***Le istruzioni qui di seguito riportate si riferiscono ad un sistema operativo Ubuntu; la parte gcc della procedura di generazione della libreria dinamica segue meccanismi differenti in funzione del sistema operativo.***

## *Dichiarazione dei metodi Nativi*

Per dichiarare i metodi nativi è necessario usare la parola chiave **native** come modificatore del metodo e chiudere il metodo con **;** come si fa con i metodi delle interfacce.

public **native** int getValore()**;**

## *Uso del comando javah*

Il comando **javah** è uno dei comandi principali del bin di java, come **javac** e **javadoc**. Javah riceve come input il nome di una classe java (ccomprensiva del package, esattamente come il comando java) ed estrae un **header c/c++** contenente la versione javadoc.

**javah** esempio01.Esempio01

Per lanciarlo è necessario aprire una shell. Se si sta utilizzando eclipse, i file .class si trovano quasi sicuramente in una cartella **bin** e il comando javah deve essere lanciato da lì. Sempre in eclipse si suggerisce di creare una cartella jni e di spostare il **jni.h** in quella cartella: tutto il codice **c/c++** verrà costruito all'interno di quella cartella. Da shell sarà possibile effettuare questo spostamento con

**mv** esempio01_Esempio01.java ../jni/esempio01_Esempio01.java

## *Scrittura del codice c/c++ e generazione della libreria dinamica*

La scrittura del codice c/c++ può avvenire con l'ausilio di QTCreator, ma la sua compilazione dovrà avvenire da **shell**.
Il file prodotto da javah è un .h, ed è necessario realizzare un .cpp contenente l'implementazione dei metodi nativi java.
Se si usa uno o più file cpp di supporto, è necessario compilarli separatamente, indicando al compilatore l'opzione -c, con la quale viene prodotto il file .o, ma non avviene il processo di linking:

**gcc -c** filecpp.cpp

Quando invece si compila il file il file di libreria, useremo l'opzione **-shared -fPIC** . Inoltre è necessario indicare il nome del file di libreria; le librerie linux iniziano sempre con **lib** e terminano sempre con **.so**, quindi se il nome della libreria è nativelib, bisognerà compilare con -o **libnativelib.so**

**gcc -o libnativelib.so -shared** nativelib.cpp

In entrambi i casi è probabile che sia necessario fornire al gcc la locazione degli header jni. Per fare questo, è necessario aggiungere l'opzione **-I/JAVA_HOME/include**.

**gcc -c** filecpp.cpp
**gcc -o libnativelib.so -shared -I/JAVA_HOME/include** nativelib.cpp

## *Utilizzo della libreria dinamica*

Per utilizzare la libreria dinamica è necessario caricare la libreria dinamica dal codice java:

```
static{
        System.loadLibrary("nativelib");
}
```

Inoltre è necessario indicare alla java virtual machine di accedere alle librerie dinamiche; per fare questo in **eclipse** è necessario accedere alle **Properties** del progetto java e in **Java Build Path->Libraries** modificare la **Native Library Location** da usare in combinazione con il JRE inserendo la cartella contenente la libreria nativa precedentemente generata.