插线时的颜色对应。



3. 程序编写：

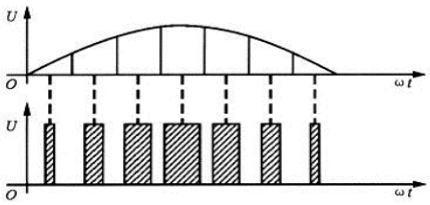
这段程序中我们接触到了两个新的模块——**函数**以及**模拟输出**。

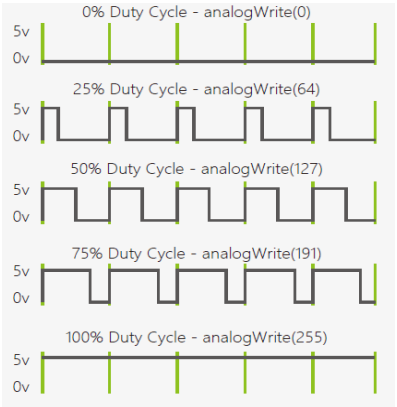
函数：我们最初见到函数是在数学当中。y=f(x)是函数的一种一般形式，它接受变量 x 的值，经过对应法则 f 的处理，向人返回结果值 y。Mixly 程序中的函数，可以类比理解。程

序中的 fadeOn 函数接受了变量 time 的值，对其进行了操作，并向程序的其余部分返回一个值。

有区别的是，这里的变量 time，我们称之为“参数”，函数返回的值，是“空值”，因为这个函数的目的，不在于返回一个数值，而在于对于硬件进行输出操作。另外，函数的内部，也可以定义变量（如图中的 value），这个变量只在函数执行的时候存在，一旦函数执行结束，value 也就不存在了。函数的一次执行叫做函数的调用，在一个函数中，可以调用其他函数，甚至可以调用自己。

模拟输出：脉宽调制（PWM：Pulse Width Modulation）输出：它是一种对模拟信号电平进行数字编码的方法，简单来说就是通过一个时钟周期内高低电平的不同占空比来表征模拟信号，如右图就是一个具体的编码样例。



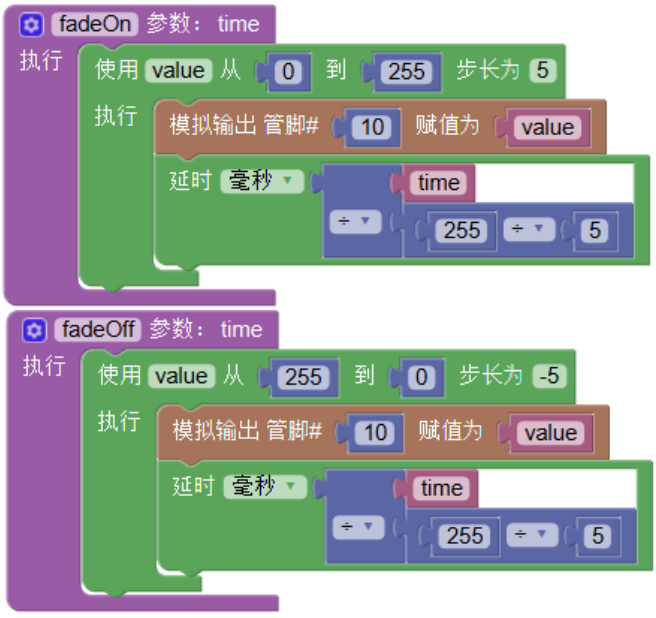


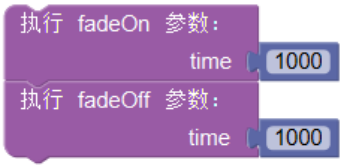
Arduino 使用 analogWrite(int value)输出 PWM 信号，其中的 value 取值范围是 0-255，效果如右图所示。

Arduino 主控板只有有限个 GPIO 管脚支持 PWM。

观察一下 Arduino 板，查看数字引脚，你会发现其中 6 个引脚（3、5、6、9、10、11）旁标有“~”，这些引脚不同于其他引脚，因为它们可以输出 PWM 信号。

不难看出，上面这个函数的功能是让 LED 灯逐渐变亮。要实现任务要求，还需要一个让 LED 逐渐变暗的函数。完整的程序如下图。



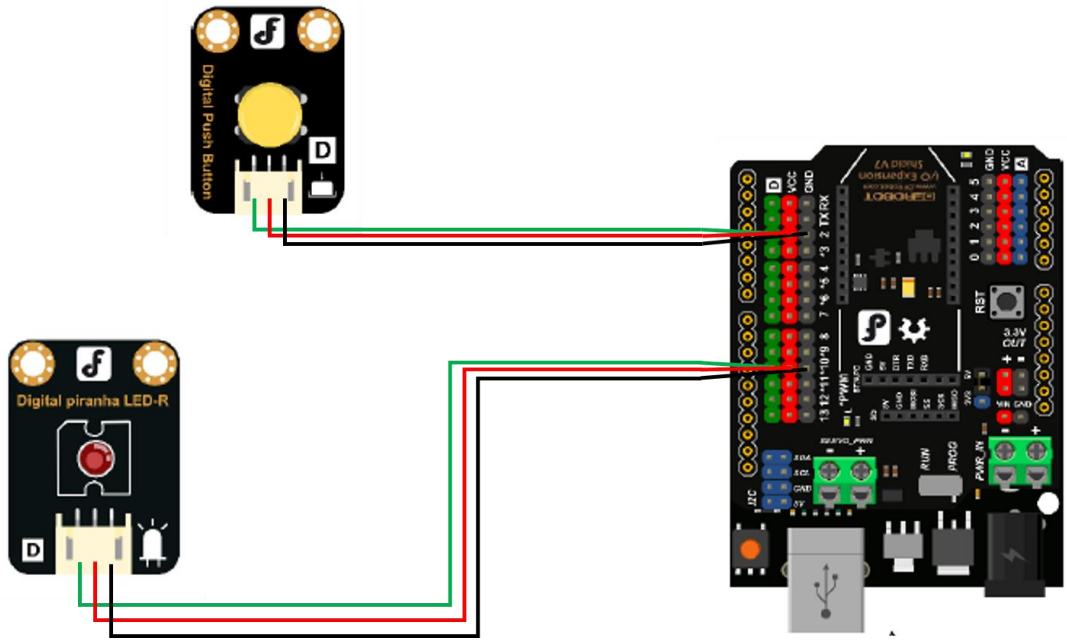


左侧两个紫色块内的程序需要自行编写，它们的功能是定义两个函数。程序的主体是右侧的两个模块，它们是对左侧的函数的调用。

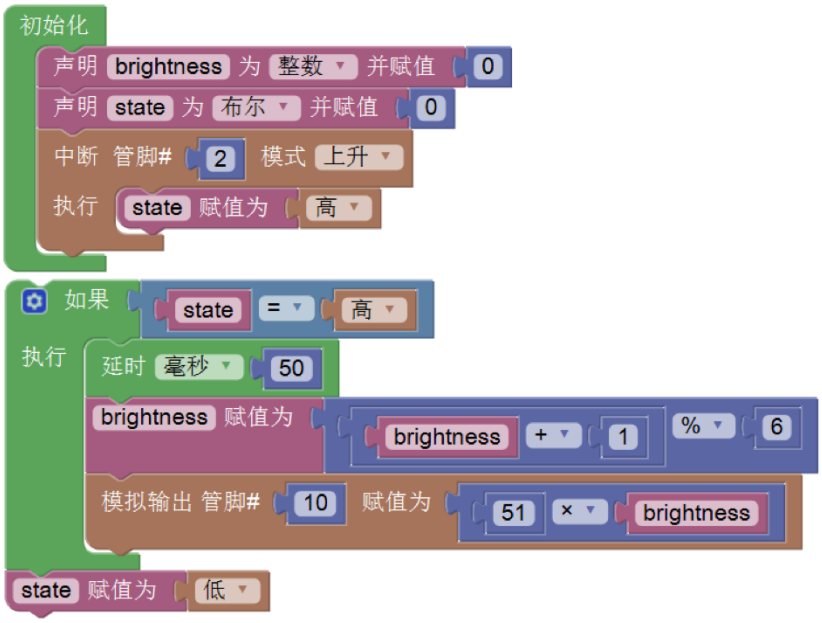
任务 2——制作 5 档可调灯

1. 实现的功能：通过按钮，让灯的亮度发生变化。

2. 硬件连接：按钮→2；LED→10。注意插线时的颜色对应。



3. 程序编写：



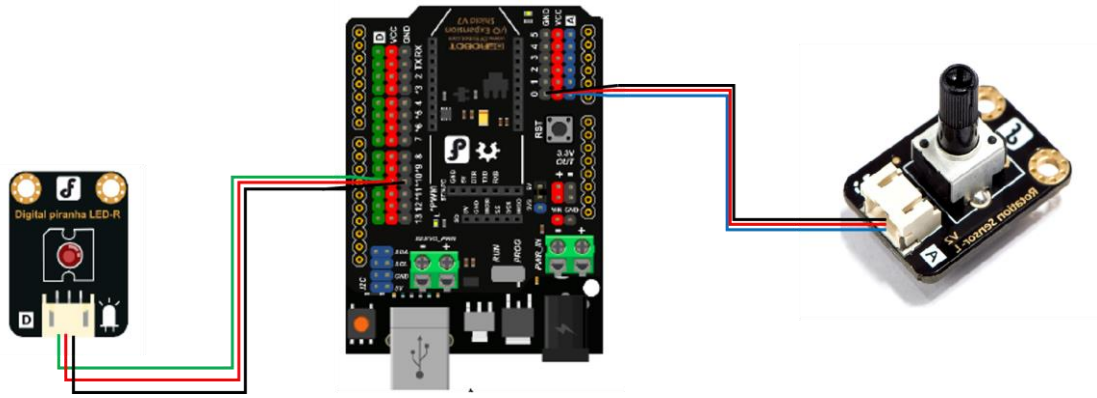
程序中“brightness 赋值为”一句里，对变量进行了运算操作：(brightness+1) %6。

这里的“%”是一个运算符号，叫做“取余”，这个式子的结果是 brightness+1 后的数除以 6 得到的余数。

任务 3——制作旋钮可调灯

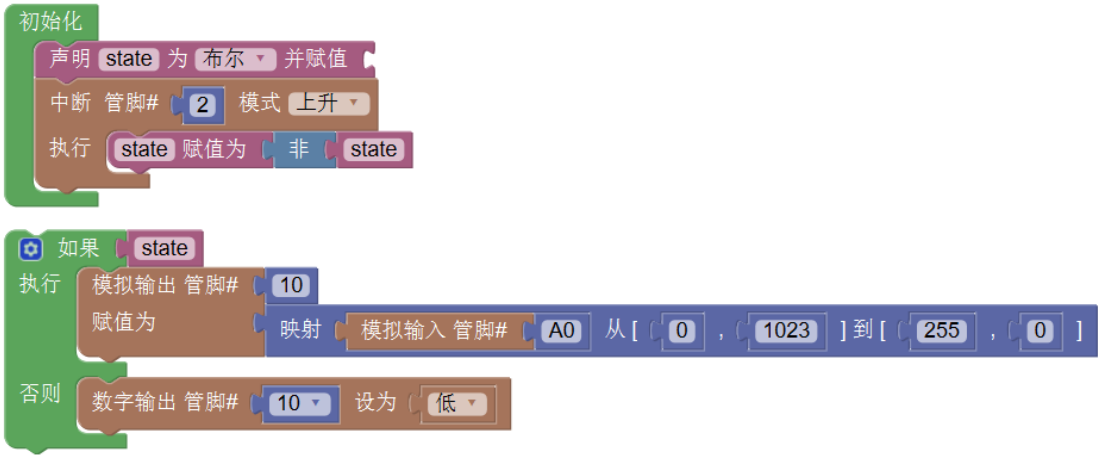
这里，我们用到了一个新的元件——**模拟角度电位器**，也叫“滑动变阻器”。通过调节旋钮，可以改变它接入电路的阻值大小。将其连到主控板支持模拟输入的接口上，就可以把阻值作为模拟信号输入到主控板上。主控板根据输入值的大小，确定输出的值（在这里，输入值大，输出值也大；也可能另外一些程序希望输出值随着输入值变大而减小。）

1. 硬件连接：模拟角度电位器→A0；LED→10。注意插线时的颜色对应。



需要注意的是，主控板支持的模拟输入信号的大小范围是 0~1023。然而，模拟输出大小是 0~255。因此，模拟输入的数值，不能直接进行模拟输出，我们需要一种办法，能够把 0~1023 内的数，按比例缩小，转化成 0~255 之间的数，再模拟输出。方法如下：

2. 整个程序：



知识点小结

元件

1. 模拟角度电位器

Mixly 块

1. 模拟输入
2. 自定义函数
3. 映射

项目六

智能灯

本节任务

任务 1——声控灯

顾名思义，我们需要通过声音代替按钮形式的开关。如何识别声音信号呢？我们需要使用一个新的原件——**模拟声音传感器**。

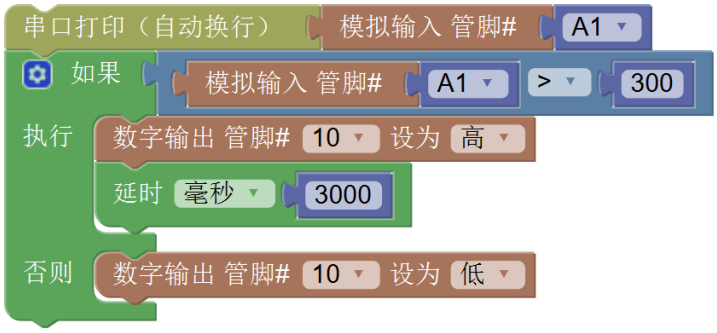
模拟声音传感器可以将声音的响度转化成模拟信号。在 Arduino 主控板上，仍然是输入 0~1023 的数值。

1. 实现的功能：有响声灯亮并延时一段时间。

2. 硬件连接：模拟声音传感器→A0；LED→10。注意插线时的颜色对应。

【配图】

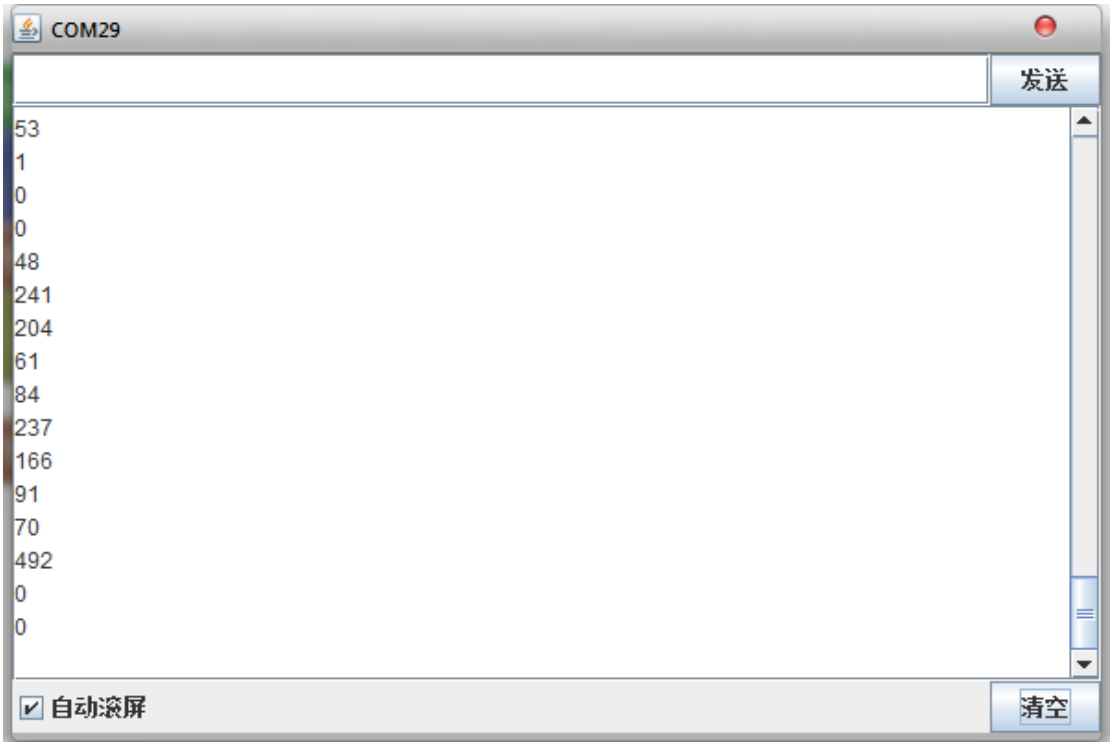
3. 程序编写



我们看到，程序中使用了一个新的模块——**串口打印**

模拟输入的信号是通过串口传给 Arduino 主控板的，使用“串口打印”可以把当前通过串口的数据显示出来。显示在哪呢？在串口监视器里。

单击串口监视器，会弹出一个新窗口，这个窗口中显示上传的数据：



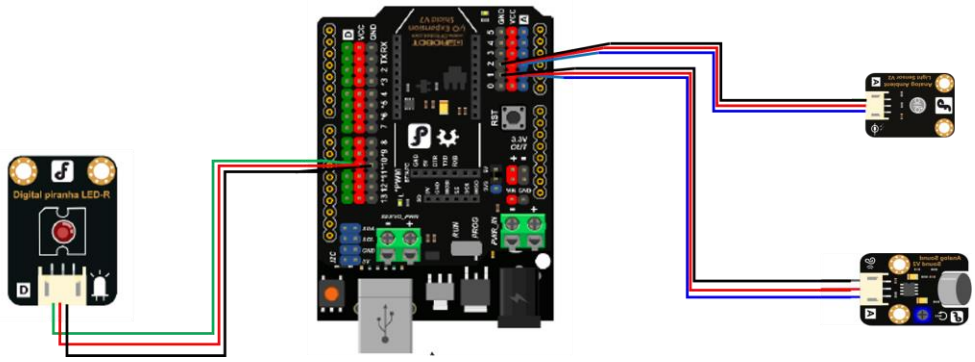
任务 2——制作楼道灯

生活中我们是否用到了声控灯呢？对声控灯比较常见的应用，就是楼道灯。不过，在白天，楼道在阳光下并不昏暗的时候，声音再大，楼道灯也是不会亮的。而任务 1 中的声控灯则不论周围环境的亮度如何，只要声音超过了设定值便会亮灯。这是因为，控制楼道灯的开关，不只有模拟声音传感器，还有**模拟光线传感器**。

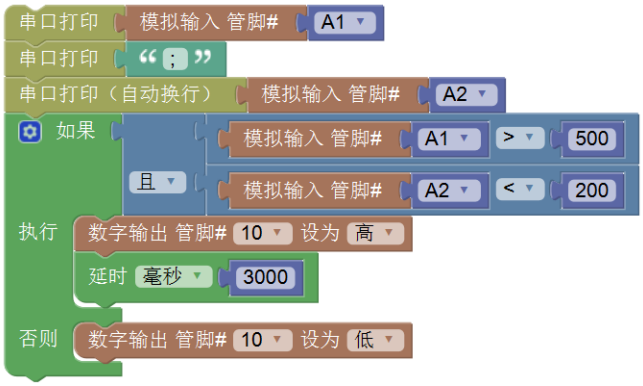
模拟光线传感器可以将周围的亮度转化为模拟信号，输入到 Arduino 主控板上。

1. 实现的功能：当亮度暗且有声音时，灯才亮

2. 硬件连接：模拟声音传感器→A0；模拟光线传感器→A1；LED→10。注意插线时的颜色对应。



3. 程序编写



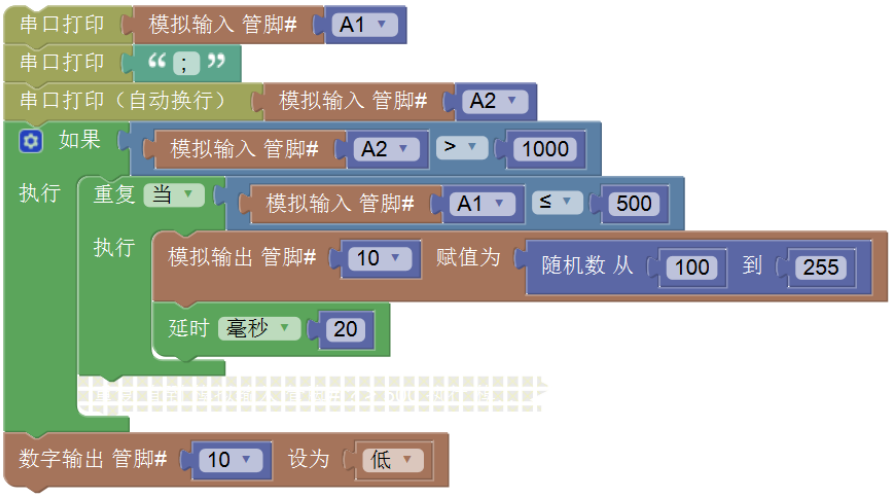
任务 3——制作电子蜡烛

1. 实现的功能：当亮度暗时，蜡烛“点亮”，火苗闪烁，有吹蜡烛的声音的时候，蜡烛熄灭。

2. 硬件连接：与任务 2 相同。

3. 程序编写：

使用数值菜单中“随机数”模块，可以模拟真实蜡烛火苗闪动的效果。



项目七

创意门铃

说到门铃，都会想起自家门铃“叮咚”的响声。制作一个门铃，需要一个能够发出声音的电子原件——**蜂鸣器**。

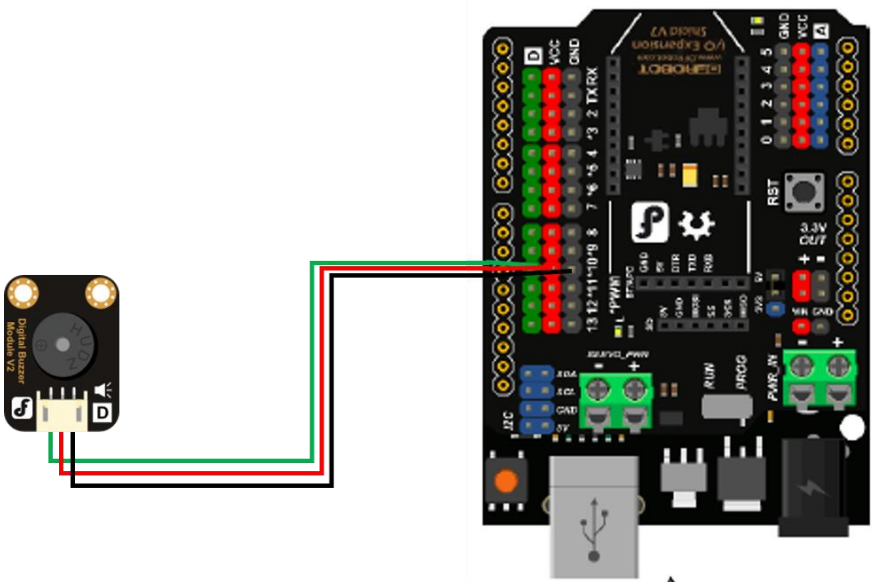
蜂鸣器可以**依照设定的频率发出不同音高的声音**。日常生活中蜂鸣器的应用非常广泛，比如刷公交卡时滴滴的提示音，警车，救护车等发出的警报声。

这个项目中，我们也要做门铃。与普通门铃不同的是，这个门铃可以按照你的想法来工作，比如门铃可以播放一段音乐，或者只有当门外的人连按两次时，门内的人才会被告知。你还可以再添加一些新奇的想法，快来试试吧！

本节任务

任务 1——让蜂鸣器发声

1. 硬件连接：蜂鸣器→8。注意插线时的颜色对应。



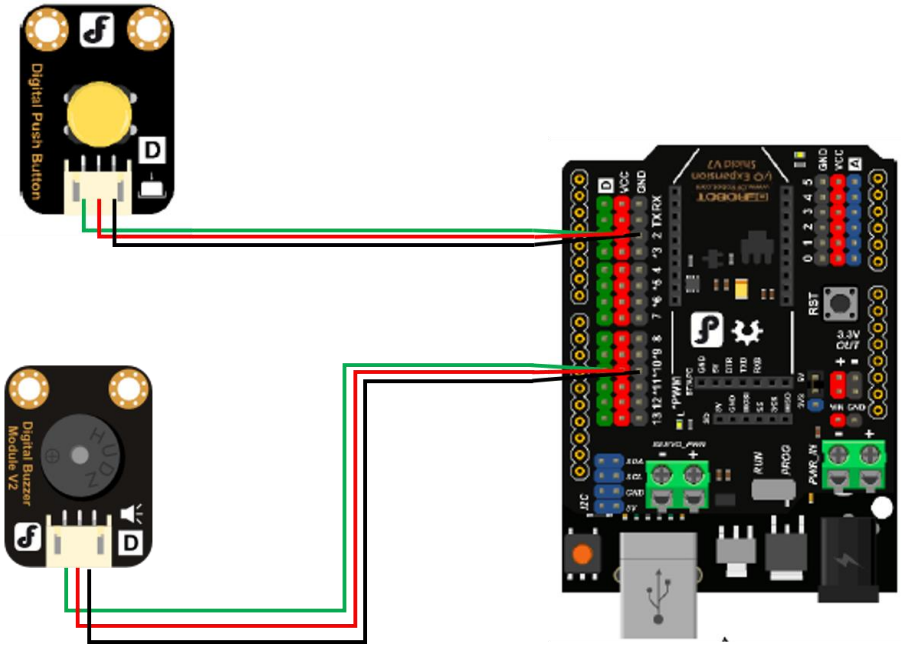
2. 程序编写

注意在播放完声音后加上结束声音的程序。

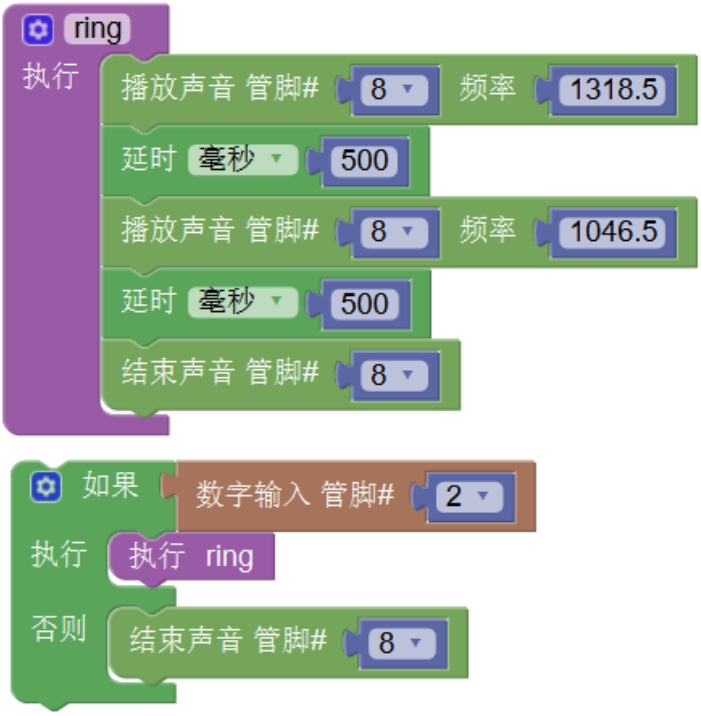
任务 2——制作简易门铃

1. 实现的功能：按下按钮，蜂鸣器发出“叮咚”声。

2. 硬件连接：按钮→3；蜂鸣器→8。注意插线时的颜色对应。



3. 程序编写：



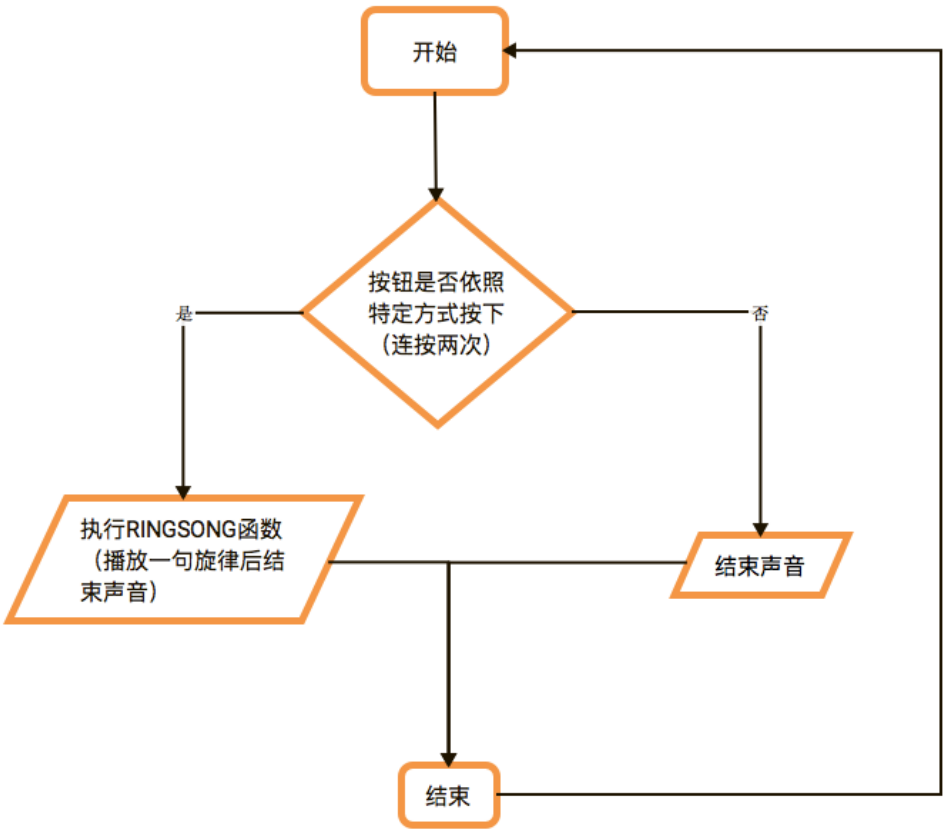
课后作业

通过前六个项目的学习，你大概已经熟悉 Arduino 的硬件连接以及 Mixly 的模块化编程了。从本节开始，我们将增加课后作业一项，给出任务要求和思路，但是不提供连接方式和程序。程序需要由你自己来完成了，相信你能够做到！

制作一个音乐门铃：每次按下按钮，播放一句旋律。

制作一个暗号门铃：用户依照特定方式按按钮（如一秒内按两次），门铃才会响。

1. 思路解析



- **如何检测按键是否按特定方式按下（以 1 秒内 2 次按下为例）？**

为了检测一秒内按键是否正好按下两次，不是数秒内，也不是一次或三次。在准备工作上，我们需要有一个 millis 函数来计时一秒，还需要有一个变量 times 来计次数。此外，在编程思路上，我们还需要考虑什么时候开始计时，不妨设为第一次检测到按键按下时；什么时候结束计时，即一秒之后；在这段时间内，我们需要反复检测（循环检测）按

键是否按下并记录次数。若一秒内正好按下两次，则符合要求，可进行相应反馈；否则，反复进行上述检测。

- **如何使蜂鸣器播放一句旋律？**

如果要播放一句较长的旋律，将每个音写成“叮咚”的播放形式无疑是很繁琐的。不难发现，我们要播放的音只有七个，只是重复和不同的组合而已，所以，怎样能使编程简化呢？在上面的程序中，我们每次都要获取 tonelist 中的某一项，而我们不妨将这些项写成一个 xxx 数组，用一个循环函数，每次获取 xxx 数组中的某一项对应的声音并播放，这样代码就简洁了许多。

项目八
小小作曲家

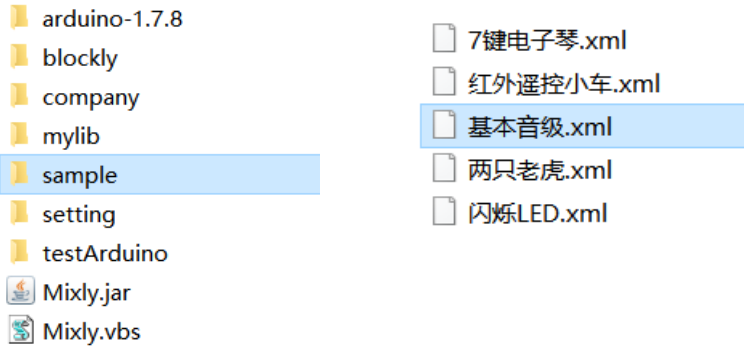
项目七的末尾，我们留了一项课后作业——让蜂鸣器播放一段旋律。不知道你是否实现了？本节中，我们将向你介绍一种用 Mixly 播放音乐的方法。这一节中，你能够了解到一些有关音乐的基本知识，以及它们如何与 Arduino 和 Mixly 结合，并试着自己编一段小旋律。

本节任务

任务 1——简单音阶

1. 实现的功能：播放一组音阶
2. 硬件连接：蜂鸣器→2。
3. 程序编写

这个任务是一个参考任务，我们已经写好了一段程序。请你打开 Mixly，然后点击“打开”按钮，在 Mixly 所在文件夹中找到 sample 文件夹，选中“基本音级.xml”，打开。



打开后可看到如下程序：

这段程序实现了一个简单的功能：按顺序播放一组音阶（do, re, mi, fa, sol, la, si）。程序的开头定义了一个含有 7 个元素的数组，每一个元素对应着一个音的频率。

关于音乐的基本知识：一般来讲，一个音有以下要素：音高、音色、响度。在乐曲中，时值也是一个重要的要素。

声音与物体的震动：声音是由物体的震动产生的。

音高与频率：一个音的音高，由物体振动的频率决定。振动频率指物体振动的快慢，频率越高，物体振动越快，音高越高。

音色与材质：不同的物体的材质不同，振动时产生的声音的“色彩”不相同。

响度与振幅：一个音色响度，由物体振动的幅度决定。物体振动幅度越大，响度越大。

音名：前面已经知道，音调的高低是由频率决定的。对于音乐来说，人们已经发现，用一些特定的频率的音，演奏出来的曲子比较悦耳。人们也将这些频率用英文字母代替，方便记忆。这个英文字母，就是它代表的那个音的音名。（例如，规定 A=440Hz，那么，A 就是一个音名）音名是唯一的，任何情况下，一个音名只能代表同一个频率。（如 A 永远代表 440Hz）。

唱名：音乐课上，你肯定听到过 do re mi fa sol la si 吧？这些就是唱名。因为唱出来比较简单。” do ” 可以对应不同的频率，可以是 440Hz，也可以是 880Hz，也可以是……。所以可以看出音名和唱名的区别——一个音名值对应一个频率，一个唱名可以对应多个频率。

时值：一个音的长短

任务 2——简单乐曲

1. 实现的功能：播放简单乐曲
2. 硬件连接：蜂鸣器→10。
3. 程序编写

同样，为了方便你们更好地了解音乐的基本知识，我们写好了一段程序，找到上一个任务中的“sample”文件夹，打开其中的“两只老虎.xml”文件。可看到如下程序：

首先来看我们定义的 4 个数组：

- “tonelist”数组与任务 1 中的数组相同，记录了从 do 到 si 的频率；

- “musiclist”数组就像乐谱一样，按顺序存储了其中的“唱名”；
- “highlist”表示这个音符所在的八度（0 代表当前音高，1 代表高 8 度，等等）；
- “rhythmlist”数组表示每一个音符的时值，即这个音符的长短。

接下来是程序段：

- 程序的整体是一个循环，每一次循环代表一个音符。
- 那一行很长的播放声音的程序，是要算出蜂鸣器播放的声音频率。方法是：读出“musiclist”数组中的编号，再根据这个编号找到“tonelist”数组中对应的频率，得到这个频率以后，看是否需要调高八度或调低八度（当一个音变成高八度时，频率变成 2 倍，低八度时频率变成二分之一）。
- 延时程序块则是获取了每一个音符对应的时值。

任务 3——自编旋律

任务二中介绍了编写一段旋律的方法——设定每一个音符的唱名、八度、时间，并将它们按顺序排列在数组当中。接下来，就要由你自己来编写一段美丽的旋律啦！

1. 实现的功能：播放自创旋律
2. 硬件连接：蜂鸣器→10
3. 程序编写：请参考“两只老虎.xml”文件

项目九

噪音计

生活中我们可以看到一些仪器上指针——时钟、电压表、电流表、汽车上的转速表……Arduino 套件中也有一个元件可以模拟指针，它就是**舵机**。

- **舵机是由直流电机、减速齿轮组、传感器和控制电路组成的一套自动控制系统**
- **通过发送信号，指定输出轴旋转角度**
- **套件中舵机最大旋转角度：180°**

这是 Mixly 软件中控制舵机的程序块：

本节任务

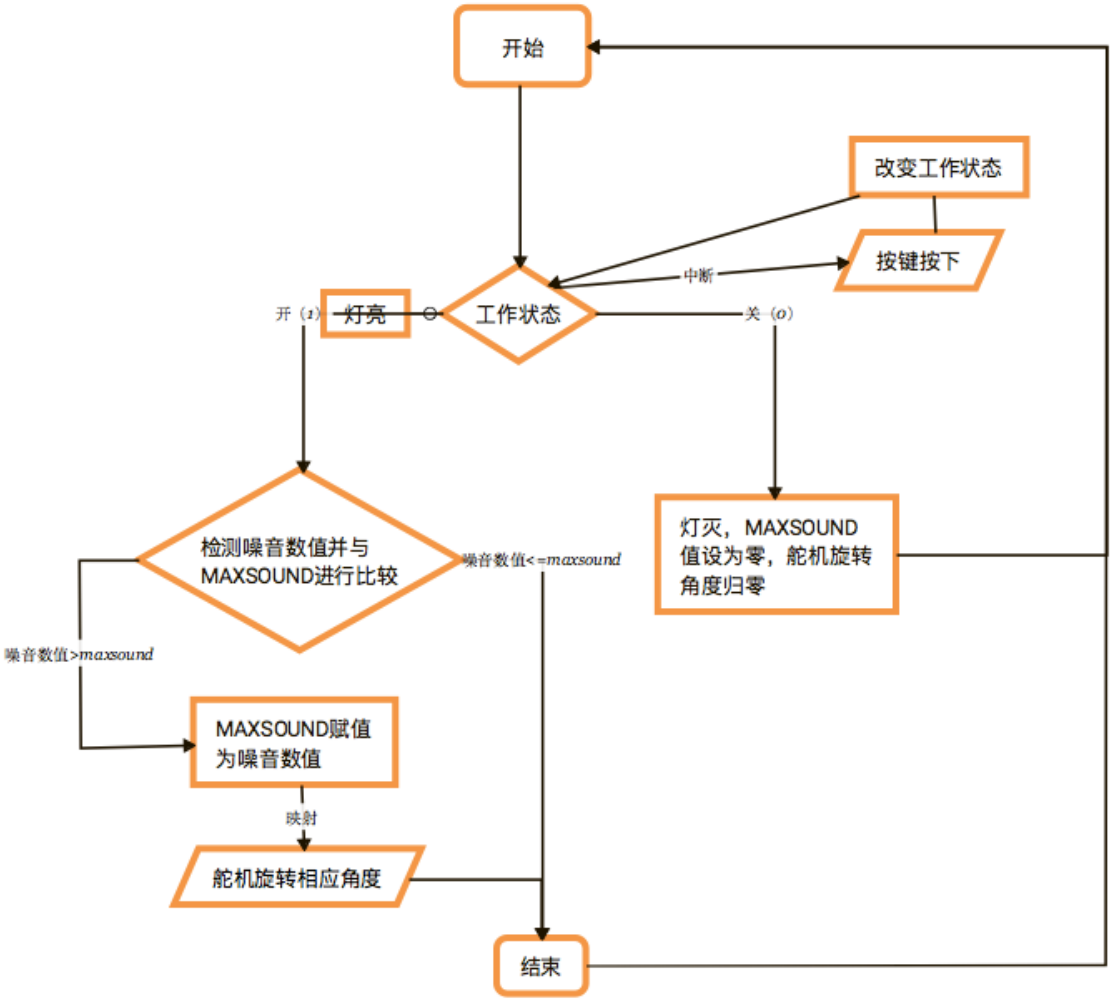
任务 1——制作一个噪音计

1. 实现的功能：根据噪音的大小，使舵机转动到不同位置。
2. 硬件连接：舵机→9；模拟声音传感器→A1。注意插线时的颜色对应。
3. 程序编写：

任务 2——探测最大噪音

1. 实现的功能：探测一段时间内噪音的最大值。用户按下按键后开始工作，只记录最大值。再次按下按键停止记录，并回到初始位置。

2. 思路解析

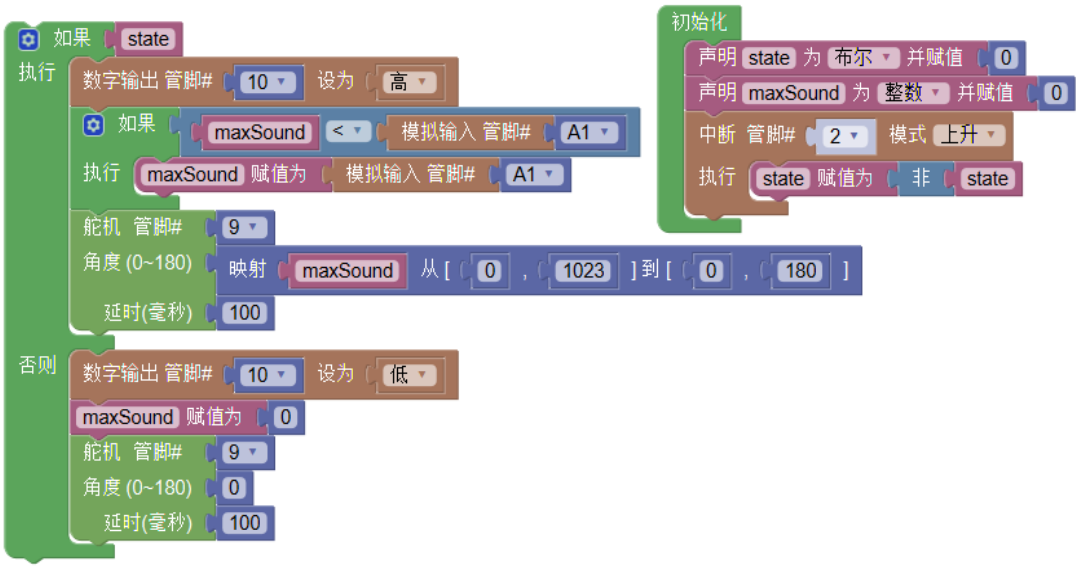


如何使按键随时改变工作状态——中断函数

在这个程序中，我们主要需要用一个 state 变量来判断工作状态，若 state 值为 1（即开）各部分按要求工作，若 state 值为 0（即关）则所有部分归零；其次，在工作过程中我们还需随时检测按键有没有被按下以改变 state 的值；噪音计和舵机的工作状态与按键状态是两个可以独立运行的部分（没有什么明显的逻辑关系语句可以把两个部分建立联系），因此，我们考虑用中断函数，即在工作或停止工作过程中随时检测按键状态（在用按键控制复杂的元件的开关状态时经常使用中断函数）。

3. 硬件连接：舵机→9；模拟声音传感器→A1。注意插线时的颜色对应。

4. 程序编写：



项目十

遥控灯

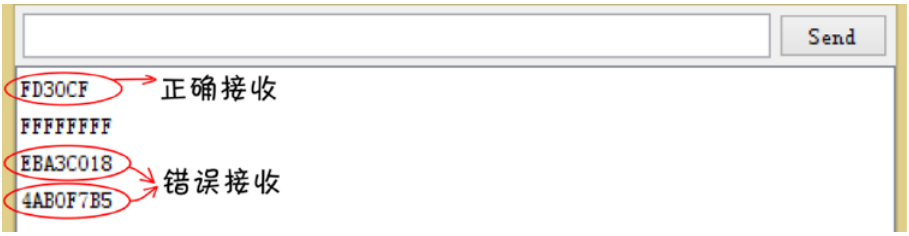
想象自己躺在床上，该睡觉了，灯的开关却在几米外的门口，你不愿离开温暖的被窝，在寒冷中跋涉漫长的几米去关灯。你需要一个遥控器。

红外遥控器（发出和接收红外光的电子器件）：

- **现实世界的大多数遥控器都是红外的，如电视机遥控器，机顶盒遥控器等。**
- **任何一个遥控系统都由发射器和接收器两部分组成。**

- **每个按钮都有一个特定的 16 进制代码，都以 FD 开头，在接收过程中有可能出错。**
- **读取按键对应的代码值：**

- **编译、上传后，打开“串口监视器”**



本节任务

任务 1——制作一个红外遥控灯

1. 实现的功能：按一下开关键控制灯的亮灭。
2. 硬件连接

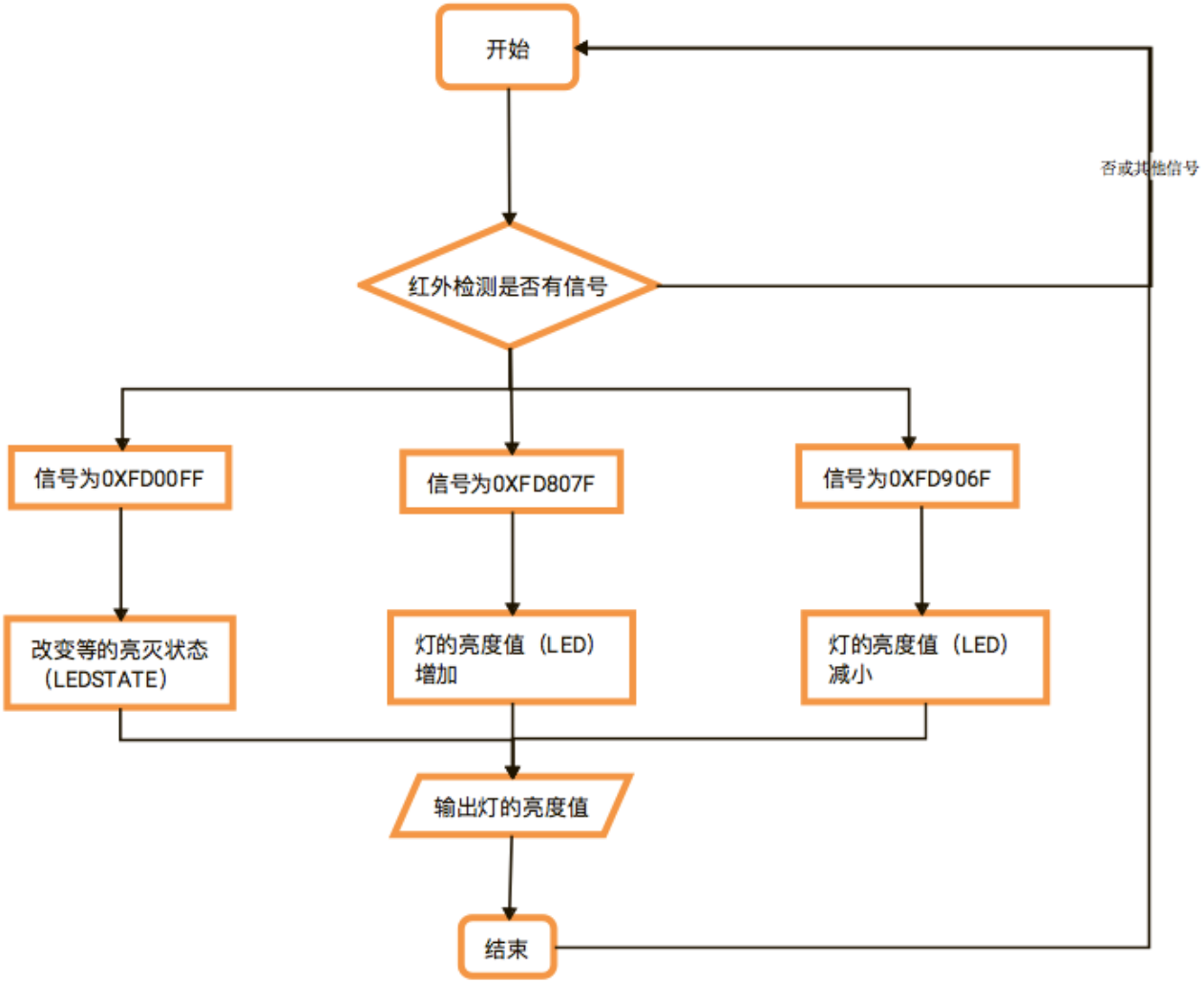
- **取出一个 LED 模块，用连接线将其与 10 号管脚对应的三个管脚相连（注意插线时颜色的对应）**
- **取出一个红外遥控模块，用连接线将其与 5 号管脚对应的三个管脚相连（注意插线时颜色的对应）**

3. 程序编写：

课后作业

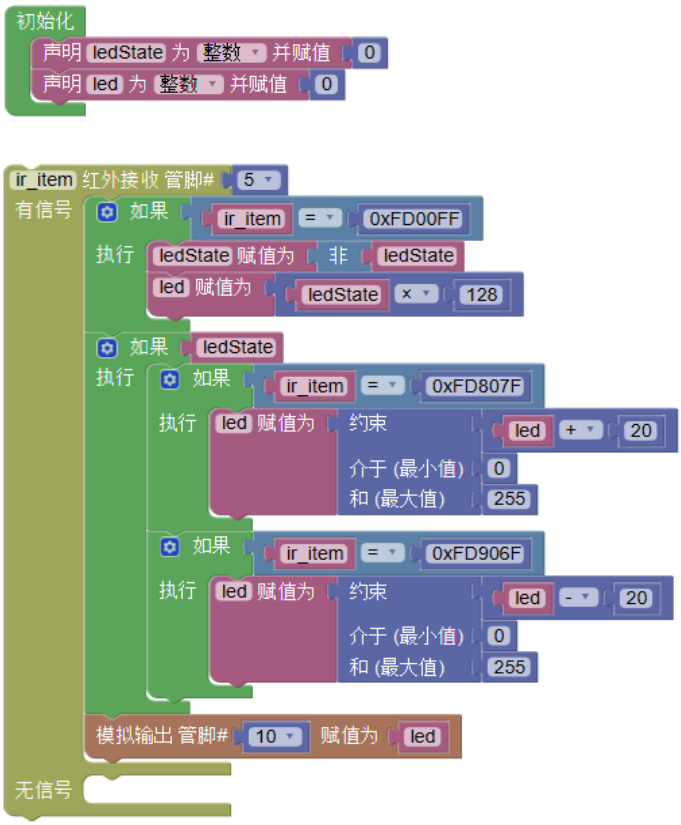
1. 制作一个红外调光器：使用“+、-”对应光值的明暗变化。遥控多个 LED 灯（编程提示为一个）。切换不同的闪灯模式。等等.....

思路解析



如何使用加减按键控制 led 的亮度？

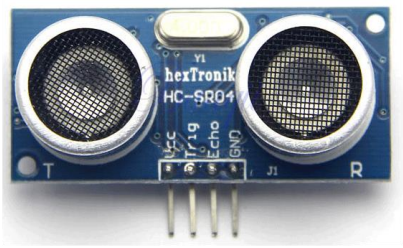
要用加减控制 led 灯的亮度，首先，我们能确定 led 灯需要使用模拟输出，其数值范围是 0-255，因此在控制加减时我们需要用到约束函数是数值范围不会越界。其次，我们需要用一个 ledstate 变量来控制灯的开关状态（为更符合实际使用方便，可以为 led 灯设置一个初始亮度值）；还需要有一个 led 变量来记录操作过程中灯的亮度值。有了这些准备之后，我们只需用简单的逻辑关系把程序编写出来就 OK 了。


初始化
声明 ledState 为 整数 并赋值 0
声明 led 为 整数 并赋值 0
ir_item 红外接收 管脚# 5
有信号
如果 ir_item = 0xFD00FF
执行 ledState 赋值为 非 ledState
led 赋值为 ledState x 128
如果 ledState
执行 如果 ir_item = 0xFD807F
执行 led 赋值为 约束 led + 20
介于 (最小值) 0
和 (最大值) 255
如果 ir_item = 0xFD906F
执行 led 赋值为 约束 led - 20
介于 (最小值) 0
和 (最大值) 255
模拟输出 管脚# 10 赋值为 led
无信号

项目十一
另类电子琴

电子琴想必大家都见过，但是，你是否玩过我们即将介绍的另类电子琴呢？这个电子琴里有一个重要的元件——超声波测距仪。

超声波测距仪：



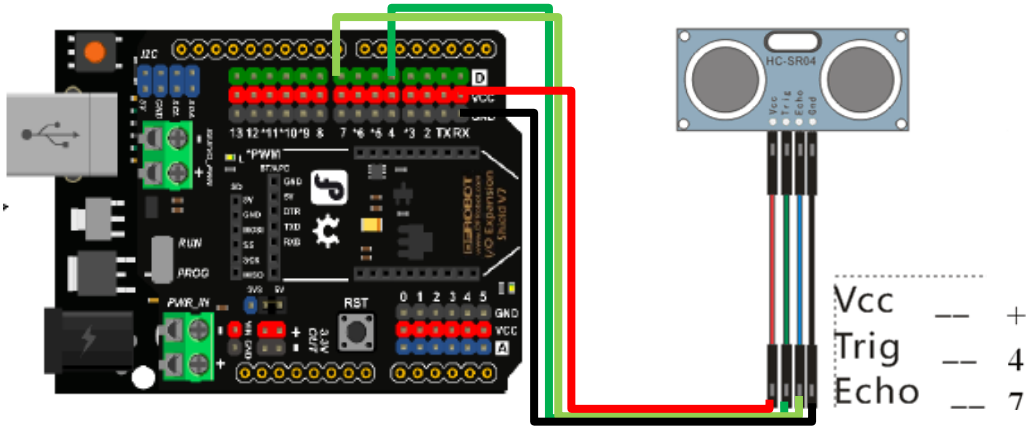
1. 超声波是一种振动频率高于声波的机械波，具有频率高、波长短、绕射现象小，特别是方向性好、能够成为射线而定向传播等特点。
2. 由于超声波指向性强，能量消耗缓慢，在介质中传播的距离较远，因而超声波经常用于距离的测量，如测距仪和物位测量仪等都可以通过超声波来实现。
3. 测距原理：通过超声波发射装置发出超声波，根据接收器接到超声波时的时间差就可以知道距离了。超声波发射器向某一方向发射超声波，在发射时刻的同时开始计时，超声波在空气中传播，途中碰到障碍物就立即返回来，超声波接收器收到反射波就立即停止计时。超声波在空气中的传播速度为 340m/s，根据计时器记录的时间 t，就可以计算出发射点距障碍物的距离(s)，即：s=340t/2。

下面这段程序，能够将测到的距离存储到一个变量里。

本节任务

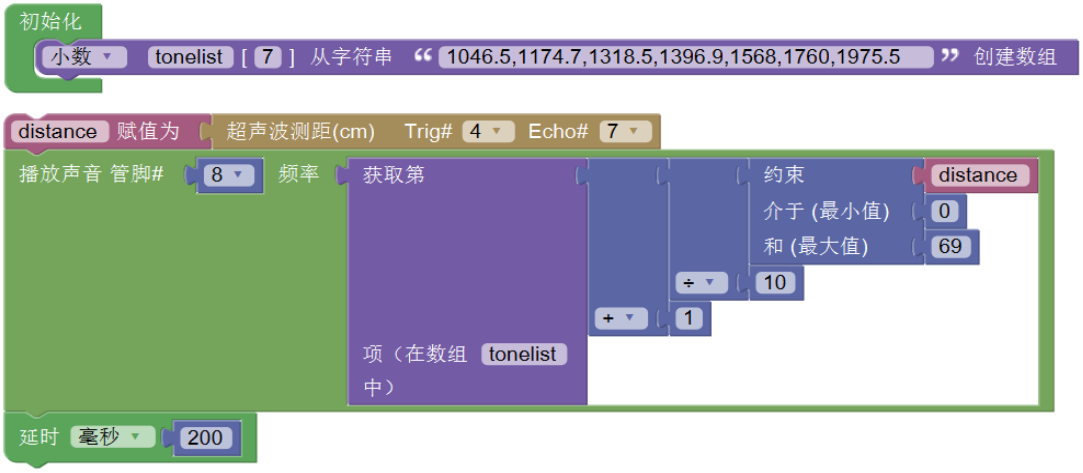
任务 1——制作另类电子琴

1. 实现的功能：根据距离变化发出不同的音
2. 硬件连接：蜂鸣器→8；超声波→（下图）



针脚	接线
VCC	5V
Trig	数字口#4（发射端）
Echo	数字口#7（接收端）
GND	GND

3. Mixly 编程



课后作业

1. 制作一个电子琴：使它发出更多的音

项目十二

超声波测距仪

在上一个电子琴任务中，我们通过距离的变化，控制琴的音高。相应地，我们也能通过音高的变化，大致感受手与元件的距离。不过，我们能不能直接看到距离是多少呢？怎样把测得的数字显示出来呢？这就需要一个液晶显示屏了

液晶显示模块：

- **每行 16 个字符，共 2 行**
- **若出现乱码，可在每次输出之前清屏**

本节任务

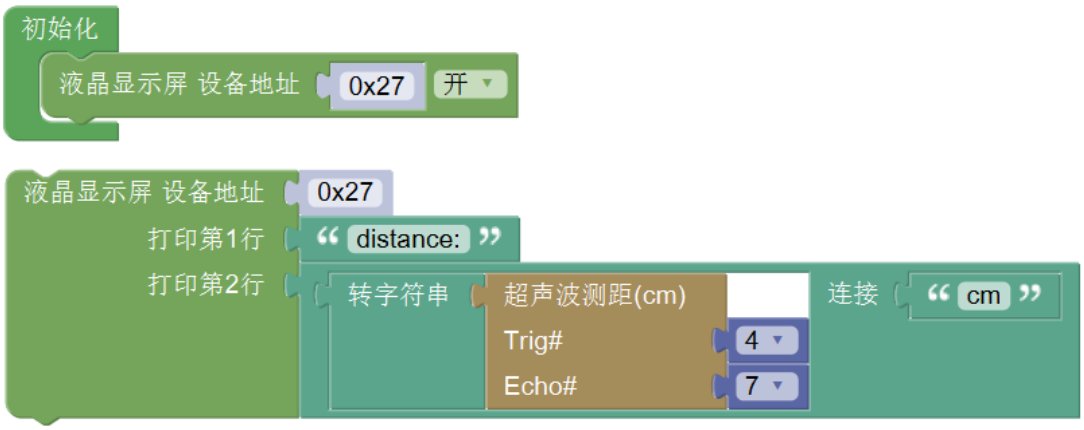
任务 1——制作一个超声波测距仪，并在屏幕上显示距离

1. 硬件连接

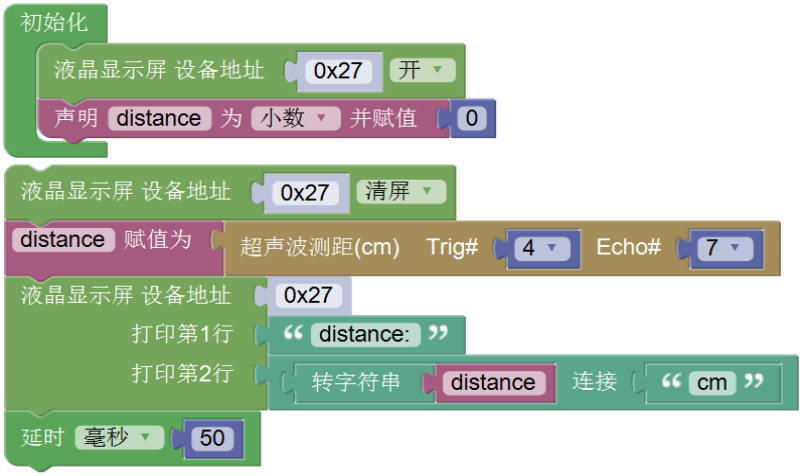
- **取出一个超声波测距模块，用连接线按第十一课的连接方式相连**
- **取出一个液晶显示模块，用连接线按下图所示的连接方式相连**

针脚	接线
VCC	5V
GND	GND
SDA	A4
SCL	A5

2. Mixly 编程



请你观察上面的程序是否能正确输出？如果不能，想一想应如何修改？



项目十三

公园人数计数系统

本节中，我们不再学习新的元件，而是将前面所学的液晶显示屏和红外接近开关联系起来，做一个综合项目，公园人数计数系统。我们经常见到媒体报道某个景区在某个假日接待多少人次，创下历史新高，不知道你有没有好奇过景区到底是如何完成这项统计的呢？我相信，在学习了本节之后，你一定可以找到一种解决办法。本节是一个难度较大的项目，但在你已经掌握了前面的知识和元件的用法后，相信你一定可以通过思考和实践完成这个任务。

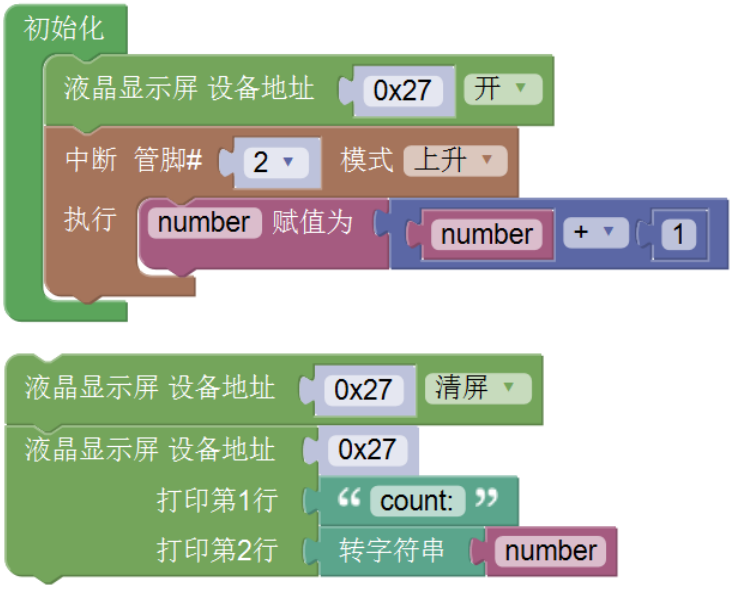
本节任务

任务 1——制作一个公园人数计数系统

1. 实现的功能：每进来一位游客，显示屏上进入公园人次加一
2. 硬件连接

- **取出一个液晶显示模块，依然采用第 11 课的连接方式**
- **取出一个红外接近开关模块，用连接线将其与 2 号管脚对应的三个管脚相连（注意插线时颜色的对应）**

3. 程序编写：



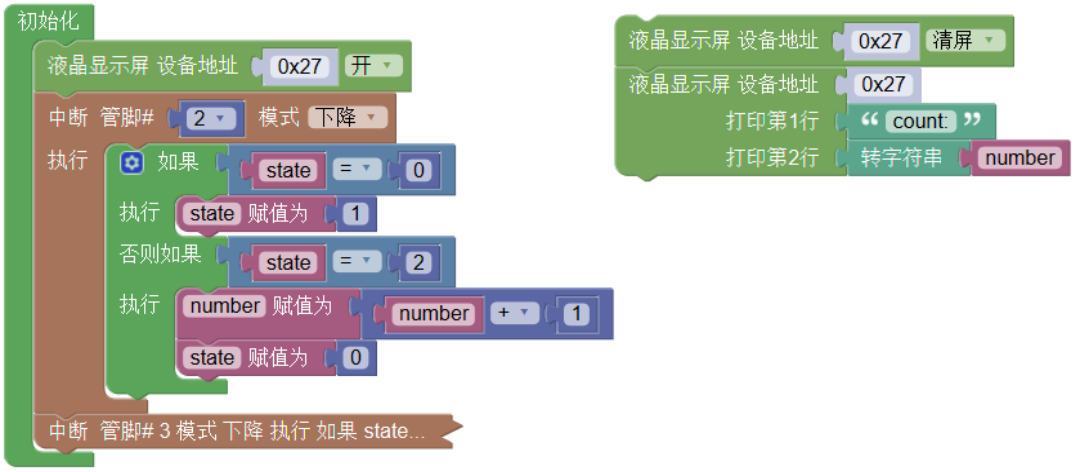
任务 2——制作一个公园人数计数系统

1. 实现的功能：每进来一位游客（2 号管脚触发中断），显示屏上公园人数加一。每出去一位游客（3 号管脚触发中断），显示屏上公园人数减一。

2. 硬件连接

- **取出一个液晶显示模块，依然采用第 11 课的连接方式**
- **取出两个红外接近开关模块，用连接线分别将其与 2 号管脚，3 号管脚对应的三个管脚相连（注意插线时颜色的对应）**

3. 程序编写

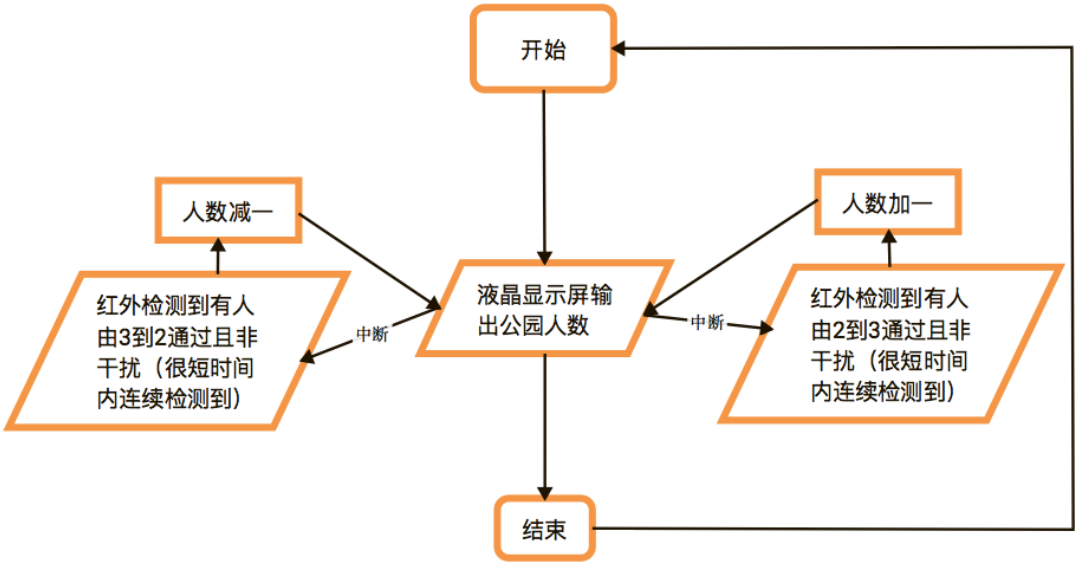


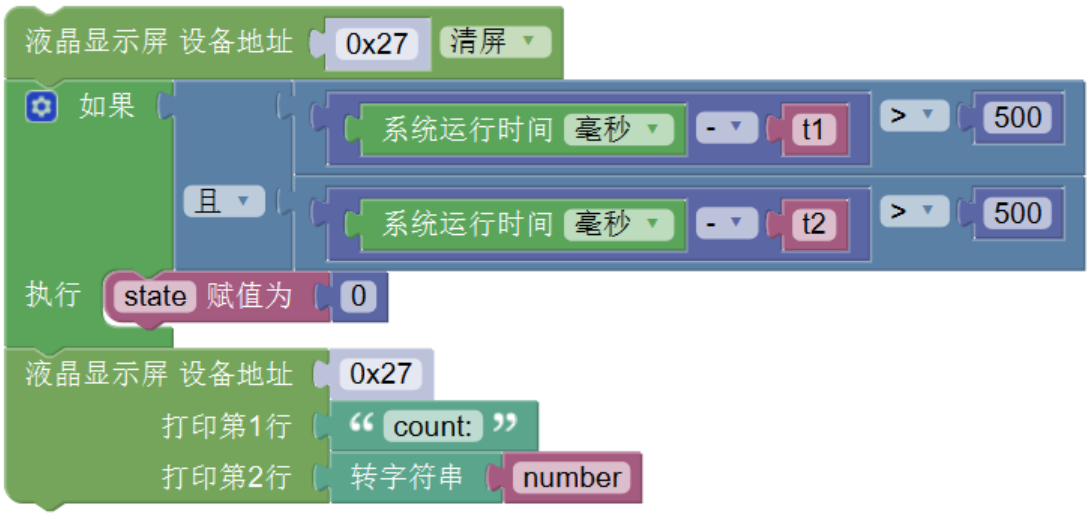
课后作业

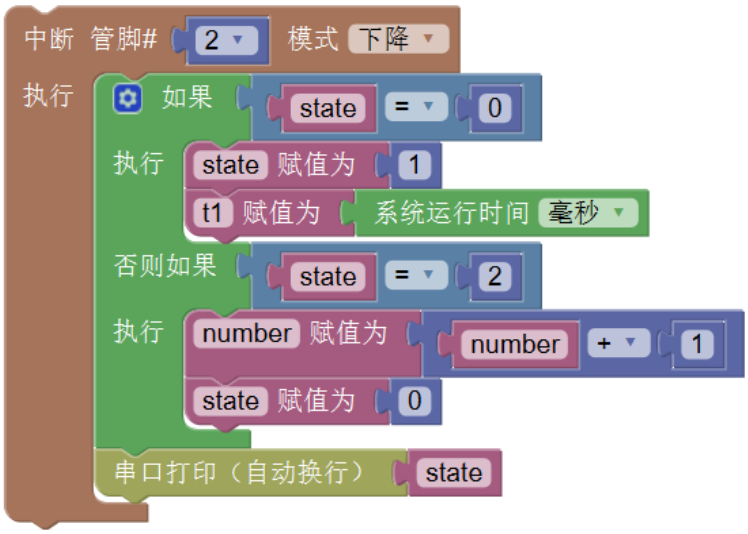
1. 任务要求

制作一个公园人数计数系统：在任务 2 的基础上，增加防干扰功能（提示：如果 2 号或 3 号管脚在一定短暂时间内数次触发中断，则不增加或减少计次）

2. 思路解析


开始
人数减一
人数加一
红外检测到有人由3到2通过且非干扰（很短时间内连续检测到）
中断
液晶显示屏输出公园人数
中断
红外检测到有人由2到3通过且非干扰（很短时间内连续检测到）
结束


液晶显示屏 设备地址 0x27 清屏
如果
系统运行时间 毫秒 - t1 > 500
且
系统运行时间 毫秒 - t2 > 500
执行 state 赋值为 0
液晶显示屏 设备地址 0x27
打印第1行 “ count: ”
打印第2行 转字符串 number


中断 管脚# 2 模式 下降
执行
如果 state = 0
执行 state 赋值为 1
t1 赋值为 系统运行时间 毫秒
否则如果 state = 2
执行 number 赋值为 number + 1
state 赋值为 0
串口打印（自动换行） state

项目十四
打地鼠游戏

在我的同龄人中，几乎所有人都玩过一个很有意思的游戏，它叫打地鼠。现在我将这个游戏介绍给你。在你的面前有五六个小洞，洞的里面藏着地鼠。它们会时不时地蹿出来，企图去周围的庄稼里偷吃。这时你就要用手中的小锤子把它们敲回洞里去，以免庄稼受到伤害。一开始它们一只一只地出现，到了后来，它们越来越猖狂，两三只同时往外冒，考验你的反应速度的时候到了……

本节任务

任务 1——打地鼠初步

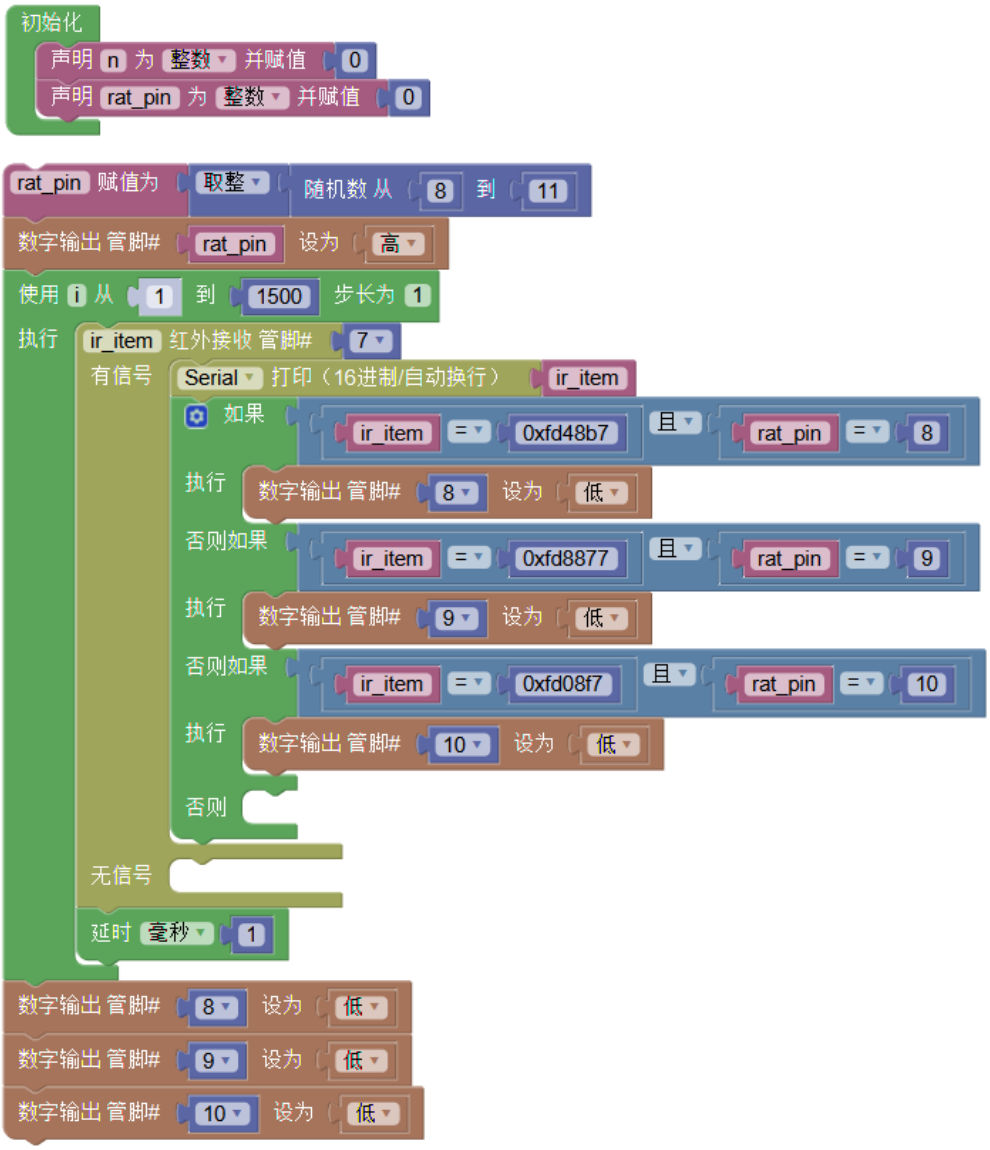
1. 实现的功能：

使用套件中的 3 个 LED 灯代表 3 个洞，当灯亮时代表有地鼠从洞中钻出。让遥控器上的 3 个按键对应三个洞，按下按键即代表用锤子砸向哪个洞。

本任务中要求完成简单的打地鼠游戏——一次出现一只地鼠，玩家要在规定的时间内按下相应按键。由于这仅仅是一个初步的游戏，所以即使玩家没有在规定的时间内按下相应按键，游戏还是会继续进行——当前亮着的灯熄灭，又有新的灯亮起。

2. 硬件连接：3 个 LED→8, 9, 10

3. 程序编写：



任务 2——打地鼠进阶

1. 实现的功能：加入声音，打中高音，打错低音

在打地鼠初步的游戏中，你可能会发现，有时候，明明按下了对的按键，但是对应的 LED 灯还是亮着的。这很有可能是因为，两次生成的随机数是一样的。也就是说，在你按下按键后，灯其实熄灭了，但是进入了下一轮循环后又立刻亮了起来。

对于这个问题，有两种解决办法，一种是在程序的末尾加一个延时的模块，让所有灯保持熄灭状态一段时间，这样玩家就能够注意到下一轮即将开始。

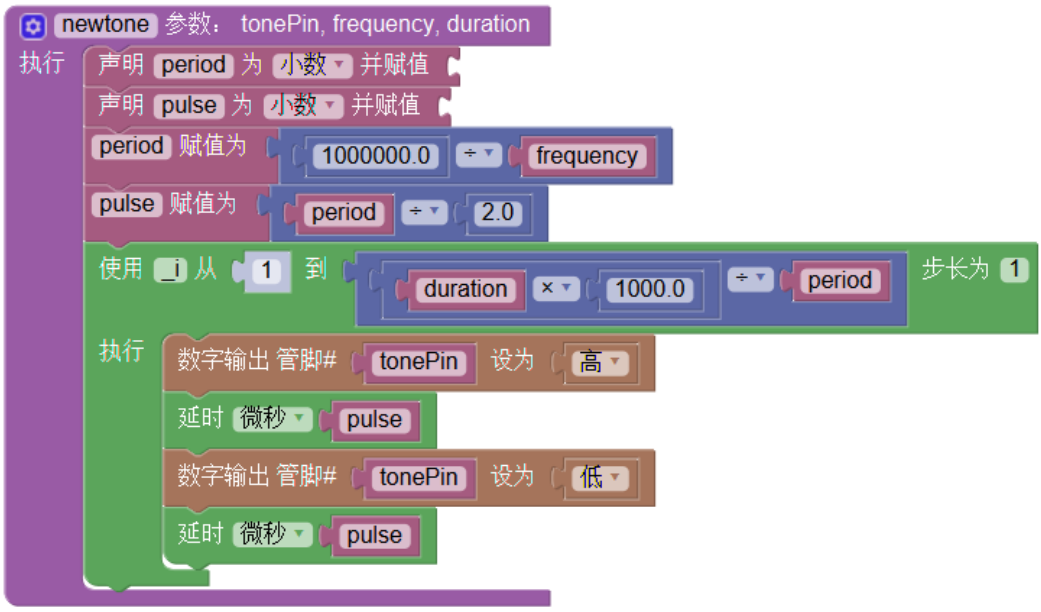
还有一种办法是，将蜂鸣器与主控板连接，当玩家打中地鼠，就让蜂鸣器播放一个很短的音，如果没打中，就播放另外一个音，以此达到向玩家反馈结果的目的。

这两种方法有能够达到目的，也有自己的优缺点，用哪一种，请你们自己做决定吧。

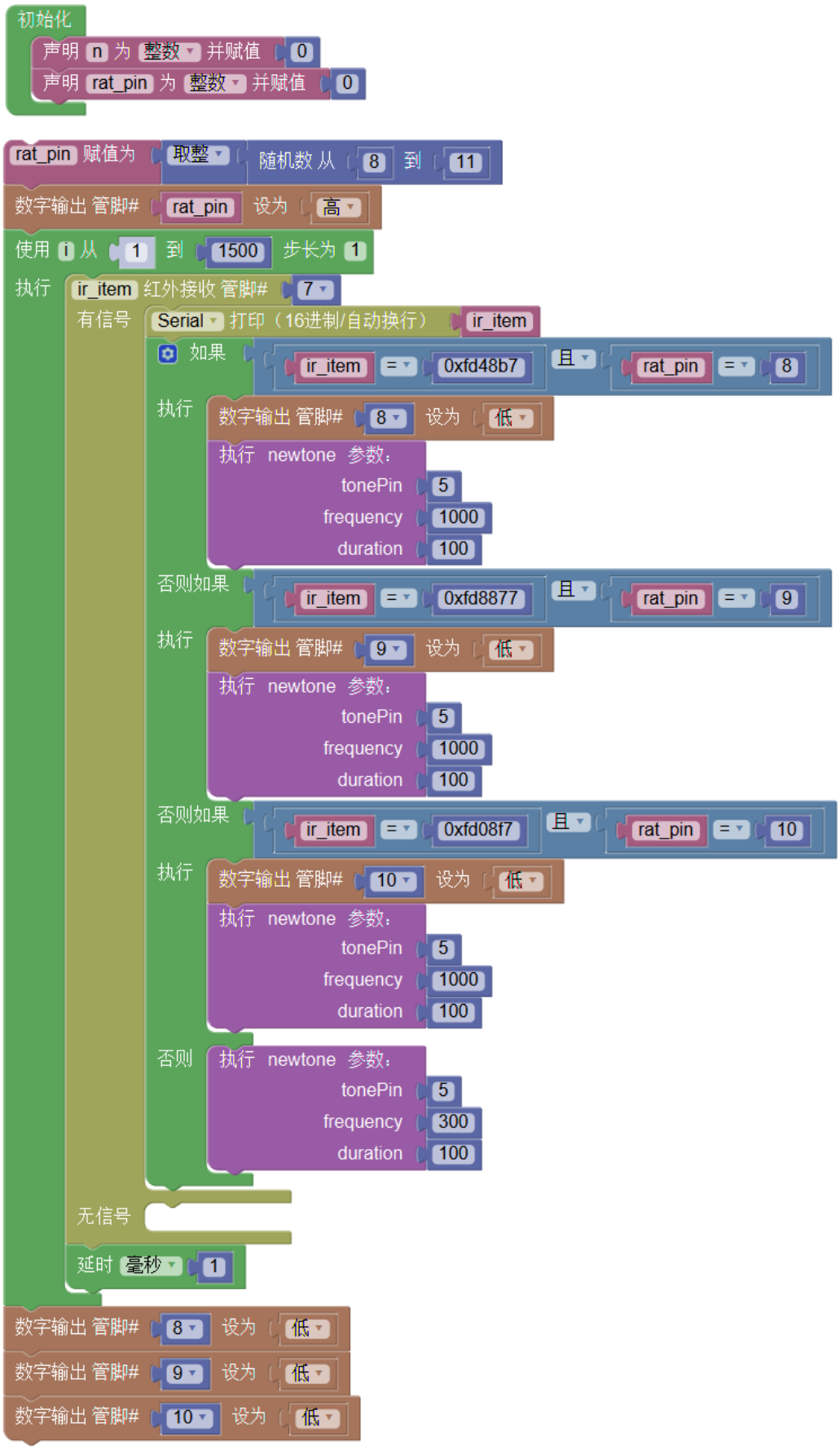
由于硬件的原因，在 arduino 上面不能同时使用红外接收的程序块和蜂鸣器播放声音的程序块。下面我们会给出一种新的方法。

2. 硬件连接：3 个 LED→8, 9, 10；蜂鸣器→5

3. 软件编写：



蜂鸣器的音高是通过频率进行调节的，在这个函数中，我们实现了让蜂鸣器根据玩家设定的频率播放不同声音的功能。这个函数的不足之处在于，播放出来的声音不是很好听。不过，我们至少可以达到播放不同音高的声音以提醒玩家的目的了。


初始化
声明 n 为 整数 并赋值 0
声明 rat_pin 为 整数 并赋值 0
rat_pin 赋值为 取整 随机数 从 8 到 11
数字输出 管脚# rat_pin 设为 高
使用 i 从 1 到 1500 步长为 1
执行
ir_item 红外接收 管脚# 7
有信号
Serial 打印（16进制/自动换行） ir_item
如果 ir_item = 0xfd48b7 且 rat_pin = 8
执行
数字输出 管脚# 8 设为 低
执行 newtone 参数：
tonePin 5
frequency 1000
duration 100
否则如果 ir_item = 0xfd8877 且 rat_pin = 9
执行
数字输出 管脚# 9 设为 低
执行 newtone 参数：
tonePin 5
frequency 1000
duration 100
否则如果 ir_item = 0xfd08f7 且 rat_pin = 10
执行
数字输出 管脚# 10 设为 低
执行 newtone 参数：
tonePin 5
frequency 1000
duration 100
否则
执行 newtone 参数：
tonePin 5
frequency 300
duration 100
无信号
延时 毫秒 1
数字输出 管脚# 8 设为 低
数字输出 管脚# 9 设为 低
数字输出 管脚# 10 设为 低

课后作业

1. 实现的功能：加入闯关计分功能，关卡数和分数在液晶屏上显示

2. 硬件连接：在任务 2 的硬件连接基础上增加液晶显示屏模块，连接方式参考项目十二（硬件连接）

3. 软件编写：参考本节任务 2 中的程序