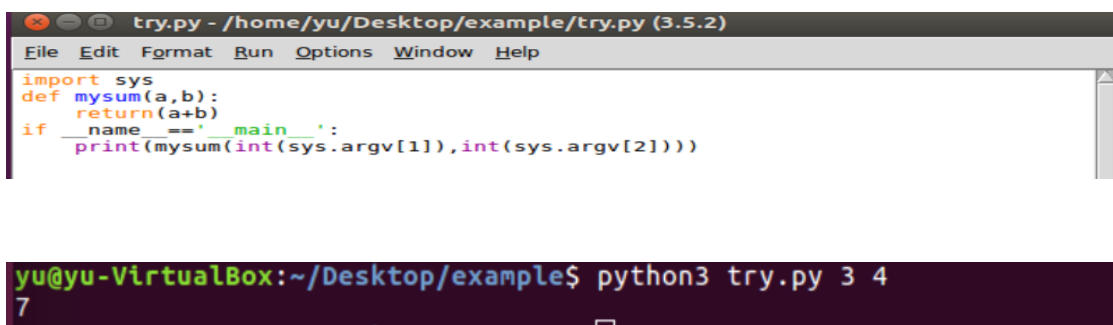


第四课：信息隐藏

我相信对于好多同学来说第三课是难理解的。在第四节课我们做一些简单的事。但是请仔细看，大作业的大部分分数都在这里了。

I. 现在我们来介绍一下 sys 库。



```
try.py - /home/yu/Desktop/example/try.py (3.5.2)
File Edit Format Run Options Window Help
import sys
def mysum(a,b):
    return(a+b)
if __name__=='__main__':
    print(mysum(int(sys.argv[1]),int(sys.argv[2])))

yu@yu-VirtualBox:~/Desktop/example$ python3 try.py 3 4
7
```

使用 sys 库，我们就可以使用参数了。注意 Sys.argv 这一个列表里的参数都是字符串类型。其中 sys.argv[0]='try.py'。

解释:if __name__=='__main__': 他的意思是如果运行这一个程序那么冒号后面的语句将被执行。他是被别的程序调用的那么冒号后的语句将不被执行。

len(sys.argv) 可以获取参数的个数

隐藏信息的工作将在这里展开

II. 好的，现在我们可以正式的实验了。我们该怎么做呢？

我们要新建一个文档，大家统一为各自的名字.py（比如，名字是张三，那么我要新建张三.py）。

在这个文档中，我会对算法做说明以减轻大家的负担。

首先我们要有这样一些函数：getheight(), getwidth(), save(newpath) 分别能够得到图片的高度，宽度和能够将图片保存到新的路径。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。

我现在默认大家都得到了 BITMAPDATA 的数据了。我们用前 9 位存储要隐藏的字符串长度。算法是用四进制存储，比如 $length=100=1 \times 4^3 + 2 \times 4^2 + 1 \times 4$

那么我们就让 $BITMAPDATA[0]=BITMAPDATA[0]-BITMAPDATA[0]\%4$

$BITMAPDATA[1]=BITMAPDATA[1]-BITMAPDATA[0]\%4+1$

$BITMAPDATA[2]=BITMAPDATA[2]-BITMAPDATA[0]\%4+2$

以此类推 直到修改到 BITMAPDATA[8] 那么我们通过保存好的数据就知道以同样方式我们可以开始在图片里存储数字符号了。我们使用 4 位来隐藏一个字母，也是通过四进制的方法。怎么使用数字来表示字母呢，请使用 `ord()` 读取符号的 `ascii` 值 请使用 `chr()` 来将数字转化字母

同学们或许现在对如何保存得到的新的图片还存在困惑。文件头和信息头在读完之后保存不要改变，可以直接 `img.write(文件头+信息头)`，因为他们本身就是二进制字节流。但是 BITMAPDATA 是整数类型，我们应该如何将他转化为二进制的字节流呢？python 对字符串有自带的对字符串变成二进制的。

实例：

```
>>> img=open('test.bmp','rb')
>>> img.read(2)
b'BM'
```

注意：b'BM' 表示它是二进制的，不等于字符串 'BM'

```
>>> im=open('test.bmp','rb')
>>> im.seek(1000)
1000
>>> xx=im.read(10)
>>> xx[1]
58
>>> xx[2]
48
>>> xx[3]
84
```

在写的时候注意

```
>>> img=open('testnew.bmp','wb')
>>>
```



使用 `Img.write(chr(数字).encode('latin1'))` 就可以将列表中的数字以二进制流的方式写入到 `testnew.bmp` 文件中了。

为什么要这样做呢？这样可以做到从 `string` 到 `bytes` 类型的转化:第二个才是我们想要的，每一个 `byte` 存储 0 到 255，也就是长度为 8 的 01 串。ff 在十六进制的大小转化成十进制为 $16*15+16=255$

```
>>> chr(255).encode()
b'\xc3\xbf'
>>> chr(255).encode('latin1')
b'\xff'
```

同学们也可以使用 `bytes([number])` 得到同样的结果

引用自维基百科：[ISO 8859-1](#)，正式编号为 [ISO/IEC 8859-1:1998](#)，又称 **Latin-1** 或“[西欧语言](#)”，是[国际标准化组织](#)内 [ISO/IEC 8859](#) 的第一个 8 位[字符集](#)。它以 [ASCII](#) 为基础，

在空置的 0xA0-0xFF 的范围内，加入 96 个[字母及符号](#)，藉以供使用[附加符号](#)的[拉丁字母](#)语言使用。曾推出过 ISO 8859-1:1987 版

作业: 完成信息隐写任务,

隐写 `python stulD.py bmpImgPath hide filename newBmpImgSavePath`

其中 `hide` 表示隐写

`bmpImgPath` 表示需要隐写的图片路径 比如 (`'test.bmp'`)

`newBmpImgSavePath` 表示隐写好的图片的保存路径 比如 (`'testnew.bmp'`)

读取隐写内容 `python stulD.py newBmpImgSavepath show`

`show` 表示 显示图片

可以参考第一课给实例

要求隐写 `hideinformation.txt` 中的信息并且将信息隐藏至 `testnew.bmp` 中。

提交内容: `stulD.py` `testnew.bmp`

下载说明: 压缩包中有一张图片，其中按照上诉规则隐写了 `word1.txt` 中的内容，同学们可以通过它来测试自己的程序。