MAC0422 – Sistemas Operacionais – 2s2015
EP2
Data de entrega: 16/10/2015

Prof. Daniel Macêdo Batista

1 Problema

A tarefa neste EP é implementar um simulador de gerência de memória com diversos algoritmos para gerência do espaço livre e para substituição de páginas. 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 gerenciador.

O simulador de gerência de memória deve receber como entrada um arquivo de trace, em texto puro, que possui como primeira linha:

```
total virtual
```

seguida de várias linhas com o seguinte formato:

```
t0 nome tf b p1 t1 p2 t2 p3 t3 [pn tn]
```

`total` é o total de memória física que deve ser simulada e `virtual` é o total de memória virtual que deve ser simulada. `t0` é o instante de tempo em segundos que um processo chega no sistema, `nome` é uma string sem espaços em branco que identifica o processo, `tf` é o instante de tempo no qual o processo é finalizado, `b` é a quantidade de memória em bytes utilizada pelo processo.

Os valores `p0`, `t0`, ... `pn`, `tn` dizem respeito às posições de memória, no espaço de endereço "local" do processo, acessadas pelo processo. `p1`, `t1` por exemplo informa que no instante de tempo `t1`, a posição `p1` é acessada pelo processo.

Todos os valores no arquivo de entrada são números inteiros.

O simulador deve finalizar sua execução assim que todos os processos do arquivo de entrada forem finalizados.

Com relação aos algoritmos para gerência do espaço livre, neste EP o simulador deve implementar os seguintes algoritmos, considerando que a o controle de qual espaço está livre e qual está ocupado é feito usando lista ligada:

1. *First Fit*
2. *Next Fit*
3. *Quick Fit*

Com relação aos algoritmos de substituição de página, neste EP o simulador deve implementar os seguintes algoritmos, considerando que cada página na memória virtual e cada quadro de página na memória física tem um tamanho de 16 bytes:

1. *Not Recently Used Page*
2. *First-In, First-Out*
3. *Second-Chance Page*
4. *Least Recently Used Page*

2 Interação com o simulador

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

`[ep2]:`

Neste prompt os seguintes comandos precisam ser implementados:

- `carrega <arquivo>`: carrega o arquivo de nome `<arquivo>` para a simulação. Pode ser tanto o caminho relativo como absoluto do arquivo;
- `espaco <num>`: informa ao simulador que ele será executado com o algoritmo de gerenciamento de espaço livre de número `<num>`, de acordo com a numeração dos algoritmos apresentada anteriormente neste documento;
- `substitui <num>`: informa ao simulador que ele será executado com o algoritmo de substituição de páginas de número `<num>`, de acordo com a numeração dos algoritmos apresentada anteriormente neste documento;
- `executa <intervalo>`: executa o simulador e imprime o estado das memórias na tela de `<intervalo>` em `<intervalo>` segundos, juntamente com o conteúdo da lista ligada que mantém o status da memória;
- `sai`: finaliza o simulador e volta para o shell do sistema operacional.

A memória deve ser simulada utilizando o arquivo `/tmp/ep2.mem` para a memória física e o arquivo `/tmp/ep2.vir` para a memória virtual. Estes arquivos devem ser criados toda vez que o simulador for inicializado e devem ter inicialmente um tamanho igual aos valores `total` e `virtual` em bytes definido no arquivo de entrada do simulador. Este arquivo deve ser um arquivo **binário** e deve conter inicialmente diversos valores -1 informando que toda a memória está livre para ser usada. À medida que a memória for sendo utilizada pelos processos simulados, as posições utilizadas por esses processos devem ser marcadas com números inteiros que identifiquem unicamente cada processo. Toda vez que um processo for carregado na memória ele deve escrever o número único que identifica ele (seria o equivalente ao PID no Linux), nas posições do arquivo corretas. Toda vez que uma posição de memória for acessada, o PID também deve ser escrito na posição correta dos arquivos.

3 Requisitos

O EP pode ser feito em qualquer linguagem de programação desde que haja compilador/interpretador livre e gratuito para GNU/Linux. A interação com o usuário deve ser feita inteiramente no prompt do EP, sem interface gráfica. Bibliotecas, funções, frameworks e similares que já implementem os algoritmos solicitados não podem ser usados. Caso sejam usados, a nota do EP será ZERO.

4 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 10 minutos resumindo alguns resultados obtidos com diversos experimentos e que sejam suficientes para mostrar que os algoritmos funcionam como esperado e possuem o desempenho, em termos de complexidade de tempo, esperado também. 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.

O desempacotamento do arquivo .tar.gz deve produzir um diretório contendo os itens. O nome do diretório deve ser ep2-membros_da_equipe. Por exemplo: ep2-joao-maria.

A entrega do .tar.gz deve ser feita através do PACA.

O EP pode ser feito individualmente ou 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 as entradas podem ser colocadas na apresentação, caso elas não ocupem muito espaço.