Hashing Aberto ou Encadeado: Exercício

Agora, implemente a função de inserção de uma chave na tabela hashing aberta em questão.

```
void inserirHash(Tabela tabela, int n)
{
  Hash* novo;
  int pos = funcaoHashing(n);
  novo = (Hash *) malloc (sizeof (Hash));
  novo->chave = n;
  novo->prox = tabela[pos];
  tabela[pos] = novo;
}
```

Hashing Aberto ou Encadeado: Exercício

```
int funcaoHashing(int num)
{
   return num % numEntradas;
}
```

Hashing Aberto ou Encadeado: Exercício

Implemente uma função que especifique se uma determinada chave esta contida na tabela hashing aberta em questão.

```
Hash* localizarHash (Tabela tabela, int num)
{
  int pos = funcaoHashing (num);
  Hash* aux;
  if (tabela[pos] != NULL)
    if (tabela [pos] -> chave==num)
      return (tabela [pos]);
    else
    {
      aux=tabela [pos]->prox;
```

Hashing Aberto ou Encadeado: Exercício

```
        while (aux != NULL && aux ->chave != num)
           aux = aux->prox;
        return (aux);
      }
   else
      return NULL;
}
```

Hashing Aberto ou Encadeado: Exercício

Implemente a função de remoção de uma chave na tabela hashing aberta em questão.

```
void excluirHash (Tabela tabela, int num) {
  int pos = funcaoHashing (num);
  Hash* aux;
  if (tabela[pos] != NULL) {
    if (tabela [pos] -> chave==num) {
      aux = tabela [pos];
      tabela [pos] = tabela [pos] ->prox;
      free (aux);
    }
    else
    {
```

```
        aux=tabela [pos]->prox;
        Hash* ant=tabela [pos];
        while (aux != NULL && aux ->chave != num) {
           ant = aux;
           aux = aux->prox;
        }
        if (aux != NULL) {
           ant->prox = aux->prox;
           free (aux);
        }
        else
           printf("\nNumero nao encontrado");
     }
  }
  else
     printf("\nNumero nao encontrado"); }
```

Hashing Aberto ou Encadeado: Exercício

Para uma melhor fixação do tópico em estudo com base na definição do TAD abaixo implemente suas funções ainda não definidas (fazer em casa).

```
#define numEntradas 8
typedef struct _Hash
{
   int chave;
   struct _Hash *prox;
} Hash;
typedef Hash* TADTabelaHash[numEntradas];
void inicializarHash(TADTabelaHash);
int funcaoHashing(int);
void inserirHash(TADTabelaHash, int);
void mostrarHash(TADTabelaHash);
Hash* localizarHash (TADTabelaHash, int);
void excluirHash (TADTabelaHash, int);
void liberarMemoria(TADTabelaHash);
```