应用：BI TY (BInary Type)
用途：推导二进制代码中变量的类型

Windows 下使用方法：
注：此工具依赖于 IDA Pro 和 python3，需提前安装依赖项
1、安装 python3.5.2，IDA Pro 和 PyQt5-5.6-gpl-Py3.5-Qt5.6.0-x32-2，并下载 BInary Type 项目文件
将代码克隆到本地之后：“运行前必装组件”中的两个安装程序先安装。

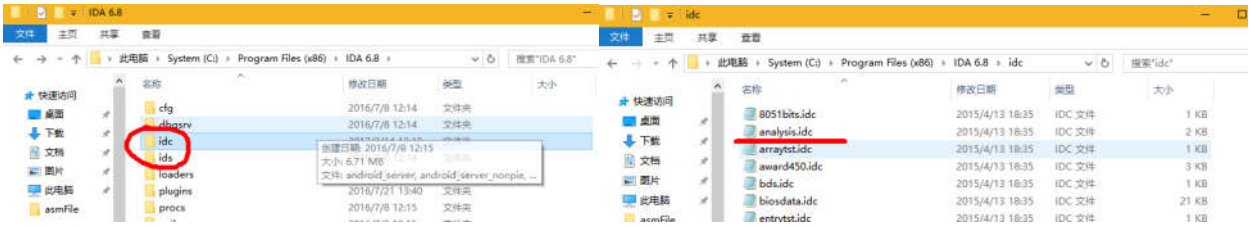
BInary Type Git 地址：	git@bitbucket.org:zhiwu/binary-type.git	（必须）
IDA Pro 下载地址：	http://pan.baidu.com/s/1bp7rOpp	（必须）
python3.5.2 下载地址：	https://www.python.org/downloads/	（必须）
PyQt5 下载地址：	https://riverbankcomputing.com/software/pyqt/download5	（非必须）

2、命令行执行 IDA Pro 分析二进制文件（需要在 IDA 的目录下）
依次执行： **idaq -c -A -S"idc\analysis.idc" "D:\test\base64.exe"**
idaq -Ohexrays:outfile:ALL -A " D:\test\base64.idb "
idaq -A -S"idc\analysis.idc" " D:\test\base64.idb "

(注意：-S 后面没有空格，-A 表示让 ida 自动运行，不需要人工干预。也就是在处理的过程中不会弹出交互窗口，但是如果从来没有使用过 ida 那么许可协议的窗口无论你是否使用这个参数都将会显示。-c 参数会删除所有与参数中指定的文件相关的数据库，并且生成一个新的数据库。-S 参数用于指定 ida 在分析完数据之后执行的 idc 脚本，该选项和参数之间没有空格，并且搜索目录为 ida 目录下的 idc 文件夹。)

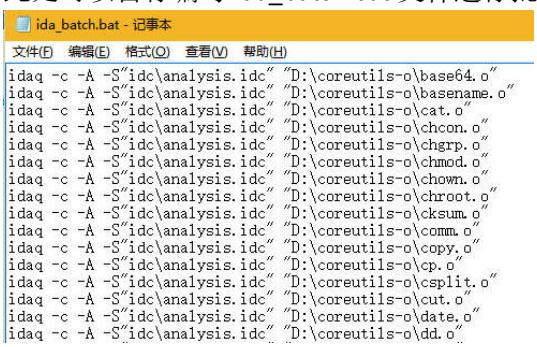
```
C:\Users\admin>cd C:\Program Files (x86)\IDA 6.8

C:\Program Files (x86)\IDA 6.8>idaq -c -A -S"idc\analysis.idc" "D:\test\base64.o"

C:\Program Files (x86)\IDA 6.8>idaq -Ohexrays:outfile:ALL -A "D:\test\base64.idb"

C:\Program Files (x86)\IDA 6.8>idaq -A -S"idc\analysis.idc" "D:\test\base64.idb"
```

“idc\analysis.idc” 表示脚本文件，原地址在 IDA 根目录下的 idc 文件夹里：



"D:\test\base64.o" 是表示二进制文件的地址，根据实际情况改变

执行完该命令行后，二进制文件的同目录下就可以得到.asm 文件
此处可以自行编写 ida_batch.bat 文件进行批处理，例如如下文件：

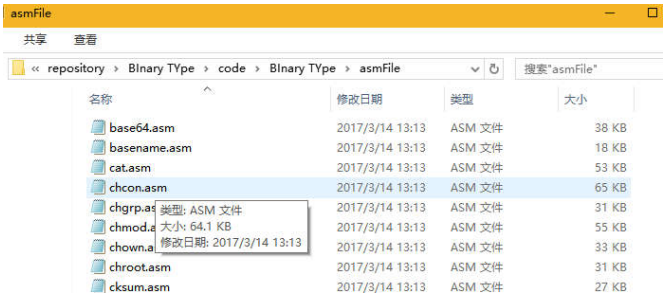


```
C:\Program Files (x86)\IDA 6.8>C:\Users\admin\Desktop\ida_batch.bat
```

批处理执行完后将.asm 文件拷贝到 Binary Type/asmFile 文件夹中即可。

3、BITY 的预处理（将.asm 中的函数片段提取出来）

将需要分析的.asm 文件放到 BITY 的 asmFile 文件夹下

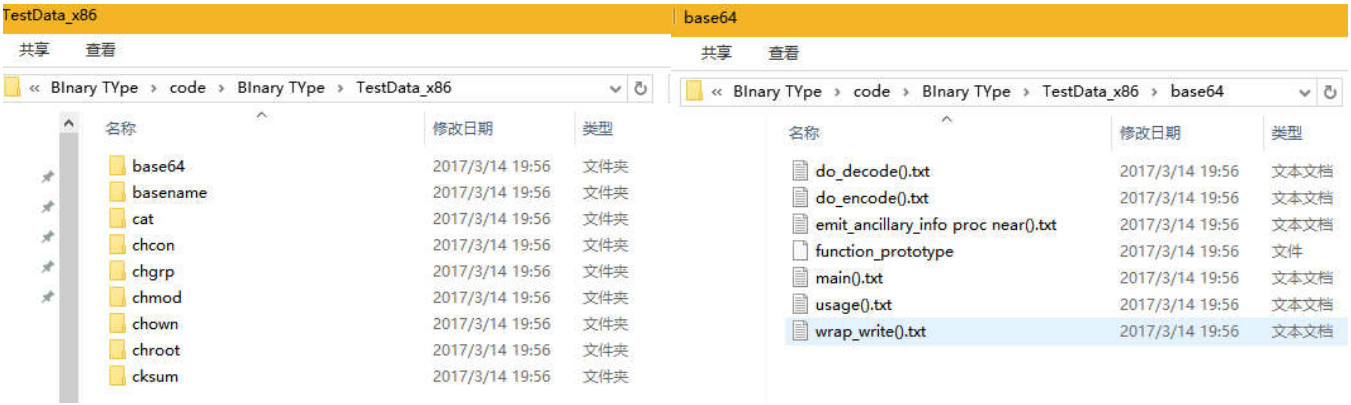


命令行下切换到 BITY 的目录，然后执行 python SplitIDAasm.py

```
D:\repository\BInary TYpe\code\BInary TYpe>python SplitIDAasm.py
        base64.asm                              splited
        basename.asm                            splited
        cat.asm                                 splited
        chcon.asm                               splited
        chgrp.asm                               splited
        chmod.asm                               splited
        chown.asm                               splited
        chroot.asm                              splited
        cksum.asm                               splited

        (All splited)
```

(如果遇到编码问题，只需要用记事本将.asm 文件打开，另存为 utf-8 格式即可)

执行完后 TestData 目录下回出现若干个文件夹，每个文件夹中存放着按函数分割的汇编代码



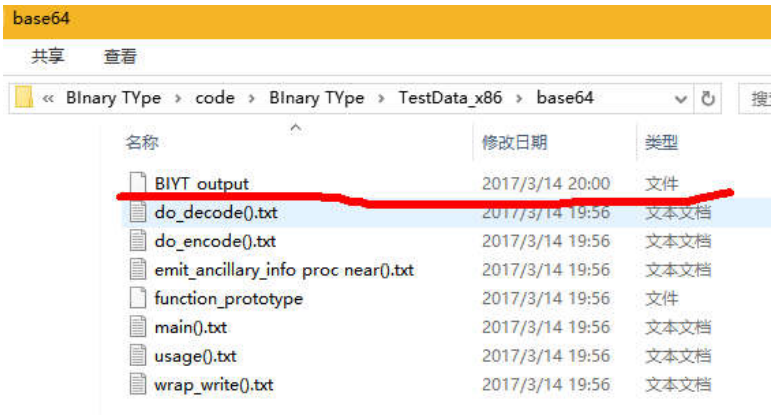
4、执行分析

命令行下执行 python Predict_SVM_Model.py

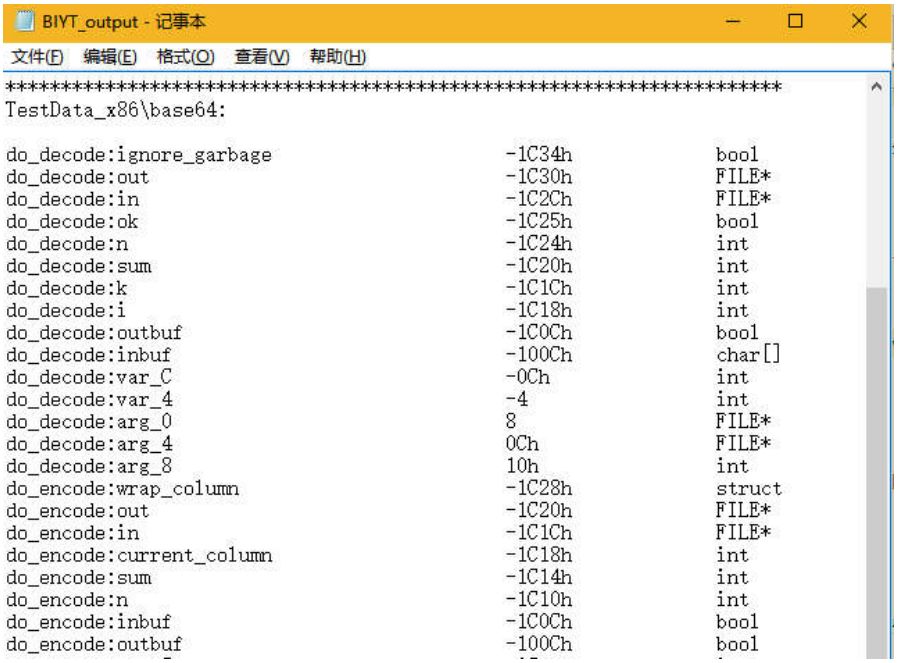
```
D:\repository\BInary TYpe\code\BInary TYpe>python Predict_SVM_Model.py
        TestData_x86\base64                     Finished
        TestData_x86\basename                   Finished
        TestData_x86\cat                        Finished
        TestData_x86\chcon                      Finished
        TestData_x86\chgrp                      Finished
        TestData_x86\chmod                      Finished
        TestData_x86\chown                      Finished
        TestData_x86\chroot                     Finished
        TestData_x86\cksum                      Finished
        (All finish)

D:\repository\BInary TYpe\code\BInary TYpe>_
```

(如果遇到编码问题，只需要用记事本将.asm 文件打开，另存为 utf-8 格式即可)

待提示结束后，再查看 TestData 目录，多了一个 BITY_output 文件，用记事本打开即为分析结果



5、查看 TestData 目录，多了一个 BITY_output 文件，用记事本打开即为分析结果，如图所示：



第一列是“函数名：变量名”，第二列是在栈中的偏移量，第三列是推导出来的类型

BITY 项目组成说明：

asmFile/	从 IDA 中得到.asm 文件，放到该文件夹中
TestData/	执行 SplitIDAasm.py 后，经过分割的.asm 文件即可用于分析（分析样本）
TrainData/	一些训练样本
SaveData /	无须理会，存放程序需要的一些中间文件
FileToVector.py	将一个汇编代码文件中的每个变量用一个向量的形式表示
Train_SVM_Model.py	训练 SVM 模型
SplitIDAasm.py	分割 asmFile 中的.asm 文件
Predict_SVM_Model.py	分析 TestData 中的数据
Sketchy_Predict_SVM_Model.py	不用管
svm.py	LibSVM 工具包
svmutil.py	LibSVM 工具包
libsvm.dll	LibSVM 动态库
libsvm.so.2	LibSVM 动态库
Normalization.py	归一化程序
TransformDwarfFile.py	提取 DwarfFile 中的 debug 信息
TransformExplorerFile.py	可将 https://gcc.godbolt.org/上经过 x86 CL 19 RC 编译的二进制码转为 BITY 可接受的文件
GUI_Predict.py	一个简单的图形界面的输入，保存代码，点击预测按钮进行预测