Question 4. [6 MARKS]

A program issues the following system calls on the original Fast File System (FFS) file system. The file system block size is 4KB and the size of an inode is 64 bytes. Assume that the file system is already mounted when the program starts and that all system calls succeed.

```
char buf[4096];
int fd = open("/a/b/c", 0); // open in read-only mode
lseek(fd, 1034*4096, 0);    // seek to position (1034*4096) from start of file
read(fd, buf, 4096);        // read 4k of data from file
```

Assume that the file buffer cache is empty. State the *minimum* number of the following types of blocks that will be read when the program above is run. Explain your answer for each block type.

1. Inode block(s)

2. Directory block(s)

3. Indirect block(s) (include single, double or triple indirect)

4. Other data block(s)

Question 5. File system consistency [9 MARKS]

On an ext2 or FFS file system, consider the operation of creating a new directory in an existing directory. Assume the existing directory occupies one block and there is enough space to add a new entry. Assume the existing directory inode and the new directory inode are in different disk blocks.

Part (a) [2 MARKS]

Which of the following blocks must be updated? Check all that apply.

- ☐ inode bitmap
- ☐ new directory inode
- ☐ new directory inode
- ☐ new directory data block
- ☐ existing directory inode
- ☐ existing directory data block

Part (b) [2 MARKS]

What data is updated in the existing directory inode?

last modified time, number of links

Part (c) [5 MARKS]

In each of the remaining questions, check all of the boxes that most closely explain what happens if a crash occurs after updating only the block(s) specified.

Inode Bitmap and Data Block Bitmap

- ☐ No inconsistency (it simply appears that the operation was not performed)
- ☐ Data leak (data block is lost for any future use)
- ☐ Inode leak (Inode is lost for any future use)
- ☐ Inconsistent inode data (Some inode field does not match what is stored in data blocks)
- ☐ Multiple file paths may point to same inode
- ☐ Something points to garbage

New Directory Inode

- ☐ No inconsistency (it simply appears that the operation was not performed)
- ☐ Data leak (data block is lost for any future use)
- ☐ Inode leak (Inode is lost for any future use)
- ☐ Multiple file paths may point to same inode
- ☐ Inconsistent inode data (Some inode field does not match what is stored in data blocks)

- ☐ Something points to garbage

Inode Bitmap, Data Block Bitmap, Existing Directory data, New Directory inode, and New Directory data

- ☐ No inconsistency (it simply appears that the operation was not performed)
- ☐ Data leak (data block is lost for any future use)
- ☐ Inode leak (Inode is lost for any future use)
- ☐ Multiple file paths may point to same inode
- ☐ Inconsistent inode data (Some inode field does not match what is stored in data blocks)
- ☐ Something points to garbage

Inode Bitmap and New Directory inode

- ☐ No inconsistency (it simply appears that the operation was not performed)
- ☐ Data leak (data block is lost for any future use)
- ☐ Inode leak (Inode is lost for any future use)
- ☐ Multiple file paths may point to same inode
- ☐ Inconsistent inode data (Some inode field does not match what is stored in data blocks)
- ☐ Something points to garbage

New Directory inode, Existing Directory inode, and Existing Directory data

- ☐ No inconsistency (it simply appears that the operation was not performed)
- ☐ Data leak (data block is lost for any future use)
- ☐ Inode leak (Inode is lost for any future use)
- ☐ Multiple file paths may point to same inode
- ☐ Inconsistent inode data (Some inode field does not match what is stored in data blocks)
- ☐ Something points to garbage

Part (e) [1 MARK]

Indexed-based file systems suffer from external fragmentation because blocks of the file may be scattered across the disk.

Part (f) [1 MARK]

Extent-based file systems may require more disk block accesses than indexed-based files systems for random access because it may need to traverse an extent to get to a particular byte in the file.

Q7. [16 marks] File systems [25 minutes]

Consider the following ext2 file system, as discussed in class (you can ignore the absence of the superblock and block group regions).

Inode bitmap (IB)	Data bitmap (DB)	Inode table (I)	Data region (D)

1) [5 marks] A user creates a new (empty) file A in a given directory. You can assume that the file name is not taken, that the directory does exist, and that the file system is consistent at this point.

i) Which of the regions above must be updated for the operation to be complete?

ii) Imagine that a crash can occur during this operation, and that we have no tool for checking file system consistency. In which order should the updates happen, such that we minimize the potential negative impact on the file system? Explain your rationale **in detail**.

2) [5 marks] Imagine a deletion operation for a file B. You can assume that the file exists and that no file system inconsistency is present at this time.

i) Which of the regions above must be updated for the operation to be complete?

ii) Imagine that a crash can occur during this operation, and that we have no tool for checking file system consistency. In which order should the updates happen, such that we minimize the potential negative impact on the file system? Explain your rationale **in detail**.

3) [3 marks] What was the main motivation behind the design of the Log-structured File System (LFS)? How are updates carried out on LFS? In your explanation, compare to how updates are carried out in the Fast File System (FFS).

4) [3 marks] Describe what is the challenge in locating data and metadata on disk in LFS. Additionally, explain in detail how this is done in practice.