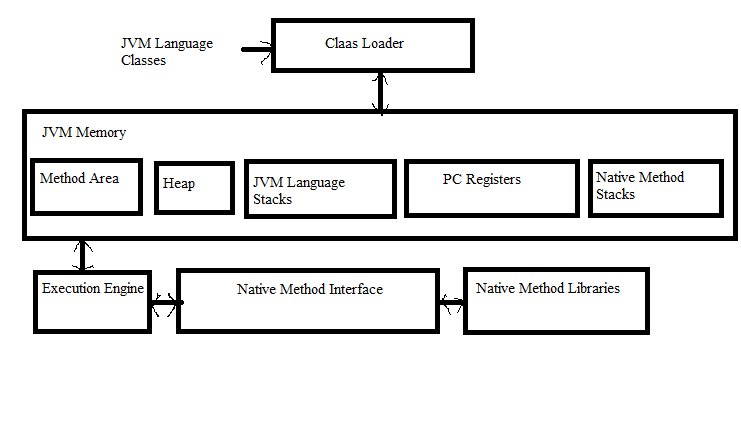
TASK 3:

1. the first suggestion that I have is replacing hash set with list as long as it is just adding object at the end and it doesn't require to check for unique objects
2. An instance of Hash table has two parameters that affect its performance: initial capacity and load factor. As long as hash tables are 10% filled, I can decrease whether the initial capacity or load factor, or decrease both of them.
3. Accessing to database has low performance and I can remove this access by writing the data offline. We can save the data offline and call them once and use them while running instead of accessing the database every time.
4. Increasing hard drive specifically in database side

TASK 4: 

1. Stack is for static memory allocation, Heap is for dynamic memory allocation
2. Variables allocated on the stack are stored directly to the memory with fast access in compile time, the heap have their memory allocated at run time and accessing this memory is slower.
3. The stack is always reserved in a LIFO order, Element of the heap have no dependencies with each other.
4. Keep track is much easier in stack rather than heap.
5. You can use the stack if you know exactly how much data you need to allocate before compile time and it is not too big, You can use heap if you don't know exactly how much data you will need at runtime or if you need to allocate a lot of data.
6. In a multi-threaded situation each thread will have its own completely independent stack but they will share the heap.
7. Stack is thread specific and Heap is application specific.