

小小掌控，无限智造。大家好，这是 N+频道，我是播报员小马。
(诶，你不是上次那小何吗，怎么换马甲了。)

这个嘛，很久以前给学生上课的时候，有个学生很皮，被我暴揍一顿(脑海里)后
我：“你真的皮得像匹野马一样”。
学生：“我是马，你是我老师，那你就是马老师咯，哈哈。”
于是马老师这个名字就传开了……

言归正传，前段时间分享了[百灵鸽 | 随身 MP3\(上\)](#)后，不少老师对我在文中提到的列表、字典的使用方法不太了解。本帖就根据我的理解，结合 mPython 里的模块，针对**列表**、**元组**、**字符串(文本)**这三种数据类型，发表一下拙见，其中不当之处，还需要大家指出改正。

一、 列表

列表(list)是 python 语言中的一种数据类型。可能不少老师接触过 C 语言，但没接触过 python，其实列表与 C 语言中的数组有些类似。

打个简单的比喻，列表其实就像我们的书包，我们可以往里面放书本(整数)、笔袋(浮点数)、水杯(字符串)，甚至还可以往里面再放一个书包(列表)。在装了这么多东西之后，还是可以只用一只手(变量名)能够拎起来。就像叮当猫的百宝袋一样，Python 的列表真的非常强大。

1.1 创建列表

我们想使用列表，首先要创建它。在 mPython 里面有很多种创建列表的方法，它们该怎么用呢？



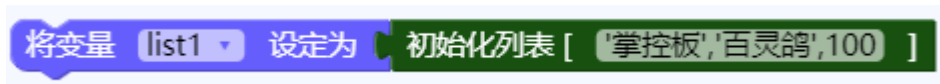
如果你暂时还想不到这个列表要装什么，可以先创建一个空列表：



如果你已经定义了几个变量，想把这几个变量装进列表里，可以这样做：



如果你已经提前想好了要存放的数据，也可以不经过变量，直接放进列表里：



或者：



这两者的代码都是一样的：



大家可以观察到，'掌控板','百灵鸽',100 被一对 [](中括号)括起来了，[]就是列表的标志
 性符号，'掌控板','百灵鸽',100 被称为列表的**元素**，列表中的每个元素用“,”隔开。

注意：编写代码的话，所有的符号都要英文输入法下的符号，中文符号 python 不认识。

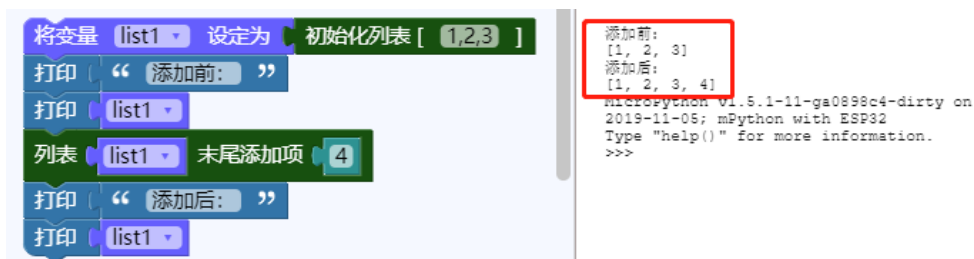
1.2 向列表添加元素

列表相当灵活，所以它的内容不可能总是固定的，在 mPython 里面添加元素的方法有三种：



末尾添加项：

顾名思义，就是在列表的最后一位，添加一个新的元素；

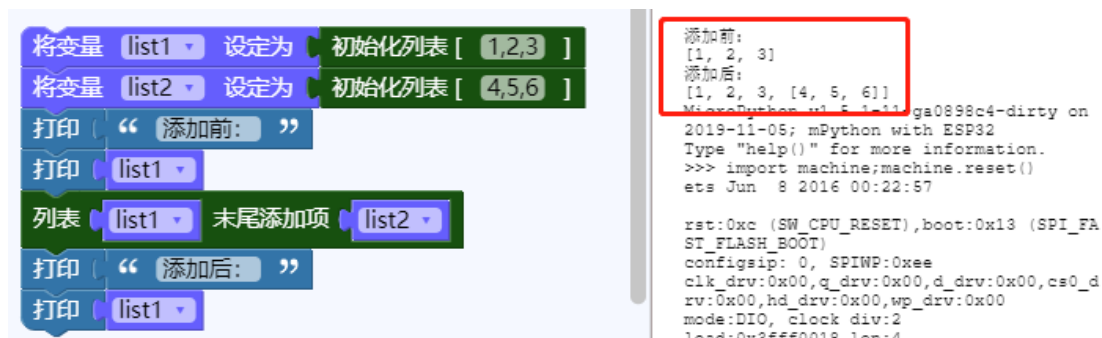


“小马，一次只加一个，我要加多个的话，行不行啊？”

这个嘛，添加项后面只有一个空格，挤不进去噢。

“那我把多个元素，组成一个新列表，不就能塞进去了嘛。”

好像是这么个道理，我们一起试一下。

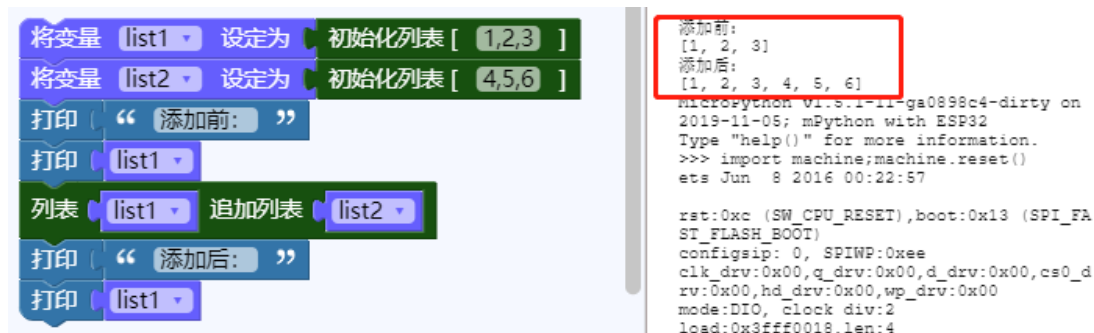


The Scratch code blocks show: '将变量 list1 设定为 初始化列表 [1,2,3]', '将变量 list2 设定为 初始化列表 [4,5,6]', '打印 “添加前:”', '打印 list1', '列表 list1 末尾添加项 list2', '打印 “添加后:”', '打印 list1'. The terminal output shows the list becoming [1, 2, 3, [4, 5, 6]] after the concatenation step.

可以观察到，list2 虽然加入到了 list1 里面，却是以整个列表的身份加入的，这不符合我们的预期。此路不同，我们换一条路。

追加列表：

在列表的末尾，接上另一个列表，两个列表合二为一，就像拼接火车车厢一样：



The Scratch code blocks show: '将变量 list1 设定为 初始化列表 [1,2,3]', '将变量 list2 设定为 初始化列表 [4,5,6]', '打印 “添加前:”', '打印 list1', '列表 list1 追加列表 list2', '打印 “添加后:”', '打印 list1'. The terminal output shows the list becoming [1, 2, 3, 4, 5, 6] after the extension step.

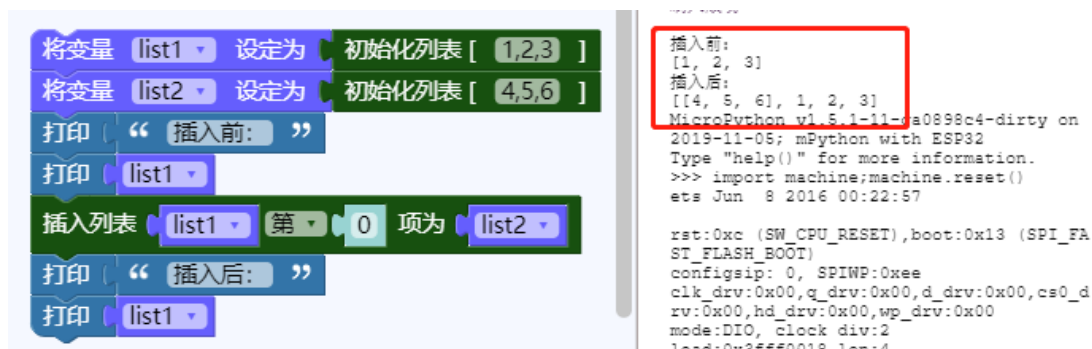
这样，我们就能实现一次性加入多个元素了。

“小马，怎么总是在最后添加啊，在其他位置添加不行吗？”

是可以的，我准备讲了，请不要抢我台词。

插入元素

在指定位置插入元素：



The Scratch code blocks show: '将变量 list1 设定为 初始化列表 [1,2,3]', '将变量 list2 设定为 初始化列表 [4,5,6]', '打印 “插入前:”', '打印 list1', '插入列表 list1 第 0 项为 list2', '打印 “插入后:”', '打印 list1'. The terminal output shows the list becoming [4, 5, 6, 1, 2, 3] after the insertion step.

可以观察到，list2 整个列表插入到 list1 的最前面了。

列表是一个有序的数据类型，和掌控板的 RGB 灯序号一样，是从 0 开始的。如果我们想插入在 1, 2 之间的话，应该改为第 1 项。因为 list2 插入后取代了 2 原来的位置，成为了第 1 项：

将变量 list1 设定为 初始化列表 [1,2,3]

将变量 list2 设定为 初始化列表 [4,5,6]

打印 “ 插入前: ”

打印 list1

插入列表 list1 第 1 项为 list2

打印 “ 插入后: ”

打印 list1

插入前:
[1, 2, 3]
插入后:
[1, [4, 5, 6], 2, 3]
MicroPython v1.5.1-11-ga0898c4-dirty on
2019-11-05; mPython with ESP32
Type "help()" for more information.
>>> import machine;machine.reset()
ets Jun 8 2016 00:22:57

rst:0xc (SW_CPU_RESET),boot:0x13 (SPI_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_d
rv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0018,len:4

1.3 从列表中获取元素

可以通过元素的索引值(序号)，从列表中获取单个元素：

将变量 list1 设定为 初始化列表 [1,2,3]

将变量 a 设定为 列表 list1 第 0 项

打印 “ 取出值: ”

打印 a

取出值:
1
MicroPython v1.5.1-11-ga0898c4-dir
2019-11-05; mPython with ESP32
Type "help()" for more information.
>>> import machine;machine.reset()
ets Jun 8 2016 00:22:57

rst:0xc (SW_CPU_RESET),boot:0x13 (ST_FLASH_BOOT)
configsip: 0, SPIWP:0xee

“小马……”

好，我知道你又想一次取出多个了，我讲

列表确实支持一次取出多个元素，但这几个元素会组合形成一个新的列表，这个操作叫做切片：

索引: 0,1,2,3,4,5,6,7

将变量 list1 设定为 初始化列表 [1,2,3,4,5,6,7,8]

将变量 a 设定为 返回列表 list1 取 第 2 项到 第 6 项

打印 “ 取出值: ”

打印 a

取出值:
[3, 4, 5, 6, 7]
MicroPython v1.5.1-11-ga0898c4-dir
2019-11-05; mPython with ESP32
Type "help()" for more information.
>>> import machine;machine.reset()
ets Jun 8 2016 00:22:57

rst:0xc (SW_CPU_RESET),boot:0x13 (ST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00

1.4 修改列表中的元素：

将变量 list1 设定为 初始化列表 [1,2,3]

打印 “ 修改前: ”

打印 list1

设列表 list1 第 1 项为 100

打印 “ 修改后: ”

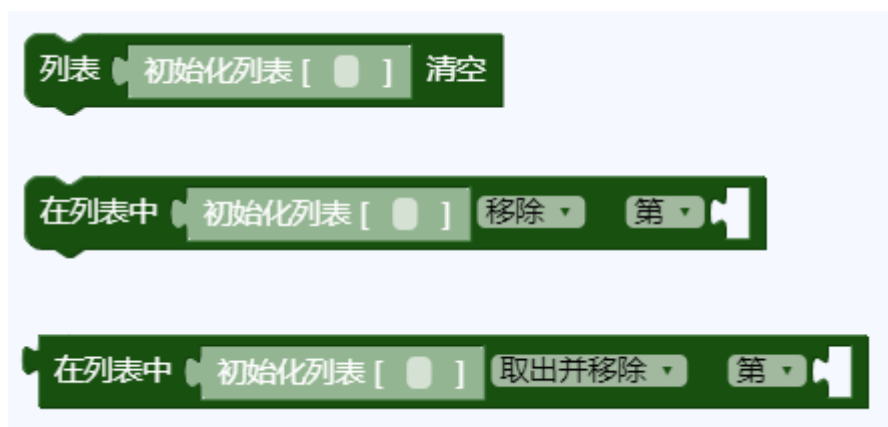
打印 list1

修改前:
[1, 2, 3]
修改后:
[1, 100, 3]
MicroPython v1.5.1-11-ga0
2019-11-05; mPython with
Type "help()" for more in
>>> import machine;machin
ets Jun 8 2016 00:22:57

rst:0xc (SW_CPU_RESET),bo
ST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d
rv:0x00,hd_drv:0x00,wp_dr

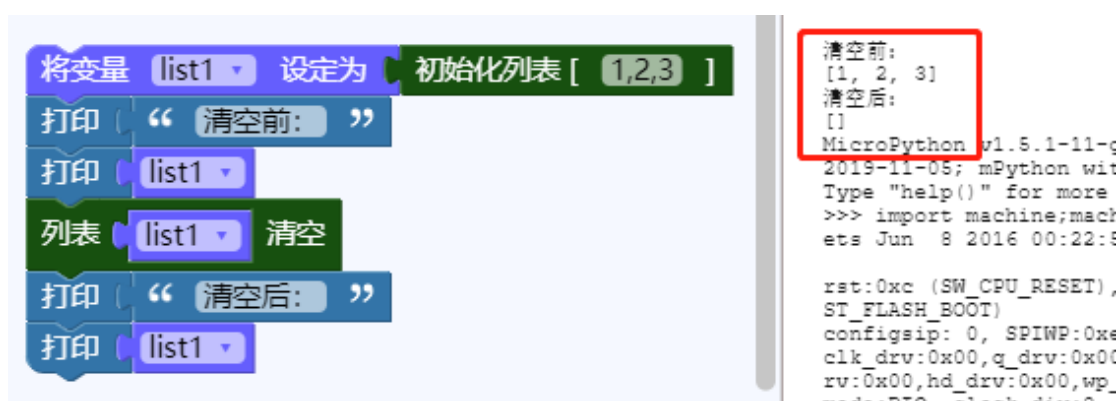
1.5 从列表中删除元素

列表删除元素在 mPython 在里也有三个模块：



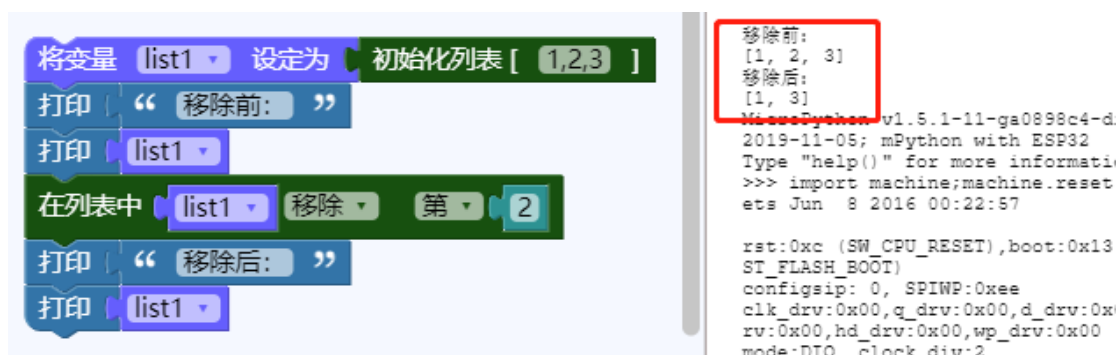
清空:

把整个列表的所有元素都清楚，留下一个空列表:

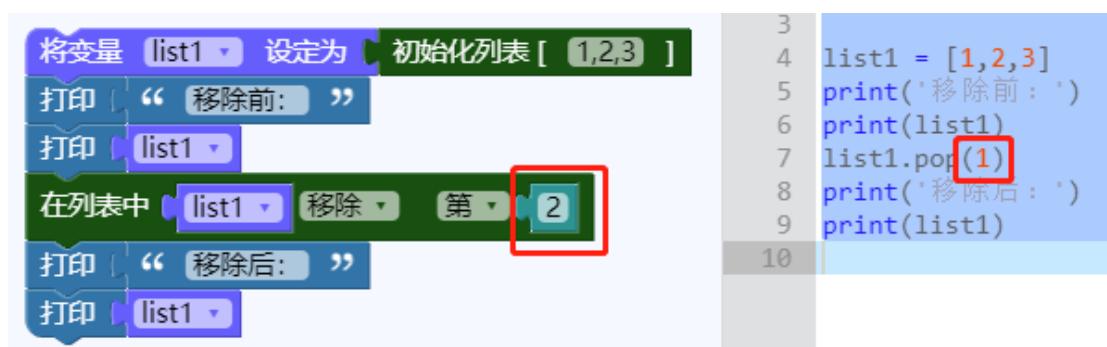


移除某一项:

移除指定位置上的元素: (注意不是索引值)

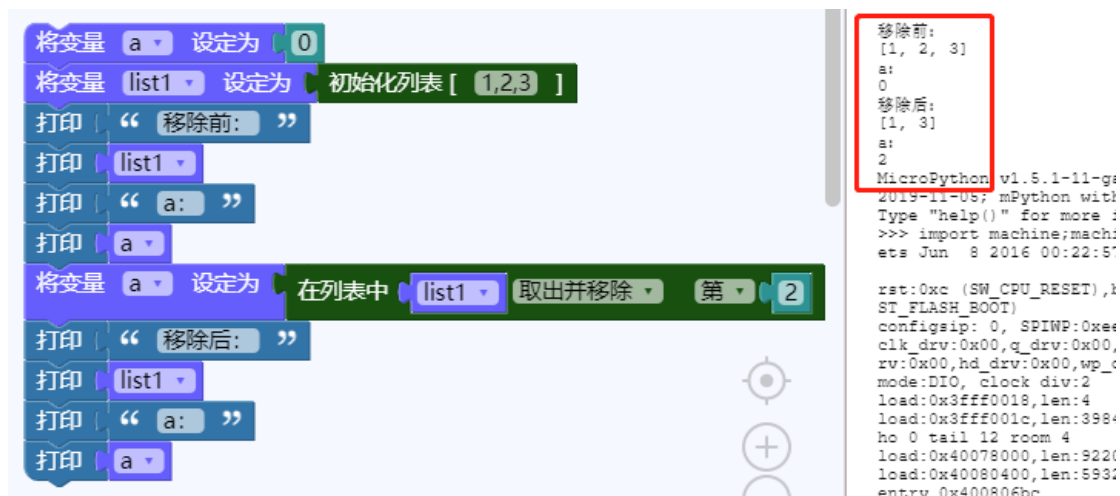


其实代码里都是用索引值从 0 开始的，这个模块估计为了使用方便，封装的时候自动-1了，我习惯代码的用起来就怪怪的。



取出并移除：

这个模块在移除的基础上，还能把移除的数据再次使用，观察模块的形状应该也能猜到用途：



The image shows a Scratch script and a terminal window. The Scratch script performs the following steps:

- 将变量 `a` 设定为 `0`
- 将变量 `list1` 设定为 `初始化列表 [1,2,3]`
- 打印 `" 移除前: "`
- 打印 `list1`
- 打印 `" a: "`
- 打印 `a`
- 将变量 `a` 设定为 `在列表中 list1 取出并移除 第 2`
- 打印 `" 移除后: "`
- 打印 `list1`
- 打印 `" a: "`
- 打印 `a`

The terminal window shows the output of the script:

```
MicroPython v1.5.1-11-g8
2019-11-08; mPython with
Type "help()" for more :
>>> import machine; machi
ets Jun  8 2016 00:22:57

rst:0xc (SW_CPU_RESET),1
ST_FLASH_BOOT)
configsip: 0, SPIWP:0xae
clk_drv:0x00,q_drv:0x00,
rv:0x00,hd_drv:0x00,wp_
mode:DIO, clock div:2
load:0x3fff0018,len:4
load:0x3fff001c,len:3984
ho 0 tail 12 room 4
load:0x40078000,len:9220
load:0x40080400,len:5936
entry 0x400806bc
```

这个模块在制作抽奖机的时候可以用到：抽出一个礼品并显示。

以上就是列表的基础用法了，剩下的模块大家可以自己动手尝试一下，用一下就知道是怎么回事了。

补充一点：

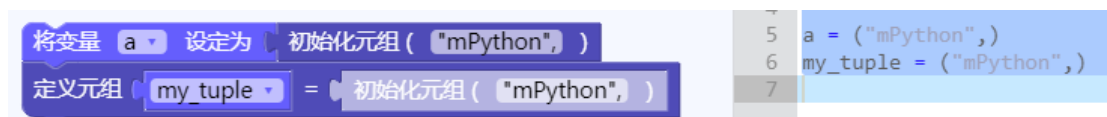


The image shows a Scratch block labeled `从文本制作列表` with a sub-block `用分隔符` set to `" , "`.

这个模块得到的虽是一个列表，但严格意义上是属于字符串的方法。所以我放到后面去讲。

二、元组

元组可以理解为上了枷锁的列表，也就是不能添加、修改、删除。能看不能动，用于保存一些重要的、不能被修改的数据。



The image shows a Scratch script and a terminal window. The Scratch script performs the following steps:

- 将变量 `a` 设定为 `初始化元组 ("mPython",)`
- 定义元组 `my_tuple` = `初始化元组 ("mPython",)`

The terminal window shows the output of the script:

```
5 a = ("mPython",)
6 my_tuple = ("mPython",)
7
```

大家观察一下，元组的标志性符号是什么？

相信很多人会回答：小括号

其实并不是，我通过 pythonIDE 中的 `type()` 函数(作用是查询数据类型)为大家演示下。

```
>>> a=(1)
>>> b=(1,2)
>>> type(a)
<class 'int'>
>>> type(b)
<class 'tuple'>
```

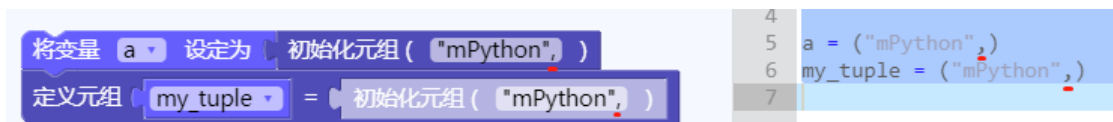
整型

元组

(1) 是一个被 python 视为一个整数，(1,2)才被 python 视为一个元组，两者的区别在哪里？

“元组里面有 2 个元素，难道一个元素就不能形成一个元组吗？”

元素多少并不是关键，图形化积木里也是一个元素，仔细观察一下，单个元素的元组里面多了什么？



没错，其实就是这个不起眼的“,”在起作用，我们再回到 pythonIDE 测试一下：

```
>>> a=(1,)
>>> type(a)
<class 'tuple'>
>>> a=1,
>>> type(a)
<class 'tuple'>
```

所以元组的标志性符号是“,”!

三、 字符串

字符串与元组一样，也是能看不能动的，字符串是以每一个字符为一个元素，引号内的所有字符(包括符号)都视为该字符串的元素。

字符串的标志性符号是引号，单引号、双引号都可以，但是要成对使用噢，棒打鸳鸯可不好。

3.1 字符串分隔



这个模块的作用是，以特定的分隔符，去分隔整个字符串。

我举个栗子：“1234, 2345, 3456, 4567”，这个字符串可以看成是 4 段组成的，每段由“,”连接。分隔之后就会形成：



可以观察到，分隔后，并不破坏原来的字符串。

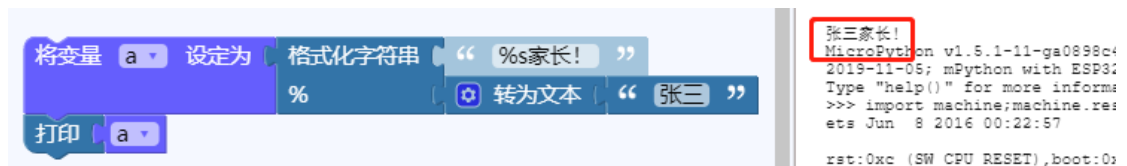
3.2 字符串格式化

这里的格式化并不是与内存一样，不是清空的意思。而是化作某种特定的格式。

我举个栗子：

袁老师要给家长们单独发送开学通知：XXX 家长您好！……

每条通知只是前面的学生名字不同，后面内容是完全相同的，这时候用格式化就会非常方便：



格式化后，“张三”会代替%s，形成字符串“张三家长！”



格式化后，“张三”会代替{}，形成字符串“张三家长！”

字符串的格式化可能有点难理解，大家多用用，多体验下。

聪明的你可以已经发现，我把列表、元组、字符串放在一块儿讲是有道理的，因为他们有很多共同点：

- 都可以通过索引得到每一个元素

- 默认索引值总是从 0 开始（灵活的 Python 还支持负数索引）

- 可以通过切片的方法得到一个范围内的元素的集合。

所以我们把它们统称为：**序列**！

好了，以上就是本帖要讲的全部内容了，非常感谢你看到这里，如果觉得帖子写的还不错，对你有帮助的话，感谢点个赞支持一下，可以的话点个关注就更好啦。

N+频道，我是小马，我们下期再见。