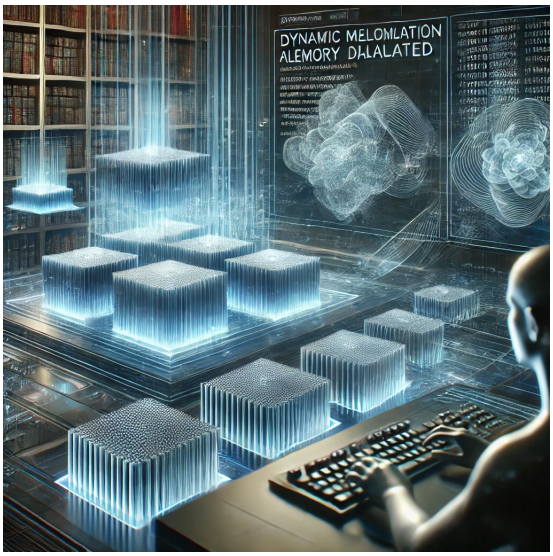
Universidade de Brasília

FCTE

Exercício Prático: Alocação Dinâmica de Memória
Disciplina: FGA0147 - Estruturas de Dados e Algoritmos
Prof. Dr. Nilton Correia da Silva
Faculdade de Ciências e Tecnologias em Engenharias - FCTE
Universidade de Brasília - UnB

# Aplicação de Alocação Dinâmica para gerenciar temperaturas de cidades.




Gama, Distrito Federal

# Sumário

**1 Introdução** **3**

**2 Tema** **4**

**3 Objetivo** **4**

**4 Condições de Contorno da Solução** **4**

**Orientações**

Exercício Prático da disciplina de Estruturas de Dados e Algoritmos. Este exercício foi elaborado considerando que sua resolução deve ser realizada de forma individual.

A solução deverá ser em linguagem C ou C++. No caso do acadêmico adotar C++, poderá se valer de uma solução orientada a objetos, contudo sua solução não deve usar classes e contêineres já prontos da linguagem C++ - por exemplo, algoritmos de ordenação, busca, objetos vector, list, matrix, string, etc.

# 1 Introdução

A alocação dinâmica de memória é uma técnica essencial na programação, especialmente útil quando lidamos com quantidades de dados que não são conhecidas antecipadamente. Em muitos problemas práticos, desde a análise de grandes conjuntos de dados até o desenvolvimento de aplicativos que se adaptam a diferentes condições de uso, o tamanho dos dados pode variar significativamente durante a execução do programa. A alocação dinâmica permite que os programas ajustem a quantidade de memória utilizada em tempo real, otimizando o uso de recursos e evitando o desperdício ou a insuficiência de memória, que poderiam levar a falhas ou desempenho degradado.

No contexto da análise de dados, por exemplo, onde o volume de informações pode crescer exponencialmente, a alocação dinâmica é vital. Ela permite que estruturas de dados como vetores e matrizes cresçam conforme necessário para acomodar novos dados. Isso é particularmente importante em áreas como aprendizado de máquina e mineração de dados, onde a capacidade de processar grandes datasets de maneira eficiente pode diretamente impactar a qualidade dos insights gerados e a velocidade das decisões baseadas em dados. Sem alocação dinâmica, os programas precisariam reservar uma grande quantidade de memória desde o início para garantir que haveria espaço suficiente, o que poderia levar a uma utilização ineficiente dos recursos do sistema.

Além disso, a alocação dinâmica promove melhores práticas de design de software, facilitando a criação de programas modulares e reutilizáveis. Por exemplo, em sistemas embutidos ou aplicações críticas, onde os recursos são limitados, a alocação dinâmica permite o desenvolvimento de componentes que consomem memória apenas quando ativamente necessários. Isso não apenas economiza recursos, mas também contribui para sistemas mais rápidos e responsivos. A habilidade de alocar e desalocar memória dinamicamente também ajuda na manutenção e na escalabilidade de aplicativos, permitindo que eles se ajustem mais facilmente a mudanças nos requisitos ou nos ambientes operacionais sem necessidade de recompilação ou redistribuição.

Neste exercício teremos a oportunidade de exercitar alocação dinâmica de memória em linguagem C/C++ em um caso concreto.

## 2 Tema

Alocação dinâmica e estruturas heterogêneas (structs).

## 3 Objetivo

Suponha que você seja um cientista de dados trabalhando em uma empresa de análise climática. Você precisa analisar as temperaturas diárias de várias cidades ao longo de um período indefinido. As temperaturas são fornecidas uma por uma e não há um número fixo de dias para análise, o que torna necessário o uso de alocação dinâmica de memória para armazenar essas temperaturas.

## 4 Condições de Contorno da Solução

Sua solução deve atender às seguintes condições:

1. Crie um arquivo **tipos.h** com os tipos abaixo:

```c
typedef struct {
    char cidade[20];
    float *temperatura;
} TMedidas;

typedef struct {
    int qtde_dias; //Quantidade maxima de medidas
    int qtde_medidas; //Qtde de medidas registradas
    TMedidas *medidas;
} TTemperaturas;
```

2. O programa deve importar o arquivo **tipos.h** e ter um menu com as seguintes opções:

   (a) Reset: Esta opção deve: ler a quantidade de dias de registros de temperaturas *qtde_dias* ,alocar o vetor de medidas e colocar *qtde_medidas* igual a zero;

   (b) Inserir Medida: Esta opção deve: ler os dados de uma medida (nome da cidade e temperatura) e adicionar esta medida ao vetor de temperaturas;

   (c) Estatística: Esta opção deve: mostrar a temperatura média, a menor temperatura (o nome da cidade e temperatura) e a maior temperatura (o nome da cidade e temperatura);

   (d) Sair: Esta opção deve: Desalocar o vetor *medidas* e encerrar o programa.