# Tecnologia em Jogos Digitais

## Estrutura de Dados

### Simulador de Gerenciamento de Memória

O sistema operacional gerencia a memória controlando a memória disponível e a memória utilizada pelos programas que estão sendo processados (processos). O objetivo deste trabalho é implementar um simulador de gerenciamento de memória.

O gerenciamento da memória será controlado utilizando duas listas**: lista de blocos livres** e **lista de blocos alocados**. Inicialmente toda a memória está disponível, quando o usuário solicitar execução de um processo, é reservado um espaço na memória, isto é representado através da alocação de um novo nó na lista de memória alocada com exatamente o tamanho de memória necessária para execução do processo.

Antes do processo iniciar a execução o seu Simulador deve verificar se há memória disponível, caso exista deverá alocar de memória e garantir que enquanto o processo estiver executando outro processo não utilize este espaço na memória, para tanto deve ser atualizada as listas de blocos livres e blocos alocados conforme o exemplo da próxima página. Para cada processo é necessário que exista um espaço na memória contíguo (=consecutivo)

**Implementação**

No início do programa será informado ao simulador a quantidade de memória disponível. O Simulador realizar as seguintes operações:

- Alocar memória para execução do processo;
- Finalizar processo, ou seja, liberar memória que o processo estava usando;
- Imprimir na tela a situação atual da memória: blocos de memórias livres e blocos de memória alocados.

A solicitação de memória será feita pelo teclado, o usuário informará o número do processo e a quantidade de memória necessária. Para liberar a memória de um processa basta informar, através do teclado, o número do processo.

As listas de blocos livres e alocadas deverão ser implementadas utilizando uma **Lista Ligada**. Quando os blocos de memórias livres não forem contígüos, eles deverão estar ordenados pelo endereço inicial da memória, e quando for solicitado a alocação de um novo bloco, a escolha será feita pelo bloco livre que melhor satisfizer o tamanho solicitado. Na lista de blocos livres serão armazenadas as seguintes informações:

- endereço inicial da memória livre
- quantidade de memória livre

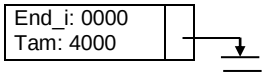
Quando o processo for finalizado o Simulador deve verificar se com o bloco de memória desalocado é possível formar com os seus vizinhos blocos maiores de memória contígüa. A lista de blocos alocados armazenará as seguintes informações:

- número do processo que está usando a memória
- endereço inicial da memória alocada
- quantidade de memória alocada

**Exemplo:**
Tamanho da Memória = 4000 bytes

**1º Instante**

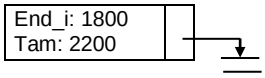
- Lista de Blocos de Memória Livre




- Lista de Blocos Memória Alocada = NULL (pois nenhum aplicativo está sendo executado)

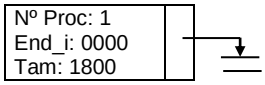
**2º Instante (alocação de memória para processo 1 que utiliza 1800 bytes de Memória)**

- Lista de Blocos de Memória Livre



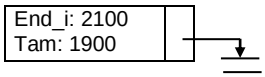

Antes de criar o bloco de memória alocada, deve-se verificar a existência da quantidade de memória solicitada. Após isso, é alterado o endereço inicial e tamanho do bloco livre, pois continuamos tendo uma memória contígua.

- Lista de Blocos Memória Alocada

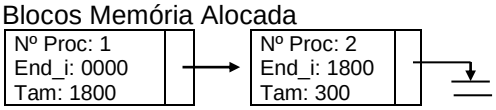



**3º Instante (alocação de memória para processo 2 que utiliza 300 bytes de Memória)**

- Lista de Blocos de Memória Livre

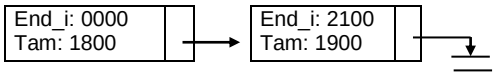



- Lista de Blocos Memória Alocada

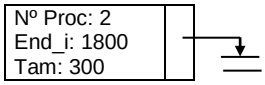



**4º Instante (finalização processo 1 que utiliza 1800 bytes de Memória)**

- Lista de Blocos de Memória Livre

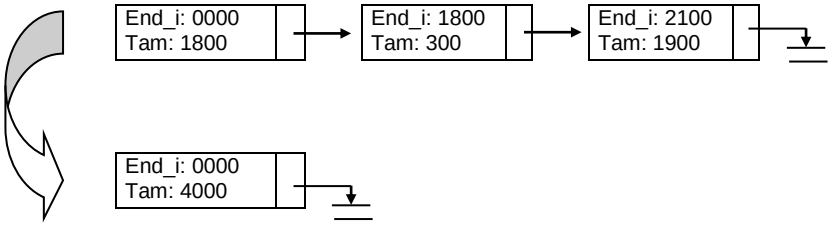



- Lista de Blocos Memória Alocada




**5º Instante (finalizar processo 2 que utiliza 300 bytes de Memória)**

- Lista de Blocos de Memória Livre




Como pode-se ver acima, o bloco desalocado torna a memória contígüa novamente, assim deve-se ter um único bloco de memória livre contígüa.

- Lista de Blocos Memória Alocada = NULL

**Observações importantes:**
O programa deve estar bem documentado e implementado na **linguagem Java** e pode ser feito em grupo de até **2 alunos**, não esqueçam de colocar o **nome dos integrantes** do grupo no programa fonte. A entrega do trabalho deve ser feita pelo *Blackboard* (**não serão aceitos trabalhos entregues via e-mail**) e será avaliado de acordo com os seguintes critérios:

- Utilização dos conhecimentos de orientação a objetos vistos na disciplina de Projeto Integrador III.
- O quão fiel é o programa quanto à descrição do enunciado;
- Funcionamento do programa, Indentação, comentários e legibilidade do código e clareza na nomenclatura de variáveis;

Como este trabalho pode ser feito em **grupo**, evidentemente você pode "*discutir*"' o problema dado com outros **grupos**, inclusive as "*dicas*"' para chegar às soluções, mas você deve ser responsável pela solução final e pelo desenvolvimento do seu programa. Ou seja, qualquer tentativa de fraude será punida com a **nota zero**.