#ifndef HASHNODE\_H

#define HASHNODE\_H

#include<string>

using namespace std;

class hashNode

{

public:

hashNode();

hashNode(int key, string value);

virtual ~hashNode();

int getKey(){ return key;}

string getValue(){ return value; }

void setNext(hashNode \*val){ next = val;}

hashNode\* getNext(){ return next; }

protected:

private:

int key;

string value;

hashNode \*next;

};

#endif // HASHNODE\_H

#include "hashNode.h"

hashNode::hashNode()

{

//ctor

}

hashNode::hashNode(int key, string value)

{

this->key = key;

this->value = value;

this->next = nullptr;

}

hashNode::~hashNode()

{

//dtor

}

#ifndef HASHTABLE\_H

#define HASHTABLE\_H

#include "hashNode.h"

class hashTable

{

public:

int const TABLE\_SIZE = 3;

hashTable();

virtual ~hashTable();

int hashFunc(int key);

void insertNode(int key, string val);

void display();

protected:

private:

hashNode \*\*table; // dung de quan ly mang con tro hashNode

};

#endif // HASHTABLE\_H

#include "hashTable.h"

#include <iostream>

#include<iomanip>

using namespace std;

hashTable::hashTable()

{

//ctor

table = new hashNode \* [TABLE\_SIZE];

for(int i = 0; i < TABLE\_SIZE; i++)

{

table[i] = nullptr;

}

}

hashTable::~hashTable()

{

//dtor

}

int hashTable::hashFunc(int key)

{

return key % TABLE\_SIZE;

}

void hashTable::insertNode(int key, string val)

{

hashNode \*tmp = new hashNode(key, val);

int location = hashFunc(key);

// insert first linked list

tmp->setNext(table[location]);

table[location] = tmp;

}

void hashTable::display()

{

// Traverse the entire hash table

for (int i=0; i < TABLE\_SIZE; ++i) {

cout << " +--------+--------+" << endl;

cout << i << " |";

hashNode\* p = table[i];

if (p == NULL ) {

// NULL record, print empty

cout << " " << setw(6) << "" << " | " << setw(6) << "" << " |";

} else {

// Print the record from the table

cout << " " << setw(6) << left << p->getKey() << " | " << setw(6) << right << p->getValue() << " |";

// Traverse and print the chain

for (p = p->getNext(); p != nullptr ; p = p->getNext()) {

cout << " --> " << "[ " << p->getKey() << " | " << p->getValue() << " ]";

}

}

cout << endl;

}

cout << " +--------+--------+" << endl << endl;

}

#include <iostream>

#include"hashNode.h"

#include"hashTable.h"

using namespace std;

int main()

{

hashTable \*ht = new hashTable();

ht->insertNode(2, "le");

ht->insertNode(1, "ngu");

ht->insertNode(5, "tran");

ht->display();

return 0;

}