
 INSTITUTO DE INFORMÁTICA UFG	Universidade Federal de Goiás Instituto de Informática Ciências da Computação	 UFG
<b>Professor:</b> Eduardo Simões de Albuquerque		
<b>Atividade:</b> exercício prático sobre gerenciamento de memória (espaço livre e alocação contígua)		

Esse exercício tem como objetivo desenvolver um pequeno sistema de simulação para compreender o gerenciamento de espaço livre através de lista encadeada e os algoritmos de alocação contígua de memória particionada dinâmica: *first fit*, *next fit (circular fit)*, e *best fit*. A seguir é descrito esse sistema hipotético, seu funcionamento e algumas estatísticas que serão extraídas do mesmo.

A memória do sistema é de 1 MB e está dividida em unidades de 4 KB. O tamanho de um processo pode variar entre 3 e 15 unidades de memória (uma unidade de memória possui 1KB). A simulação consiste em três componentes:

- **Memória** – implementa um algoritmo específico para alocação/desalocação de memória. Portanto, há 3 versões desse componente.
- **Geração de requisições** – gera requisições de alocação/desalocação.
- **Relatório de estatísticas** – exibe as estatísticas relevantes.

O componente de **memória** exporta (é acessível através) as seguintes funções:

1. `int alloc_mem (int PID, int mem_units)` – aloca `mem_units` unidades de memória para o processo cujo identificação é `PID`. Se a chamada tiver sucesso, a função retorna o número de nós atravessados na lista ligada. Caso contrário, a função retorna `-1`.
2. `int dealloc_mem(int PID)`: libera a memória alocada para o processo cujo identificador é `PID`. A função retorna `1`, em caso de sucesso, e `-1`, em caso de erro.
3. `int frag_count()`: retorna o número de buracos (fragmentos externos que tenham tamanho de 1 ou 2 unidades).

O componente de **geração de requisições** deve fornecer o PID do processo e a quantidade de unidades de memória a serem requisitadas. PID é um número sequencial que vai variar de 0 a 9.999. A quantidade de unidades de memória será gerada aleatoriamente, podendo variar 3 a 15 (apenas em valores inteiros). Para a desalocação, basta fornecer o PID. As requisições devem ser gravadas em um arquivo, para posterior leitura. Todos os algoritmos devem ser submetidos à mesma sequência aleatória. Devem ser gerados 3 cenários: baixa carga (25%), média carga (50%) e alta carga (75%). A carga é obtida através da probabilidade de uma requisição ser para alocar memória (ao invés de desalocar). Exemplo: com carga baixa, a probabilidade de uma requisição ser para alocação deve ser igual a 0,5.

O componente de **relatório de estatísticas** deve fornecer, para os 3 algoritmos, os seguintes parâmetros de desempenho: tamanho médio dos fragmentos externos, tempo médio em termos de número de nós atravessados na lista ligada até ocorrer a alocação e percentual de vezes que uma requisição de alocação falhou, pois não havia memória contígua suficiente. Esses parâmetros devem ser obtidos e atualizados a partir da inserção de cada requisição no componente de **memória**. Os valores devem ser guardados em um arquivo e posteriormente plotados usando Excel, LibreOffice ou o programa **gnuplot**.

Por fim, com base nos resultados, devem ser feitas considerações sobre cada um dos algoritmos avaliados.