通过JNI调用DLL需要对Java和C++同时进行适配

**Java端代码：**

**static** {  
 System.*loadLibrary*(**"FaceTest"**);  
}  
**public static native void** testFeatureExtract();  
**public static native void** test\_extract(**int**[] data, **int** width, **int** height);

1.System.loadLibrary("FaceTest")：加载名称为FaceTest，后缀名称是DLL或者so（windows为DLL，linux是so）的库，这个库可以放在C:\Windows\System32目录下，也可以放在工程文件的根目录下即和./src目录是同级目录下，也可将库的目录添加至系统环境变量中，即path

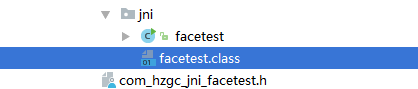
2.添加native关键字用来声明调用的是本地方法，此处不用对此做实现，因为它是调用本地方法的

3.定义好了接口以后，需要通过如下几步引出头文件

（1）通过javac xxx.java将java文件编译为.class文件

（2）再通过javah com.hzgc.jni.filename引出头文件，此处需要注意的是javah后面跟的是文件包的全路径，用.进行分割，并且filename后面不要添加后缀，比如.clss或者.java，执行javah命令最好在com目录的上级目录下执行此命令

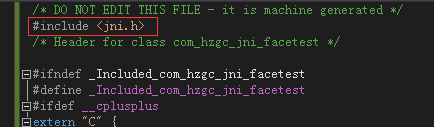
（3）执行成功后会产生如下几个文件，后缀为.h的文件就是后面用到的头文件



4.关于头文件，这是系统生成的不要改，其中JNIEXPORT的声明是为了在生成dll库时，外部可以调用的到，而自动生成的Java\_com\_hzgc\_jni\_facetest\_testFeatureExtract含义即生命是java，包名为com.hzgc.jni，类名为facetest，方法名称为testFeatureExtract

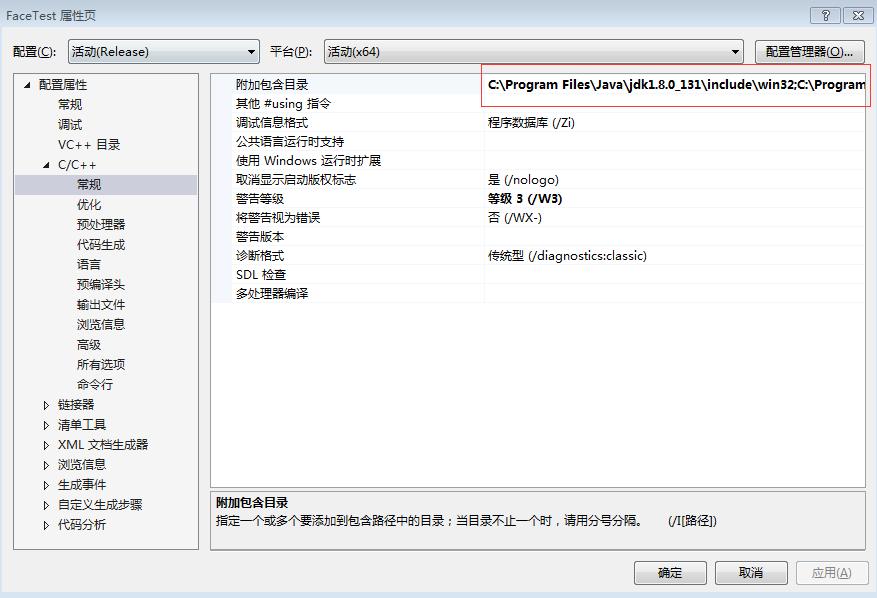
*/\* DO NOT EDIT THIS FILE - it is machine generated \*/***#include** <jni.h>  
*/\* Header for class com\_hzgc\_jni\_facetest \*/***#ifndef** \_Included\_com\_hzgc\_jni\_facetest  
**#define** \_Included\_com\_hzgc\_jni\_facetest  
**#ifdef** \_\_cplusplus  
**extern "C"** {  
**#endif***/\*  
 \* Class: com\_hzgc\_jni\_facetest  
 \* Method: testFeatureExtract  
 \* Signature: ()V  
 \*/*JNIEXPORT **void** JNICALL Java\_com\_hzgc\_jni\_facetest\_testFeatureExtract  
 (JNIEnv \*, jclass);  
  
*/\*  
 \* Class: com\_hzgc\_jni\_facetest  
 \* Method: test\_extract  
 \* Signature: ([III)V  
 \*/*JNIEXPORT **void** JNICALL Java\_com\_hzgc\_jni\_facetest\_test\_1extract  
 (JNIEnv \*, jclass, jintArray, jint, jint);  
  
**#ifdef** \_\_cplusplus  
}  
**#endif  
#endif**

5.打开c++项目，在头文件中新建头文件，名称为com\_hzgc\_jni\_facetest.h，并将上述头文件全部粘贴进来，此处可能会报错

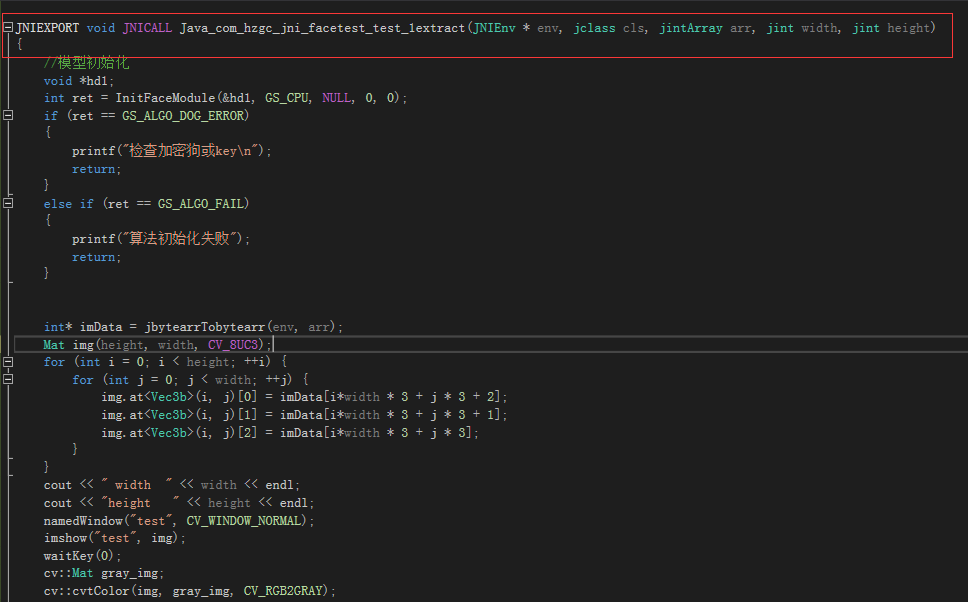


这是因为此处引用了java的头文件，需要需要配置相关的头文件，对应的路径为：C:\Program Files\Java\jdk1.8.0\_131\include和C:\Program Files\Java\jdk1.8.0\_131\include\win32

将这两个路径配置至下面红框未知，报错解决



6.针对头文件生命的方法，进行具体实现，头文件中方法和.cpp文件中实现的方法必须一致，可以直接复制粘贴过来可以保证不出错



红框中即是对方法的实现，在头文件中，方法的参数只有类型生命，没有形参，在具体实现的时候加上形参名称即可