# CMSC 131 Lab, Notes Week 11, Monday

## Notes for TAs

* 1. Every week after Wednesday’s lab please send us an email letting us know how things went in lab. Please include a brief summary of what questions you think they still have, what went well, what didn't, etc.
  2. Please do not provide these lab notes to the students.

## Lecture Example

* 1. Please review the concepts of reference copying, shallow copying and deep copying. Please see the definition of shallow copy included at the end of these notes. The lectures slides covering these topics are:

<http://www.cs.umd.edu/class/fall2015/cmsc131/content/lectures/Week10/CopyingObjects.pdf>

* 1. Go over the example in the package cdexample available in the code distribution:

<http://www.cs.umd.edu/class/fall2015/cmsc131/content/lectures/Week10/CopyingObjectsCode.zip>

## Midterm Material

Address any questions about the midterm material.

## Office Hours

* 1. Hold in-lab office hours if you have any time left. Any students without any questions can leave.
  2. Address any questions students may have. Don’t stay sitting at the front desk; go around asking students whether they have any questions.

## Shallow Copy Information

From Wikipedia: <https://en.wikipedia.org/wiki/Object_copying#Shallow_copy>

### Shallow copy[[edit](https://en.wikipedia.org/w/index.php?title=Object_copying&action=edit&section=2)]

One method of copying an object is the shallow copy. In that case a new object B is [created](https://en.wikipedia.org/wiki/Object_creation), and the fields values of A are copied over to B.[[3]](https://en.wikipedia.org/wiki/Object_copying#cite_note-3)[[4]](https://en.wikipedia.org/wiki/Object_copying#cite_note-4)[[5]](https://en.wikipedia.org/wiki/Object_copying#cite_note-5)[[6]](https://en.wikipedia.org/wiki/Object_copying#cite_note-6) If the field value is a reference to an object (e.g., a memory address) it copies the reference, hence referring to the same object as A does, and if the field value is a primitive type it copies the value of the primitive type. In languages without primitive types (where everything is an object), all fields of the copy B are references to the same objects as the fields of original A. The referenced objects are thus *shared*, so if one of these objects is modified (from A or B), the change is visible in the other. Shallow copies are simple and typically cheap, as they can be usually implemented by simply copying the bits exactly.