

## **PRÁTICA 5 – Trabalho 2 Exercício 5: Busca binária e ordenação**

**Nome:** Davi Gabriel Domingues

**Número USP:** 15447497

O objetivo desta atividade é o de ajudar uma prefeitura de uma cidade hipotética a fazer uma pesquisa sobre os domicílios na área urbana. Em cada domicílio, são coletados dados sobre renda média mensal, número total de ocupantes, endereço e número de ocupantes em idade escolar.

Para tanto, foi – se solicitado um programa, em C++, que contivesse um menu que permitisse:

- (a) O número de domicílios que serão cadastrados;
- (b) A leitura dos dados do domicílio;
- (c) A ordenação dos dados pelo método de inserção direta, de acordo com a ordem alfabética devida dos endereços;
- (d) A listagem dos endereços no programa e a impressão das informações do respectivo domicílio, cujo endereço foi digitado pelo usuário. Para tal objetivo, foi – se requisitado a utilização da busca binária (usei a versão simplificada, não a rápida);

Dessa forma, o programa a seguir foi desenvolvido, a fim de ser tomado como resposta para a situação-problema proposta na Prática:

```

1  #include <iostream>
2  #include <limits>
3  #include <string> //só basta colocar nome1 < nome2, que a comparação alfabética é automática
4  #include <cmath>
5
6  using namespace std;
7
8  struct Domicilio{
9      float rendaMensal;
10     int totalOcupantes;
11     string endereco;
12     int ocupantesIdadeEscolar;
13 };
14
15 bool preencherInformacaoDomicilio(Domicilio& domicilio) {
16     cout << endl << "Renda mensal do domicílio: ";
17     cin >> domicilio.rendaMensal;
18     if (domicilio.rendaMensal <= 0) {
19         cout << "Dados inválidos! Inserção interrompida..." << endl;
20         cin.ignore(); // Ignora a linha
21         return false;
22     }
23     cin.ignore(numeric_limits<streamsize>::max(), '\n');
24
25     cout << "Total de ocupantes do domicílio: ";
26     cin >> domicilio.totalOcupantes;
27     if (domicilio.totalOcupantes <= 0) {
28         cout << "Dados inválidos! Inserção interrompida..." << endl;
29         cin.ignore(numeric_limits<streamsize>::max(), '\n');
30         return false;
31     }
32     cin.ignore(numeric_limits<streamsize>::max(), '\n');
33
34     cout << "Endereço do domicílio: ";
35     getline(cin, domicilio.endereco);
36     if (domicilio.endereco.empty()) {
37         cout << "Dados inválidos! Inserção interrompida..." << endl;
38         return false;
39     }
40
41     cout << "Ocupantes em idade escolar do domicílio: ";
42     cin >> domicilio.ocupantesIdadeEscolar;
43     if (domicilio.ocupantesIdadeEscolar < 0 || domicilio.ocupantesIdadeEscolar >= domicilio.totalOcupantes) {
44         cout << "Dados inválidos! Inserção interrompida..." << endl;
45         cin.ignore(numeric_limits<streamsize>::max(), '\n');
46         return false;
47     }
48     cin.ignore(numeric_limits<streamsize>::max(), '\n');
49
50     return true;
51 }
52
53 void ordenacaoInformacoesDomicilioDeclarados(Domicilio domicilio[], int tamanho) { //ordenação por inserção direta
54     for (int i = 2; i <= tamanho; i++){
55         Domicilio x = domicilio[i];
56         int j = i - 1;
57
58         for (int j = i - 1; j >= 0 && x.endereco < domicilio[j].endereco; j--)
59             domicilio[j + 1] = domicilio[j];
60
61         domicilio[j + 1] = x;
62     }
63 }
64
65 void encontrarDadosDomicilio(Domicilio domicilio[], int i, string endereco_desejado) { // busca binária, i = tamanho
66     int a = 1, b = i - 1, m;
67     bool encontrado = false;
68
69     while ((a <= b) && (!encontrado)){
70         m = floor((a + b)/2);
71         if (domicilio[m].endereco == endereco_desejado){
72             cout << "Endereço localizado, dados encontrados!" << endl;

```

```

73         cout<<"Renda mensal: "<<domicilio[m].rendaMensal<<endl;
74         cout<<"Total de ocupantes: "<<domicilio[m].totalOcupantes<<endl;
75         cout<<"Ocupantes em idade escolar: "<<domicilio[m].ocupantesIdadeEscolar<<endl;
76         encontrado = true;
77     }
78
79     else{
80         if (domicilio[m].endereço > endereço_desejado)
81             b = m - 1;
82         else
83             a = m + 1;
84     }
85 }
86
87 if (!encontrado) cout<<"Endereço não localizado, dados inexistentes!"<<endl;
88 }
89
90 int main(){
91     int opcao = 0, i = 0, total_domicilios;
92     string endereço_desejado;
93
94     cout<<"Informe o total de domicilios: ";
95     cin>>total_domicilios;
96     cout<<endl;
97
98     Domicilio* domicilio = new Domicilio[total_domicilios + 1]; //vetor que comporta o sentinela
99
100     do{
101         cout<<"Informe o que deseja fazer:"<<endl;
102         cout<<"(1) Digitar os dados de um domicilio"<<endl;
103         cout<<"(2) Buscar os dados de um domicilio, a partir de um endereço"<<endl;
104         cout<<"(3) Sair"<<endl;
105         cin>>opcao;
106
107         switch(opcao){
108             case 1:
109                 if (i < total_domicilios){
110                     if (preencherInformacaoDomicilio(domicilio[i])) i++;
111                 }
112
113                 else cout<<"Número máximo de domicilios alcançado, impossível adicionar informações!"<<endl;
114                 break;
115
116             case 2:
117                 cout<<"Informe o endereço desejado para saber os dados associados: ";
118                 getline(cin, endereço_desejado);
119
120                 ordenacaoInformacoesDomicilioDeclarados(domicilio, i);
121
122                 encontrarDadosDomicilio(domicilio, i, endereço_desejado);
123                 break;
124
125             case 3:
126                 cout<<endl<<"Saindo..."<<endl;
127                 break;
128
129             default:
130                 cout<<"Opção inválida, tente novamente..."<<endl;
131         }
132
133         cin.ignore(numeric_limits<streamsize>::max(), '\n'); // limpa o buffer antes de um novo loop
134     } while(opcao != 3);
135
136     delete[] domicilio;
137
138     return 0;
139 }

```

**Código utilizado:**

```
#include <iostream>

#include <limits>

#include <string> //só basta colocar nome1 < nome2, que a comparação alfabética é automática

#include <cmath>


using namespace std;


struct Domicilio{

    float rendaMensal;

    int totalOcupantes;

    string endereco;

    int ocupantesIdadeEscolar;

};


bool preencherInformacaoDomicilio(Domicilio& domicilio) {

    cout << endl << "Renda mensal do domicilio: ";

    cin >> domicilio.rendaMensal;

    if (domicilio.rendaMensal <= 0) {

        cout << "Dado invalido! Insercao interrompida..." << endl;

        cin.ignore(); // Ignora a linha

        return false;

    }

    cin.ignore(numeric_limits<streamsize>::max(), '\n');


    cout << "Total de ocupantes do domicilio: ";

    cin >> domicilio.totalOcupantes;

    if (domicilio.totalOcupantes <= 0) {

        cout << "Dado invalido! Insercao interrompida..." << endl;

        cin.ignore(numeric_limits<streamsize>::max(), '\n');
```

```

        return false;
    }

    cin.ignore(numeric_limits<streamsize>::max(), '\n');

    cout << "Endereco do domicilio: ";

    getline(cin, domicilio.endereco);

    if (domicilio.endereco.empty()) {

        cout << "Dado invalido! Insercao interrompida..." << endl;

        return false;

    }

    cout << "Ocupantes em idade escolar do domicilio: ";

    cin >> domicilio.ocupantesIdadeEscolar;

    if (domicilio.ocupantesIdadeEscolar < 0 || domicilio.ocupantesIdadeEscolar >=
domicilio.totalOcupantes) {

        cout << "Dado invalido! Insercao interrompida..." << endl;

        cin.ignore(numeric_limits<streamsize>::max(), '\n');

        return false;

    }

    cin.ignore(numeric_limits<streamsize>::max(), '\n');

    return true;
}

```

```

void ordenacaoInformacoesDomicilioDeclarados(Domicilio domicilio[], int tamanho){
//ordenação por inserção direta

    for (int i = 2; i <= tamanho; i++){

        Domicilio x = domicilio[i];

        domicilio[0] = x;

        int j = i;

        while (j >= 0 && x.endereco < domicilio[j - 1].endereco){

```

```

        domicilio[j] = domicilio[j - 1];

        j--;
    }
    domicilio[j] = x;
}
}

```

```

void encontrarDadosDomicilio(Domicilio domicilio[], int i, string endereco_desejado){ //
busca binaria, i = tamanho

```

```

    int a = 1, b = i - 1, m;

```

```

    bool encontrado = false;

```

```

    while ((a <= b) && (!encontrado)){

```

```

        m = floor((a + b)/2);

```

```

        if (domicilio[m].endereco == endereco_desejado){

```

```

            cout<<"Endereco localizado, dados encontrados!"<<endl;

```

```

            cout<<"Renda mensal: "<<domicilio[m].rendaMensal<<endl;

```

```

            cout<<"Total de ocupantes: "<<domicilio[m].totalOcupantes<<endl;

```

```

            cout<<"Ocupantes em idade escolar:

```

```

"<<domicilio[m].ocupantesIdadeEscolar<<endl;

```

```

            encontrado = true;

```

```

        }

```

```

    else{

```

```

        if (domicilio[m].endereco > endereco_desejado)

```

```

            b = m - 1;

```

```

        else

```

```

            a = m + 1;

```

```

        }

```

```

    }

```

```

    if (!encontrado) cout<<"Endereco nao localizado, dados inexistentes!"<<endl;

```

```
}
```

```
int main(){
```

```
    int opcao = 0, i = 0, total_domicilios;
```

```
    string endereco_desejado;
```

```
    cout<<"Informe o total de domicilios: ";
```

```
    cin>>total_domicilios;
```

```
    cout<<endl;
```

```
    Domicilio* domicilio = new Domicilio[total_domicilios + 1]; //vetor que comporta o  
    sentinela
```

```
    do{
```

```
        cout<<"Informe o que deseja fazer:"<<endl;
```

```
        cout<<"(1) Digitar os dados de um domicilio"<<endl;
```

```
        cout<<"(2) Buscar os dados de um domicilio, a partir de um endereco"<<endl;
```

```
        cout<<"(3) Sair"<<endl;
```

```
        cin>>opcao;
```

```
        switch(opcao){
```

```
            case 1:
```

```
                if (i < total_domicilios){
```

```
                    if (preencherInformacaoDomicilio(domicilio[i])) i++;
```

```
                }
```

```
            else cout<<"Numero maximo de domicilios alcancado, impossivel adicionar  
informacoes!"<<endl;
```

```
                break;
```

```
            case 2:
```

```
                cout<<"Informe o endereco desejado para saber os dados associados: ";
```

```
getline(cin, endereco_desejado);

ordenacaoInformacoesDomicilioDeclarados(domicilio, i);
encontrarDadosDomicilio(domicilio, i, endereco_desejado);
break;

case 3:
    cout<<endl<<"Saindo..."<<endl;
    break;

default:
    cout<<"Opcao invalida, tente novamente..."<<endl;
}

    cin.ignore(numeric_limits<streamsize>::max(), '\n'); // Limpa o buffer antes de um
novo loop
} while(opcao != 3);

delete[] domicilio;

return 0;
}
```