```cpp


#include "shphashtable.h"
#include <string>
#include <iostream>

using namespace std;

shphashtable::shphashtable(int numBuckets) {
    size = numBuckets;
    buckets = new Node*[size]; //array of pointers that will become a linked list or nullptr

    for(int i = 0; i<numBuckets; i++)
        buckets[i] = nullptr;
}

shphashtable::~shphashtable() {
    for(int i = 0; i < size; i++) {
        while(buckets[i] != nullptr) {
            Node* save = buckets[i];
            buckets[i] = buckets[i]->next;
            delete save;
        }
    }
    delete[] buckets;
}

bool shphashtable::contains(string str) const{
    return getPointer(str) != nullptr;
}

void shphashtable::put(string str, int score) {
    if(!contains(str)) {
        int index = getHashCode(str);
        Node* head = buckets[index];
        Node* temp = new Node;
        temp->value = WordEntry(str, score);
        temp->next = head;
        buckets[index] = temp;
    }
    else {
        Node* temp = getPointer(str);
        temp->value.addScore(score);
    }
}

void shphashtable::printStats() const {
    int numBuckets = size;
    int numWords = 0;
```

```
50      int numEmptyBuckets = 0;
51      int longestChain = 0;
52      int totalWordRating = 0;
53
54      for(int i = 0; i < size; i++) {
55          Node* temp = buckets[i];
56          int chainLen = 0;
57          while(temp != nullptr) {
58              chainLen++;
59              numWords++;
60              totalWordRating += temp->value.getAvgScore();
61              temp = temp->next;
62          }
63          if(chainLen > longestChain)
64              longestChain = chainLen;
65          if(buckets[i] == nullptr)
66              numEmptyBuckets++;
67      }
68
69      double avgChainLen = 1.0*numWords/(numBuckets-numEmptyBuckets);
70
71      cout << "STATS" << endl;
72      cout << "NUM BUCKETS: " << numBuckets << endl;
73      cout << "NUM WORDS: " << numWords << endl;
74      cout << "NUM EMPTY BUCKETS: " << numEmptyBuckets << endl;
75      cout << "LONGEST CHAIN: " << longestChain << endl;
76      cout << "AVG CHAIN LEN: " << avgChainLen << endl;
77      cout << "AVG WORD SCORE: " << 1.0*totalWordRating/numWords << endl;
78
79  }
80
81  double shphashtable::getAverage(string word) const{
82      Node* head = getPointer(word);
83      if(head == nullptr)
84          return 2.0;
85      return head->value.getAvgScore();
86
87  }
88
89  int shphashtable::getHashCode(string str) const {
90      hash<string> hashFunc;
91      return hashFunc(str)%size;
92  }
93
94  shphashtable::Node* shphashtable::getPointer(string str) const {
95      int index = getHashCode(str);
96      Node* head = buckets[index];
97      Node* temp = head;
98      while(temp != nullptr) {
99          if(temp->value.getWord() == str)
```

```
100             return temp;
101         temp = temp->next;
102     }
103     return nullptr;
104 }
105
```