



Ordenação - Vetores

Função qsort

A função `qsort()` é uma função em C utilizada para ordenação de arrays.

O protótipo da função é

`void qsort(void* base, size_t nroEl, size_t tamanho, func_compara)`

`base` é um ponteiro para o primeiro elemento do array (ou seja, o array),

`nroEl` é o número de elementos do array,

`tamanho` é o tamanho em bytes de cada elemento do array e

`func_compara` é uma função utilizada para comparar dois elementos do array.

Função de comparação

A função **qsort()** pode ser utilizada para qualquer tipo de dado e por isso ela utiliza um **void*** como base. Dessa forma, é necessário elaborar uma função que explica como fazer a comparação entre dois elementos do array.

A função de comparação deve ter o protótipo genérico:

int <nome> (const void*, const void*)

Essa função deverá retornar:

- Valor **menor do que zero**: se o primeiro valor for menor do que o segundo.
- **Zero**: se os valores forem iguais.
- Valor **maior do que zero**: se o primeiro valor for maior do que o segundo.

Ordenação de vetor de inteiros

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

/* retorna negativo se a < b
   retorna zero se a == b
   retorna positivo se a > b */
int compara(const void* a, const void* b) {
    int* n1 = (int*) a;
    int* n2 = (int*) b;
    return *n1 - *n2;
}

int main() {
    int i;
    int v[] = {1, 3, 5, 7, 9, 2, 4, 6, 8};
    printf("Antes da ordenação: \n");
    for (i = 0; i < 9; i++) {
        printf("%d ", v[i]);
    }
    qsort(v, 9, sizeof(int), compara);
    printf("\nApos ordenação: \n");
    for (i = 0; i < 9; i++) {
        printf("%d ", v[i]);
    }
    printf("\n");
    return 0;
}
```

Obs: a função recebe ponteiros para 2 elementos do array de inteiros, logo os ponteiros genéricos (void*) devem ser convertidos para (int*)
Array de int => cast para int*

Função de comparação

```
/* retorna negativo se a > b
   retorna zero se a == b
   retorna positivo se a < b */
int comparaDecrescente(const void* a, const void* b) {
    return -(compara(a,b));
}
```

Função de comparação decrescente

Ordenação de vetor de *Strings*

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

```
int compara(const void* a, const void* b) {
    char** s1 = (char**) a;
    char** s2 = (char**) b;
    return strcmp(*s1, *s2);
}
```

Função de comparação

```
int main() {
    int i;
    char* v[] = {"ana", "carlos", "bob", "joão", "oscar", "maria"};
    printf("Antes da ordenação: \n");
    for (i = 0; i < 6; i++) {
        printf("%s ", v[i]);
    }
    qsort(v, 6, sizeof(char*), compara);
    printf("\nApos ordenação: \n");
    for (i = 0; i < 6; i++) {
        printf("%s ", v[i]);
    }
    printf("\n");
    return 0;
}
```

Obs: a função recebe ponteiros para 2 elementos do array de strings (char *), logo os ponteiros genéricos (void*) devem ser convertidos para (char**)

Array de char* => cast para char**

Lembrete:

int strcmp(const char *str1, const char *str2)

- retorna < 0 se str1 menor que str2.
- retorna > 0 se str1 maior que str2.
- retorna 0 se str1 e str2 são iguais.

Ordenação de vetor de *Structs*

```
int main (){
    int i;
    Produto produtos[] = {{1, "Coca", 3, 3.5}, {2, "Pepsi", 2, 3},
                           {3, "Guaraná", 4, 4.0}, {4, "Fanta", 1, 2.80}};


    printf("Antes da ordenação: \n");
    for (i = 0; i < 4; i++) {
        printf("Codigo: %d, ", produtos[i].codigo);
        printf("Nome: %s, ", produtos[i].nome);
        printf("Quantidade: %d, ", produtos[i].quantidade);
        printf("Preço: %.2f\n", produtos[i].preco);
    }

    qsort(produtos, 4, sizeof(Produto), compara);
    printf("\nApos ordenação: \n");
    for (i = 0; i < 4; i++) {
        printf("Codigo: %d, ", produtos[i].codigo);
        printf("Nome: %s, ", produtos[i].nome);
        printf("Quantidade: %d, ", produtos[i].quantidade);
        printf("Preço: %.2f\n", produtos[i].preco);
    }

    return 0;
}
```

Obs: a função recebe ponteiros para 2 elementos do array de Produtos, logo os ponteiros genéricos (void*) devem ser convertidos para (Produto*)

Array de Produto => cast para Produto*



```
typedef struct {
    int codigo;
    char nome[51];
    int quantidade;
    float preco;
} Produto;

int compara(const void* a, const void* b) {
    Produto* p1 = (Produto*) a;
    Produto* p2 = (Produto*) b;
    return strcmp(p1->nome, p2->nome);
}
```



Saiba mais

- <https://youtu.be/HtvfgqO0IM4>
- <https://www.galirows.com.br/meublog/programacao/utilizacao-funcao-qsort/>
- https://www.tutorialspoint.com/c_standard_library/c_function_qsort.htm