Aluno: Henrique Misael Machado **RA:** 99660

LISTA DE EXERCÍCIOS 14

Estrutura de Dados

Wesley Romão

Exercício 1:

Uma colisão em tabela hash é quando as chaves, de elementos distintos, resultam na mesma imagem da função hash. Quando ocorre, contorna-se o problema criando outra estrutura de dados em cada elemento do vetor de hash. Nesta escolha, há a opção de *hashing aberto* ou *hashing fechado*. A diferença entre eles é que, enquanto no *hashing fechado* utilizamos estruturas de vetores, estáticas, com limite de tamanho; no *hashing aberto* opta-se por estruturas de dados baseadas em ponteiros, ou seja, dinamicamente alocadas. Um exemplo de hash aberta é uma que possui um vetor, e cada elemento deste vetor é uma lista. Um exemplo de hash fechada seria uma matriz: um vetor onde cada elemento é outro vetor.

Exercício 2:

Utilizando a função hash f(k)=(2k+5)%11 em uma hash de tamanho 11, e inserindo os elementos 12, 44, 13, 88, 23, 94, 11, 39, 20, 16 e 5, obtemos a seguinte disposição:

Posições vazias da hash: 0, 2, 3, 8 e 10.
Posição 1: 20.
Posição 4: 16, 5.
Posição 5: 44, 88, 11.
Posição 6: 94, 39.
Posição 7: 12, 23.
Posição 9: 13.

Exercício 3:

Numa hash de tamanho 11, com função hash f(k)=k%11, temos:

a) ao inserir as chaves 4, 17, 13, 35, 25, 11, 2, 10, 32 a seguinte disposição das chaves:
Posições vazias da hash: 1, 5, 7, 8, 9
Posição 0: 11.
Posição 2: 13, 35, 2.
Posição 3: 25.
Posição 4: 4.
Posição 6: 17.
Posição 10: 10, 32.

b) ao remover as chaves 25 e 11:
Posições vazias da hash: 0, 1, 3, 5, 7, 8, 9
Posição 2: 13, 35, 2.
Posição 4: 4.
Posição 6: 17.
Posição 10: 10, 32.

c) ao inserir as chaves 40, 3.
Posições vazias da hash: 0, 1, 5, 7, 8, 9
Posição 2: 13, 35, 2.

Posição 3: 3.
Posição 4: 4.
Posição 6: 17.
Posição 7: 40.
Posição 10: 10, 32.

Exercício 4:

A função HASH para uma string, que multiplica o valor ASCII do caractere em cada posição por 128 elevado à sua posição (com a primeira posição valendo 0), é a seguinte:

```
int funcaoHash( char String[] )
{
   unsigned long hash, hashTemp;
   int tamanho, i;

   tamanho = strlen( String );

   for( hash = 0, i = 0; i < tamanho; i++ )
   {
      printf( "\n %d * 128^%d", String[i], tamanho-i-1 );
      hashTemp = String[ i ] * pow(128, tamanho-i-1);
      hash += hashTemp;
   }
   return hash;
}
```