**Object and Reference Manipulation**

Consider the following (high-level) JVM instructions from [JVM Spec](http://lara.epfl.ch/w/compilation:jvm_spec):

* **new**
* **ifnull**
* **ifnonnull**
* **getfield**
* **putfield**

In lower level code, these instructions are implemented by mapping objects into chunks of memory. The addresses of dynamically allocated objects are returned by a memory allocator (malloc in C) or garbage collector in Java. Such memory management routines implement an illusion of a large graph-like memory, by finding free space in RAM and freeing up space that is no longer used.

To access individual fields, x.f is compiled into something like

mem[x+offset(f)]

We compute offset(f) by adding up the sizes needed to store all fields declared before ‘f’, for example. To improve efficiency of memory access, compiler often align the addresses of fields at the word boundaries, which makes objects larger by improves performance.