EECS 345 Homework #7 - due 11/29/05

1. Using the heap representation described in class, implement a mark-and-sweep garbage collector for Tiny-3 in Common Lisp. Your garbage collector should be triggered when `cons` requires a new cons cell and there is no space left in the heap. Your collector may use the `mark` field of a cons cell structure. Note that your garbage collector only needs to collect cells allocated by `cons`. Tiny can continue to use Common Lisp lists for programs, environments and closures. Note also that, while a full garbage collector would be able to deal with closures that occur anywhere in the environment, your implementation only needs to deal with closures defined in the global environment.

   Hint: You might want your collector to return the location of the first cons cell it frees up, so that the `allocate-cons` function can either allocate that newly-found cell or return an error if garbage collection yield no free cells. You will have to modify `allocate-cons` slightly to make that work.

2. Implement BLOCK and RETURN-FROM in Tiny-4, where `(block` *name form*`)` evaluates *form* and `(return-from` *name form*`)` immediately returns the value of *form* from the lexically enclosing block *name*, or signals an error if there is no such block. If no RETURN-FROM is executed within a BLOCK, BLOCK just returns the value of the *form* in its body.

   Hint: In the Tiny environment, associate each block name with a tag and use Common Lisp's `catch` and `throw` to implement BLOCK and RETURN-FROM, respectively.