

Sep 28, 09 8:19

ArrayListHash.java

Page 1/2

```

import java.util.*;

public class ArrayListHash implements IMapper {

    5     private static int SIZE = 1000000;

    private class Combo {
        int value;
        String key;
    10     public Combo(String key, int value){
            this.key = key;
            this.value = value;
        }
    }

    15     private ArrayList<ArrayList<Combo>> myTable;
    private int mySize;

    20     public ArrayListHash(){
        myTable = new ArrayList<ArrayList<Combo>>(SIZE);
        for(int k=0; k < SIZE; k++){
            myTable.add(null);
        }
    25     mySize = 0;

    private int getHash(String key){
        30     return Math.abs(key.hashCode()) % SIZE;
    }

    private Combo getCombo(String key){
        int bucketIndex = getHash(key);
        ArrayList<Combo> list = myTable.get(bucketIndex);
    35     if (list == null){
        return null;
    }
    else {
        for(Combo c : list){
            40     if (c.key.equals(key)){
                return c;
            }
        }
        return null;
    }
    45 }

    public boolean containsKey(String key){
    50     return getCombo(key) != null;
    }

    public int get(String key){
        return getCombo(key).value;
    }
    55

    public void put(String key, int value){
        Combo c = getCombo(key);
        if (c != null){
            c.value = value;
        }
    60     else {
        int bucketIndex = getHash(key);
        ArrayList<Combo> list = myTable.get(bucketIndex);
        if (list == null){
            list = new ArrayList<Combo>();
            65     myTable.set(bucketIndex, list);
        }
        list.add(new Combo(key,value));
        mySize++;
    70     }
    }

    public void printAll(){

```

Sep 28, 09 8:19

ArrayListHash.java

Page 2/2

```

    75     for(ArrayList<Combo> list : myTable){
        if (list == null) continue;
        for(Combo c : list){
            System.out.println(c.key+"\t"+c.value);
        }
    }
    80 }

    public int size(){
        return mySize;
    }
    85 }

```

Sep 28, 09 8:19

HashMain.java

Page 1/1

```

import java.io.*;
import java.util.Scanner;

import javax.swing.JFileChooser;
5
public class HashMain {

    private static JFileChooser ourChooser = new JFileChooser(System
10        .getProperties().getProperty("user.dir"));

    public static void storeAll(IMapper mapper, Scanner s) {

15        double start = System.currentTimeMillis();
        while (s.hasNext()){
            String str = s.next();
            if (mapper.containsKey(str)){
                mapper.put(str, mapper.get(str)+1);
20            }
            else {
                mapper.put(str,1);
            }
        }
25        double end = System.currentTimeMillis();
        double time = (end-start)/1000.0;
        System.out.printf("%s\t%.13f\t%d\n",
            mapper.getClass().getName(),time,mapper.size());
    }
30

    public static void main(String[] args) throws FileNotFoundException{
        int retval = ourChooser.showOpenDialog(null);

35        if (retval == JFileChooser.APPROVE_OPTION) {
            File f = ourChooser.getSelectedFile();
            Scanner s1 = new Scanner(f);
            Scanner s2 = new Scanner(f);
            Scanner s3 = new Scanner(f);
40            IMapper hash1 = new ArrayListHash();
            IMapper hash2 = new UtilMapHash();
            IMapper hash3 = new LinkHash();
            HashMain.storeAll(hash1,s1);
            HashMain.storeAll(hash2, s2);
            HashMain.storeAll(hash3, s3);
45        }
    }
}

```

Sep 28, 09 8:19

IMapper.java

Page 1/1

```

/**
 * Simple Map interface to experiment with
 * tradeoffs in low-level implementations
 * of Maps in Java as part of Compsci 100.
5  * @author ola
 *
 */

public interface IMapper {
10    public abstract boolean containsKey(String key);
    public abstract int get(String key);

15    public abstract void put(String key, int value);
    public abstract void printAll();
    public abstract int size();
20 }

```

Sep 28, 09 8:19

LinkHash.java

Page 1/2

```

public class LinkHash implements IMapper {
    private static int SIZE = 100000;
5
    private class Node{
        String key;
        int value;
        Node next;
10        public Node(String key, int value){
            this.key = key;
            this.value = value;
            this.next = null;
        }
15    }
    private Node[] myTable;
    private int mySize;

    public LinkHash(){
20        myTable = new Node[SIZE];
    }

    private int getHash(String key){
        return Math.abs(key.hashCode()) % SIZE;
25    }

    private Node getNode(String key){
        int bucketIndex = getHash(key);
        Node node = myTable[bucketIndex];
30        while (node != null){
            if (node.key.equals(key)) return node;
            node = node.next;
        }
        return null;
35    }
    public boolean containsKey(String key) {
        return getNode(key) != null;
    }

    public int get(String key) {
40        return getNode(key).value;
    }

    public void printAll() {
45        for(int k=0; k < myTable.length; k++){
            if (myTable[k] != null){
                Node list = myTable[k];
                while (list != null){
                    System.out.printf("%s\t%d\n", list.key, list.valu
50 e);
                    list = list.next;
                }
            }
        }
55    }

    public void put(String key, int value) {
        int bucketIndex = getHash(key);
        Node node = myTable[bucketIndex];
        while (node != null){
60            if (node.key.equals(key)) {
                node.value = value;
                return;
            }
            node = node.next;
65        }
        node = new Node(key, value);
        node.next = myTable[bucketIndex];
        myTable[bucketIndex] = node;
        mySize++;
70    }

    public int size() {

```

Sep 28, 09 8:19

LinkHash.java

Page 2/2

```

        return mySize;
    }
75 }

```

Sep 28, 09 8:19

UtilMapHash.java

Page 1/1

```
import java.util.HashMap;
import java.util.Map;

public class UtilMapHash implements IMapper{
5     private HashMap<String,Integer> myMap;

    public UtilMapHash(){
        myMap = new HashMap<String,Integer>();
10    }

    public boolean containsKey(String key) {
        return myMap.containsKey(key);
    }

15    public int get(String key) {
        return myMap.get(key);
    }

20    public void printAll() {
        for(Map.Entry<String,Integer> entry : myMap.entrySet()){
            System.out.printf("%s\t%d\n",entry.getKey(),entry.getValue());
        }
    }

25    public void put(String key, int value) {
        myMap.put(key,value);
    }

30    public int size(){
        return myMap.size();
    }
}
```