Instructions

There are five questions in this exam (some of which include sub-sections). The point value for each question give a rough indication of the expected length of your answers, and the amount of time and thought that I would expect you to devote to the question. These questions do not have a clear right or wrong answer; the intent is that you consider the issues raised by the problem, and apply the concepts that you have learned in this course. It is important that you justify your reasoning for the answers that you give, and explicitly state any assumptions that you make.

You have five days in which to complete the exam, but I wouldn't expect that you spend more than four or five hours actually preparing your responses. However you may want to take some time to consider the questions before sitting down to write up your answers. I'd suggest that you download and read the entire exam early in the week, and then mull over the questions for a while. Remember, though, that the Friday deadline is fixed and that there will be no extensions.

I will be grading the exam based on how well you demonstrate your understanding of the subject, not on your grammar, spelling or prose. It is perfectly acceptable to use bullet points if that helps you to articulate your thoughts more precisely. You are not expected to write code for the exam.

Please type your responses, and submit them as a text, Word (`.doc` or `.docx`) or PDF file. Clearly indicate which response is associated with which question, and format your answers in the same order as the questions on this sheet. Submit your completed response file using Canvas (just as you have for previous assignments). If you encounter any difficulties in submitting your exam, please e-mail your response file to me at `phil_mcgachey@harvard.edu`.

This take-home exam is open notes and open book. You may use any information presented during class (including the lecture slides and videos), and any resources linked to from the course Canvas site (including the Java Language Specification and the Java Virtual Machine specification, linked from the Modules section). **You may not use other web resources**; it is unacceptable to search the web for answers to the questions, or inspiration for your own answers. If you are in doubt about whether a resource is acceptable to use, contact me before using it. **You must complete the exam on your own**.

To affirm that you followed these instructions, your exam submission must include the statement:

I certify that I used only permissible resources, and did not collaborate with anybody while taking this exam.

Question 1: Application Startup (75 points)

You are the tech lead on a development team that maintains a major commercial Java Virtual Machine. Your sales organization have complained that they are losing contracts because your VM under-performs on a certain benchmark suite. Upon analyzing the benchmarks, you find that the startup time for your VM is to blame; the benchmarks put a heavy weighting on the work performed in the first few tens of seconds of the VM's run.

You have been instructed by management to improve performance in the early stages of the VM's run. Describe some of the issues that affect VM startup time, and discuss how you might design or configure your VM to minimize early performance costs.

Question 2: Memory Management (125 points)

You have been hired as a consultant for a firm that develops home automation technology. They sell electrical components (outlets, switches, thermostats, etc.) that are installed in the home, and over the years have developed a large and complex server application that allows homeowners to monitor and control their devices. This platform is primarily implemented in Java, but has significant native extensions (implemented using JNI) that handle some low-level aspects of the system. It currently requires that the homeowner have a Linux server running 24/7 to handle scheduled events.

However, with the availability of cheap low-powered processor boards they want to manufacture and sell a stand-alone controller that runs their software without the overhead of full server. Their code is sufficiently complex that re-implementing their entire software platform is unfeasible, so they have decided instead to build a Java VM that targets their controller. They hope to be able to offset some of the development costs by selling the controller and VM to other companies who need an embedded JVM solution.

Your job is to design the memory management component for this new VM. You must design both the allocator and the garbage collector. Identify and describe the requirements for a memory manager for this platform, and how those factors either constrain or guide your choice in algorithm. Discuss the features of various GC algorithms, including the trade-offs that you would consider while choosing a design. Finally, conclude by proposing the memory manager algorithm that you would implement.

The following are some of the features of the platform:

- Single core processor with two hardware threads.
- Memory is word-addressed, with a 32-bit word size.
- Relatively small main memory. The available memory is around two to three times the working set of the type of application expected to run on the platform.
- A single level of cache (a split L1 cache, i.e. one that separates instructions and data).
- Relatively long latency between cache and main memory.

Question 3: Refactoring (125 points)

You are the tech lead for a development group that has built a large Java application. Application performance is becoming a major issue, and your management has set aside some time in your release schedule to make performance improvements. At a brainstorming meeting, members of your team suggest using the time in the following ways:

1. Refactor the code so that it interacts with as many classes as possible early in the application's run, forcing the class loader to load all classes early.
2. Minimize the use of object orientation, in order to maximize the number of static method calls.
3. Re-implement parts of the code in a native language and integrate with the VM using JNI.
4. Implement an object pooling strategy that pre-allocates and re-uses objects rather than creating new instances on demand.
5. Reduce the size of synchronized sections, and replace explicit monitor acquisitions with atomic operations.

This question has two parts:

(a) Expand on each of the proposals, describing the rationale behind each and give your opinion on whether it is a useful optimization.

(b) You (quite reasonably) decide that you don't have enough information to make a choice. Describe what data you would like your team to gather by profiling the application in order to make an informed decision.

Question 4: Highly Concurrent JVM (75 points)

You are designing a new custom Java Virtual Machine that aims to maximize performance for highly concurrent applications. You will be targeting large server machines that have lots of memory and many hardware threads. However, you expect that the applications that will target your VM will have many more Java threads than the server has hardware threads.

You want to optimize the VM for concurrent throughput, even at the expense of single-thread performance. You must implement the JVM specification (i.e. you cannot change the memory model, and your system must implement priority scheduling). For each of the following components, discuss any design decisions that you might make to emphasize multi-threaded performance:

(a) The implementation of monitors, including the contention manager that determines which thread should obtain a monitor or be scheduled from a set of waiting threads.

(b) The object model for heap-allocated objects and arrays, including the per-object locking strategy.

(c) The scheduling mechanism.

Question 5: Defeating Decompilation (100 points)

A decompiler is a computer program that attempts to recreate the original source code for an application by examining its executable or (as in the case of Java) its bytecode. Decompilation for Java is a mature field of study, and commercial decompilers exist that can do an impressive job of regenerating lost source code. Part of their implementation involves pattern recognition (Java compilers generally emit similar code for similar structures; we saw some of these patterns in class, including the bytecode structure of a loop or a `finally` block). However, the majority of their information comes from the debug data stored in the class file.

Java bytecode lends itself well to decompilation, but this is not necessarily an advantage for some application developers. Consider proprietary commercial software that is distributed as a `.jar` file. Such code may contain trade secret algorithms, or implementations of proprietary protocols or file formats. Unfortunately for the authors of such software it is impossible to completely defeat decompilers (the same is true for native code, although decompiling an executable is far harder than a `.jar` file). However that does not mean that developers have to make life easy for those who decompile their code.

Imagine that you were to build an application that post-processes valid Java class files to make them harder to decompile. Propose a series of transformations that you would make to the class file, with the following conditions:

- The resulting class file must still be valid, according to the JVM specification.
- The semantics of the program must not change (i.e. your modifications should not affect the behavior of the original application).
- You should not modify the `Code` attribute of a method.

Beyond those constraints you are free to rewrite the class files however you wish in order to make them harder to decompile (or to make them less useful once they have inevitably been decompiled). Be explicit - for this question you should refer to Chapter 4 of the Java Virtual Machine specification

`http://docs.oracle.com/javase/specs/jvms/se7/html/jvms-4.html`.