We recently started testing GC.compact effectiveness in production, and one challenge we faced was to ensure that C extensions were compatible with it.

Here's two examples of C-extensions which caused various issues, and their respective fixes:

- [https://github.com/Shopify/liquid-c/pull/55](https://github.com/Shopify/liquid-c/pull/55)
- [https://github.com/brianmario/mysql2/pull/1115](https://github.com/brianmario/mysql2/pull/1115)

Every time the fix is quite straightforward, my problem is that it's almost impossible to write a reliable test case for it.

With liquid-c I was able to reproduce the issue fairly constantly by calling GC.compact after loading the extension, but for some reason I was totally unable to do the same with mysql2. And even in production, the issue would only happen on a small number of processes.

This makes me believe that having a debug method to move all objects on the heap would be very useful in this scenarios. There is already several GC.verify_* method intended to be used in debug scenarios, so there's precedent.

I think something like GC.move_all_the_heap would make such testing much easier. e.g.

```ruby
require 'c-ext'
GC.move_all_the_heap
# run the library tests
```

cc tenderlovemaking (Aaron Patterson)

### History

**#1 - 05/07/2020 04:03 PM - tenderlovemaking (Aaron Patterson)**

We currently have this functionality, but the API isn't as nice as what you propose (and the naming isn't great either).

You can do GC.verify_compaction_references(toward: :empty, double_heap: true). It will double the size of the heap, then pack towards empty pages which will ensure that any object that can move will move. Maybe we should change the name to debug_compaction with options?

I'd like to add some other debugging options. For example, if an object doesn't update references correctly another object can be allocated in to the slot. E.g:

A -> B
C -> B

Maybe B moves and C updates it's references but A does not. If a new object D is allocated in the slot where B used to live, then we could end up with:

A -> D
C -> B

I have a way to debug this locally, but no good solution for upstream yet. It's probably off topic for this issue, but my point is that we have multiple techniques for debugging this so I think a method that takes options is best.