第三课:抽象化模块化

接下来是一个具有挑战性的任务

请同学们好好思考下面的一段话,或许在实验报告或者考试中有所体现:

在编写程序过程中,可以将完成重复的工作的语句提取出来,将其编写为函数。我们可以方便的调用函数来完成重复的工作。在调 bug 的时候,若发现函数出现问题,只需在函数部分修改即可。我们在调用函数时,我们也不用关心,函数具体是怎么编写的,只需要提供输入,得到输出即可。

由此看来,函数是一个很好的抽象,同时也是程序中的一个模块。同学们想想模块化与抽象化的定义,是否能够对它们有更深刻的理解。这些我相信这些理解都会在大家在编写程序之后得到加深。

Ⅳ. 自定义功能单元(1): 函数

声明函数,在使用一个自定义的函数的时候你必须告诉计算机它是啥吧。在 python 中我们使用 def 声明一个函数。

形式如下:

实例:

def<函数名>(参数列表):

<函数语句>

return <返回值>

注意 参数列表和返回值不是必须的,return 可以有返回值也可以没有,甚至 return 也可以没有。有些函数不需要传递参数,也没有返回值。

```
>>> def hello():
...    print('hello,world')
...
>>> hello()
hello,world
>>> def sum(a,b):
...    result=0
...    for i in range(a,b+1):
...        result=result+i
...    return(result)
...
>>> sum(1,10)
```

(2).变量的作用域

内置作用域: python 预先定义的; 全局作用域: 所编写的整个程序; 局部作用域: 某个函数内部范围。

每次执行函数,都会创建一个新的名字空间,这个名字空间就是局部作用域,同一函数不 同时间运行,其作用域是独立的。在函数内已经声明的变量名,在函数外声明后还可以使

用。并且在程序运行过程中互不影响。

实例:

```
>>> def example():
... a=0
... a=a+3
... print('in the function a:',a)
...
>>> a='external'
>>> print('in the global scope a:',a)
in the global scope a: external
>>> example()
in the function a: 3
>>> print('in the global scope a:',a)
in the global scope a: external
```

还有一种是全局变量,实例: 但是要用 global 来告诉计算机 a 是全局变量

```
>>> def example2():
... global a
... a=a+3
... print('in function a:',a)
...
>>> a=3
>>> example2()
in function a: 6
>>>
```

I.介绍 bmp 图片构成。

到现在为止,同学们应该已经具备了编写这个实验代码的必要能力了。但是同学们对于整个图片的构成还不是很清楚。图片为什么是字节流?我怎么样才能对这些字节流进行操作?

首先我来介绍一下 bmp 图片格式:

Bmp 图片可以分为三个部分:

1.BIT MAP FILE HEADER (14byte)

前两 byte 为 BM 是 bmp 图片的特征。同学们可以根据这个来判断文件是不是 BMP 图片。

2.BIT MAP INFO HEADER (不固定 byte)

构成: 引用自维基百科

Size (bytes)	Meaning (windows)
4	The size of this header 重要
4	The bitmap width in pixels(integer) 重要(256 进制第一位为最高位)
4	The bitmap height in pixels(integer)重要(256 进制第一位为最高
	位)
2	The number of color planes 本次实验不用
2	The number of bits per pixel, which is the color depth of the
	image. Typical values are 1 4 8 16 24 32 (we have to deal
	with the 24 one) 可不用
4	The compression method being used 可不用
4	The image size 可不用
4	The horizontal resolution of the image 可不用
4	The vertical resolution of the image 可不用

4	The number of colors in the color palette 可不用
4	The number of import colors used 可不用

以上是 bmp 格式图片在 windows 下的头文件,在 ubuntu 下,本次实验所需的与 windows 中一致。如何得到宽和高,可以使用进制转换的思想。它是 4 位十六进制。

3. BITMAPDATA

长度: height*width*3 代表 B, G, R, B, G, R............. 一个 byte 就代表一个 0~255 的一个整数。即 read(1) 隐藏信息的工作将在这里展开

II.在 python 中,文件的打开,读入和关闭的方法。

file.open() file.read() file.close() 记得在最后要 close 我们来尝试读一张图片:

我在 python 交互式界面,打开了一张名字为 test.bmp 的图片,并且读了前 200 个 byte,以列表的形式储存。list(im.read(200))注意:读到第 200 位时,指针就留在了第 两百零一位,也就是说,继续读下去就读 201 位了。如果想要指针返回原点 可以使用 im.seek(0) 0 表示第一位,im.seek(14) 并表示指针指向第十五位



注意在本实验中 r:只读 b:二进制数据模式 w:写

我们在读图片时使用 open(path,'rb') path 为 bmp 文件所在(比如 test.mp) 'rb'表示以只读模式打开二进制文件。

我们在写图片时使用 open(newpath,'wb') newpath 为你要保存图片的路径(比如 'testnew.bmp') 'wb'表示以写的模式写二进制文件。

在下图,我们看到了一系列的数字,这些数字是什么类型,图像里是以这样的十进制数存储的吗????同学们可以试着用十进制数写一张图片,看看会遇到什么问题

```
>>> im=open('test.bmp','rb')
>>> list(im.read(200))
[66, 77, 54, 249, 21, 0, 0, 0, 0, 0, 54, 0, 0, 0, 195, 14, 0, 0, 0, 88, 2, 0, 0, 1, 0, 24, 0, 0, 0, 0, 0, 249, 21, 0, 195, 14, 0, 0, 195, 14, 0, 0, 0, 0, 0, 0, 0, 0, 0, 169, 150, 138, 217, 195, 184, 230, 208, 196, 234, 212, 20
0, 229, 208, 193, 231, 210, 195, 232, 211, 200, 222, 210, 203, 227, 217, 215, 21
9, 217, 212, 192, 190, 185, 102, 99, 100, 103, 97, 104, 110, 107, 110, 131, 129, 128, 147, 144, 144, 145, 140, 141, 132, 127, 128, 131, 128, 128, 133, 130, 129, 128, 124, 125, 121, 119, 118, 130, 127, 125, 135, 133, 127, 130, 129, 126, 122, 120, 119, 115, 112, 111, 110, 109, 109, 109, 110, 108, 110, 108, 110, 108, 110, 101, 108, 109, 109, 110, 108, 111, 109, 108, 110, 111, 109, 107, 105, 106, 106, 101, 103, 112, 109, 109, 117, 114, 115, 123, 131, 122, 118, 120, 121, 117, 111, 111, 111, 112, 112, 109, 122, 123, 120, 121, 122, 118, 124, 125]
```

同一张图,去除了 list 我们得到了真正组成这张图的元素,bytes 类型,二进制,注意在类似字符串前有一个 b 就是代表他是这样的类型。

List 在搞鬼,但也给我们提供了方便,我们读取 BITMAPDATA 时候是不是方便很多了呢?我们可以直接对这些数字进行运算,但是注意不可以改变头文件。

```
>>> im=open('test.bmp','rb')
>>> im.read(1)
b'B'
>>> im.read(2)
b'M6'
>>> im.read(15)
b'\xf9\x15\x00\x00\x00\x00\x00\x00(\x00\x00\x00'
```

Homework:: 在 yourname.py 中,我们要有这样一些函数: getheight(), getwidth() 分别能够得到输入图片的高度,宽度。最重要的是含有一个函数 GetBITMAPDATA 来获取 BITMAPDATA 的列表。列表中是代表像素值的数字。实现功能: 读入一张图片,判断是否为 bmp 文件,如果是则输出该图片的宽,高,大小,和 BITMAPDATA 值。

提交: 1.pdf 文档,包含对于函数抽象的理解(要多次读取图片的高度,或者程序出错要修改有何好处 100 字左右)。2.文件名为 yourname.py 的 python 源程序。