COM111 — Alocação Dinâmica

Hokama

UNIFEI

13 de Setembro de 2021

Disclaimer

Esses slides foram adaptados dos slides elaborados pelo Prof. Eduardo C. Xavier para a disciplina MC102 da UNICAMP.

Erros Comuns ao Usar Alocação Dinâmica

- Você pode fazer ponteiros distintos apontarem para uma mesma região de memória.
  - Tome cuidado para não utilizar um ponteiro se a sua região de memória foi desalocada!

```
double *v1, *v2;

v1 = malloc(100 * sizeof(double));
v2 = v1;
free(v1);

for(i=0; i<n; i++)
  v2[i] = i*i;
```

- O código acima está errado e pode causar erros durante a execução, já que **v2** está acessando posições de memória que foram liberadas!

Erros Comuns ao Usar Alocação Dinâmica

- O programa abaixo imprime resultados diferentes dependendo se comentamos ou não o comando **free(v1)**. Por que?

```
#include <stdio.h>
#include <stdlib.h>

int main(){
  double *v1, *v2, *v3;
  int i;

  v1 = malloc(100 * sizeof(double));
  v2 = v1;

  for(i=0; i<100; i++)
    v2[i] = i;
  free(v1); //Comente e nao comente este comando

  v3 = calloc(100,sizeof(double));
  for(i=0; i<100; i++)
    printf("%.2lf\n", v2[i]);
  free(v3);

}
```

Exercício

Qual o resultado da execução do programa abaixo? Ocorre algum erro?

```c
#include <stdio.h>
#include <stdlib.h>

int * misterio(int n){
  int i, *vet;

  vet = malloc(n*sizeof(int));
  vet[0] = 1;
  for(i=1; i<n; i++){
    vet[i] = i*vet[i-1];
  }
  return vet;
}

int main(){
  int i, n, *v;

  printf("Digite n:");
  scanf("%d", &n);

  v = misterio(n);

  for(i=0;i<n;i++)
    printf("%d\n", v[i]);

  free(v);
}
```


Exercício

- Faça um programa que leia a dimensão *n* de um vetor, em seguida aloca dinamicamente dois vetores do tipo *double* de dimensão *n*, faz a leitura de cada vetor e finalmente e imprime o resultado da soma dos dois vetores.

Exercício

- Faça uma função que recebe como parâmetro dois vetores de inteiros representando conjuntos de números inteiros, e devolve um outro vetor com o resultado da união dos dois conjuntos. O vetor resultante deve ser alocado dinamicamente. A função também deve retornar o tamanho do conjunto resultante.
- O protótipo da função é

```c
int * uniao(int v1[], int n1,
            int v2[], int n2, int *tamanho);
```

onde **n1** e **n2** indicam o número de elementos em **v1** e **v2** respectivamente.

Exemplo de Ponteiros e Alocação Dinâmica

Vamos criar uma aplicação que cria um vetor dinâmico com funções para implementar as seguintes operações:

- Inclusão de um elemento no final do vetor.
- Exclusão da primeira ocorrência de um elemento no vetor.
- Impressão do vetor.

Exemplo de Ponteiros e Alocação Dinâmica

- O tamanho do vetor deve se ajustar automaticamente: se elementos são inseridos devemos "aumentar"o tamanho do vetor para inclusão de novos elementos, e se elementos forem removidos devemos "diminuir"o tamanho vetor.
- Temos duas variáveis associadas ao vetor:
  - **size**: denota quantos elementos estão armazenados no vetor.
  - **maxSize**: denota o tamanho alocado do vetor.

Exemplo de Ponteiros e Alocação Dinâmica

Temos as seguintes regras para ajuste do tamanho alocado do vetor:

- O vetor deve ter tamanho alocado no mínimo igual a 4.
- Se o vetor ficar cheio, então devemos alocar um novo vetor com o dobro do tamanho atual.
- Se o número de elementos armazenados no vetor for menor do que 1/4 do tamanho alocado do vetor, então devemos alocar um novo vetor com metade do tamanho atual.

Exemplo de Ponteiros e Alocação Dinâmica

Implementaremos as seguintes funções:

- `int * initVet(int *size, int *maxSize);`

  Aloca um vetor inicial de tamanho 4, setando **size** com valor 0, **maxSize** com valor 4, e devolve o endereço do vetor alocado.

- `void printVet(int *v, int size, int maxSize);`

  Imprime o conteúdo e tamanhos associados ao vetor **v**.

- `int * addVet(int *v, int *size, int *maxSize, int e);`

  Adiciona o elemento **e** no final do vetor **v**. Caso não haja espaço, um novo vetor com o dobro do tamanho deve ser alocado. A função sempre retorna o endereço do vetor, sendo um novo alocado ou não. Além disso os valores de **size** e **maxSize** devem ser atualizados.

Exemplo de Ponteiros e Alocação Dinâmica

Implementaremos as seguintes funções:

- `int find(int *v, int size, int e);`

  Determina se o elemento **e** está presente ou não no vetor **v**. Caso esteja presente, retorna a posição da primeira ocorrência de **e** em **v**. Caso não esteja presente, retorna -1.

- `int * removeVet(int *v, int *size, int *maxSize, int e);`

  Remove a primeira ocorrência do elemento **e** do vetor **v** caso este esteja presente. O valor de **size** deve ser decrementado de 1. Caso o número de elementos armazenados seja menor do que $\frac{1}{4}$maxSize, então um novo vetor de tamanho $\frac{1}{2}$maxSize deve ser alocado no lugar de **v**. A função sempre retorna o endereço inicial do vetor alocado, sendo um novo vetor alocado ou não.