2.3/hash.h

```c
#ifndef HASH_H
#define HASH_H

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "LSE.h"

typedef struct{
    Lista **tabela;
    int tam, qtd;
}Hash;


Hash* criaHash(int t){
    Hash* h;
    h = (Hash*) malloc (sizeof(Hash));
    if(h != NULL){
        h->tam = t; h->qtd = 0;
        h->tabela = (Lista**) malloc (t*sizeof(Lista*));
        if(h->tabela == NULL) return NULL;
        int i;
        for(i = 0; i<t; i++)
            h->tabela[i] = NULL;
    }
    return h;
}


void destroiHash(Hash *h){
    if(h != NULL){
        int i;
        for(i = 0; i<h->tam; i++)
            if(h->tabela[i] != NULL)
                destroiLista(h->tabela[i]);
        free(h->tabela);
        free(h);
    }
}

int chaveDivisao(int chave, int tam){
    return (chave & 0x7FFFFFFF) % tam;
}

int chaveMultiplicacao(int chave, int tam){
    float A = 0.6180339887; //constante: 0 < A < 1
    float val = chave * A;
    val = val - (int) val;
    return (int) (tam * val);
}

int chaveDobra(int chave, int tam){
    int pos, n_bits = 30;

    int p = 1;
    int r = p << n_bits;
    while((chave & r) != r){ n_bits--; r = p << n_bits; }
```

```
58
59      n_bits++;
60      pos = chave;
61      while(pos > tam){
62          int metade_bits = n_bits/2;
63          int parte1 = pos >> metade_bits;
64          parte1 = parte1 << metade_bits;
65          int parte2 = pos ^ parte1;
66          parte1 = pos >> metade_bits;
67          pos = parte1 ^ parte2;
68          n_bits = n_bits/2;
69      }
70      return pos;
71  }
72
73  int valorString(char *str){
74      int i, valor = 1;
75      int tam = strlen(str);
76      for(i=0; i<tam; i++)
77          valor = 31*valor + (i+1)*((int) str[i]);
78      return valor;
79  }
80
81  int insereHashLSE(Hash* h, int elem){
82      if(h == NULL) return 0;
83      int pos = chaveDivisao(elem, h->tam);
84      if(h->tabela[pos] == NULL)
85          h->tabela[pos] = criaLista();
86      insereIni(h->tabela[pos], elem);
87      h->qtd++;
88      return 1;
89  }
90
91  int buscaHashLSE(Hash* h, int elem, int *p){
92      if(h == NULL) return 0;
93      int pos = chaveDivisao(elem, h->tam);
94      if(h->tabela[pos] == NULL) return 0;
95      return listaBuscaElem(h->tabela[pos], elem, p);
96  }
97
98  void imprimeHash(Hash *h){
99      if(h == NULL) return;
100     int i;
101     for(i=0; i<h->tam; i++){
102         printf("%d: ", i);
103         if(h->tabela[i] == NULL) printf("NULL\n");
104         else imprimeLista(h->tabela[i]);
105     }
106 }
107
108 #endif
```