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:

```
#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){ //não está ordenando... estranho.....
//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;
}
```

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;
}
```