**实 验 报 告**

|  |  |
| --- | --- |
| **课程名称** | **数据结构（Java）** |
| **实验项目** | **实验一 线性表** |
|  |  |

|  |  |  |
| --- | --- | --- |
| **学院** |  | **计算机学院** |
| **系别** |  | **软件工程** |
| **班级/学号** |  | **软工2101/2021011561** |
| **学生姓名** |  | **陈志浩** |
| **实验日期** |  | **2022.10.08** |
| **成绩** |  |  |
| **指导教师** |  | **蒋玉茹** |

## 实验一 线性表

**一、实验目的**

1．自定义线性表数据结构

2．在实际问题中应用线性表结构

**二、实验内容及要求**

**【参考指导教师给的文档写】**

**三、系统设计**

**1.概述**

**本实验一共有7个Java文件，1个Python文件，本次云通过使用Github中的Python开源库WordCloud生成，在Java文件中有5个类，1个接口列表展示如下：**

* **Reader类**
* **seqlist类**
* **world\_seqlist类**
* **WordFreq类**
* **Writer类**
* **LList接口**

**2.Reader类的描述**

**属性及含义:**

**Reader类无属性**

**方法及功能：**

**Reader类的返回类型是String数组，Reader类通过BufferReader读取路透社.txt文本中的内容，当.readline()不为零时，持续读入。新建一个String 类型的变量s，将读入的数据进行粘合，最后创建String类型的数组demo用于接收s.split()的数据，最后返回demo数组。**

**3.seqlist类的描述**

**属性及含义：**

**seqlist类含有两个属性，分别为Object类型的数组和int类型的len用于表示线性表的长度**

**方法及功能：**

**seqlist一共有10个方法分别是：**

**1.void clean()方法 清空线性表**

**2.boolean isEmpty()方法 判断线性表是否为空**

**3.int size()方法 返回线性表的大小**

**4.T get(int i)方法 获取第i号位置上的元素**

**5.boolean set(int i, Object x)方法 将第i号位置元素设置为x**

**6.void add(int I, Object x)方法 在第i号位置添加x**

**7.T remove(int i)方法 删除第i号位置上的元素并返回该元素**

**8.T indexOf(T x)方法 查找x是否在线性表中**

**9.boolean find(T x)方法 查找x是否在线性表中，若在线性表中则返回true反之返回false**

**10.String toSting()方法 重构toString方法，使线性表能够被println语句输出**

**3.word\_seqlist类的描述**

**属性及含义：**

**word\_seqlist类无属性**

**方法及功能：**

**creat\_seqlist(seqlist<WordFreq> list, int a)方法 需要传入两个值分别为含有WordFreq对象的seqlist数组，以及整形a，当输入的a等于0时，该方法将会返回停用词线性表；当输入为1时，该方法将会返回不含停用词的线性表**

**4.WordFreq类的描述**

**属性及含义：**

**WordFreq类含有两个属性分别为world和value，分别表示被统计词，被统计词的词频**

**方法及功能：**

**WordFreq类一共有四个方法分别为：**

**1.void set\_WordFreq()方法 当调用该方法时value的值增加1**

**2.String get\_world()方法 当调用此方法时返回被统计词**

**3.int get\_value()方法 当调用此方法时返回被统计词的词频**

**4.String toString()方法 重构toString方法，使WordFreq中的元素能够被println语句输出**

**5.Writer类的描述**

**属性及含义：**

**Writer类中无属性**

**方法及功能：**

**Writer类是将词与词频按行一次存入txt文件中方便Python进行读取。Writer类中的void write\_file(seqlist<WordFreq> list)没有返回类型，该方法接收一个元素类型为WordFreq的线性表,遍历线性表并调用WordFreq的toString方法，将得到的字符串存进字符串数组中，接着使用Java IO操作将字符串数组中的元素写入txt文件中保存。**

**6.LList接口的描述**

**属性及含义：**

**该接口含有9个方法：**

**1.void clean(); // 清空线性表  
2.boolean isEmpty(); // 判断线性表是否为空  
3.int size(); // 返回线性表的长度  
4.T get(int i); // 返回第i个元素  
5.boolean set(int i, T x); // 将第i号位置元素设置为x  
6.void add(int i, T x); // 在第i号位置插入元素x  
7.T remove(int i); // 删除第i号元素并返回被删除的对象  
8.T indexOf(T x); // 查找首次出现的x元素  
9.boolean find(T x); // 查找首次出现的x元素**

**7.词云生成程序main.py**

**该程序调用Python中的imageio库，matplotlib.pyplot库，以及WordCloud库。该程序首先通过open函数读取Java Writer类生成的txt文件，通过分割函数，分别将词和词频存入两个不同的列表，随后调用zip函数将两个列表变成一个字典，使用WordCloud中的fit\_words函数，使WorldCloud根据词频生成词云图，最后调用matplotlib.pyplot库，将词云图显示出来。**

**四、类图**

**表格

描述已自动生成**

**四、系统环境**

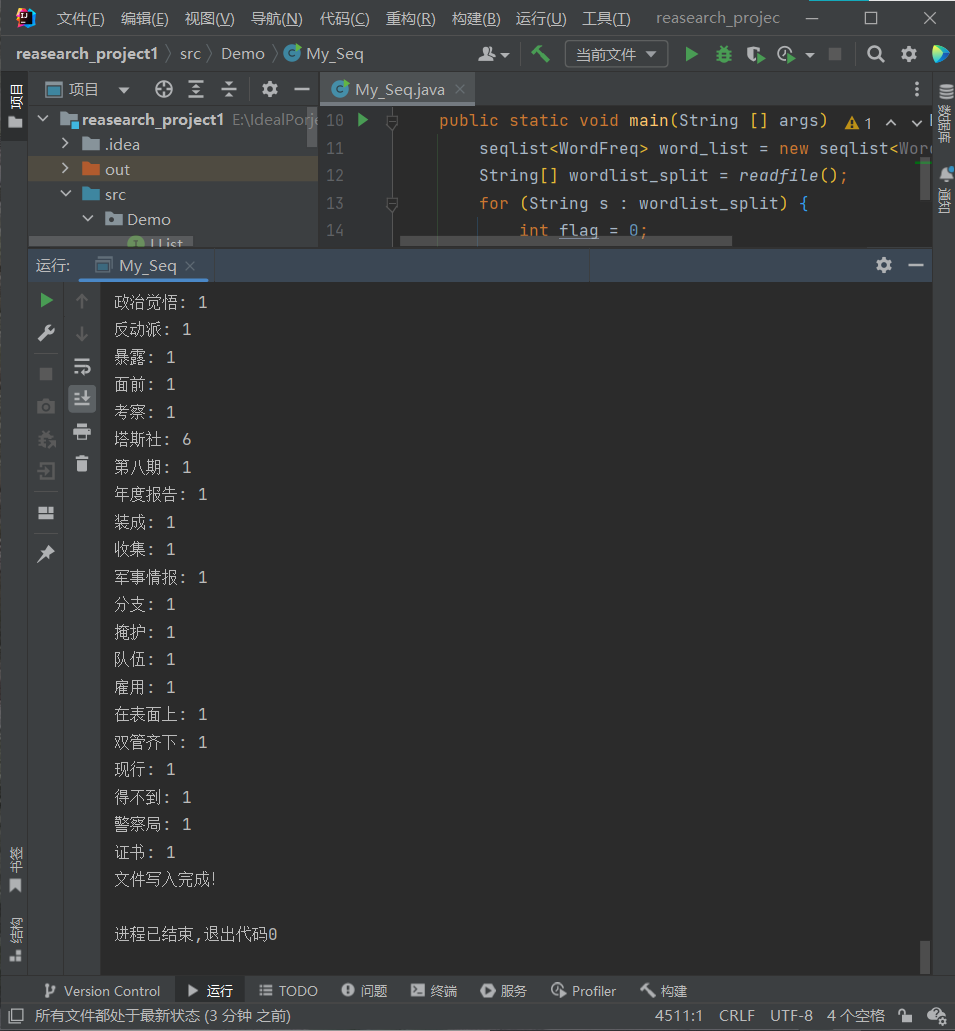
**1.操作系统及版本 Windows 10 家庭中文版**

**2.JDK及版本 java version “1.8.0\_15”**

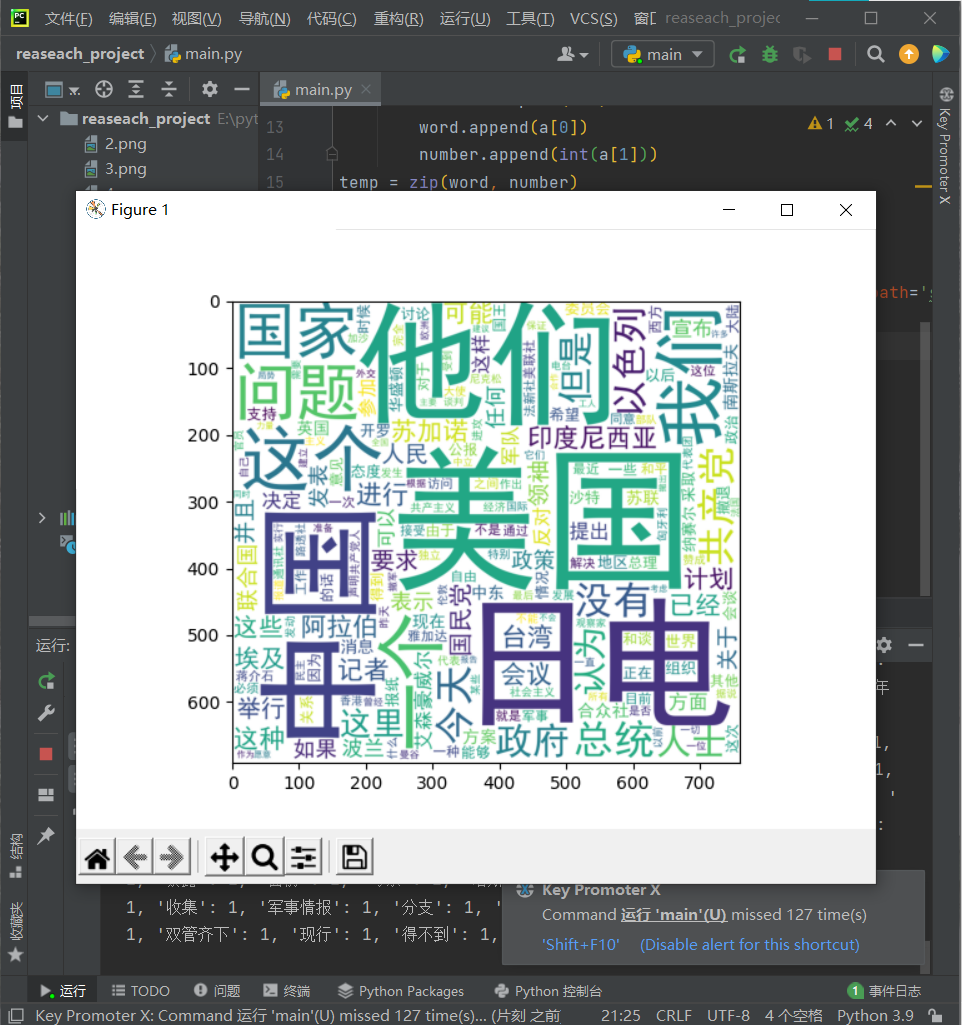
**3.其他JAR包等 无**

**五、程序运行过程**

**【步骤1】将已分好词的txt文件复制到src目录下，运行src目录下的My\_Seq.java文件，运行结果如下：**

****

**【步骤2】运行python文件：main.py得到词云图，运行结果如下：**

****

**六、编写程序中遇到的问题总结**

**【依次列举编程中遇到的问题及解决过程】**

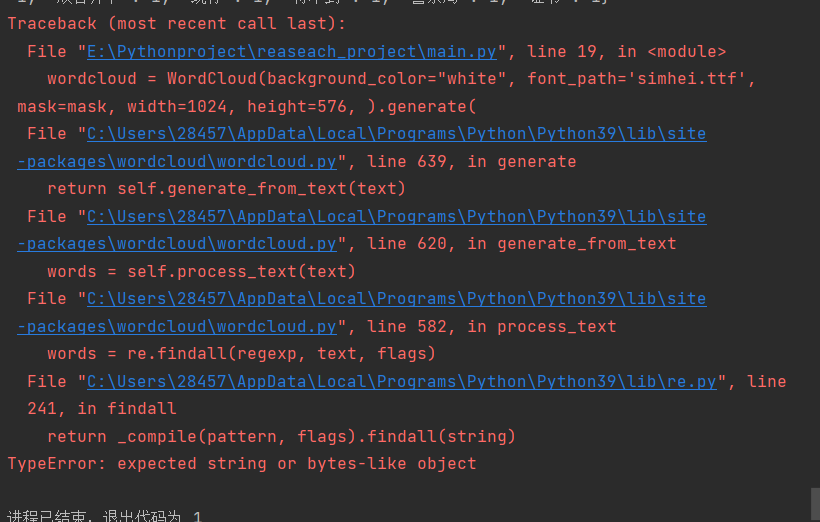
**1.第一次实现词频统计时使用了其他数据结构**

**在刚开始实现词频统计时，未使用动态键，我首先通过将读入的字符串分割，得到字符串，通过将数组转化成list在转化成set进行去重，然后将集合中的元素依次添加到线性表中，随后再次对字符串数字进行遍历，并将线性表中相应的词频加1。解决办法：通过使用动态键将元素添加到线性表中，首先遍历被分割的字符数组，同时嵌套一个for循环依次，嵌套的for循环遍历线性表，当线性表中元素与字符串元素相同时，该词对应的词频加1。**

**2.调用WordCloud生成词云时报错**

**WordCloud库生成词云时，有两个函数分别为.generate()和.fit\_words()函数，其中.generate()函数需要传入一段文本才能生成词云，我将压缩得到的数组传递给了.generate()函数发生了错误，导致词云生成失败，解决方法将.generate()函数变为.fit\_word()函数。**

**报错信息：**

****

**七、结果展示**



**图片包含 图示

描述已自动生成**

**八、参考文献**

**1.csdn wordcloud生成词云图(含形状颜色设置)** [(57条消息) wordcloud生成词云图（含形状、颜色设置）\_IRON POTATO的博客-CSDN博客\_wordcloud生成词云](https://blog.csdn.net/qq_48003414/article/details/117427378?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522166494903016781432930239%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request_id=166494903016781432930239&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~top_positive~default-1-117427378-null-null.142%5ev51%5econtrol,201%5ev3%5eadd_ask&utm_term=wordcloud&spm=1018.2226.3001.4187)

**九、源代码**

|  |
| --- |
| **接口： LList** |
| package Demo;  public interface LList<T> {  void clean(); // 清空线性表  boolean isEmpty(); // 判断线性表是否为空  int size(); // 返回线性表的长度  T get(int i); // 返回第i个元素  boolean set(int i, T x); // 将第i号位置元素设置为x  void add(int i, T x); // 在第i号位置插入元素x  T remove(int i); // 删除第i号元素并返回被删除的对象  T indexOf(T x); // 查找首次出现的x元素  boolean find(T x); // 查找首次出现的x元素 } |

|  |
| --- |
| **类：My\_Seq** |
| package Demo;  import java.io.IOException; import java.util.\*; import static Demo.Reader.*readfile*; import static Demo.Writer.*write\_file*; import static Demo.word\_seqlist.*creat\_seqlist*;  public class My\_Seq {  public static void main(String [] args) throws IOException {  seqlist<WordFreq> word\_list = new seqlist<WordFreq>();  String[] wordlist\_split = *readfile*();  for (String s : wordlist\_split) {  int flag = 0;  for (int j = 0; j < word\_list.size(); j++) {  if (Objects.*equals*(word\_list.get(j).get\_world(), s)) {  word\_list.get(j).set\_WordFreq();  flag = 1;  }  }  if (flag == 0) {  word\_list.add(word\_list.size(), new WordFreq(s, 1));  }  }  seqlist<WordFreq> stop\_words\_list1 = *creat\_seqlist*(word\_list, 1);  for(int i = 0; i< stop\_words\_list1.size(); i++) {  System.*out*.println(stop\_words\_list1.value[i].toString());  }  *write\_file*(stop\_words\_list1);  } } |

|  |
| --- |
| **类：Reader** |
| package Demo;  import java.io.\*;  public class Reader {  public static String[] readfile() throws IOException {  String encoding = "utf-8";  File file=new File("E:\\IdealPorject\\reasearch\_project1\\src\\2.txt");  BufferedReader in = new BufferedReader(  new InputStreamReader(  new FileInputStream(file)  )  );  String s = "";  String line;  while((line=in.readLine())!=null){  s += line;  }  String[] demo = s.split(" ");  /\*for(String a : demo){  System.out.println(a);  }\*/  // System.out.println(demo);  return demo;  }  public static void main(String [] args) throws IOException {  *readfile*();  } } |

|  |
| --- |
| **类：seqlist** |
| package Demo;  import Demo.LList;  public class seqlist<T> implements LList<T> {  public Object[] value;  public int len;  public seqlist(int max\_size){  this.value = new Object[max\_size]; // 创建数组  this.len = 0;  }  public seqlist(){  this.value = new Object[9999]; // 创建数组  this.len = 0;  }    @Override  public void clean() { // 清空线性表  this.len = 0;  }   @Override  public boolean isEmpty() {  return this.len == 0; // 判断线性表是否为空  }   @Override  public int size() {  return this.len; // 返回线性表的大小  }   @Override  public T get(int i) {  T a;  if(i>=0 && i<this.len){  a =(T)this.value[i]; // 获取第i个位置上的元素  }else{  a = null;  }  return a;  }   @Override  public boolean set(int i, Object x) {  if(x==null){ // x为空返回false  return false;  }  if(i>=0 && i < this.len) {  this.value[i] = x;  return true;  }else{  throw new IndexOutOfBoundsException("index outof bound");  }  }   @Override  public void add(int i, Object x) {  if(x == null){  return ;  }  if(this.len == value.length){  Object[] temp = this.value;  this.value = new Object[temp.length \* 2];  for(int j = 0; j< temp.length; j++){  this.value[j] = temp[j]; // 将temp中的代码复制到value中  }  }  if(i<0){  i=0;  }  if(i>this.len){  i = this.len;  }  for(int j=this.len-1; j>=i;j++){  this.value[j+1] = this.value[j]; // i之后元素向后移动  }  this.value[i] = x;  this.len++;  }   @Override  public T remove(int i) {  if(this.len==0 || i<0 || i>=this.size()){  return null;  }  T a = (T)this.value[i];  for(int j = this.len-1; j>=i; j--){  this.value[i] = this.value[i+1]; // 后面的元素向前移动  }  this.value[this.len-1] = null; // 让之前最后一个元素为零  this.len--;  return a;  }   @Override  public T indexOf(T x) {  for(int i = 0; i<this.len; i++){ // 查找x是否在线性表中  if(x.equals(this.value[i])){ // 判断是否相等  return (T)this.value[i]; // 强制类型转换  }  }  return null;  }   @Override  public boolean find(T x) {  for(int i = 0; i<this.len; i++){ // 查找x是否在线性表中  if(x.equals(this.value[i])){ // 判断是否相等  return true; // 强制类型转换  }  }  return false;  }   @Override  public String toString() { // 重构toString方法  String a = "";  for(int i = 0; i<this.len; i++){  a += this.value[i] + " ";  }  return a;  } } |

|  |
| --- |
| **类：word\_seqlist** |
| package Demo;  import java.util.Objects;  public class word\_seqlist {  public static seqlist<WordFreq> creat\_seqlist(seqlist<WordFreq> list, int a) {  seqlist<WordFreq> word\_list = new seqlist<WordFreq>();  if(a==0) {  for (int i = 0; i < list.size(); i++) {  if (list.get(i).get\_world().length() <= 1) {  WordFreq temp = list.get(i);  word\_list.add(list.size(), temp);  }  }  }  if(a==1) {  for (int i = 0; i < list.size(); i++) {  if (list.get(i).get\_world().length() > 1) {  WordFreq temp = list.get(i);  word\_list.add(list.size(), temp);  }  }  }  return word\_list;  } } |

|  |
| --- |
| **类：WordFreq** |
| package Demo;  public class WordFreq {  private String world;  private int value;   public WordFreq(){  this.world = "";  this.value = 0;  }   public WordFreq(String world, int value){  this.world = world;  this.value = value;  }   public WordFreq(String world){  this.world = world;  this.value = 1;  }   public void set\_WordFreq(){  this.value++;  }   public String get\_world(){  return this.world;  }  public int get\_value(){  return this.value;  }   @Override  public String toString() {  return this.world + ": " + this.value;  } } |

|  |
| --- |
| **类：Writer** |
| package Demo;  import java.io.BufferedReader; import java.io.BufferedWriter; import java.io.FileWriter; import java.io.IOException;  public class Writer {  public static void write\_file(seqlist<WordFreq> list){  String[] string\_list = new String[list.size()];  for(int i = 0; i< list.size(); i++) {  string\_list[i] = (list.value[i].toString());  }  try{  BufferedWriter out = new BufferedWriter(  new FileWriter("wordlist.txt"));  String s = "";  for (String value : string\_list) {  s = value + "\n";  out.write(s);  }  out.close();  System.*out*.println("文件写入完成！");  }catch (IOException ignored){}   }  } |

|  |
| --- |
| **Python程序： main.py** |
| import imageio import matplotlib.pyplot as plt from wordcloud import WordCloud   file = "E:\\IdealPorject\\reasearch\_project1\\wordlist.txt" with open(file, "r") as f:  word = []  number = []  data = f.readlines()  for line in data:  a = line.split(":")  word.append(a[0])  number.append(int(a[1])) temp = zip(word, number) final\_dic = dict(temp) print(final\_dic) mask = imageio.v3.imread("7.png") wordcloud = WordCloud(background\_color="white", font\_path='simhei.ttf', mask=mask, width=1024, height=576, ).fit\_words(  final\_dic) wordcloud.to\_file('demo7.jpg') plt.imshow(wordcloud) plt.show() |