

# Relatório — Implementação de Gerenciador de Heap em Assembly

Beatriz Camargo e Giuliano Tavares

## 1. Introdução

O projeto teve como objetivo implementar um gerenciador de heap simples em linguagem Assembly AMD64 (NASM), capaz de realizar operações básicas de alocação e liberação de memória dinâmica. A implementação reproduz, em nível baixo, o funcionamento de um alocador de memória como *malloc* e *free*, manipulando diretamente o *heap pointer* do processo por meio da chamada de sistema *brk*.

## 2. Estratégia de Implementação

A estrutura de cada bloco de memória contém:

- Um byte de controle indicando se o bloco está livre (0) ou ocupado (1);
- Um campo de 8 bytes com o tamanho do bloco (tanto no cabeçalho quanto no rodapé).

[uso:1B][tamanho:8B][dados:N bytes][tamanho:8B]

- Função *memory\_alloc*:

Implementa a estratégia **worst-fit**, que seleciona o maior bloco livre disponível para atender à requisição. O algoritmo percorre a heap, localiza o pior encaixe, e:

- Se não houver bloco livre adequado, expande a heap com *brk* e cria um novo bloco.
- Se houver espaço excedente ( $\geq 18$  bytes), realiza o **split**, criando um novo bloco livre após o alocado.

- Função *memory\_free*:

Ao liberar um bloco, a função marca-o como livre e realiza **fusão (merge)** com blocos adjacentes também livres, tanto **atrás** quanto **à frente**, evitando fragmentação. Caso o bloco liberado esteja no final da heap, é feita a **redução da heap** via *brk*, retornando memória ao sistema. Também há tratamento de **erro** para ponteiros nulos e **duplo free**, retornando **1** nesses casos.

## 3. Testes e Resultados

Os testes foram realizados com um programa em C que executa uma sequência de alocações e liberações para verificar todos os casos: alocação inicial e subsequente; alocação com split e sem split; fusão à frente, atrás e em ambos os lados; redução do topo da heap (shrink); tentativas de *double free* e ponteiro nulo.

Todos os testes foram executados com sucesso, validando o comportamento esperado do gerenciador.