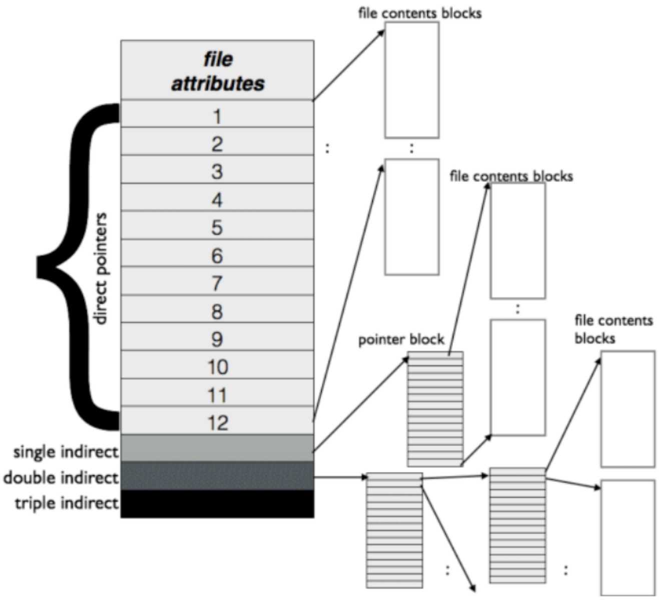
1. a) 1) 4 - inode blocks. 1 for the file `c`, and 3 for the directdories `/`, `a`, `b`
   2) 3 - directory blocks - one for root `/`, one for `a`, the other for `b`
   3) 1 - single indirect block as far as we know. The file definitely has more than 12 blocks (# of data blocks pointed by direct pounters), but less than 1036 (# of data blocks pointed by direct pointers and single indirect pointers). We are reading block 1034.
   4) 1 - data block for file `c`

   b) All of the above

Notes

- **Inode**



  - Is short form of **index node**
  - Describes a file system object such as file or data
  - Contains all information about a file/directory, including
    - File Type,
    - Size
    - Number of blocks allocated to it
    - Protection information
    - Time information (e.g time created, time modified)
    - Location of data blocks residing on disk

References

1) Wikipedia, Inode, link
2) Machanick, Philip. (2016). Teaching Operating Systems: Just Enough Abstraction. 642. 10.1007/978-3-319-47680-3_10., link

c) Size, the location of data blocks that reside on disk

Notes

- I wonder what information about blocks inode has. Is it total number of blocks both inode and data, or just data?
- I struggled a bit on this one. I should find an easier way to remember which information inode has

d) **Notes**

- I wonder how system call for deleting file in UNIX works
- I wonder how system call for adding file in UNIX works
- **Creash Scenarios**
  - When only new data block is written to disk
    - This is fine in system's point of view
    - No inode points to it (it doesn't contain any information about file)
    - No bitmap points to it
    - Is as if write never occured
  - When only the updated inode is written to disk
  - When only inode bitmap is written to disk
  - When only data bitmap is written to disk