

PRÁTICA 6 – Trabalho 2 Exercício 6: Ordenação por BubbleSort

Nome: Davi Gabriel Domingues

Número USP: 15447497

O objetivo desta prática é o de implementar um programa de ordenação, em C++, o qual manipula registros de pessoas com os campos: nome (tipo string) e idade (tipo int). Para isso, foi-se requisitado o uso da ordenação por BubbleSort, além da requisição de colocar as informações em ordem alfabética, conforme os nomes e as idades devidamente ordenados, considerando que o BubbleSort é um método de ordenação estável.

Sendo assim, foi – se desenvolvido este código:

```
1  #include <iostream>
2  #include <string>
3
4  using namespace std;
5
6  struct Domicilio { //struct que armazena as informações pedidas no exercício.
7      int idade;
8      string nome;
9  };
10
11 void preenchimentoInformacoes(Domicilio *domicilio, int N){ //atribuição das informações de cada struct.
12     for (int i = 1; i <= N; i++){
13         cout<<"Idade do ocupante: ";
14         cin>>domicilio[i].idade;
15         cin.ignore();
16
17         while (domicilio[i].idade <= 0){ //verificação de dados
18             cout<<"Informe uma idade valida: ";
19             cin>>domicilio[i].idade;
20             cin.ignore();
21         }
22
23         cout<<"Nome do ocupante: ";
24         getline(cin, domicilio[i].nome);
25
26         while (domicilio[i].nome.empty()){ //verificação de dados.
```

```

27         cout<<"Informe um nome valido: ";
28         getline(cin, domicilio[i].nome);
29     }
30
31     cout<<"Dados do domicilio "<< i << " cadastrados com sucesso!"<<endl<<endl;
32 }
33
34
35 void ordenacaoBubbleSort(Domicilio *domicilio, int N){ //não está ordenando... estranho....
36     //o fato do método ser estável garante maior facilidade de ordenação para os nomes iguais, em caso de idades iguais
37     Domicilio x_auxiliar, y_auxiliar;
38
39     for (int i = 2; i <= N; i++){ //ordenação dos nomes
40         for (int j = N; j >= i; j--){
41             if (domicilio[j - 1].nome > domicilio[j].nome){
42                 y_auxiliar = domicilio[j - 1];
43                 domicilio[j - 1] = domicilio[j];
44                 domicilio[j] = y_auxiliar;
45             }
46         }
47     }
48
49     for (int i = 2; i <= N; i++){ /*ordenação das idades --> ordenação dos nomes, em caso de idades iguais, já é inclusa,
50     porque o método de ordenação é estável.*/
51         for (int j = N; j >= i; j--){
52             if (domicilio[j - 1].idade > domicilio[j].idade){
53                 x_auxiliar = domicilio[j - 1];
54                 domicilio[j - 1] = domicilio[j];
55                 domicilio[j] = x_auxiliar;
56             }
57         }
58     }
59
60     for (int i = 1; i <= N; i++) //impressão das informações.
61         cout<<"["<<domicilio[i].idade<<", "<<domicilio[i].nome<<"]"<<endl;
62 }
63
64 int main(){
65     int N;
66
67     cout<<"Informe o total de domicilios que deseja administrar: ";
68     cin>>N;
69     cout<<endl;
70
71     Domicilio *domicilio = new Domicilio[N + 1]; //vai de 0 a N, ou seja, tem N + 1 termos.
72
73     preenchimentoInformacoes(domicilio, N);
74     ordenacaoBubbleSort(domicilio, N);
75
76     delete[] domicilio;
77
78     return 0;
79 }
80

```

Código escrito:

#include <iostream>

#include <string>

```
using namespace std;
```

```
struct Domicilio { //struct que armazena as informações pedidas no  
exercício.
```

```
    int idade;
```

```
    string nome;
```

```
};
```

```
void preenchimentoInformacoes(Domicilio *domicilio, int N){  
//atribuição das informações de cada struct.
```

```
    for (int i = 1; i <= N; i++){
```

```
        cout<<"Idade do ocupante: ";
```

```
        cin>>domicilio[i].idade;
```

```
        cin.ignore();
```

```
        while (domicilio[i].idade <= 0){ //verificação de dados
```

```
            cout<<"Informe uma idade valida: ";
```

```
            cin>>domicilio[i].idade;
```

```
            cin.ignore();
```

```
        }
```

```
        cout<<"Nome do ocupante: ";
```

```
        getline(cin, domicilio[i].nome);
```

```
        while (domicilio[i].nome.empty()){ //verificação de dados.
```

```
            cout<<"Informe um nome valido: ";
```

```

        getline(cin, domicilio[i].nome);
    }

    cout<<"Dados do domicilio "<< i << " cadastrados com
    sucesso!"<<endl<<endl;
}
}

void ordenacaoBubbleSort(Domicilio *domicilio, int N){
//o fato do método ser estável garante maior facilidade de ordenação
para os nomes iguais, em caso de idades iguais

    Domicilio x_auxiliar, y_auxiliar;

    for (int i = 2; i <= N; i++){ //ordenação dos nomes
        for (int j = N; j >= i; j--){
            if (domicilio[j - 1].nome > domicilio[j].nome){
                y_auxiliar = domicilio[j - 1];
                domicilio[j - 1] = domicilio[j];
                domicilio[j] = y_auxiliar;
            }
        }
    }
}

    for (int i = 2; i <= N; i++){ /*ordenação das idades --> ordenação dos
    nomes, em caso de idades iguais, já é inclusa,
    porque o método de ordenação é estável.*/
        for (int j = N; j >= i; j--){

```

```

        if (domicilio[j - 1].idade > domicilio[j].idade){
            x_auxiliar = domicilio[j - 1];
            domicilio[j - 1] = domicilio[j];
            domicilio[j] = x_auxiliar;
        }
    }
}

```

```

for (int i = 1; i <= N; i++) //impressão das informações.
    cout<<"["<<domicilio[i].idade<<",
"<<domicilio[i].nome<<"]"<<endl;
}

```

```

int main(){
    int N;

```

```

    cout<<"Informe o total de domicilios que deseja administrar: ";
    cin>>N;
    cout<<endl;

```

```

    Domicilio *domicilio = new Domicilio[N + 1]; //vai de 0 a N, ou seja,
    tem N + 1 termos.

```

```

    preenchimentoInformacoes(domicilio, N);
    ordenacaoBubbleSort(domicilio, N);

```

```

    delete[] domicilio;

```

```
return 0;
```

```
}
```