Explicação Detalhada:

```
#define MAX_NAME_LEN 50
#define FILENAME "usuarios.txt"
#define SHIFT 3
```

Se trata das definiçõe do codigo MAX_NAME_LEN 50, define o tamanho do nome de usuário

```
typedef struct {
int id;
char nome[MAX_NAME_LEN];
} Usuario;
```

Aqui, definimos a estrutura que contém:

- `id`: um número inteiro para identificar o usuário.
- `nome`: uma string para armazenar o nome do usuário, com limite de caracteres definido por `MAX_NAME_LEN`.

> Explicação detalhada.zip

```
void cifraDeCesar(char *texto) {
for (int i = 0; texto[i] != '\0'; i++) {
if ((texto[i] >= 'a' && texto[i] <= 'z') || (texto[i] >= 'A' && texto[i] <= 'Z')) {
texto[i] += SHIFT;
}
}
}
```

- Esta função aplica a cifra de César, que desloca cada letra do nome do usuário em `SHIFT` posições no alfabeto.
- Esse deslocamento é feito apenas para caracteres alfabéticos (`a-z` e `A-Z`).

```
void descifraDeCesar(char *texto) {
for (int i = 0; texto[i] != '\0'; i++) {
if ((texto[i] >= 'a' && texto[i] <= 'z') || (texto[i] >= 'A' && texto[i] <= 'Z')) {
texto[i] -= SHIFT;
}
}
}
```

Esta função faz o oposto de `cifraDeCesar`: ela "move" cada letra `SHIFT` posições para trás, restaurando o texto original.

```
void incluirUsuario() {
Usuario usuario;
FILE *file = fopen(FILENAME, "a+");
if (file == NULL) {
printf("Erro ao abrir o arquivo!\n");
return;
}
}
printf("Digite o ID do usuario: ");
scanf("%d", &
usuario.id);
printf("Digite o nome do usuario: ");
scanf("%49s", usuario.nome);
cifraDeCesar(usuario.nome);
fprintf(file, "%d %s\n",
usuario.id, usuario.nome);
fclose(file);
printf("Usuario incluído com sucesso!\\n");
}
```

Esta função permite adicionar um novo usuário ao sistema.

1. Abre o arquivo em modo de adição (`"a+"`).
2. Solicita ao usuário um `id` e um `nome`.
3. Aplica a cifra de César ao `nome` antes de salvar.
4. Escreve o `id` e o nome criptografado no arquivo.
5. Fecha o arquivo e exibe uma mensagem de confirmação.

```
void listarUsuarios() {
Usuario usuario;
FILE *file = fopen(FILENAME, "r");
if (file == NULL) {
printf("Nenhum usuario encontrado.\n");
return;
}
printf("Lista de usuarios:\n");
while (fscanf(file, "%d %s", &
usuario.id, usuario.nome) != EOF) {
descifraDeCesar(usuario.nome);
printf("ID: %d, Nome: %s\n",
usuario.id, usuario.nome);
}
fclose(file);
}
```

Esta função lista todos os usuários armazenados no arquivo.

1. Abre o arquivo em modo de leitura (`"r"`).
2. Lê cada `id` e `nome` criptografado do arquivo.
3. Descriptografa o `nome` e exibe o `id` e o `nome` do usuário.
4. Fecha o arquivo.

```
void removerDuplicados() {
Usuario usuarios[100];
```

```
int totalUsuarios = 0;
Usuario usuarioAtual;
FILE *file = fopen(FILENAME, "r");
FILE *tempFile = fopen("temp.txt", "w");

if (file == NULL || tempFile == NULL) {
    printf("Erro ao abrir o arquivo!\\n");
    if (file != NULL) fclose(file);
    if (tempFile != NULL) fclose(tempFile);
    return;
}

while (fscanf(file, "%d %s", &usuarioAtual.id, usuarioAtual.nome) != EOF) {
descifraDeCesar(usuarioAtual.nome);

    int duplicado = 0;
    for (int i = 0; i < totalUsuarios; i++) {
        if (usuarios[i].id == usuarioAtual.id) {
            printf("ID duplicado encontrado e removido: ID %d\\n", usuarioAtual.id);
            duplicado = 1;
            break;
        }
    }

if (!duplicado) {
usuarios[totalUsuarios++] = usuarioAtual;
cifraDeCesar(usuarioAtual.nome);
fprintf(tempFile, "%d %s\n",
usuarioAtual.id, usuarioAtual.nome);
}
}

fclose(file);
fclose(tempFile);

remove(FILENAME);
rename("temp.txt", FILENAME);


}
```

Esta função remove usuários com id duplicados.

1. Armazena cada usuário em um array (`usuarios`) e verifica duplicatas.
2. Se o `id` já existir, o usuário é ignorado; caso contrário, é gravado em um arquivo temporário (`temp.txt`).
3. Substitui o arquivo original (`usuarios.txt`) pelo arquivo temporário após a operação.

```
void alterarUsuario() {
Usuario usuario;
int id, found = 0;
FILE *file = fopen(FILENAME, "r");
FILE *tempFile = fopen("temp.txt", "w");
if (file == NULL || tempFile == NULL) {
    printf("Erro ao abrir o arquivo!\\n");
    if (file != NULL) fclose(file);
    if (tempFile != NULL) fclose(tempFile);
    return;
}

printf("Digite o ID do usuario que deseja alterar: ");
scanf("%d", &id);

while (fscanf(file, "%d %s", &usuario.id, usuario.nome) != EOF) {
    descifraDeCesar(usuario.nome);
    if (usuario.id == id) {
        found = 1;
        printf("Digite o novo nome do usuario: ");
        scanf("%49s", usuario.nome);
        cifraDeCesar(usuario.nome);
    } else {
        cifraDeCesar(usuario.nome);
    }
    fprintf(tempFile, "%d %s\\n", usuario.id, usuario.nome);
}

fclose(file);
fclose(tempFile);
```

```
remove(FILENAME);
rename("temp.txt", FILENAME);

if (found) {
    printf("Usuario alterado com sucesso!\\n");
} else {
    printf("Usuario nao encontrado.\\n");
}
}
```

Esta função permite alterar o nome de um usuário existente.

1. Solicita o `id` do usuário a ser alterado e abre o arquivo.
2. Procura o `id` e altera o `nome` se encontrado.
3. Salva as alterações em um arquivo temporário, renomeando-o depois.

`excluirUsuario`

Esta função remove um usuário do arquivo com base no `id` informado.

`main` e Menu

O `main` exibe o menu de opções e executa a função correspondente ao que o usuário escolhe.