Project 6 Frequently Asked Questions

1. **Q: I am getting weird things - nothing on the indirect, magic number not working, etc.? What is wrong?**

   Double check that you have the latest versions of fs.h, disk.c, disk.h, and shell.c. Your images should also generate the following md5 checksums (get via md5sum):

File	MD5 Checksum
image.10	2aa2486a6ad298aaa201b7d93b7e1780
image.25	7907400fc56464a49d601f015ed533b6
image.100	27a2a41d6e0f575a6b18b9a113c6ff5d
shell.c	1d253c651b3a174762e6b65f73c8c164
fs.h	fbf70334857c70fd00db0381915e7549
disk.h	c7916d0d0cd7af76beb8337758dd3bf7
disk.c	56ce8fc59cfadae0d850ee89f0c3f6c2

2. **Q: The visual has 5 direct pointers per inode but the setting via fs.h has 3 per inode. Which is it?**

   A: It is three per inode as listed in the code.

3. **Q: What is the correct magic number?**

   A: It should be 0x30341003. The fs.c base code should output the magic number.

4. **Q: I think that I have debug working. Can you give me some sample output so that I can check my answer?**

   A: Sure, here is what our solution produces for image.10:

```
    simplefs> debug
superblock:
    10 blocks
    2 inode blocks
    256 inodes
inode 2:
    size: 9546 bytes
    direct blocks: 3 4 5
inode 4:
    size: 1005 bytes
    direct blocks: 6
```

A: Sure, here is what our solution produces for image.25:

```
 simplefs> debug
superblock:
    25 blocks
    3 inode blocks
    384 inodes
inode 1:
    size: 1523 bytes
    direct blocks: 4
inode 3:
    size: 67478 bytes
    direct blocks: 5 6 7
    indirect block: 8
    indirect data blocks: 9 10 11 12 13 14 15 16 17 18 19 20 21 22
```

5. **Q: The project description says that format should set aside 10 percent of the blocks for inodes, but the example filesystems have 10 percent plus one block. Which is correct?**

   A: Both are correct! In a real filesystem, the number of inode blocks is chosen when the disk is formatted, so an operating system must be prepared to deal with whatever number of inode blocks are specified in the superblock. However, your format routine should always compute ten percent.

6. **Q: What should happen if the user tries to format a disk that is already mounted?**

   A: In this case, the format call should do nothing and return failure.

7. **Q: Is it necessary for format to zero out every block on the disk?**

   A: No. Format should do the minimum work necessary to create a blank filesystem. A blank filesystem should have a valid superblock and no valid inodes. It doesn't matter what's in the data blocks, since no inodes refer to them.

   Now, if you were really worried about someone stealing your disk and then writing their own code to examine the data in blocks that were not in use, then a really aggressive filesystem would actually fill unused blocks with zeroes during a format or a delete.

8. **Q: Can you delete an inode that is not being used i.e. not valid?**

   A: A delete on an invalid inode should do nothing and return failure.