**深 圳 大 学 实 验 报 告**

**课程名称： 编译原理**

**实验项目名称： 实验一、高级语言及其文法**

**学院： 计算机与软件学院**

**专业： 软件工程**

**指导教师： 尹剑飞**

**报告人： 张欣杰 学号：2020151091 班级： 软工02**

**实验时间： 2023年2月20日至3月11日**

**实验报告提交时间： 2023年 3 月 10 日**

**教务处制**

|  |
| --- |
| **实验目的与要求：**  **目的：**针对分词、停用词过滤等文法分析应用问题，设计并实现相应的解决方案，再通过设计文法相关类族，以及实现文法化简等方法，既加深对抽象的文法、推导、语言等形式语言理论基础概念的理解与掌握，也加强对面向对象程序编写能力和计算思维的培养。  **要求:**  **第一部分： 分词**  输入：输入是一个文本文件（见附件a.txt），里面的内容是符合C语言语法、词法要求的源代码，例如“position = initial + rate \* 60;”；  输出：设计并实现对输入文件的处理（禁用正则表达式），以获得如下输出，输出是一个文本文件，对前面例子，里面内容是：  (标识符，position)  (赋值运算符，=)  (标识符，initial)  (加法运算符，+)  (标识符，rate)  (乘法运算符，\*)  (整数，60)  (分隔符，; )  **第二部分：停用词过滤，**分为“产生随机停用词”、“产生随机待过滤文本”和“过滤停用词”3个小实验。  产生随机停用词：生成一个文件长度和单词平均长度可控的文本文件，要求字符集为26个字母，每行生产一个单词，每个单词的长度随机，全部单词的平均长度可控制，通常为4、5（非严格要求），总共输出的文件长度可控制。  产生随机待过滤文本：生成一个文件大小和单词平均长度可控的文本文件，要求  字符集为26个字母+“ ”（空格）+“\r\n”（回车换行）；从文件开始一直到结束，持续输出，不额外换行（除非遇到生成的\r\n），总共输出的文件大小可控制。  过滤停用词：写一个程序，快速扫描待过滤文本，然后将待过滤文本中出现的所有停用词，替换为“\*\*”，要求禁用正则表达式，不要误杀（例如，若停用词包括“abc”，那么“abcd”等不应该被误杀。  **第三部分：文法分类**  设计一个面向对象程序，实现下述功能：  1）可以从一个文本文件读入一个文法，从而在内存中将一个文法表示为产生式数组类ProductionArray（或文法Grammar类），而每个产生式类Production又可表示为非终止符类（VNSymbol）和产生式右侧的文法符号数组SymbolList（终止符VTSymbol和非终止符VNSymbol组成的数组）。ProductionArray支持一组方法，包括：  1.1）ProductionArray loadFrom(String fileName)/boolean writeTo(String fileName)：从一个文本文件装载并实例化一个ProductionArray对象（或将一个ProductionArray对象写到一个文本文件中）  1.2）boolean isVT(String symbol)/bolean isVN(String symbol)：判断一个符号串是否是终止符还是非终止符  1.3) String getGrammarType(): 判断该文法是RG、CFG、CSG还是PSG。 |
| **方法、步骤：**  要完成本实验，依据实验要求进行分解，需要完成的实验步骤是：   1. 如何读写文件?   读写文法文件内容，需要用到文件IO，查阅、复习文件IO操作。  在C++中，引入头文件<fstream>或者<ifstream>，即文件流类，该头文件中的代码内部对<<输入流进行了运算符重载，同样，在<fstream>和<ofstream>中也对<<输出流进行了运算符重载；所以，所以，当fstream或者ifstream类对象打开文件之后，就可以直接借助>>输入流运算符，读取文件中存储的字符（或字符串）；当fstream或者ofstream类对象打开文件后，可以直接借助<<输出流运算符向文件中写入字符（或字符串）。再pyrhon中可以使用open()函数打开指定文件，然后使用read()方法可以读文件，然后write()可以对文件进行写操作。   1. 如何分词？   第一部分实验要求进行分词，源代码中的单词可以分成3类，一类是约定的保留字，一类是普通标识符，最后一类是数字   * 1. 如何识别保留字？   在源代码中，通过观察，我们不难发现，保留字都是纯英文符号的，因此在读入文件的时候，若遇到字母，可以先使用一个变量s存储读入的字符串，若遇到空格或者换行的时候，则说明当前字符串是一个完整的单词；当读入一个完整的字符串之后，将已经得到的字符串s与我们编写的保留字数组进行比对，如果s与该数组中的关键字相同，则该字符串s为保留字。除此之外，符号‘#’之后的字符串也必定为保留，例如{”include” , ”define”}等等。  伪代码实现：  If input.get()==Alpha:  JudgeKeyword(input)  JudgeKeyword(ifstream &input)  If input.get()==Alpha:  str+=input.get()  input.get()  End if  If str==Keyword  Output(“Keyword”, str)   * 1. 如何识别标识符？   标识符的识别与关键字识别方法相似，只不过有可能字符串中可能出现下划线或者数字。  伪代码实现：  If input.get()==Alpha||’\_’:  JudgeIdWord(input)  JudgeIdWord(ifstream &input)  If input.get()==Alpha || ’ ’ || Digit:  str+=input.get()  input.get()  End if  Output(“IdWord”, str)   * 1. 如何识别数字？   在标识符和关键字中，由于都不能使用数字开头，因此使用数字开头的即为字符串即为普通数字而不是标识符中的关键字  伪代码实现：  If input.get()==Digit:  JudgeDigit(input)  JudgeKeyword(ifstream &input)  If input.get()==Digit:  str+=input.get()  input.get()  End if  Output(“NUM”, str)  特别的，如果你有多种方法来识别这3类单词，那这些方法有何优缺点？分析并做出你的选择。  在识别关键字和标识符的时候，除了上述的方法之外，我还想到可以使用容器map对关键字和标识符进行哈希，即将代码中存在的关键字和标识符使用map<string,int>进行一一映射，将存在的字符串的map值标为1，然后在读入代码文本的时候，每读入一个字符串，若不是普通数字，则使用map[str]来判定该字符串str是否为哈希表中已存在的字符串，即map[str]=1；这种方法相对于上面的方法来说时间复杂度会大大降低，因为在比对的时候只需要比对一次，不需要像数组那样一一遍历；但是空间复杂度会相较于原本的方法大，应该说是一种空间换时间的方法。   1. 如何产生符合要求的随机停用词和待处理文本   第二部分实验要求先产生随机的停用词和待处理文本，主要是如何产生符合要求的这些词或文本？   * 1. 如何产生随机停用词？   在python中有一个random模块，random模块中的randint(a,b)函数，该函数需要两个参数a和b，该函数的作用是在[a,b]范围内随机产生一个整数，然后使用chr(random.randint(97,122))随机产生a-z中的随机字母，多次循环将每个字符放入字符串str中即可得到一个随机停用词。  伪代码：  import random  create\_word()  for i = 0 to len:  str+=chr(random.randint(97,122))  return str   * 1. 如何控制随机停用词的平均长度（而不是固定长度）？   在python中有一个random模块，random模块中的randint(a,b)函数，该函数需要两个参数a和b，该函数的作用是在[a,b]范围内随机产生一个整数，使用这个函数生成一个整数n，进行n此随机产生一个字符的操作，即可将单个单词的长度限定在指定的范围内。  伪代码：  import random  create\_word()  word\_len=random,randint(2,8)#生成长度为2-8的单词  for i = 0 to len:  str+=chr(random.randint(97,122))  return str   * 1. 如何产生待处理文本中的“段落”？   使用随机数生成0-27范围内的一个整数，0~25代表a-z,26代表空格，27嗲表换行符，这样既可做到段落的转换。  import random  create\_word()  word\_len=random,randint(2,8)#生成长度为2-8的单词  for i = 0 to len:  ch\_form=random.randint(0,27)  if ch\_form==26:  str+=’ ’  elif ch\_form==27:  str+=’\r’  else:  str+=chr(97+ch\_form)  return str   * 1. 如何控制待处理文本的长度？   在函数外部定义一个全局变量cnt用来计算生成的单词的总长度，当cnt大于自己限定的一个值的时候则退出循环，文本生成完成，这样即可控制生成的文本的总长度。  import random  cnt=0  create\_word\_txt()  global cnt  line=random.randint(a,b)  for i = 0 to line:  word\_list=create\_word\_list()  for j in word\_list:  cnt+=len(j)  str+=j  if cnt>=n:  break  return str  create\_word\_list()  num=random.randint(10,15)#生成单词个数为10-15的单词列表  word\_list=[]  for i = 0 to num:  word\_list.append(create\_word())  return word\_list   1. 如何过滤停用词   第二部分实验最后要求把待处理文本中出现的停用词替换为“\*\*”，那你如何准确、快速判断出文本中的停用词？  首先定义 create\_txt 函数，生成一个包含100个随机单词的单词列表文件 word\_list.txt生成 100 个随机单词将随机单词写入word\_list.txt文件，读取单词列表文件 word\_list.txt，将单词存储为列表 word\_list，读取需要过滤的文本文件 test.txt，将每行文本存储为列表 passage，将每行文本以空格分割为单词，并删除每行最后一个元素（为空格），遍历 passage 中的每个单词，如果单词在单词列表 word\_list 中，将其替换为 \* 号，将替换后的单词写入文本文件 passage.txt，输出“文本过滤完成”的提示信息。  伪代码：  fp=open(‘word\_list.txt’)  Word\_list=fp.read().split(‘ ’)  fp=open(‘test\_list.txt’)  passage=fp.readline()  for i = 0 to len(passage)  passage[i]=passage[i].split(‘ ’)  for i = 0 to len(passage)  for j = 0 to len(passage[i])  if passage[i][j] in word\_list  passage[i][j] = ’\*’\*(len(passage[i][j]))   1. 在第三部分的实验中，有哪些数据对象？数据对象是指具有一组（不可分离的）数据属性的实体，例如：产生式是一个数据对象，其拥有的哪些属性？试说明在第三部分的实验中，这些数据对象是如何相互调用的？   ProductionArray 中包含产生式列表，可以通过调用其中的产生式对象来获取非终止符对象或符号列表对象; 非终止符对象可以通过其名称属性调用到该非终止符的产生式列表，即获取由该非终止符左侧的产生式列表; 产生式对象中包含符号列表对象，可以通过其调用其中的终止符或非终止符对象; 文法对象中包含产生式列表，可以通过其中的产生式列表获取到每个产生式对象，也可以通过文法类型属性获取文法类型。 |
| **实验过程及内容：**   1. 文法分类代码实现 2. 初始化     使用production类来分别存储产生式的左部和右部，Grammer类用来分割产生式，将变量，终结符，产生式和开始符分别用V,T,P,S进行存储。   1. 存储文法的要求   V中的的变量必为大写字母；终结符T必须满足V∩T≠空；产生式需要满足格式A->B；第一行必须为变量V，第二行必须为终结符T，最后一行必须为开始符号S，第三行到倒数第二行为产生式P。   1. Init()函数对变量进行初始化分割   第一行是V，第二行是T，第三到倒数第二行是P，最后一行是S。按照这个顺序将grammer类中的变量进行初始化     1. 检查四元组语法   首选需要对变量V进行检查，如果V中的字母不为大写字母，则输出变量不合法信息    对终结符的进行检查    对其实符号进行检查    检查产生式格式    检查左部    检查右部     1. 文法分类   1型文法    2型文法    3型文法         1. 文法分类主程序   首先对文法的合法性进行判断，如果合法，则进行文法分类  由于1、2、3型文法之间的关系为1型文法∈2型文法∈3型文法，因此首先需要对3型文法进行判断然后再进行2型文法判断最后在进行1型文法判断，如果文法是合法的，但是不属于1、2、3型文法，则该文法类型为0型文法。 |
| **实验结论：**   1. 测试用例的设计与说明   设置一个1型文法进行判断    验证结果    设置一个2型文法进行验证    验证结果    设置1个3型文法进行验证    验证结果    设置一个不符合文法要求的文本进行验证    验证结果    设置0型文法进行验证    验证结果     1. 测试结果的说明与分析   对测试（实验）结果进行分析说明  根据实验所设置的测试样本可知，程序运行结果基本正确，能够正确分辨出该文本文件中的式子是否为文法，能够将合法的文法按照定义分出0，1，2，3型文法。 |
| **心得体会：**  通过本次实验，我学会了如何编写代码对C++文件进行关键字、标识符、运算符等符号的分类，学会了如何产生随机停用词，并对停用词进行过滤。  在最后一部分，掌握了文法的分类标准，并能够编写程序对给定的文本文件中的文法进行分类操作。  理解：文法（Grammar）指的是一套用于生成语言的规则，包括产生式、终结符、非终结符、起始符等要素。文法是形式语言的基础，用于描述计算机语言、自然语言等等各种语言。  直接推导（Immediate Derivation）是指将一个非终结符号替换成它的某个产生式的过程，该产生式中所有非终结符都被替换成它们对应的产生式，得到一个新的句型。  推导（Derivation）是指从起始符号开始，通过一系列直接推导得到的一串句型的过程。在推导的过程中，每一步都是将某个非终结符号替换成它的某个产生式，并最终得到一个由终结符组成的句子。  句型（Sentence Form）指的是由文法中的非终结符号和终结符号构成的一个字符串，表示一种潜在的句子。  句子（Sentence）指的是由文法中的终结符号构成的一个字符串，它是一个可以被语法分析器接受的语言字符串。  语言（Language）指的是由一个或多个句子构成的集合，这些句子都符合一定的语法规则。在计算机科学中，通常将语言分为形式语言和自然语言两类。形式语言是由计算机科学家和数学家设计的、用于与计算机交互的语言，如编程语言、数据描述语言等；自然语言则是人们平时使用的语言，如中文、英文等。 |
| 指导教师批阅意见：  成绩评定：  指导教师签字：  年 月 日 |
| 备注： |