MAC0422 – Sistemas Operacionais – 2s2020
EP3
Data de entrega: 7/12/2020 até 8:00 da manhã

Prof. Daniel Macêdo Batista

1 Problema

A tarefa neste EP é implementar um simulador de sistemas de arquivos com vários comandos que simulem ações com arquivos dentro de tal sistema. O código pode ser escrito em qualquer linguagem de programação desde que haja compilador livre e gratuito para GNU/Linux. A interação com o usuário deve ser feita inteiramente no shell, sem interface gráfica, usando um prompt de comandos para enviar comandos para o simulador.

Os arquivos simulados no sistema de arquivos podem ser arquivos regulares ou diretórios e precisam ter os seguintes atributos armazenados:

- Nome
- Tamanho em bytes (menos no caso de diretórios)
- Instante de tempo em que foi criado
- Instante de tempo em que foi modificado
- Instante de tempo em que foi acessado

Além é claro dos dados de tais arquivos. No EP seu sistema de arquivos tratará apenas com arquivos em texto puro que serão copiados de algum local do sistema de arquivos real do computador.

2 Interação com o simulador

Quando executado na linha de comando (sem parâmetros) o simulador deve fornecer o prompt:

```
[ep3]:
```

Neste prompt os seguintes comandos precisam ser implementados:

- `mount arquivo`: monta o sistema de arquivos simulado que está no arquivo `arquivo`. Por exemplo, se o seu sistema de arquivos simulado estiver em `/tmp/unidadesimulada`, no seu computador, o comando será: `mount /tmp/unidadesimulada`. Se o `arquivo` já existir, o sistema de arquivos nele deve ser lido e a simulação deve continuar com o conteúdo já existente. Se o `arquivo` não existir, um novo sistema de arquivos simulado deve ser criado. Os comandos a seguir só podem ser executados caso o arquivo simulado tenha sido aberto com sucesso.

- `cp origem destino`: cria dentro do sistema de arquivos simulado uma cópia do arquivo `origem` que está em algum sistema de arquivos real dentro do seu computador. No sistema de arquivos simulado a cópia de `origem` será salva em `destino`. Tanto `origem` quanto `destino` devem ser informados com o caminho completo tanto dentro do sistema de arquivos real quanto no simulado[1]. Esse comando será executado apenas para copiar arquivos em texto puro.
- `mkdir diretorio`: cria o diretorio `diretorio`.
- `rmdir diretorio`: apaga o diretório `diretorio`. Se o diretório não estiver vazio, apaga tudo que estiver embaixo e avisa para o usuário tudo que tiver sido apagado
- `cat arquivo`: mostra o conteúdo do arquivo `arquivo` na tela.
- `touch arquivo`: se o arquivo `arquivo` existir, atualiza o instante de tempo de acesso para o instante de tempo atual. Se não existir, cria um arquivo vazio.
- `rm arquivo`: remove o arquivo `arquivo`.
- `ls diretorio`: lista os arquivos e diretórios que estejam "embaixo" do diretório `diretorio`. Na listagem que será exibida, para todos os arquivos existentes, deverão ser exibidos: nome, tamanho em bytes e data de última modificação. Diretórios devem ser exibidos com alguma informação a mais que os diferencie como diretórios.
- `find diretorio arquivo`: busca **a partir de `diretorio`** se há algum arquivo com nome `arquivo`. Todos os arquivos encontrados devem ser exibidos na tela com os seus caminhos completos (apenas o nome dos arquivos, não o conteúdo)
- `df`: imprime na tela as seguintes informações do sistema de arquivos: quantidade de diretórios, quantidade de arquivos, espaço livre, espaço desperdiçado (considerando o espaço a mais necessário para cada arquivo ocupar exatamente múltiplos do tamanho de 1 bloco)
- `umount`: desmonta o sistema de arquivos
- `sai`: sai do simulador

Note que, uma vez finalizado o simulador, o arquivo que simula o sistema de arquivos não pode ser apagado. Ele deve continuar no estado em que foi deixado na última execução.

3 Requisitos

O sistema de arquivos simulado equivale a uma única partição em um disco sem a necessidade de simular questões referentes a boot de um sistema operacional. Dessa forma, nenhum campo relativo a boot precisa ser simulado.

O sistema de arquivos simulado tem que suportar até 100MB, tanto de dados válidos quanto de metadados. Os tamanhos de blocos devem ser de 4KB. Se a execução de algum comando extrapolar o limite de 100MB, ela deve ser interrompida e uma mensagem de erro deve ser exibida.

[1]Em todos os comandos a seguir, quando houver necessidade de especificar o nome de algum arquivo ou diretório, ele deve ser fornecido com o caminho completo

Com relação a diretórios, o sistema deve ter uma estrutura hierárquica começando no '/'. O caractere '/' deve ser o caractere separador entre diretórios.

Com relação ao armazenamento dos arquivos, deve ser feito usando FAT.

Com relação à implementação de diretórios, deve ser feita de modo que cada diretório seja uma lista com 1 entrada para cada arquivo dentro do diretório.

Com relação à gerência de espaço livre, deve ser feita usando bitmap.

Toda a simulação do sistema de arquivos e todas as operações nesse sistema devem ser implementadas do zero. EPs que utilizem bibliotecas ou similares que já implementem um sistema de arquivos ou os comandos terão nota ZERO

4 Experimentos

Após finalizada a implementação do seu EP você deverá realizar experimentos que meçam o tempo médio (média de 30 execuções com intervalo de confiança e nível de confiança de 95%) das seguintes operações:

1. Cópia de um arquivo de 1MB no '/'
2. Cópia de um arquivo de 10MB no '/'
3. Cópia de um arquivo de 30MB no '/'
4. Remoção de um arquivo de 1MB no '/'
5. Remoção de um arquivo de 10MB no '/'
6. Remoção de um arquivo de 30MB no '/'
7. Remoção completa de um diretório "pai" com 30 níveis de hierarquia abaixo dele e sem arquivo regular em nenhum dos subdiretórios
8. Remoção completa de um diretório "pai" com 30 níveis de hierarquia abaixo dele e com centenas de arquivos regulares em todos os subdiretórios

Com o sistema de arquivos em 3 estados diferentes:

1. Sistema de arquivos vazio
2. Sistema de arquivos com 10MB ocupados
3. Sistema de arquivos com 50MB ocupados

5 Sobre a entrega

Deve ser entregue um arquivo .tar.gz contendo os itens listados abaixo. EPs que não contenham **todos** os itens abaixo **exatamente como pedido** terão nota ZERO e não serão corrigidos. **A depender da qualidade do conteúdo entregue**, mesmo que o EP seja entregue, **ele pode ser considerado como não entregue, o que mudará o cálculo da média final**:

- código-fonte;
- arquivo LEIAME **em formato texto puro** explicando como compilar e executar o programa;
- Makefile ou similar para facilitar a compilação do código-fonte e a geração do binário do programa;
- apresentação **em .pdf** para ser apresentada em no máximo 15 minutos resumindo os resultados obtidos nos experimentos (use gráficos!). Informações sobre o hardware e o SO do computador usado nos experimentos também devem ser fornecidas. Outras informações que forem julgadas como importantes, principalmente relacionadas com decisões de projeto que não ficaram especificadas no enunciado também podem ser apresentadas nos slides. **Esses slides não serão apresentados. Eles devem ser preparados supondo que você teria que apresentá-los.**

O desempacotamento do arquivo .tar.gz deve produzir um diretório contendo os itens. O nome do diretório deve ser ep3-membros_da_equipe. Por exemplo: ep3-joao-maria. EPs que não gerem um diretório ou que gerem o diretório com o nome errado perderão 2,0 pontos.

A entrega do .tar.gz deve ser feita através do e-Disciplinas.

O EP deve ser feito em dupla.

Obs.: não inclua no .tar.gz itens que não foram pedidos neste enunciado. Relatórios, saídas para diversas execuções e entradas usadas para testar o programa não devem ser entregues. No máximo um resumo sobre as entradas usadas pode ser colocado na apresentação, caso isso não ocupe muito espaço.