T-18 SPOS

Name:- Ankit kailasrao pawar.

Roll No:- 20U535. (Div-5)

Date / /

Page

Assignment No:- 02

* title:- Create the dynamic library for mathematical operation and write an application to test native Interface

Objectives:- • To study Dynamic link Library
• To Learn Java native Interface

* Problem statement:- Write a program to create the dynamic link library for any mathematical operatn and write an application program to test it.

Outcomes:- ① understand how to create DLL.
② perform mathematical operatn and test it.
③ Use of appropriate header file for JNI impleotn.

* Software Requirement:-
1) operating system: ubunt or fedora.
2) JDK/JRE, JVM.
3) Eclipse.

→ Hardware Requirement:- ① 64 bit machine
② 4 GB or 8 GB RAM.
③ 500 GB or 1TB HDD.

→ Theory:- A dynamic link Library (DLL) is collection of small program that can be loaded when needed by larger program and used to the same time. small program let the larger program communicate with specific device, such as printer, or scanner, with a packaged all a DLL program, which is usually referred to as a DLL file that support specific device operatn are known as device driver. A DLL file is often given a "dll" filename suffix, DLL files is often given a "d" file suffix dynamically linked with program

Date / /
Page

that does them during program. exceution eather than being complied into main program. ex the advantage of DLL, file is space is used in random access memory, because the file don't get loaded into RAM, together, with the main program. when a DLL. file its user is editing a document in the microsoft word. the printer out file does not need to be Loaded into RAM, if the user decide to the print the document the word application causes the printer DLL file to be loaded and Ean.

- java Native Interface :- The java Native Interface (JNI) is a programming framework that enables java. code in a java virtual machine (JVM) to call and be called by native application and libraries, written in other language such as, c, C++, and assembly, JNI enables programmer to write native programing language to be accessible to java application, many of the standerd library classes.

- depend on JNI to provide the functionality to the developer and the user e.g. file I/o and sound library allow the all java application to access this functionality in a safe and the platform - independent manner.

- An Interface that allow java to Incorrect with code written in another language.

- Motivation for JNI :-
  → Code Reusability
  Reuse existing / legacy code with java.
  → performance :-
  - Native code used to Inteepted mode.
  - Allows java to SAP into low level o/s, H/W routines.

- JNI Code is not portable.

* JNI Components :-

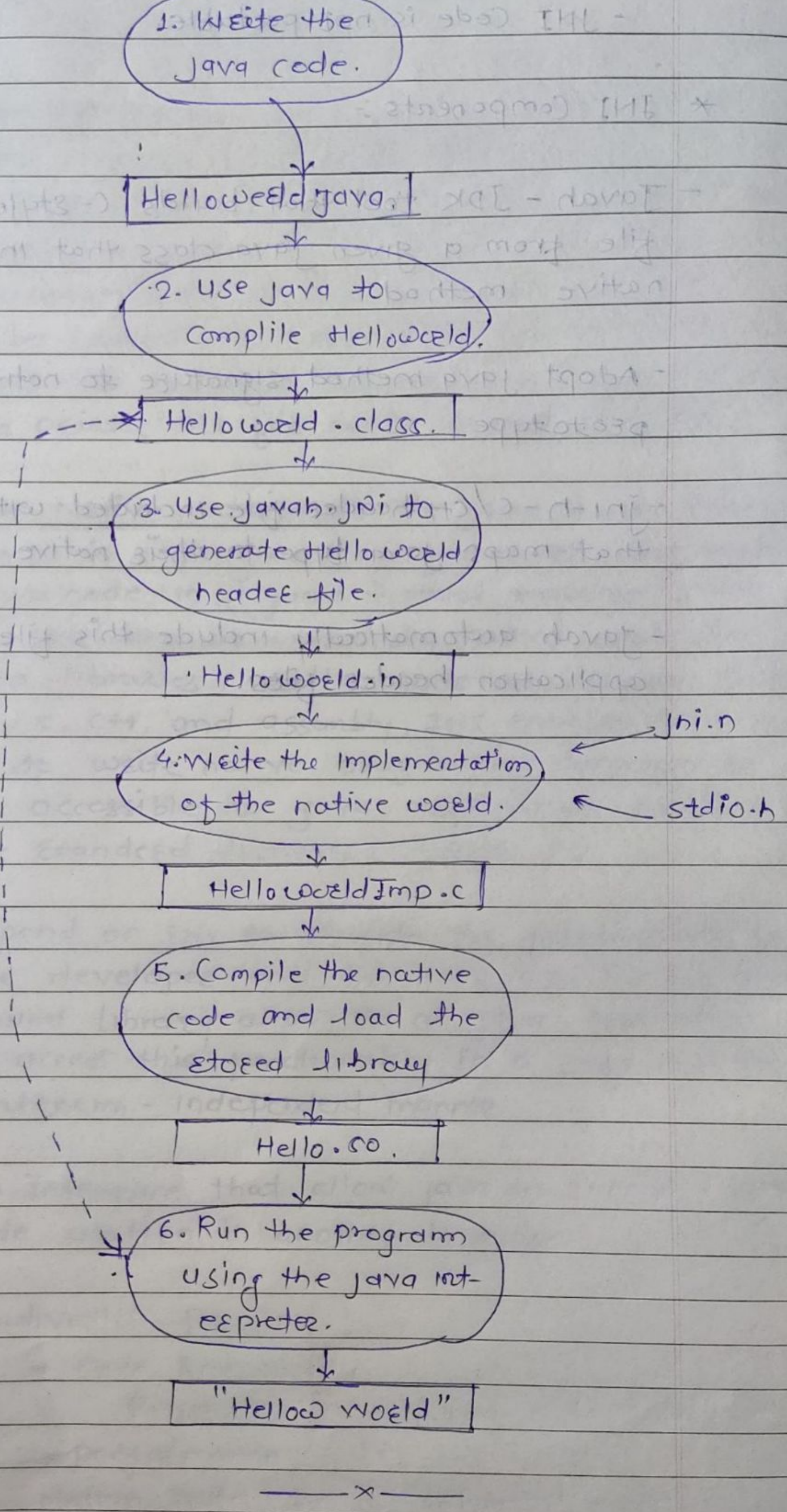
- Javah - JDK tool that builds C-style header file from a given java class that include the native method.

- Adopt java method signature to native function prototype

- jni-h - C/C++ header file included with the JDK that maps java type to their native counterparts.

- javah automatically include this file in the application header files.

T-18


1. Weite the Java code.
HelloweEld.java.
2. Use Java to compile Helloweeld.
Helloweeld . class.
3. Use javah.jni to generate Helloweeld headeE file.
Helloweeld.in
4. Weite the Implementation of the native weeld.
jni.h
stdio.h
HelloweeldImp.c
5. Compile the native code and load the stoEed libraey
Hello.so.
6. Run the program using the java int-eEpretee.
"Hellow weeld"

——×——