QUESTÃO 1-

```
#include <iostream>
#include <string>
#include <vector>

using namespace std;

struct Ramal {
    int numero;
    int categoria;
    string setor;
    int edificio;
    string sala;
    string responsavelRamal;
    string responsavelConta;
};

bool categoriaValida(int categoria) {
    return categoria >= 0 && categoria <= 6;
}

bool numeroRamalValido(int numero) {
    return numero >= 1000 && numero <= 2999;
}

Ramal cadastrarRamal() {
    Ramal r;

    do {
        cout << "Número do Ramal (1000-2999): ";
        cin >> r.numero;
    } while (!numeroRamalValido(r.numero));

    do {
        cout << "Categoria (0-6): ";
        cin >> r.categoria;
    } while (!categoriaValida(r.categoria));

    cout << "Setor: ";
    cin >> r.setor;

    do {
        cout << "Edifício de Localização (500 ou 505): ";
        cin >> r.edificio;
    } while (r.edificio != 500 && r.edificio != 505);

    cout << "Sala: ";
```

```
    cin >> r.sala;

    cout << "Responsável pelo Ramal: ";
    cin >> r.responsavelRamal;

    cout << "Responsável pelo Ateste da Conta Telefônica: ";
    cin >> r.responsavelConta;

    return r;
}

void imprimirRamal(const Ramal& r) {
    cout << "\n--- Detalhes do Ramal ---" << endl;
    cout << "Número do Ramal: " << r.numero << endl;
    cout << "Categoria: " << r.categoria << endl;
    cout << "Setor: " << r.setor << endl;
    cout << "Edifício de Localização: " << r.edificio << endl;
    cout << "Sala: " << r.sala << endl;
    cout << "Responsável pelo Ramal: " << r.responsavelRamal << endl;
    cout << "Responsável pelo Ateste da Conta Telefônica: " << r.responsavelConta << endl;
}

int main() {
    vector<Ramal> listaRamais;
    char continuar;

    do {
        Ramal novoRamal = cadastrarRamal();
        listaRamais.push_back(novoRamal);

        cout << "Deseja cadastrar outro ramal? (s/n): ";
        cin >> continuar;
    } while (continuar == 's' || continuar == 'S');


    for (const auto& ramal : listaRamais) {
        imprimirRamal(ramal);
    }

    return 0;
}
```

```
Número do Ramal (1000-2999): 2563
Categoria (0-6): 5
Setor: 1
Edifício de Localização (500 ou 505): 502
Edifício de Localização (500 ou 505): 500
Sala: 2
Responsável pelo Ramal: 6
Responsável pelo Ateste da Conta Telefônica: 4
Deseja cadastrar outro ramal? (s/n): n

--- Detalhes do Ramal ---
Número do Ramal: 2563
Categoria: 5
Setor: 1
Edifício de Localização: 500
Sala: 2
Responsável pelo Ramal: 6
Responsável pelo Ateste da Conta Telefônica: 4
```

QUESTÃO 2-

```
#include <iostream>
#include <string>
#include <vector>
#include <set>
#include <algorithm>

using namespace std;

struct Ramal {
    int numero;
    int categoria;
    string setor;
    int edificio;
    string sala;
    string responsavelRamal;
    string responsavelConta;
};

bool categoriaValida(int categoria) {
    return categoria >= 0 && categoria <= 6;
}

bool numeroRamalValido(int numero) {
    return numero >= 1000 && numero <= 2999;
}

Ramal cadastrarRamal() {
    Ramal r;
```

```
    do {
        cout << "Número do Ramal (1000-2999): ";
        cin >> r.numero;
    } while (!numeroRamalValido(r.numero));

    do {
        cout << "Categoria (0-6): ";
        cin >> r.categoria;
    } while (!categoriaValida(r.categoria));

    cout << "Setor: ";
    cin >> r.setor;

    do {
        cout << "Edifício de Localização (500 ou 505): ";
        cin >> r.edificio;
    } while (r.edificio != 500 && r.edificio != 505);

    cout << "Sala: ";
    cin >> r.sala;

    cout << "Responsável pelo Ramal: ";
    cin >> r.responsavelRamal;

    cout << "Responsável pelo Ateste da Conta Telefônica: ";
    cin >> r.responsavelConta;

    return r;
}

void imprimirRamal(const Ramal& r) {
    cout << "\n--- Detalhes do Ramal ---" << endl;
    cout << "Número do Ramal: " << r.numero << endl;
    cout << "Categoria: " << r.categoria << endl;
    cout << "Setor: " << r.setor << endl;
    cout << "Edifício de Localização: " << r.edificio << endl;
    cout << "Sala: " << r.sala << endl;
    cout << "Responsável pelo Ramal: " << r.responsavelRamal << endl;
    cout << "Responsável pelo Ateste da Conta Telefônica: " << r.responsavelConta << endl;
}

template <typename T>
void listarUnicos(const vector<Ramal>& listaRamais, T Ramal::*campo) {
    set<T> unicos;
    for (const auto& ramal : listaRamais) {
        unicos.insert(ramal.*campo);
    }
```

```
    for (const auto& item : unicos) {
        cout << item << endl;
    }
}

void pesquisarRamais(const vector<Ramal>& listaRamais) {
    int escolha;
    cout << "Métodos de Pesquisa:" << endl;
    cout << "1. Número do Ramal" << endl;
    cout << "2. Categoria" << endl;
    cout << "3. Setor" << endl;
    cout << "4. Edifício de Localização" << endl;
    cout << "5. Sala" << endl;
    cout << "6. Responsável pelo Ramal" << endl;
    cout << "7. Responsável pelo Ateste da Conta Telefônica" << endl;
    cout << "Escolha um método de pesquisa: ";
    cin >> escolha;

    switch (escolha) {
        case 1:
            listarUnicos(listaRamais, &Ramal::numero);
            break;
        case 2:
            listarUnicos(listaRamais, &Ramal::categoria);
            break;
        case 3:
            listarUnicos(listaRamais, &Ramal::setor);
            break;
        case 4:
            listarUnicos(listaRamais, &Ramal::edificio);
            break;
        case 5:
            listarUnicos(listaRamais, &Ramal::sala);
            break;
        case 6:
            listarUnicos(listaRamais, &Ramal::responsavelRamal);
            break;
        case 7:
            listarUnicos(listaRamais, &Ramal::responsavelConta);
            break;
        default:
            cout << "Método de pesquisa inválido!" << endl;
            return;
    }

    cout << "Digite a opção desejada para ver os detalhes: ";
    if (escolha == 1 || escolha == 2 || escolha == 4) {
        int opcao;
```

```
        cin >> opcao;
        for (const auto& ramal : listaRamais) {
            if ((escolha == 1 && ramal.numero == opcao) ||
                (escolha == 2 && ramal.categoria == opcao) ||
                (escolha == 4 && ramal.edificio == opcao)) {
                imprimirRamal(ramal);
            }
        }
    } else {
        string opcao;
        cin >> opcao;
        for (const auto& ramal : listaRamais) {
            if ((escolha == 3 && ramal.setor == opcao) ||
                (escolha == 5 && ramal.sala == opcao) ||
                (escolha == 6 && ramal.responsavelRamal == opcao) ||
                (escolha == 7 && ramal.responsavelConta == opcao)) {
                imprimirRamal(ramal);
            }
        }
    }
}

int main() {
    vector<Ramal> listaRamais;
    char continuar;

    do {
        Ramal novoRamal = cadastrarRamal();
        auto it = find_if(listaRamais.begin(), listaRamais.end(), [&](const Ramal& r) {
            return r.numero == novoRamal.numero;
        });

        if (it == listaRamais.end()) {
            listaRamais.push_back(novoRamal);
        } else {
            cout << "Ramal já cadastrado!" << endl;
        }

        cout << "Deseja cadastrar outro ramal? (s/n): ";
        cin >> continuar;
    } while (continuar == 's' || continuar == 'S');

    char opcaoPesquisa;
    cout << "Deseja realizar uma pesquisa? (s/n): ";
    cin >> opcaoPesquisa;
    if (opcaoPesquisa == 's' || opcaoPesquisa == 'S') {
        pesquisarRamais(listaRamais);
    }
```

```
    return 0;
}
```

```
Número do Ramal (1000-2999): 1000
Categoria (0-6): 1
Setor: 1
Edifício de Localização (500 ou 505): 500
Sala: 1
Responsável pelo Ramal: 1
Responsável pelo Ateste da Conta Telefônica: 1
Deseja cadastrar outro ramal? (s/n): n
Deseja realizar uma pesquisa? (s/n): s
Métodos de Pesquisa:
1. Número do Ramal
2. Categoria
3. Setor
4. Edifício de Localização
5. Sala
6. Responsável pelo Ramal
7. Responsável pelo Ateste da Conta Telefônica
Escolha um método de pesquisa: 1
1000
Digite a opção desejada para ver os detalhes: 1000

--- Detalhes do Ramal ---
Número do Ramal: 1000
Categoria: 1
Setor: 1
Edifício de Localização: 500
Sala: 1
Responsável pelo Ramal: 1
Responsável pelo Ateste da Conta Telefônica: 1
```