Universidade do Oeste Paulista
UNOESTE
Presidente Prudente - SP
Campus I Rua José Bongiovani, 700
Cidade Universitária CEP 19050 900
Fone(18) 3229 1000

Faculdade de Informática de Presidente Prudente
Fone (18) 3229 1060
www.unoeste.br/fipp
e-mail: fipp@fipp.unoeste.br
CAMPUS I

Simulação do Sistema de Arquivos utilizado Inode

Instruções:

- o projeto deve ser realizado por **três 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 sem uso de unions ou outras estruturas complexas. Deve-se usar apenas struct, lista encadeada e pilha.**

Descrição:

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

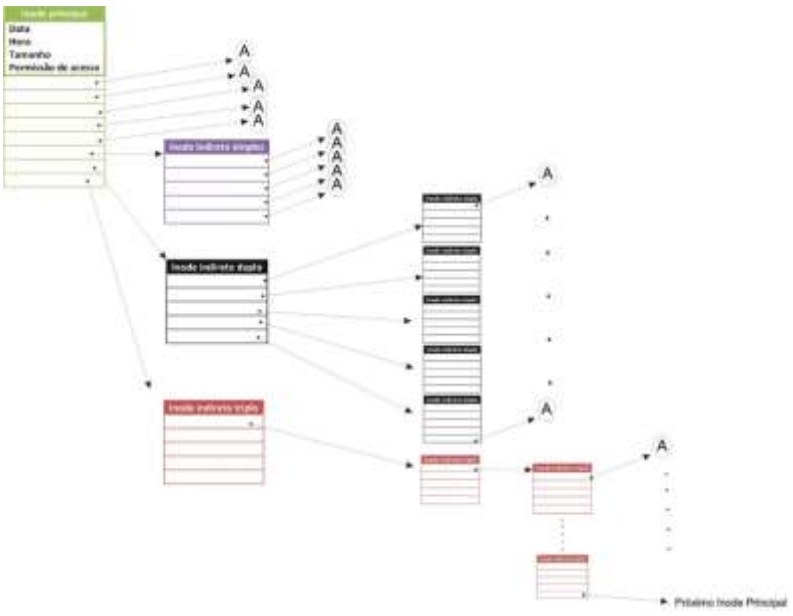
O tamanho do disco é de 1000 blocos de 10 bytes cada (10 mega), mas o usuário pode definir outro tamanho no início da execução. Conforme se pode observar na figura abaixo, tem-se um arquivo de diretório com as entradas de diretório contendo o nome do arquivo e seu ponteiro para o inode (figura a). Cada bloco do disco consegue armazenar um arquivo de diretório com no máximo 10 entradas, caso tenha mais que 10, deverá ser guardado em mais de um bloco.

Na figura b apresenta-se um arquivo de inodes. Cada inode ocupa um bloco em disco e o inode principal possui um header de atributos com os seguintes campos: as permissões (10 bits, onde o primeiro pode ser – ou d ou l, seguido dos bits de permissão: usuário dono (u) RWX, seguido do grupo (g) RWX e dos outros (o) RWX), data, hora, tamanho, nome do usuário, nome do grupo e contador de links físicos. O inode principal 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. Já o inode de extensão possui apenas 5 ponteiros.

Figura (a)

Arquivo Raiz /	inicio
.	1
..	1
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)



Os inodes e arquivos ficam gravados em blocos em disco conforme a figura c abaixo.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
F	A	F	I	I	F	I	A	A	A	A	A	A	A	I	A	I	A	F	A	A	A	F	A	A	F	A	A	A	I	F	I	A	B	F	I	A	B	A	A

40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
A	A	F	A	B	I	A	F	A	F	F	A	F	A	F	B	F	F	A	A	A	F	I	A	F	F	F	I	F	F	A	F	A	I	F	F	A	F	F	F

Figura C.

Como se pode visualizar na figura c, um bloco pode ser do tipo **F** que indica bloco livre (Free), **B** que indica bloco defeituoso (Bad), **I** que indica inode de arquivo (principal ou extensão) e **A** que indica arquivos (arquivos regulares ou arquivos de diretório).

A figura D abaixo apresenta o funcionamento básico do sistema de arquivos com inode. Com base no arquivo de diretório da raiz (*/*) já carregado na memória, o usuário tem acesso a seus arquivos locais e a outros diretórios contidos na raiz. Vamos exemplificar o funcionamento para uma entrada de diretório correspondente a um arquivo e um diretório.

- 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 existe 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.

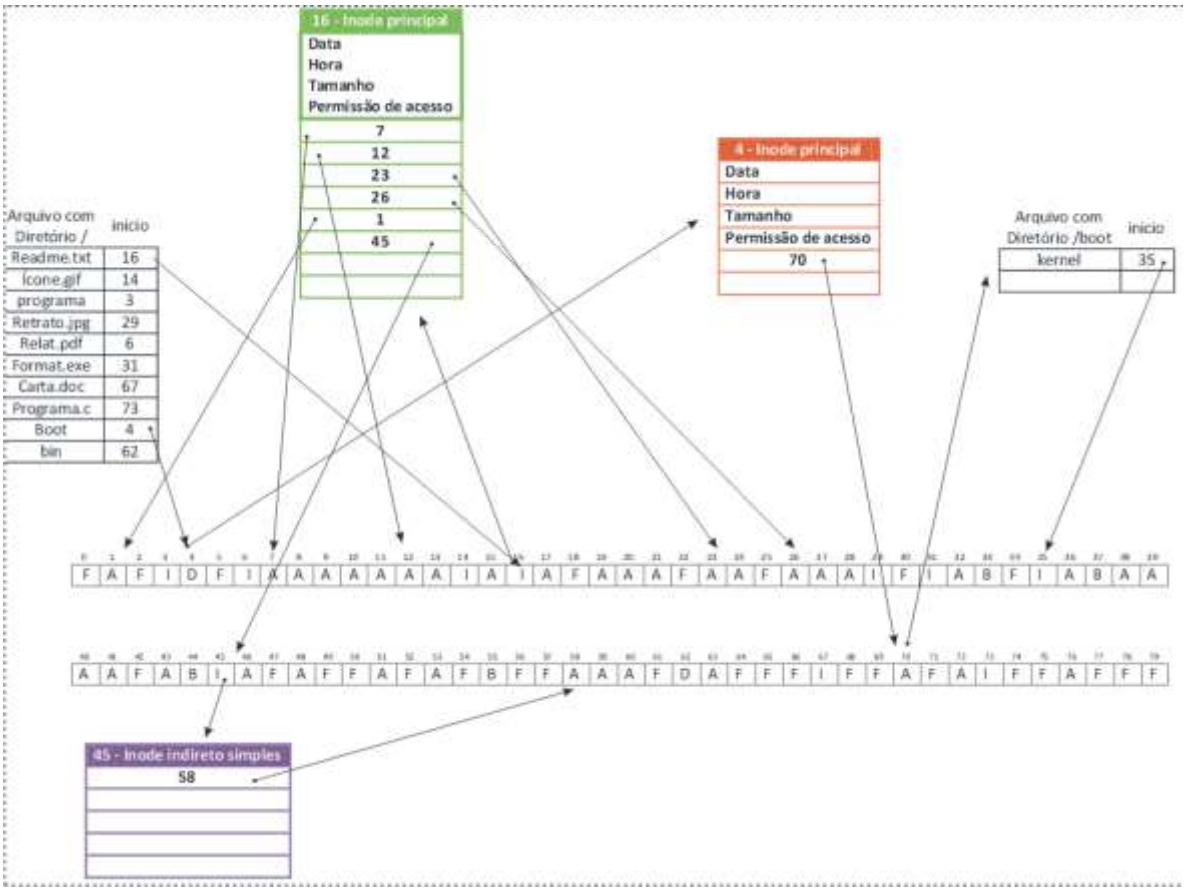


Figura D.

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 utilizando os seguintes comandos:

- chmod (+)(-)ugo RWX	Alterar as permissões de acesso a arquivos e diretórios
- vi nomeArquivo	Visualizar um arquivo regular. Se o arquivo na estiver corrompido apenas deve ser apresentado um msg dizendo que o arquivo foi visualizado. Não é para abrir o arquivo realmente.
- ls	Listar os nomes dos arquivos/diretório do arquivo de diretório
- ls -l	Listar os nomes dos arquivos/diretórios 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 de forma absoluta ou relativa

- link –h nomeArquivoOrigem NomeArquivoDestino	criar link fisico
- 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 NomeAruivo TamanhoBytes	Criar um arquivo Regular
- df	Apresentar em bytes espaço livre e ocupados do disco

A aplicação deverá permitir que o usuário:

a) Defina inicialmente a quantidade de blocos do disco;
b) 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 retorno um número de bloco livre. Toda vez que um arquivo for deletado, seus blocos deve retornar para os livres. Utilize uma pilha como estrutura;
c) 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;
d) Escolher um bloco pelo seu número para ser colocado como do tipo B – defeituoso;
e) Delete um arquivo existente liberando todos os seus blocos (coloca F nos blocos) e retornando os blocos para a lista de blocos livres;
f) Crie uma função que permite criar arquivos ou diretórios em qualquer lugar da arvore de diretórios a escolha do usuário;
g) Permitir ao usuário criar links simbólicos e links físicos. Vale lembrar que o link simbólico deve ter um arquivo comum contendo o caminho da raiz até a entrada do diretório do arquivo que se quer criar o link. Já o link físico deve fazer a entrada apontar para o inode do arquivo desejado. Não esqueçam de verificar as permissões se pode ter acesso.

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;