**SAURO 2000**

**Sistema Acadêmico Ultra Rápido e Operacional**

Versão 1.0

**10/06/2024**

# **Introdução**

### Programa sauro-alunos

O programa sauro-alunos é projetado para gerenciar os dados pessoais dos alunos. Ele lê informações de um arquivo chamado alunos.csv, que contém detalhes como nome, sobrenome, data de nascimento e número de matrícula de cada aluno. Este programa permite a visualização, inserção, modificação e exclusão de registros de alunos, facilitando o gerenciamento eficiente dos dados dos alunos.

Funcionalidades principais:

* Leitura de dados: Carrega os dados dos alunos a partir do arquivo CSV.
* Cadastro de novos alunos: Permite adicionar novos alunos ao sistema.
* Atualização de registros: Permite modificar os detalhes dos alunos existentes.
* Exclusão de alunos: Remove registros de alunos do sistema.

### Programa sauro-cursos

O programa sauro-cursos é responsável pelo gerenciamento dos cursos oferecidos. Ele lê um arquivo chamado cursos.csv, que lista todos os cursos disponíveis junto com seus respectivos códigos. Este programa permite a administração dos cursos, incluindo a adição de novos cursos, atualização de informações existentes e exclusão de cursos não mais oferecidos.

Funcionalidades principais:

* Leitura de dados: Carrega a lista de cursos a partir do arquivo CSV.
* Cadastro de novos cursos: Permite adicionar novos cursos ao sistema.
* Atualização de registros: Permite modificar os detalhes dos cursos existentes.
* Exclusão de cursos: Remove cursos do sistema.

### Programa sauro-matriculas

O programa sauro-matriculas gerencia as matrículas dos alunos nos diversos cursos oferecidos pela instituição. Ele lê e escreve em um arquivo chamado cursos-alunos.csv, que registra quais alunos estão matriculados em quais cursos. Este programa facilita o processo de inscrição dos alunos, permitindo a visualização dos cursos matriculados, bem como a adição ou remoção de matrículas.

Funcionalidades principais:

* Leitura de dados: Carrega os dados das matrículas a partir do arquivo CSV.
* Matrícula de alunos em cursos: Permite inscrever alunos em cursos específicos
* Visualização de matrículas: Exibe os cursos em que um aluno está matriculado.
* Remoção de matrículas: Permite cancelar a inscrição de um aluno em um curso.

# Documentação do Sistema **SAURO 2000**

# M**ódulo** Alunos

### Introdução

Este documento descreve como utilizar o menu principal do programa **SAURO 2000** Módulo Alunos. O programa permite cadastrar, listar, alterar matrícula, eliminar e buscar alunos por matrícula.

## Menu Principal

### 1. Cadastrar Aluno

### **Descrição**: Permite cadastrar um novo aluno.

### **Passos**:

### Escolha a opção 1 no menu.

### Digite o nome do aluno quando solicitado.

### Digite o sobrenome do aluno quando solicitado.

### Digite a data de nascimento do aluno no formato DD-MM-YYYY quando solicitado.

### Digite a matrícula do aluno quando solicitado.

### Uma mensagem de confirmação será exibida: "Aluno adicionado com sucesso!"

### 2. Exibir Lista de Alunos

### **Descrição**: Lista todos os alunos cadastrados.

### **Passos**:

1. Escolha a opção 2 no menu.
2. A lista de alunos será exibida. Caso não haja alunos cadastrados, será exibida a mensagem: "Sem registros de alunos cadastrados!"

### 3. Alterar Matrícula de Aluno

### **Descrição**: Permite alterar a matrícula de um aluno.

Passos:

1. Escolha a opção 3 no menu.
2. Digite a matrícula atual do aluno quando solicitado.
3. Digite a nova matrícula do aluno quando solicitado.
4. Uma mensagem de confirmação será exibida: "Matrícula alterada com sucesso!" ou "Aluno com matrícula [matrícula] não encontrado!" se a matrícula atual não for encontrada.

### 4. Eliminar Aluno

### **Descrição**: Permite eliminar um aluno do cadastro.

Passos:

1. Escolha a opção 4 no menu
2. Digite a matrícula do aluno a ser removido quando solicitado.
3. Uma mensagem de confirmação será exibida: "Aluno com matrícula [matrícula] removido com sucesso!" ou "Aluno com matrícula [matrícula] não encontrado!" se a matrícula não for encontrada.

### 5. Buscar Aluno por Matrícula

### **Descrição**: Permite buscar e exibir os dados de um aluno pela matrícula.

Passos:

1. Escolha a opção 5 no menu.
2. Digite a matrícula do aluno quando solicitado.
3. Os dados do aluno serão exibidos. Caso a matrícula não seja encontrada, será exibida a mensagem: "Aluno com matrícula XXX não encontrado!"

### 6. Sair

### **Descrição**: Sai do programa.

Passos:

1. Escolha a opção 6 no menu.
2. Uma mensagem de despedida será exibida: "Saindo do programa..."

### Estrutura de Dados

**Estrutura Aluno**

struct Aluno {

char nomeAluno[20];

char sobrenomeAluno[20];

char dataNascimentoAluno[11];

int matriculaAluno;

struct Aluno\* seguinte;

};

### Funções Implementadas

* createAluno: Cria um novo aluno.
* addAluno: Adiciona um aluno à lista e salva no arquivo CSV.
* saveAlunosToCSV: Salva a lista de alunos no arquivo CSV.
* printAlunos: Exibe a lista de alunos.
* printAlunoPorMatricula: Exibe os dados de um aluno pela matrícula.
* freeAlunos: Libera a memória alocada para a lista de alunos.
* cadastrarAluno: Função auxiliar para cadastrar um novo aluno.
* eliminarAluno: Elimina um aluno da lista e do arquivo CSV.
* alterarCodigoCurso: Altera a matrícula de um aluno.

# Documentação do Sistema **SAURO 2000**

# M**ódulo Curs**os

### Introdução

Este programa de cadastro de cursos permite ao usuário gerenciar um registro de dados de cursos, incluindo a adição, visualização, remoção, alteração e busca de cursos pelo código. A seguir, estão descritas as funcionalidades e instruções sobre como utilizar o menu do programa.

Essa documentação fornece uma visão geral de como utilizar cada funcionalidade do menu para gerenciar cursos. Certifique-se de seguir as instruções corretamente para realizar as operações desejadas.

#### Menu Principal

Ao iniciar o programa, o menu principal será exibido com as seguintes opções:

#### Opções do Menu

1. Cadastrar Curso

Descrição: Permite cadastrar um novo curso, solicitando o nome e o código do curso.

Passos:

* 1. Selecione a opção 1 no menu.
  2. Digite o nome do curso quando solicitado.
  3. Digite o código do curso quando solicitado.
  4. O curso será adicionado e salvo no arquivo cursos.csv.

1. Exibir Lista de Cursos

Descrição: Exibe a lista de todos os cursos cadastrados.

Passos:

* 1. Selecione a opção 2 no menu.
  2. A lista de cursos será carregada do arquivo cursos.csv e exibida na tela.

1. Eliminar Curso

Descrição: Remove um curso do cadastro, utilizando o código do curso.

Passos:

* 1. Selecione a opção 3 no menu.
  2. Digite o código do curso a ser removido quando solicitado.
  3. Se o curso for encontrado, ele será removido do cadastro e o arquivo cursos.csv será atualizado.

1. Alterar Código do Curso

Descrição: Altera o código de um curso existente.

Passos:

* 1. Selecione a opção 4 no menu.
  2. Digite o código atual do curso quando solicitado.
  3. Digite o novo código do curso quando solicitado.
  4. Se o curso for encontrado, o código será alterado e o arquivo cursos.csv será atualizado.

1. Buscar Curso por Código

Descrição: Busca e exibe as informações de um curso específico, utilizando o código do curso.

Passos:

* 1. Selecione a opção 5 no menu.
  2. Digite o código do curso a ser buscado quando solicitado.
  3. Se o curso for encontrado, suas informações serão exibidas.

1. Sair

Descrição: Encerra o programa.

Passos:

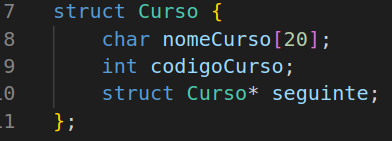
* 1. Selecione a opção 6 no menu.
  2. O programa será encerrado.

#### Exemplo de Utilização

1. Cadastrar um Curso
   1. Usuário escolhe a opção 1 no menu.
   2. Digita "Matemática" como nome do curso.
   3. Digita "1" como código do curso.
   4. Mensagem: "Curso adicionado com sucesso!"
2. Exibir Lista de Cursos
3. Usuário escolhe a opção 2 no menu.
4. Lista de cursos é exibida.
5. Eliminar um Curso
   1. Usuário escolhe a opção 3 no menu.
   2. Digita "1" como código do curso.
   3. Mensagem: "Curso com código 1 removido com sucesso!"
6. Alterar Código de um Curso
   1. Usuário escolhe a opção 4 no menu.
   2. Digita "1" como código atual do curso.
   3. Digita "2" como novo código do curso.
   4. Mensagem: "Código do curso alterado com sucesso!"
7. Buscar Curso por Código
   1. Usuário escolhe a opção 5 no menu.
   2. Digita "2" como código do curso.
   3. Informações do curso são exibidas.
8. Sair do Programa
   1. Usuário escolhe a opção 6 no menu.
   2. Mensagem: "Saindo do programa…"

### Lista Simplesmente Encadeada

1. Estrutura do Nó: Cada nó da lista é representado pela estrutura Curso.



Cada Curso possui:

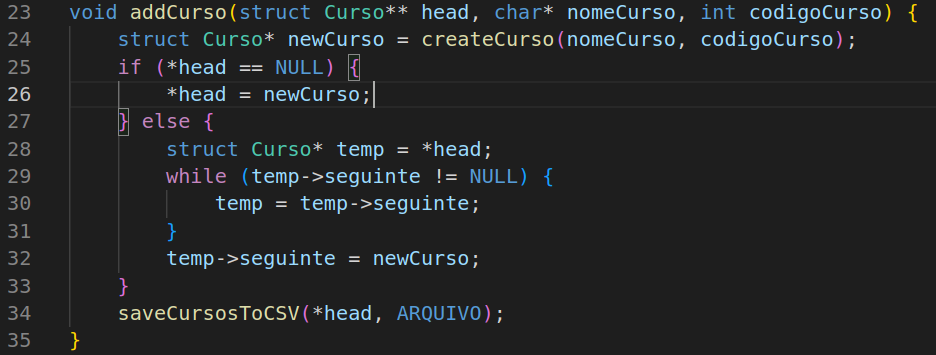
* nomeCurso: Uma string que armazena o nome do curso.
* codigoCurso: Um inteiro que armazena o código do curso.
* seguinte: Um ponteiro para o próximo nó (curso) na lista.

1. Head da Lista: A variável cursoHead na função main() é usada para manter a referência ao primeiro nó da lista.

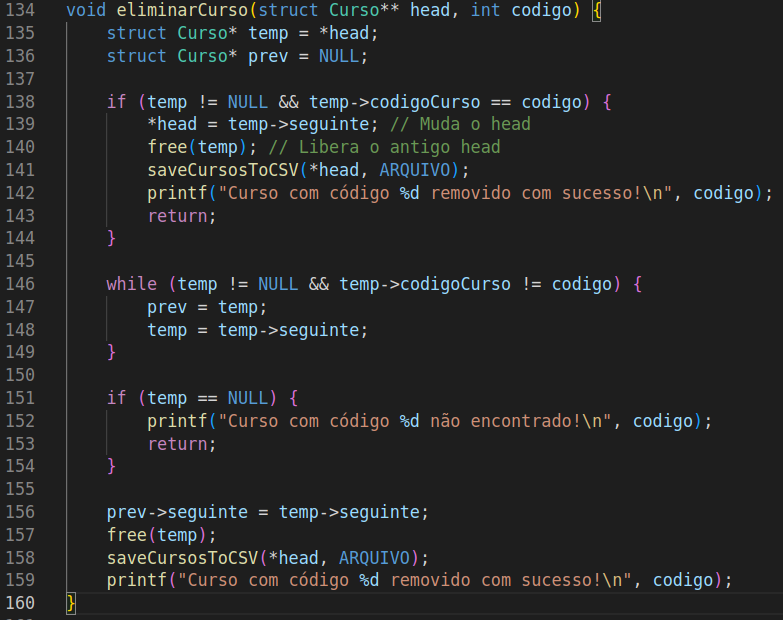


### Operações na lista simplesmente encadeada

1. Adicionar um Curso: A função addCurso insere um novo curso no final da lista.

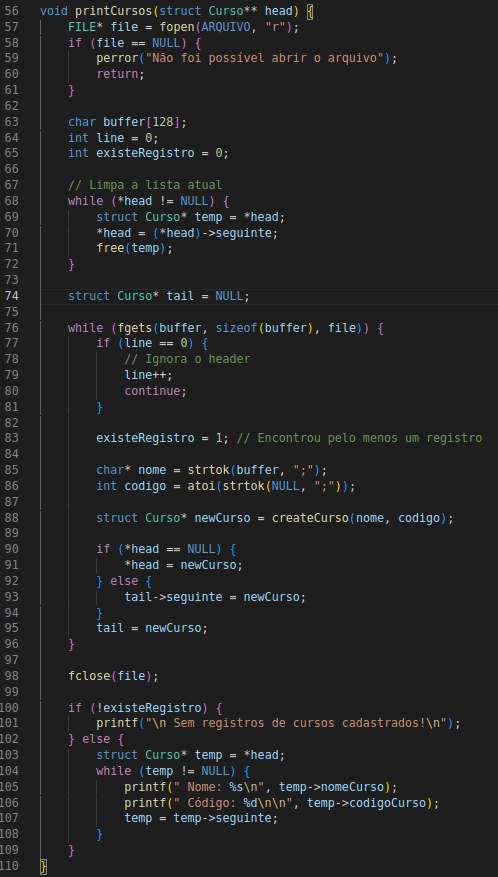
A função cria um novo nó (newCurso) e o adiciona no final da lista. Se a lista estiver vazia (\*head == NULL), o novo curso se torna o head da lista. Caso contrário, a função percorre a lista até encontrar o último nó e adiciona o novo curso após ele.

1. Remover um Curso: A função eliminarCurso remove um curso da lista com base no código do curso.

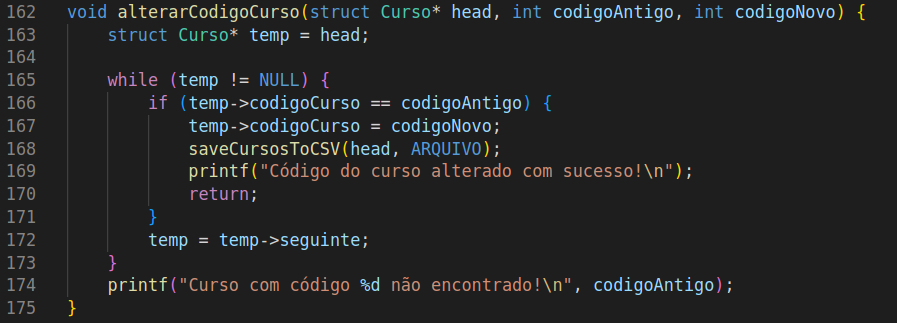


A função procura o nó com o código especificado. Se o nó estiver no head da lista, o head é ajustado para apontar para o próximo nó. Caso contrário, a função encontra o nó e ajusta o ponteiro do nó anterior (prev) para apontar para o próximo nó do nó removido.

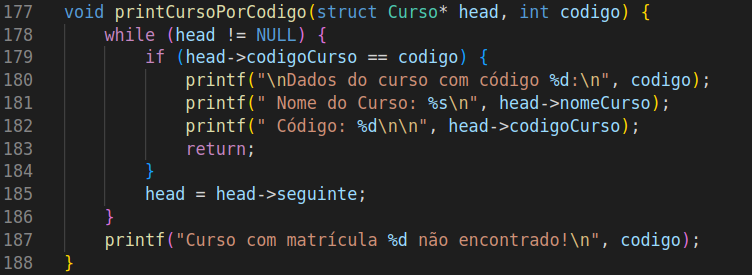
1. Imprimir Cursos: A função printCursos carrega os cursos do arquivo e imprime a lista. Lê os cursos do arquivo, limpa a lista atual e preenche a lista com os dados do arquivo, imprimindo cada curso ao final.



1. Alterar Código de um Curso: A função alterarCodigoCurso altera o código de um curso existente.

A função percorre a lista procurando o curso com o código especificado (codigoAntigo). Se encontrado, o código é alterado para codigoNovo e a lista é salva no arquivo.

1. Buscar Curso por Código: A função printCursoPorCodigo imprime os dados de um curso com base no código.

A função percorre a lista e imprime os dados do curso que corresponde ao código fornecido.

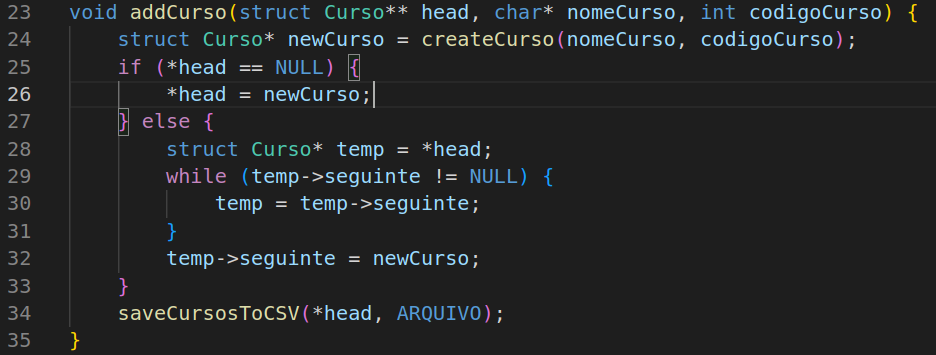
### Conclusão

No contexto do programa, a lista simplesmente encadeada é usada para armazenar e manipular os cursos. As operações de inserção, remoção, alteração e busca são realizadas através do gerenciamento dos ponteiros que conectam os nós da lista. Cada função é projetada para realizar uma operação específica na lista, garantindo a integridade e a persistência dos dados.

### Inserção de Nós em Listas Simplesmente Encadeadas

A inserção de nós em listas simplesmente encadeadas é uma operação fundamental que permite adicionar novos elementos à lista. No contexto do programa, a inserção de um novo curso é realizada através da função addCurso. A seguir, em detalhes como essa função funciona e como a inserção é realizada.

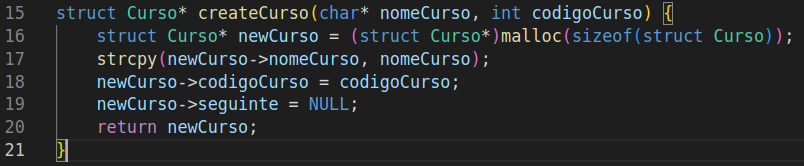
### Estrutura da Função addCurso



### Passos de inserção de um novo nó

1. Criação do novo nó:

A função createCurso é chamada para criar um novo nó (curso) com o nome e código fornecidos. Esta função aloca memória para o novo nó e inicializa seus campos.

 Aqui, createCurso aloca memória usando malloc e define os valores de nomeCurso, codigoCurso e o ponteiro seguinte (que inicialmente é NULL).

1. Verificação da Cabeça da Lista:

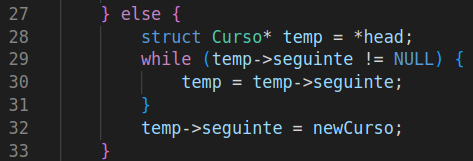
A função verifica se o head da lista (\*head) é NULL, o que significa que a lista está vazia. Se a lista estiver vazia, o novo nó (newCurso) se torna o head da lista.



1. Percorrendo a Lista para Encontrar o Fim:

Se a lista não estiver vazia, a função percorre a lista até encontrar o último nó. Isso é feito usando um ponteiro temporário (temp) que começa no head da lista e avança para o próximo nó até que o ponteiro seguinte de um nó seja NULL.

Neste trecho, temp percorre a lista. Quando temp->seguinte é NULL, temp aponta para o último nó da lista. Então, o ponteiro seguinte deste nó é ajustado para apontar para o novo nó (newCurso), efetivamente adicionando o novo nó ao final da lista.



# Documentação do Sistema **SAURO 2000**

# M**ódulo Cadastro de Matrículas**

### Descrição das Funções e Estruturas

1. Aluno
   * nome: Nome do aluno.
   * sobrenome: Sobrenome do aluno.
   * dataNascimento: Data de nascimento do aluno.
   * matricula: Matrícula do aluno.
2. Curso
   * nomeCurso: Nome do curso.
   * codigoCurso: Código do curso.
3. MatriculaCurso
   * matricula: Matrícula do aluno.
   * nomeCurso: Nome do curso.
   * codigoCurso: Código do curso.
4. cadastrarAlunoNoCurso
   * Função para cadastrar um aluno em um curso e salvar no arquivo.
5. bubbleSort
   * Função para ordenar um array de alunos por matrícula usando Bubble Sort.
6. listarMatriculas
   * Função para listar todas as matrículas disponíveis.
7. listarCursos
   * Função para listar todos os cursos disponíveis.
8. compararMatriculas
   * Função de comparação para ordenar matrículas usando qsort.
9. pesquisarCursosPorMatricula
   * Função para pesquisar e exibir os cursos onde um aluno está matriculado usando pesquisa binária.

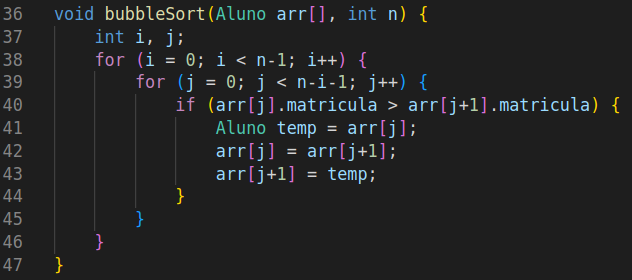
### Descrição do Programa Principal

O programa principal realiza as seguintes operações:

1. Abre os arquivos alunos.csv, cursos.csv, cursos-alunos.csv para leitura e cursos-alunos.csv para escrita.
2. Lê e armazena os dados dos arquivos CSV em arrays de estruturas (Aluno, Curso, MatriculaCurso).
3. Ordena os dados dos alunos por matrícula usando Bubble Sort.
4. Ordena os dados das matrículas e cursos usando qsort para permitir a pesquisa binária.
5. Oferece um menu interativo ao usuário para:
   * Cadastrar um aluno em um curso.
   * Pesquisar cursos de um aluno específico.
   * Sair do programa.
6. Escreve os dados de matrícula no arquivo cursos-alunos.csv.
7. Fecha todos os arquivos abertos e encerra o programa.

### Bubble Sort no contexto do programa

No contexto do programa, ele é utilizado para ordenar as matrículas dos alunos de maneira simples e direta.

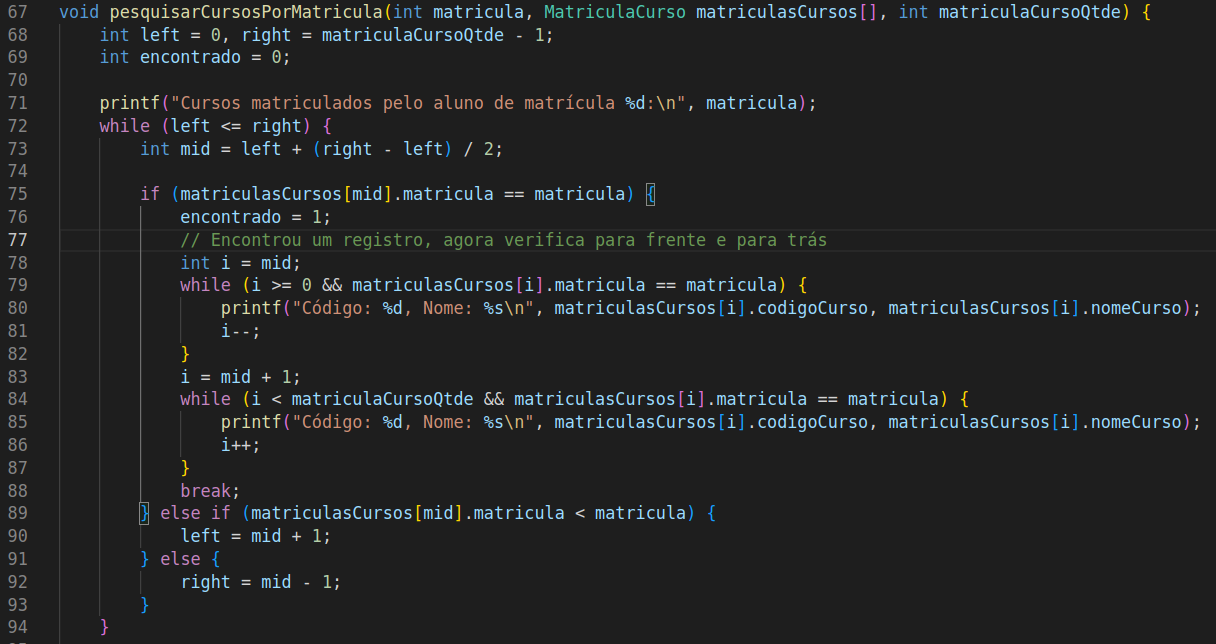


### Função bubbleSort

1. Inicialização dos laços:
   1. O laço externo (for (i = 0; i < n-1; i++)) percorre a lista inteira e diminui a faixa de comparação a cada iteração.
   2. O laço interno (for (j = 0; j < n-i-1; j++)) percorre a lista até o ponto já ordenado, comparando elementos adjacentes.
2. Comparação e troca:
3. if (arr[j].matricula > arr[j+1].matricula): Verifica se os elementos adjacentes estão na ordem correta.
4. Se não estiverem, troca os elementos usando uma variável temporária (temp).
5. Repetição:
   1. Repete o processo até que toda a lista esteja ordenada.

### Pesquisa bin**ária** no contexto do programa

A pesquisa binária é utilizada na função pesquisarCursosPorMatricula para buscar e exibir os cursos nos quais um aluno está matriculado, com base na sua matrícula.



### Função pesquisarCursosPorMatricula

1. Inicialização:
   1. left é inicializado para 0 (início do array).
   2. right é inicializado para matriculaCursoQtde - 1 (fim do array).
2. **Loop da Pesquisa Binária:**
   1. Enquanto left for menor ou igual a right, o algoritmo continua procurando:



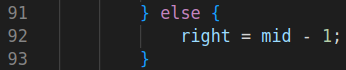
1. Cálculo do Meio:
   1. O índice do meio (mid) é calculado:



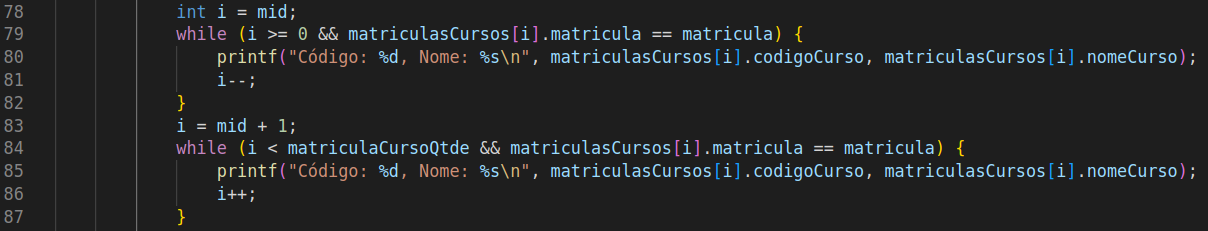
1. Comparação do Elemento do Meio:
   1. Se a matrícula no meio (matriculasCursos[mid].matricula) for igual à matrícula procurada:
   2. Se a matrícula no meio for menor que a procurada, descarta a metade esquerda:



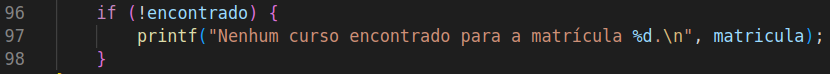
* 1. Se a matrícula no meio for maior que a procurada, descarta a metade direita:



1. Encontrar e Exibir Todos os Cursos:
   1. Se a matrícula for encontrada (encontrado = 1):
   2. Exibe os cursos associados à matrícula encontrada:



1. Resultado da Pesquisa:
   1. Se a matrícula não for encontrada após a pesquisa, exibe uma mensagem:



A pesquisa binária é eficiente para encontrar elementos em uma lista ordenada, reduzindo a complexidade de tempo de O(n) para O(log n). No contexto do programa, ela permite encontrar rapidamente todos os cursos associados a uma matrícula específica, verificando eficientemente a existência e listando os cursos correspondentes.