Visão Geral do Programa

O programa implementa um sistema de gerenciamento de alunos, permitindo incluir alunos, listar alunos e sair. Ele utiliza um vetor de structs para armazenar os dados dos alunos e organiza as funções principais de acordo com as ações do menu.

Detalhamento do Código

1. **Definições e Struct**

```
#define MAX_ALUNOS 50

typedef struct {
    char nome[50];
    double nota;
} Aluno;
```

- **MAX_ALUNOS**: Define a capacidade máxima do vetor de alunos (50).
- **Aluno**: Uma struct que contém o nome do aluno (até 50 caracteres) e a nota do aluno (tipo **double**).

2. **Função Principal main**

```
int main() {
    setlocale(LC_ALL, "portuguese");

    Aluno alunos[MAX_ALUNOS];
    int num_alunos = 0;
    int opcao;

    do {
        exibir_menu();
        scanf("%d", &opcao);
        getchar();  // Consumir o newline deixado pelo scanf

        switch (opcao) {
            case 1:
                incluir_aluno(alunos, &num_alunos);
```

```
            break;
        case 2:
            listar_alunos(alunos, num_alunos);
            break;
        case 3:
            printf("Saindo do programa...\n");
            break;
        default:
            printf("Opção inválida. Tente novamente.\n");
      }
    } while (opcao != 3);

    return 0;
}
```

- **Configuração da Localidade**: **setlocale(LC_ALL, "portuguese");** ajusta a localidade para exibir caracteres especiais corretamente.
- **Variáveis**: **alunos** (vetor de structs **Aluno**), **num_alunos** (contador de alunos) e **opcao** (seleção do menu).
- **Loop do Menu**: Um **do-while** loop exibe o menu, lê a opção e executa a ação correspondente até que o usuário escolha sair (opção 3).

3. **Função exibir_menu**

```
void exibir_menu() {
    printf("\nMenu:\n");
    printf("1. Incluir aluno\n");
    printf("2. Listar alunos\n");
    printf("3. Sair\n");
    printf("Escolha uma opção: ");
}
```

- Exibe o menu com as opções disponíveis para o usuário.

4. **Função incluir_aluno**

```
void incluir_aluno(Aluno alunos[], int *num_alunos) {
    if (*num_alunos >= MAX_ALUNOS) {
        printf("Número máximo de alunos atingido.\n");
        return;
    }

    printf("Digite o nome do aluno: ");
    fgets(alunos[*num_alunos].nome, sizeof(alunos[*num_alunos].nome), stdin);
    alunos[*num_alunos].nome[strcspn(alunos[*num_alunos].nome, "\n")] = 0; // Remove newline

    printf("Digite a nota do aluno: ");
    scanf("%lf", &alunos[*num_alunos].nota);
    getchar();  // Consumir o newline deixado pelo scanf

    (*num_alunos)++;
}
```

- **Checagem de Limite**: Verifica se o número máximo de alunos foi atingido.
- **Entrada do Nome**: Lê o nome do aluno usando **fgets** e remove o newline final.
- **Entrada da Nota**: Lê a nota do aluno usando **scanf**.
- **Incremento do Contador**: Incrementa o contador de alunos.

5. **Função listar_alunos**

```
void listar_alunos(Aluno alunos[], int num_alunos) {
    if (num_alunos == 0) {
        printf("Nenhum aluno cadastrado.\n");
        return;
    }

    ordenar_alunos_por_nome(alunos, num_alunos);

    printf("\nLista de Alunos:\n");
    for (int i = 0; i < num_alunos; i++) {
```

```
        printf("Nome: %s\n", alunos[i].nome);
        printf("Nota: %.2f\n", alunos[i].nota);
        if (alunos[i].nota >= 6.0) {
            printf("Situação: Aprovado\n");
        } else {
            printf("Situação: Reprovado\n");
        }
        printf("----------------------------------------\n");
    }
}
```

- **Verificação de Alunos**: Verifica se há alunos cadastrados.
- **Ordenação**: Chama a função de ordenação para ordenar os alunos por nome.
- **Impressão dos Alunos**: Imprime os detalhes de cada aluno, incluindo a situação (aprovado ou reprovado) com base na nota.

6. **Função ordenar_alunos_por_nome**

```
void ordenar_alunos_por_nome(Aluno alunos[], int num_alunos) {
    int troca;
    Aluno temp;

    for (int i = 0; i < num_alunos - 1; i++) {
        troca = 0;
        for (int j = 0; j < num_alunos - i - 1; j++) {
            if (strcmp(alunos[j].nome, alunos[j + 1].nome) > 0) {
                temp = alunos[j];
                alunos[j] = alunos[j + 1];
                alunos[j + 1] = temp;
                troca = 1;
            }
        }
        if (!troca) break;
    }
}
```

- **Bubble Sort**: Implementa o algoritmo Bubble Sort para ordenar os alunos pelo nome em ordem ascendente.
- **Troca**: Realiza a troca dos elementos se necessário.

Fluxo do Programa

1. **Inicialização**:
    - Configura a localidade e inicializa as variáveis.
2. **Menu e Seleção de Opção**:
    - Exibe o menu e aguarda a escolha do usuário.
    - Dependendo da escolha, chama as funções apropriadas:
        - **Incluir Aluno**: Adiciona um novo aluno ao vetor.
        - **Listar Alunos**: Ordena e exibe os alunos.
        - **Sair**: Termina o programa.
3. **Entrada e Processamento dos Dados**:
    - Lê os dados dos alunos e os armazena no vetor.
    - Ordena o vetor de alunos antes de exibir a lista.
4. **Exibição dos Dados**:
    - Mostra os alunos ordenados pelo nome e suas respectivas notas e situações.

Considerações Finais

O programa é modular, organizado em funções que realizam tarefas específicas, o que facilita a manutenção e a expansão futura. Ele utiliza as funcionalidades básicas de entrada e saída do C, bem como manipulação de strings e ordenação, de forma eficiente.