```java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.FileReader;


class HashTable{
  private static final int IntCap=10007;
  private static final double FactorThreshold=0.2;
  private String[] table;
  private int size;
  private int collisions;

  public HashTable(){
    table=new String[IntCap];
    size=0;
    collisions=0;
  }
  public void insert(String word) {
      int index = hash(word);
      if (table[index] == null) {
          table[index] = word;
      } else {
          // Handle collision using linear probing
          collisions++;
          while (table[index] != null) {
              index = (index + 1) % table.length;
          }
          table[index] = word;
      }
      size++;

      // Resize the table if necessary
      if ((double) size / table.length > FactorThreshold) {
          resize();
      }
  }
  private void resize(){
    String[] oldTable=table;
    table=new String[2*oldTable.length];
    size=0;
```

```
    collisions=0;
    for(String key:oldTable){
      if(key!=null){
        insert(key);
      }
    }
  }
public int hash(String key){
  int hash=0;
  for(int i=0;i<key.length();i++){
    hash=(31*hash+key.charAt(i))%table.length;
  }
  return hash;
}


public double collisionPercentage(){
  return ((double)collisions/size)*100;
}


}




public class HashT {
  public static void main(String[] args) {

HashTable HashTable=new HashTable();
 try{
  BufferedReader br=new BufferedReader(new
FileReader("google-10000-english-usa.txt"));
    String line;
    while((line=br.readLine())!=null){
      HashTable.insert(line.trim());
    }
   br.close();
  }
    catch(IOException e){
```

```
    e.printStackTrace();
    }

    System.out.println("Collisions percentage: " +
HashTable.collisionPercentage() + "%");


  }
}
```