

## Simulação do Sistema de Arquivos utilizado Inode

### Instruções:

- o projeto deve ser realizado por **dois alunos**;
- ver datas de entrega e apresentação na AVA;
- Tem o valor de 2,0 pontos na média final;
- Deverá ser entregue via AVA – Este deverá ser utilizado no dia para apresentação;
  - Deverá ser entregue um arquivo zipado com o código fonte e um documento do word com as informações do grupo e detalhes do código fonte.
- O grupo deverá implementar em linguagem de **programação C++**.

### Descrição:

Este projeto prático tem como finalidade realizar a simulação referente a implementação do Sistema de Arquivos utilizado Inode.

Visando restringir o tamanho do disco este possui 1000 blocos de 10 bytes cada. Conforme se pode observar na figura abaixo, tem-se um arquivo de diretório com as entradas de diretório contendo o nome do arquivo e seu ponteiro para o inode (figura a). Na figura b apresenta-se um arquivo de inodes.

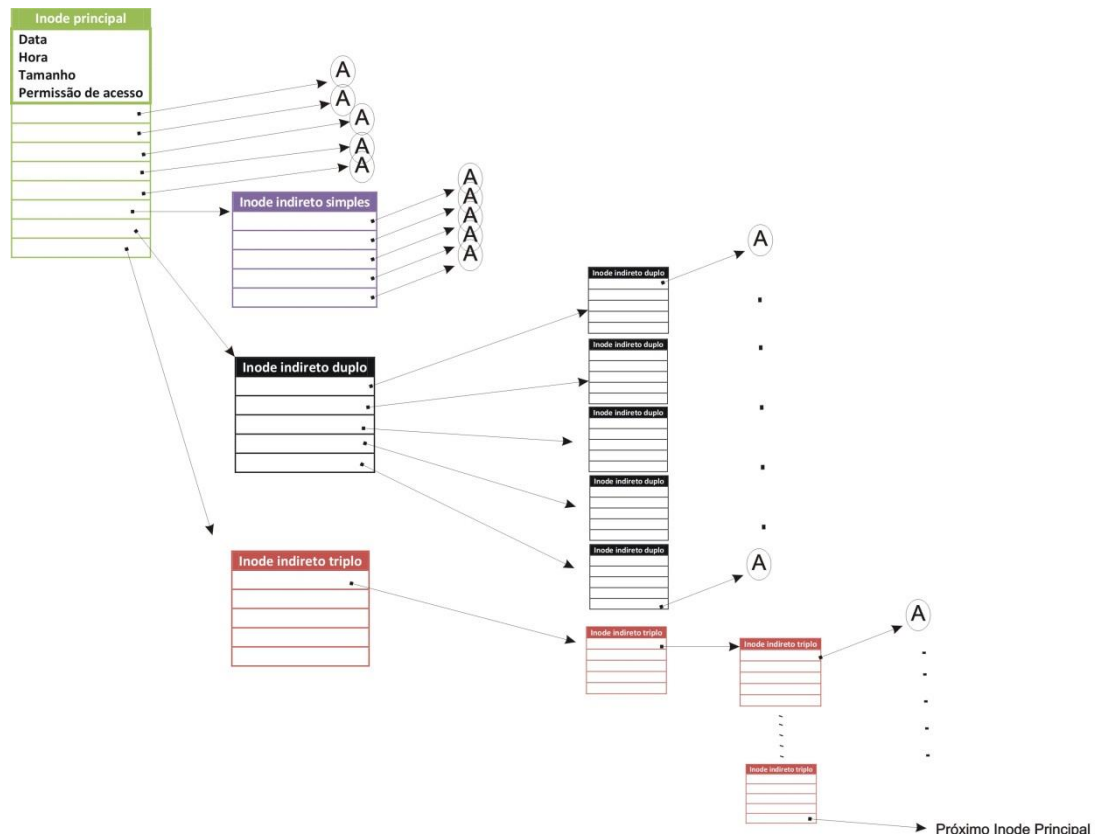
Cada Inode possui 8 ponteiros, sendo os 5 primeiros para alocação direta, o 6º ponteiro para alocação simples indireta; o 7º para alocação dupla indireta e o 8º para alocação tripla-indireta. Cada inode ocupa um bloco em disco. Cada inode principal pois um área de atributos com os seguintes campos: data, hora, tamanho, permissões (10 bits, onde o primeiro pode ser – ou d ou l, seguido dos do usuários (u) RWX, seguido do grupo (g)RWX e dos outros (o)RWX), contador de links físicos, tipo do bloco;

Figura (a)

Arquivo Raiz / início

Readme.txt	16
Ícone.gif	14
programa	3
Retrato.jpg	29
Relat.pdf	6
Format.exe	31
Carta.doc	67
Programa.c	73
Boot	4
bin	62

Figura (b)



[illegible]

- O arquivo Readme.txt tem início no bloco do disco número 16. Neste bloco existe um inode (indicado pela letra I) definido como inode principal número 16. Neste inode temos alguns atributos referentes a este arquivo bem como os índices dos blocos correspondentes ao arquivo. Neste caso, o arquivo possui 6 blocos (7, 12, 23, 26, 1 e 58). Como nosso sistema de arquivo possui inodes com 5 entradas diretas, os cinco primeiros blocos ficam no índice direto (7, 12, 23, 26 e 1). O sexto bloco (58) necessitará ser indexado com o uso do inode indireto simples. Entretanto, para indexar o inode indireto simples, no inode principal deve existir uma entrada, que neste caso é para o bloco 45. Neste bloco está o inode indireto simples e nele está a entrada 58 para o último bloco do arquivo.
- A entrada de diretório Boot tem início no bloco número 4. Neste bloco temos um inode de diretório (indicado pela letra D) definido como inode principal número 4. Neste inode temos alguns atributos referentes a este diretório bem como os índices dos blocos correspondentes. Neste caso, o arquivo de diretório ocupa apenas um bloco número com 70. Ao ler este arquivo temos acesso ao conteúdo do diretório boot. Neste arquivo de diretório temos uma entrada para o arquivo kernel que tem seu início no bloco 35. Neste bloco é encontrado o inode de arquivo para este arquivo.

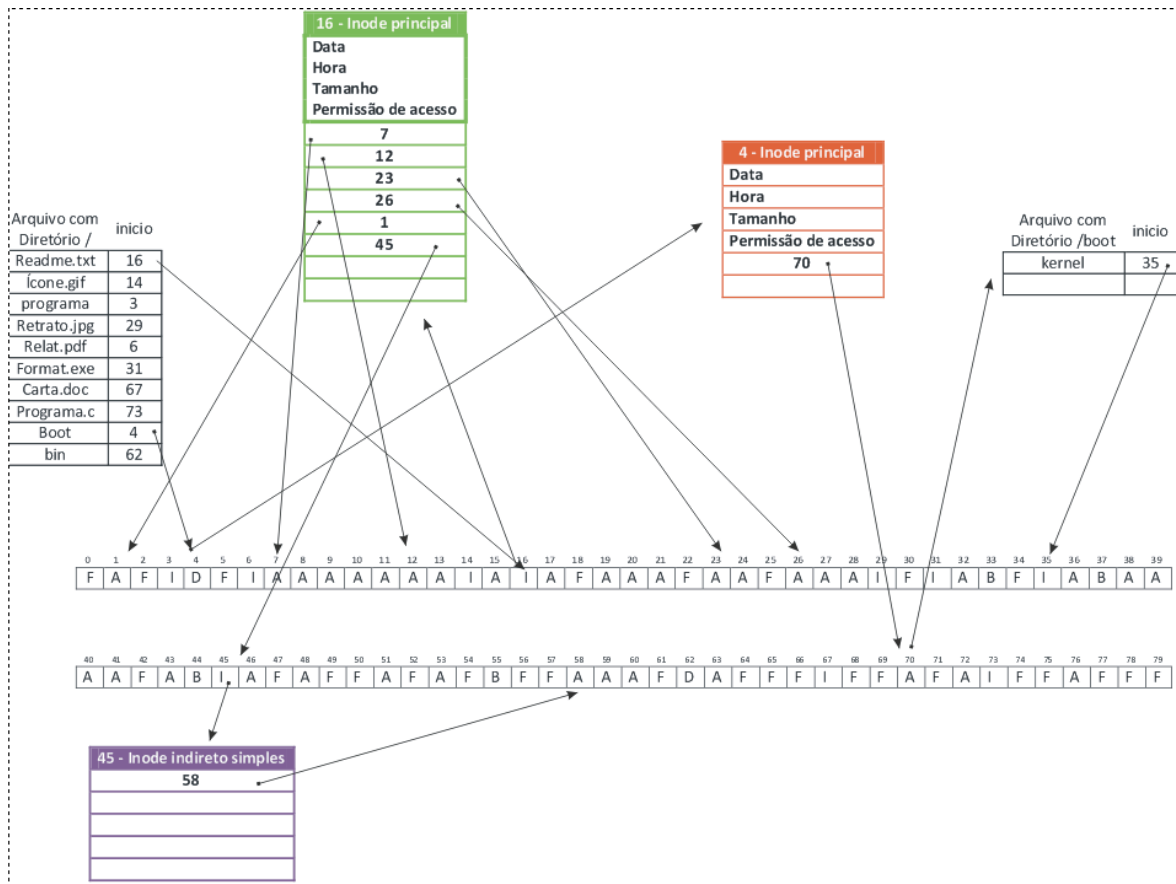


Figura D.

Os demais arquivos e diretório encontrado na raiz do disco possuem os seguintes blocos:

- Ícone.gif: 14-63-51-46
- programa: 3-19
- Retrato.jpg: 29-36-20-33
- Relat.pdf: 6-17-28-11-53-59-72-10-13
- Format.exe: 31-37
- Carta.doc: 67-60-44
- Programa.c: 73-27-55
- bin: 62-39

Por meio de um prompt de comando o usuário poderá manipular o sistema de arquivo:

- |  |   |
|--|---|
| - chmod (+)(-)ugo RWX                          | alterar as permissões de acesso a arquivos e diretórios |
| - vi nomeArquivo                               | visualizar um arquivo regular                           |
| - ls   | listar os nomes dos arquivos no diretório               |
| - ls -l  | listar os nomes dos arquivos com seus atributos         |
| - mkdir NomeDir                                | criar diretórios  |
| - rmdir NomeDir                                | deletar diretórios que estejam vazios                   |
| - rm NomeArq                                   | deletar arquivos  |
| - cd NomeDir ou . ou ..                        | navegar nos diretórios                                  |
| - link -h nomeArquivoOrigem NomeArquivoDestino | criar link físico                                       |
| - link -s nomeArquivoOrigem NomeArquivoDestino | criar link simbólico                                    |
| - unlink -h                                    | remover link físico                                     |
| - unlink -s                                    | remover link simbólico                                  |
| - bad numeroBloco                              | transformar um bloco em Bad                             |
| - touch NomeArquivo TamanhoBytes               | Criar um arquivo Regular                                |

apresentar em bytes espaço livre e ocupados do disco

A aplicação deverá permitir que o usuário:

- Defina inicialmente a quantidade de blocos do disco;
- Implemente uma solução para recuperar os blocos livres do disco que retorne uma posição livre aleatoriamente. Toda vez que um novo bloco for necessário esta função deve retornar um número de bloco livre. Toda vez que um arquivo for deletado, seus blocos deve retornar para os livres. Utilize uma pilha como estrutura;
- Cadastrar um novo arquivo ou diretório: nome de arquivo ou diretório e o seu tamanho em bytes, alocando este arquivo no disco, caso haja espaço. Lembre-se que se deve criar os inodes e inserir seus índices de forma automática. Pode ser necessário usar inodes indiretos simples, duplos ou triplos. No caso de ser um diretório, acrescente neste diretório pelo menos dois arquivos;
- Escolher um bloco pelo seu número para ser colocado como do tipo B – defeituoso;
- Delete um arquivo existente liberando todos os seus blocos (coloca F nos blocos) e retornando os blocos para a lista de blocos livres;
- Crie uma função que permite criar arquivos ou diretórios em qualquer lugar da árvore de diretórios a escolha do usuário;
- Permitir ao usuário criar links simbólicos e links físicos;

O usuário também poderá escolher visualizar alguns relatórios:

1. o número de blocos ocupados por um arquivo escolhido por usuário;
2. o tamanho (em blocos) do maior arquivo que ainda pode ser criado nesse disco;
3. quais arquivos estão íntegros e quais estão corrompidos por blocos defeituosos (badblocks);
4. apresentar quantos blocos do disco estão perdidos e quais são eles, ou seja, não são usados por arquivos e nem estão marcados como livres ou defeituosos. Não esqueça de apresentar o espaço de disco perdido em bytes;
5. Imprima todos os blocos em seu estado atual, igual a figura b.
6. Visualize os arquivos e diretórios alocados, apresentando o nome e seus números do bloco correspondentes identificando o tipo. Esta visualização deve ser algo semelhante ao Windows explore. Não precisa ser gráfico.
7. Visualizar a árvore de diretório igual a figura D, apresentado as estruturas com seus atribuídos, nomes e números;
8. Visualizar os links simbólicos e físicos criados, detalhando as entradas de diretório, inodes e números;