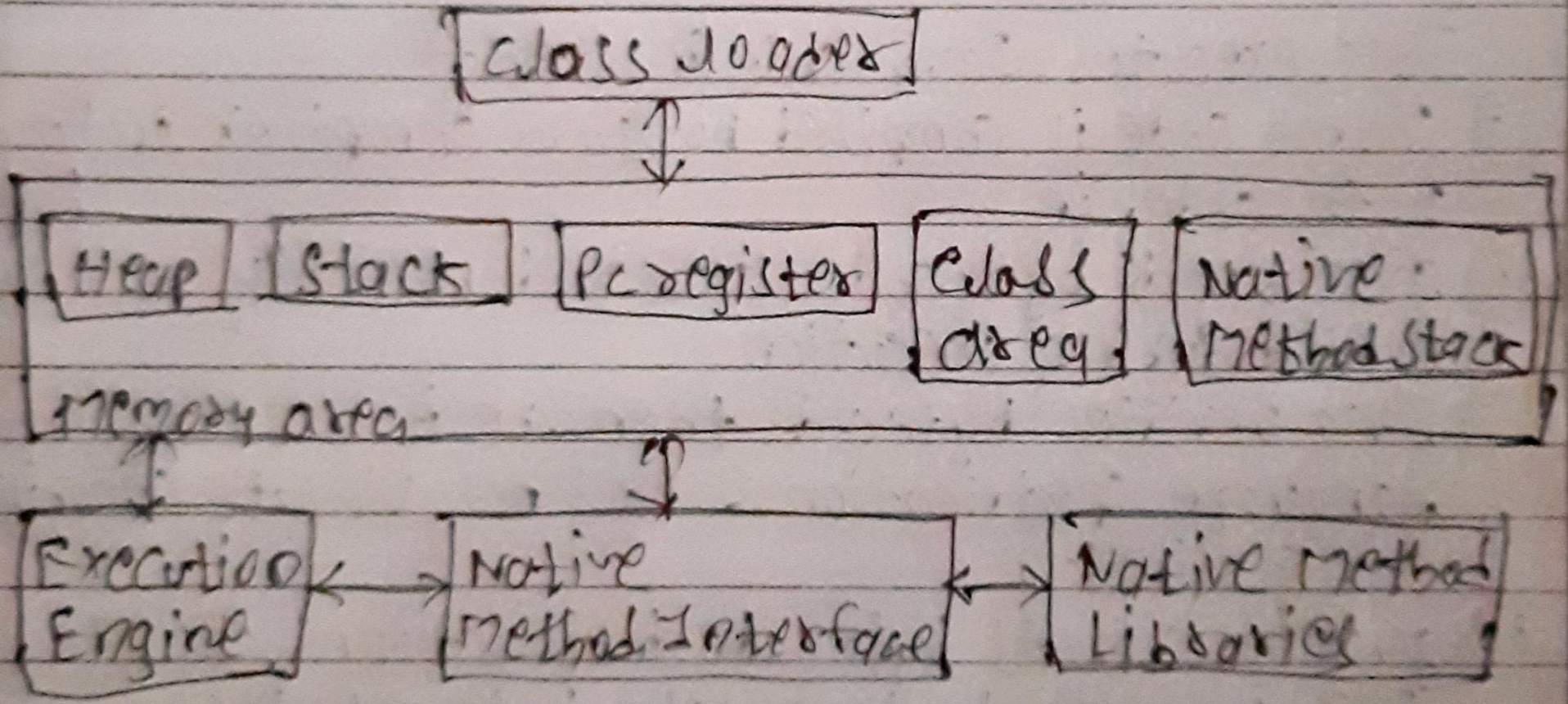
Q6) Describe architecture of JVM.

→



Class Loader →

- Loads class files into memory. It is responsible for finding & loading class files from file system, network & other sources & then converting them into binary form.

Runtime Data Area –

This is area where data structure used by JVM are allocated. It consists of several components.

- Method area →

It stores class metadata, including method bytecode, fields & method information, constant pool & static variables.

- Heap →

Here objects are created by Java program are allocated.

It's runtime data area shared among all threads & it's ~~the runtime~~ managed by garbage collector for memory management.

Stack →

Each thread in application has its own stack, which stores method invocations & local variables. It's last-in, first-out (LIFO) data structure.

PC Registers →

Each thread has its own program counter (PC) register, which holds address of currently executing instruction.

Native Method Stack →

It's used for native method execution (methods written in language other than Java).

Execution Engine-
It executes Java bytecode. It has 2 compone
- Interpreter →
It reads by bytecode & execute the codes
- Reading native machine code instructions.
It's portable but relatively slow.

- Just-In-Time (JIT) compiler →
- JIT compiles frequently executed bytecode into native machine code at runtime, optimizing performance compiled code is then cached for future use.

Native method Interface (JNI)-
It enables Java code to call & be called by native applications & libraries written in other languages. It bridge between Java runtime & native code.

Native method Libraries-
These contains native methods that provide functionality beyond JRE.