Decoding the classes: What you need to know to use.

## Inode:

Inode is the node that holds metadata of the file. Most importantly for this assignment, the pointers to its data blocks. This is denoted by the array: ptr[]

- ptr[] is the 13 slot array that holds the location on disk of the blocks (direct or indirect) (0-9 should direct blocks and 10,11,12 should be indirect blocks) that make the file.
- holes are entries in ptr[] that are not yet used. They are denoted with the integer 0.

## FreeMap:

FreeMap what keeps track of which blocks on the disk are unused. When writing new blocks of data, you must use the freemap to obtain unused blocks to write to if the block doesn't exist already(ie. if it is a hole).

- freemap.find() returns the location on disk of an unused block.
- freemap.save() is used to "commit" found blocks to be used. It is important to tell the freemap that the blocks that it gave you should not be given again in the future.

## DirectBlock:

DirectBlocks are blocks that hold data. What you will be returning in the method you are implementing. To understand the assignment, familiarize yourself with the constructors of DirectBlock and what the parameters mean.

- DirectBlock.hole is a specific type of direct block indicating that the DIRECT block that was searched for does not exist. (ie. if there is a 0 in the Inode ptr[])

## IndirectBlock:

IndirectBlocks are blocks that hold the locations of other (both indirect and direct) blocks on disk.

- IndirectBlock.COUNT is a field denoting the number of Block locations the IndirectBlock can hold.
- indirectblockobject.ptr[] is the array that holds the blocknum(on disk) of other blocks (comparable ptr[] for inode). The length of ptr[] is IndirectBlock.COUNT.

## Disk:

The disk is what you will use to read from existing blocks and write to both new and existing blocks. There are two methods you must be familiar with.

- disk.read(int blocknum, Block block) reads the contents of the block in location blocknum on disk into the provided block (either indirect or direct)
  data from Disk →Block object
- disk.write(int blocknum, Block block) writes the given Block "block" into the block in disk with location blocknum.
  data from Block object →Disk