```
/**
 * Interface for reasoning about different methods for determining
 * number of different Strings in an array.
 *<P>
 * Intended as part of a CS2/Compsci 100 course.
 *<P>
 * @author Owen Astrachan
 * @version Created on Aug 25, 2004
 * @version Revised, Aug 29, 2006
 */

public interface IUniqueCounter {

        /**
         * Returns the number of different/unique elements of list.
         * @param list is collection of strings
         * @return number of different strings in list
         */
    public int uniqueCount(String[] list);
}
```



```
/**
 * Simple, efficient implementation of IUniqueCounter.
 * <P>
 * @author Owen Astrachan
 * @version January 3, 2006
 */
import java.util.*;

public class SetUniqueCounter implements IUniqueCounter {

    public int uniqueCount(String[] list) {
        TreeSet<String> set = new TreeSet<String>();
        set.addAll(Arrays.asList(list));
        return set.size();
    }

    @Override
    public String toString() {
        return "set";
    }
}
```

```
/**
 * This inefficient code has a bug, it is not correct.
 * <P>
 *
 * @author Owen Astrachan
 * @version January 3, 2006
 * @version Aug 29, 2006
 */
public class SlowUniqueCounter implements IUniqueCounter {

    public int uniqueCount(String[] list) {
        int count = 0;
        int diffSize = list.length;

        // Invariant: all words in range list[0..k] are unique
        // Elements in list[diffSize..) are duplicates of those in
        // list[0..diffSize)

        for (int k = 0; k < diffSize; k++) {
            String word = list[k];
            count++;
            for (int j = k + 1; j < diffSize; j++) {
                if (list[j].equals(word)) {
                    list[j] = list[diffSize - 1];
                    diffSize--;
                }
            }
        }
        return count;
    }

    @Override
    public String toString() {
        return "slow";
    }

}
```



```
/**
 * Reasonably efficient with small bug.
 * @author Owen Astrachan
 * @version Aug 29, 2006
 */

import java.util.Arrays;

public class SortingUniqueCounter implements IUniqueCounter {

    public int uniqueCount(String[] list) {
        Arrays.sort(list);
        String last = list[0];
        int count = 0;
        // Invariant: count is number of unique words in list[0..k)

        for (int k = 1; k < list.length; k++) {
            if (!list[k].equals(last)) {
                last = list[k];
                count++;
            }
        }
        return count;
    }

    @Override
    public String toString() {
        return "sort";
    }

}
```

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;
import java.util.ArrayList;

public class UniqueDemo {

    public static void main(String[] args) {
        try {
            Scanner s = new Scanner(new File("/Users/ola/kjv10.txt"));
            ArrayList<String> list = new ArrayList<String>();
            while (s.hasNext()) {
                list.add(s.next());
            }
            String[] arr = list.toArray(new String[0]);
            IUniqueCounter uc = new SetUniqueCounter();
            int count = uc.uniqueCount(arr);
            System.out.println("unique count: " + count);
        } catch (FileNotFoundException e) {
            System.out.println("file not found");
            e.printStackTrace();
        }
    }

}
```



```
/**
 * @author Owen Astrachan
 * @version August 24, 2004
 * @version Januaray 3, 2006
 *
 */

import java.io.File;
import java.io.FileNotFoundException;
import java.util.*;

import javax.swing.JFileChooser;

public class UniqueTester {
    private static JFileChooser ourChooser = new JFileChooser(System
            .getProperties().getProperty("user.dir"));

    public static void test(IUniqueCounter uc, String[] list) {

        double start = System.currentTimeMillis();
        int count = uc.uniqueCount(list);
        double end = System.currentTimeMillis();
        System.out.print(count + " unique words in ");
        System.out.println((end - start) / 1000 + " seconds using " + uc);
    }

    public static int wordCount(Scanner s) {
        int a = 0;
        while (s.hasNext()) {
            a++;
            s.next();
        }
        return a;
    }

    public static void main(String[] args) throws FileNotFoundException {

        int retval = ourChooser.showOpenDialog(null);

        if (retval == JFileChooser.APPROVE_OPTION) {
            File f = ourChooser.getSelectedFile();
            Scanner s = new Scanner(f);
            ArrayList<String> list = new ArrayList<String>();

            while (s.hasNext()) {
                list.add(s.next());
            }

            String[] array = (String[]) list.toArray(new String[0]);
            String[] copy1 = (String[]) array.clone();
            String[] copy2 = (String[]) array.clone();

            IUniqueCounter uc1 = new SlowUniqueCounter();
            IUniqueCounter uc2 = new SortingUniqueCounter();
            IUniqueCounter uc3 = new SetUniqueCounter();

            System.out.print("reading " + f.getName());
            System.out.println(" # words = " + array.length);
            test(uc1, array);
            test(uc2, copy1);
            test(uc3, copy2);
        } else {
            System.out.println("operation cancelled");
        }
        System.exit(0);
    }
}
```

```
/**
 * @author Owen Astrachan
 * @version August 24, 2004
 * @version Januaray 3, 2006
 *
 */

import java.util.*;

import javax.swing.JFileChooser;

public class UniqueTimer {

    public static double test(IUniqueCounter uc, String[] list) {

        double start = System.currentTimeMillis();
        int count = uc.uniqueCount(list);
        double end = System.currentTimeMillis();
        return (end - start) / 1000.0;
    }

    public static String[] allSame(int n) {
        String[] list = new String[n];
        for (int k = 0; k < n; k++) {
            list[k] = String.valueOf(n);
        }
        return list;
    }

    public static String[] allDifferent(int n) {
        String[] list = new String[n];
        for (int k = 0; k < n; k++) {
            list[k] = String.valueOf(k);
        }
        // Collections.shuffle(Arrays.asList(list));
        return list;
    }

    public static void main(String[] args) {

        String[] same = allSame(1500000);
        String[] diff = allDifferent(1500000);
        IUniqueCounter uc1 = new SortingUniqueCounter();
        IUniqueCounter uc2 = new SetUniqueCounter();

        System.out
                .println("size\t" + uc1 + "\t" + uc2 + "\n-------------------");

        for (int k = 100000; k <= 1500000; k += 100000) {

            String[] copy1 = new String[k];
            String[] copy2 = new String[k];
            System.arraycopy(same, 0, copy1, 0, k);
            System.arraycopy(same, 0, copy2, 0, k);

            double sortTime = test(uc1, copy1);
            double setTime = test(uc2, copy2);
            System.out.println(k + "\t" + sortTime + "\t" + setTime);
        }
    }
}
```