

# Homework # 7

## due Monday, March 23, 11:00 AM

### 1 Garbage Collection

For this homework, you will implement a mark-sweep collector. The framework of the problem is provided in two files: `Client.java` which *uses* the heap manager, and `HeapManager.java` which defines it. These files are available in `$CLASSHOME/src/homework7`. The `HeapManager` class is incomplete. You need to define the code to look for a block in the free list, and the code to perform the mark-sweep garbage collection.

Unlike the code in the book, each cell in the memory is either an `Integer` object or a `Pointer` object. In order to tell which you have, you will need to use `instanceof` checks and casts. For instance, to determine if the memory pointed to by a pointer is an `Integer` object, one may write:

```
p.peek(0) instanceof Integer
```

To assume that it is an `Integer` object, one writes:

```
(Integer)p.peek(0)
```

An `Integer` object is not very useful. In order to get the `int` value out, one will need to write

```
((Integer)p.peek(0)).intValue()
```

(The `peek` method is defined in the class `HeapManager.Pointer` and accesses memory using the pointer `p`.) In this system, we represent null pointers by a `Pointer` object with a `-1` address.

In a mark-sweep collector (described on page 265), the collector starts from the “root” pointer (which will be the current frame pointer) and traverses the heap links marking every heap location that is accessible. It uses the size of allocated blocks, and traverses `Pointer` values, but not `Integer` values. You may assume that all non-null pointers point to the beginning of a block. The allocator should ensure that the size of the block is stored just before the beginning of each block.

### 2 Submitting Your Work

Put the changed `HeapManager.java` code in your `homework7` directory.