日期:2021.11.15
责编:高大力 组版:陈学兵 责校:杨志刚

“多线程”编程应用三例

文/山东省招远第一中学 牟晓东

在计算机编程中,“线程”(thread)指的是一组可以在程序中独立执行的指令集合,它是代码执行的最小单位。如果程序在运行过程中只有一个线程的话,那么下一个任务必须要等到上一个任务结束后才能进行,这是一种低效的“串行”流程;引入“多线程”运行机制后,就可以在主线程执行任务的同时“并行”执行其他的任务,几乎不需要等待时间,从而极大提高程序的运行效率。值得一提的是,“多线程”与程序调用函数并不相同,因为函数的调用是“阻塞”执行方式——必须等函数正常执行结束后才会继续执行后面的程序代码(否则会一直处于等待中)。在此分别以Python代码编程和树莓派、掌控板图形化编程为例,演示“多线程”编程的应用。

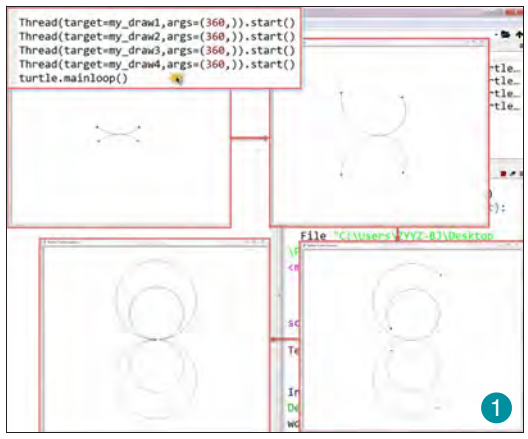
一、Python:四支“画笔”同时作画

在Spyder编辑器中新建“Python多线程画图.py”文件,首先通过“import turtle”和“from threading import Thread”将海龟库模块和线程库模块导入;接着,设置好“画布”的大小为800×600像素:“turtle.screensize(800,600)”,并且通过turtle模块的Turtle()函数建立t1-t4四个Turtle对象;然后自定义四个画图函数,以my_draw1(a)为例:

传入的参数a控制循环的次序:“for i in range(a):”,循环体只包括“t1.forward(2)”和“t1.left(1)”两行代码,作用是控制t1(Turtle对象)向前走两个像素再向左转1度。比如调用该函数时传入的参数a值为180,则会画一个180度的“半圆”(画笔颜色默认为黑色)。其余的my_draw2(b)、my_draw3(c)和my_draw4(d)三个函数基本类似,只是多了一行“t2.pencolor('red')”设置画笔颜色的代码。

接下来,同时开启四个线程,调用目标分别是四个Turtle对象的画图函数,传入的参数均为360(画360度的空心圆):“Thread(target=my_draw1,args=(360,)).start()”、“Thread(target=my_draw2,args=(360,)).start()”、“Thread(target=my_draw3,args=(360,)).start()”和“Thread(target=my_draw4,args=(360,)).start()”,特别要注意args后的参数必须是元组形式“(360,)”(逗号不能省略);最后,添加“turtle.mainloop()”无限循环方式处理事件语句。

程序代码保存后运行测试,在弹出的“Python Turtle Graphics”窗口中出现了四个小箭头,同时分别朝着设定的方向运动,最终画出了四个颜色相异的“内切”和“外切”圆(如图1)。



1

二、树莓派:三色灯带不同步闪烁

首先,将一条可编程ws281x灯带通过古德微扩展板的18号引脚与树莓派连接;接着,登录古德微机器人网站进入“积木”编程区进行图形化编程。

在主程序中对灯带先进行初始化,然后顺序添加三个子线程,名称为“灯带红色”、“灯带绿色”和“灯带蓝色”,分别对应三个同名的函数;每个函数均控制整条灯带60个灯珠的三分之一部分,其中的“灯带红色”函数的功能是将1-20号灯珠先设置为发红光,0.1秒钟后再熄灭并持续0.1秒钟;“灯带绿色”函数的功能是将21-40号灯珠先设置为发绿光,0.2秒钟后再熄灭并持续0.2秒钟;而“灯带蓝色”函数的功能则是将剩下的41-60号灯珠先设置为发蓝光,0.4秒钟后再熄灭并持续0.4秒钟(如图2)。

如果主程序不是采用“多线程”而是函数的直接调用方式,其运行效果就会是在1-20号灯珠闪烁红光的0.2秒钟(两个0.1秒钟)周期内,其余的40个灯珠是全熄灭状态;接下来,在21-40号灯珠闪烁绿光的0.4秒钟周期内,前20个和后20个灯珠同样是全熄灭状态;最后,在41-60号灯珠闪烁蓝光的0.8秒钟周期内,前40个灯珠也是处于全熄灭状态的。

2

使用“多线程”编程的话,会有什么样的展示效果呢?将程序保存后再点击“运行”按钮,出现了三组灯珠互不干扰地以各自的频率进行不同步闪烁的效果,而不是函数式的阻塞等待的执行方式。

三、掌控板:LED变色灯音乐节拍器

运行Mind+,先点击左下角的“扩展”按钮,将掌控板和“功能模块”中的“多线程”添加至主界面;返回后,在“ESP32主程序”下的“循环执行”结构中添加启动三个子线程,其中的“子线程1”只有一行“播放音乐”语句,并且其重复模式为“无限循环”,音乐可自行选择Mind+内置的曲目(比如PRELUDE);“子线程2”实现的功能是控制OLED显示屏根据声音传感器的检测数据实时输出多个柱状音量的动态波形图,包括设置线条的宽度和两个坐标轴数据的绘制,特别需要注意的是,将“读取麦克风声音强度”数据进行映射运算——从0-4095映射为从50-0,并且需要添加“清屏”语句模块(“屏幕显示为‘全黑’”);“子线程3”实现的功能是控制三支LED灯进行有规律的变色闪烁,发光颜色的随机变化是由三个“在0和255之间取随机数”的RGB值来动态实现的,两个“等待0.2秒”的语句模拟控制的是LED灯的闪烁频率,可根据所选曲目节奏的快慢来多次调试。

将程序保存为“多线程节拍.sb3”,然后连接好掌控板,再点击“上传到设备”按钮进行程序的测试。当左下角出现“上传成功”的提示后,掌控板开始有了“反应”:蜂鸣器循环播放程序设置好的音乐曲目,同时三支LED灯在不断变换颜色地闪烁,而且在OLED显示屏上有柱状音量动态波形图在不停地随音乐的音量大小而跳动(如图3)。


3

趣味数学——加油问题

文/陈新龙

在辅导学生竞赛的时候看到一道比较有意思的题目:小明打算在“十一”假期骑摩托车自驾游。摩托车每次加满油后可以行驶100公里。小明在自家附近的加油站加满油就上路了,上路之后还要顺序经过“1号-6号”六个加油站,每个加油站到上一个加油站距离分别为50、80、39、60、40、32公里。由于“十一”期间加油站有很多车要加油,所以小明希望尽量减少加油次数同时顺利到达目的地。那么请问小明需要在哪些加油站停靠加油,才能使得沿途加油次数最少呢?

因为如果摩托车剩下的油不够行驶到下一个加油站时,就必须要在这个加油站加油。结合Scratch编程思想一起来解决这道有趣的奥数题吧。

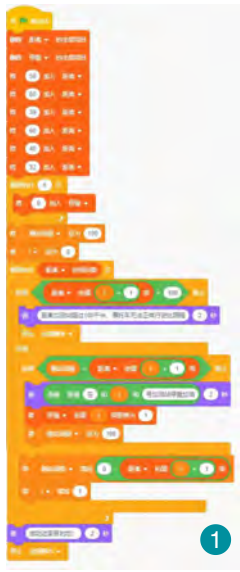
首先需要创建两个列表。列表“距离”用来存放各个加油站之间的距离数值,将50、80、39、60、40、32公里数添加入列表中。

列表“停靠”存放六个元素,用于标记在哪个加油站停靠加油,默认“停靠”列表中六个元素初始值为0。当停靠某个站点时将对应值从0修改为1代表停靠此地加油。

根据题目的要求小明不必在所有站点加油。例如小明在第一个加油站加油,因为出发时油箱是满的,可以行驶100公里,到达第一个目的地的时候,只能行驶50公里了,因为距离第二个加油站还有80公里,所以必须将油箱加满才能继续前行……

使用当油箱的油满足到下一个加油站时就不加油的逻辑,计算出小明停靠加油的合理方案,使得加油次数最少。

新增加两个变量“剩余油量”(统计当前还可以行驶公里数,初始值为100)和变量“i”(列表中的序号,也可以理解成第几个加油站,初始值为0)。


1

重复循环执行6次(列表“距离”的项目数)提取出“距离”列表中每一项的数值进行比较,如果“距离”列表中的任何一项的数字超出了100,那么弹出提示“距离加油站超过100千米,摩托车无法正常行驶此路程”并且停止全部脚本。

如果剩余的油量小于“距离”列表中当前的项数时,需要小明在第i号加油站停靠加油,并且在“停靠”列表中进行对应标记(对应初始值从0变为1),将剩余油量重新设置为100,表示在此加油站将油加满。

在行驶的过程中需要注意两点:剩余的油量是不断地减少的(当前剩余油量值减去距离的第i+1项的值),序号i是不断地增加的(每次增加1),直至到达最终目的地(图1)。

最终小明需要在1号、2号、4号加油站停靠加油,才能顺利到达最终的目的地。

简单的奥数题目结合有趣的Scratch编程摩擦出了不一样的火花,那么聪明的你是否解答出正确答案了呢(图2)?

2