



SHANGHAI UNIVERSITY

**<编译原理>实验报告**

|  |  |
| --- | --- |
| **学 院** | **计算机工程与科学学院** |
| **组 号** | **10** |
| **实验题号** | **一** |
| **日期** | **2023.3.29** |

|  |  |  |  |
| --- | --- | --- | --- |
| 学号 | 姓名 | 主要工作 | 贡献因子 |
| 20124652 | 徐驰 | 报告撰写，代码测试 | 0.25 |
| 20124698 | 陈世锦 | 代码优化，使之优美 | 0.25 |
| 20124699 | 李子浩 | 代码构思，初步实现 | 0.25 |
| 20124703 | 邵启承 | 代码优化，使之高效 | 0.25 |

**<编译原理>实验报告**

**一、实验目的与要求**

* 根据 PL/0 语言的文法规范，编写 PL/0 语言的词法分析程序。
* 通过设计调试词法分析程序，实现从源程序中分出各种单词的方法加深对课堂教学的理解；提高词法分析方法的实践能力。 
* 掌握从源程序文件中读取有效字符的方法和产生源程序的内部表示文件的方法。
* 掌握词法分析的实现方法。 
* 上机调试编出的词法分析程序。

**二、实验环境**

微机 CPU P4 以上，256M 以上内存，安装好 C 语言，或 C++，或 Visual C++

**三、实验内容**

输入 PL/0 语言程序，输出程序中各个单词符号（关键字、专用符号以及其它标记）。

**四、实验内容的设计与实现**

**说明：**选择本实验中最有程序设计技巧或特色的并具有独立功能的源代码片断，并对其算法的技巧或特色进行必要的文字说明。

1. **头文件说明**

#include<bits/stdc++.h> 常用的头文件，包含了一些常用的标准库头文件。

#include<fstream> 头文件包含了与文件输入输出相关的类和函数。

#include<string> 头文件包含了 string 类型及其相关的函数和操作符。

#include<map> 头文件包含了关联式容器 map 及其相关的函数和操作符。

#include<vector> 头文件包含了序列式容器 vector 及其相关的函数和操作符。

#include<iostream> 头文件包含了输入输出流相关的类和函数。

#include<sstream> 头文件包含了 stringstream 类及其相关的函数和操作符。

1. **文件读入和初始化**

首先按行读入，然后对字符进行切分，并对读入的每个英文字符都转为小写，以实现不区分大小写。然后对读入的每一个字符进行判断其是否为标识符。

下面为代码示例以及详细说明：

void reshape(string& s) {  
 for (int i = 0; i < s.size(); i++) {  
 if (s[i] >= 'A' && s[i] <= 'Z')  
 s[i] = 'a' + (s[i] - 'A');  
 }  
}  
  
string basic[] = {  
 "begin", "call", "const", "do","end",  
 "if","odd","procedure","read",  
 "then", "var", "while", "write"  
};  
  
int main() {  
 ifstream infile("in.txt"); // 创建一个输入文件流  
 ofstream outfile("out.txt"); // 创建一个输出文件流  
 string str, s1, now = "";  
 map<string, int>ans;  
 vector<string>show;  
 while (getline(infile, str)) {  
 stringstream ss(str);  
 while (ss >> s1) {  
 reshape(s1);  
 标识符判断；  
 }  
 }  
 for (auto i : show) {  
 cout << "(" << i << ": " << ans[i] << ")" << endl;  
 outfile << "(" << i << ": " << ans[i] << ")" << endl;  
 }  
 infile.close(); // 关闭文件  
 outfile.close();  
 return 0;  
}

* 初始化

**string basic[]** 是一个字符串数组，包含了 PL/0 语言中的基本关键字（begin, call, const, do, end, if, odd, procedure, read, then, var, while, write）。

**map<string, int>ans** 创建了一个关联式容器，用于存储每个标识符出现的次数。

**vector<string>show** 创建了一个序列式容器，用于存储出现过的标识符。

* 文件操作

**ofstream outfile("out.txt")** 创建了一个输出文件流对象，将输出结果写入到名为 "out.txt" 的文件中。**ifstream infile("in.txt")** 创建了一个输入文件流对象，读取输入数据来自名为 "in.txt" 的文件。

**while (getline(infile, str))** 通过输入文件流对象从文件中循环按行读取数据，存储在字符串变量 str 中。

**stringstream ss(str)** 创建了一个 stringstream 对象，用于将字符串 str 当做输入流进行读取操作。

**while (ss >> s1)** 通过 stringstream 对象从输入流中读取一个字符串 s1，该循环用于处理每行中的所有单词，再调用**reshape()** 函数将字符串中的大写字母转换为小写字母，以实现不区分大小写。

1. **标识符判断及统计**

对于每个字符串s1，按字符读入，若查询到一个新的以字母开头的字符串，对于字母与数字，压入当前字符串now中。

注：int j = 0 定义变量 j，用于标记当前是否查询到了一个以字母开头的字符串。

int j = 0; //是否查询到一个新的以字母开头的字符串  
 int len = s1.size();  
 for (int i = 0; i < len; i++) {  
 if (s1[i] >= 'a' && s1[i] <= 'z') {  
 if (!j) { //该字符串以字母开头  
 j = 1;  
 now = "";  
 now += s1[i];  
 }  
 else  
 now += s1[i];  
 }  
 else if (s1[i] >= '0' && s1[i] <= '9') {  
 if (j) { //已经字母开头  
 now += s1[i];  
 }  
 }

若遇到标点符号等，表明字符串s1可能含有多个标识符，需要继续循环判断，而对于每个有可能的标识符，需要与basic[]比较判断其是否为基本字，若不是基本字，则为标识符并统计个数。

注：int ok = 0 定义变量 ok，用于标记当前查询到的字符串是否为基本关键字。

else { //碰到了标点符号等  
 if (j) {  
 int ok = 0; //不在基本字中  
 for (int jj = 0; jj <= 12; jj++) {  
 if (now == basic[jj]) {  
 ok = 1;  
 break;  
 }  
 }  
 if (!ok) {  
 ans[now]++;  
 if (ans[now] == 1)  
 show.push\_back(now);  
 }  
 j = 0;  
 }  
 }  
 }

若未遇到标点符号，表明字符串s1中仅有一个可能的标识符，将其与basic[]比较判断是否为基本字，若不是基本字，则为标识符并统计个数。

if (j) {  
 int ok = 0; //不在基本字中  
 for (int jj = 0; jj <= 12; jj++) {  
 if (now == basic[jj]) {  
 ok = 1;  
 break;  
 }  
 }  
 if (!ok) {  
 ans[now]++;  
 if (ans[now] == 1)  
 show.push\_back(now);  
 }  
 j = 0;  
 }

**if (ans[now] == 1)** 如果当前查询到的标识符是第一次出现，则将其加入到序列式容器 show 中。

1. **文件输出**

for (auto i : show) {  
 cout << "(" << i << ": " << ans[i] << ")" << endl;  
 outfile << "(" << i << ": " << ans[i] << ")" << endl;  
}  
infile.close(); // 关闭文件  
outfile.close();

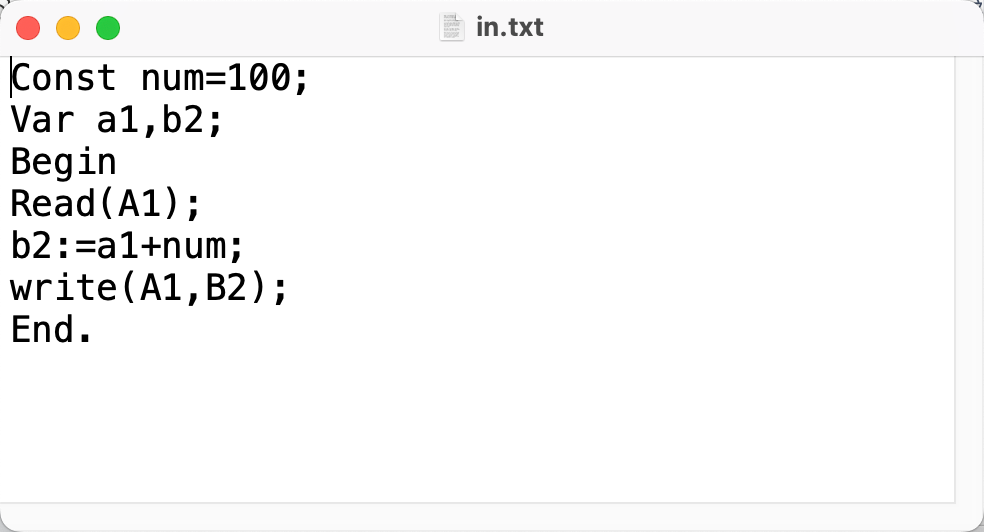
**for (auto i : show)** 循环遍历show，输出每个标识符及其出现次数。

**infile.close()** 关闭输入文件流对象。

**outfile.close()** 关闭输出文件流对象。

1. **测试用例与实验结果**

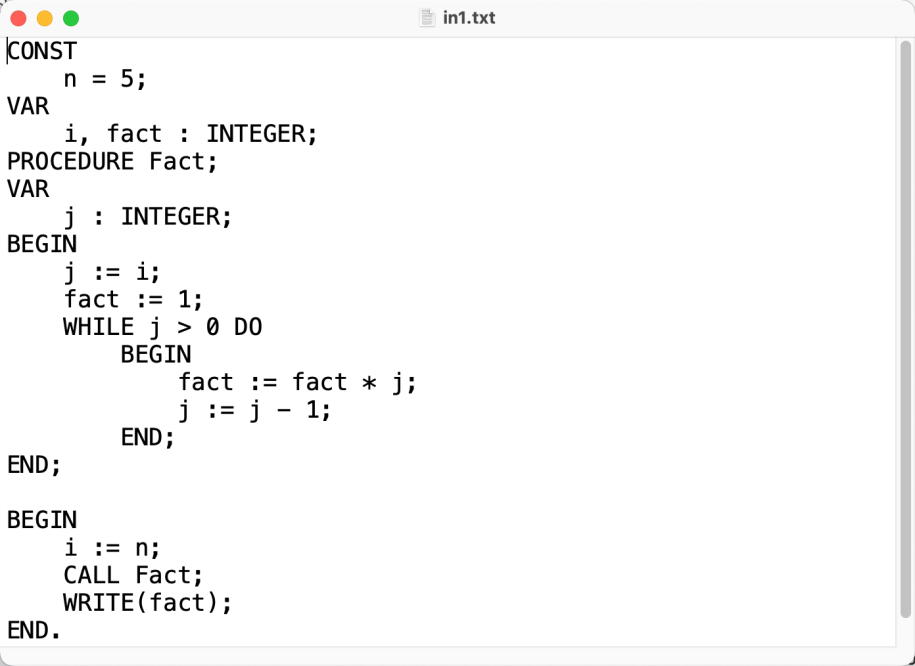
**测试用例1：**



**运行结果：**



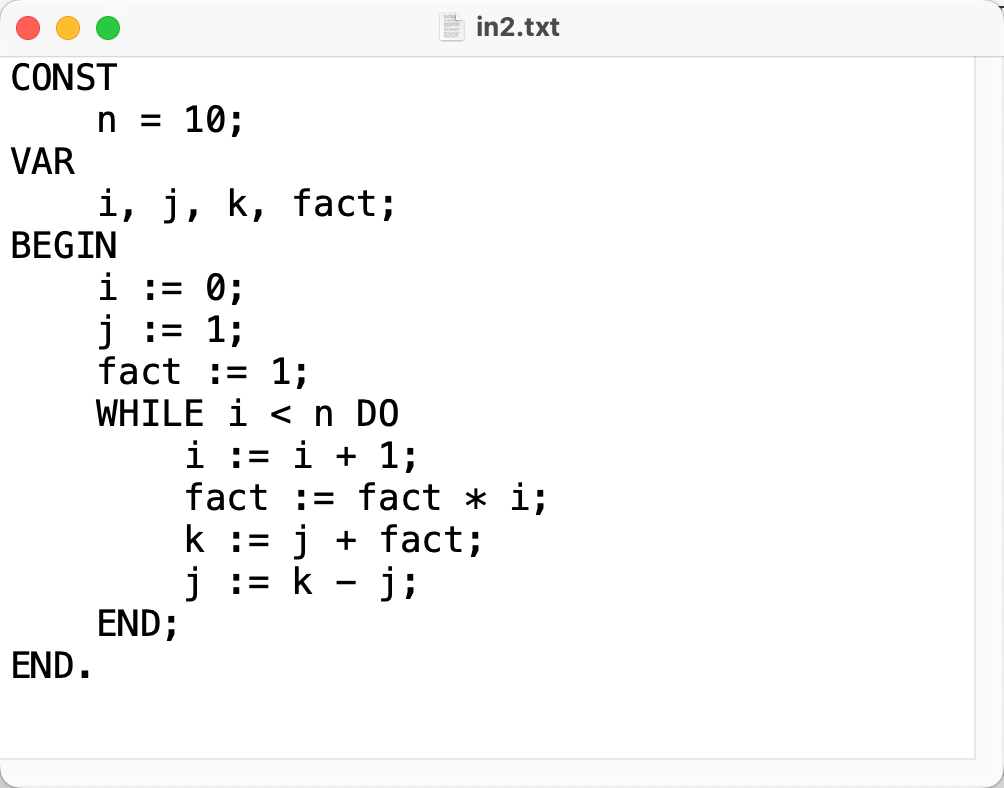
**测试用例2：**



**运行结果：**



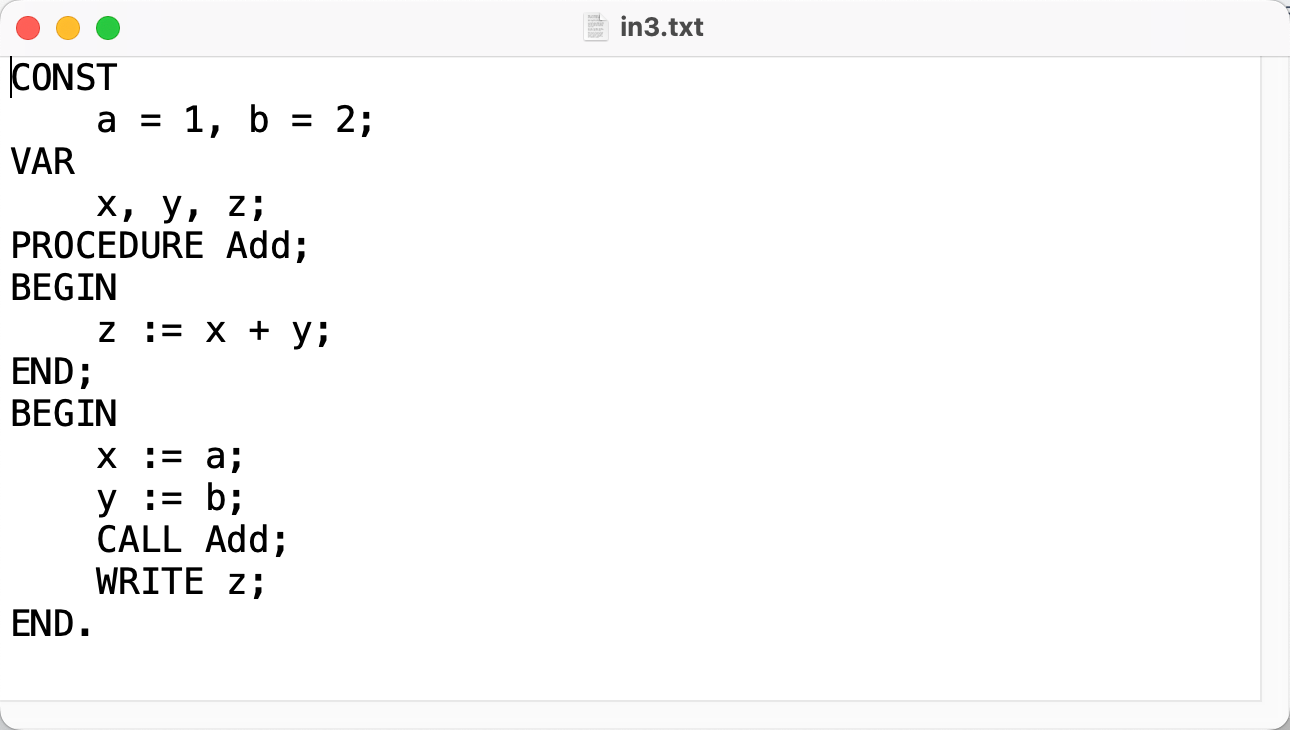
**测试用例3：**



**运行结果：**



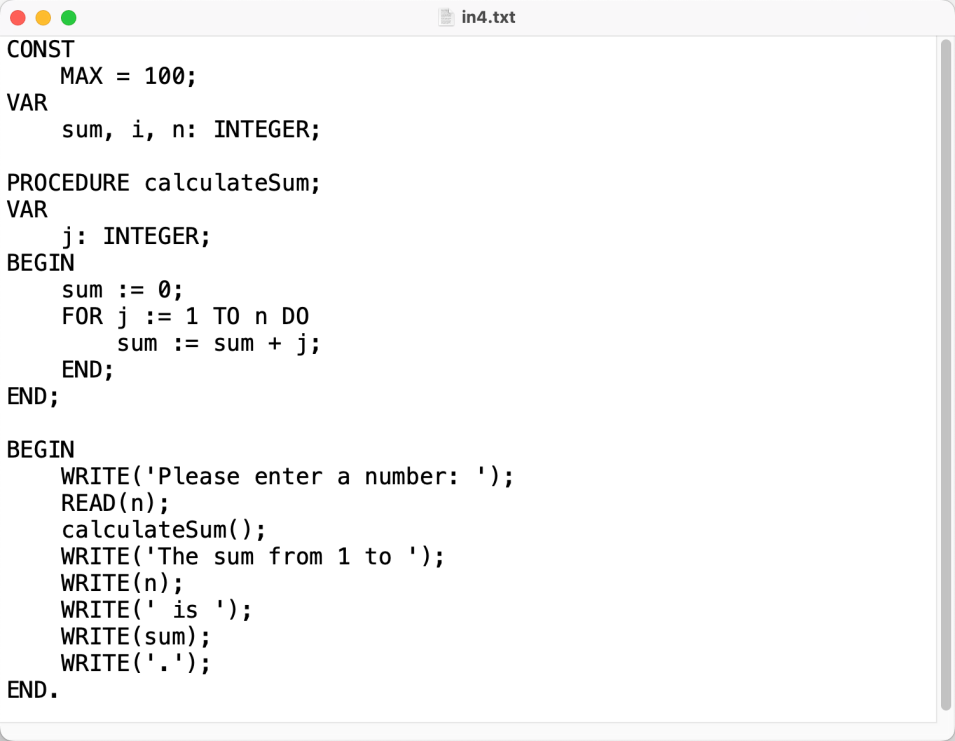
**测试用例4：**



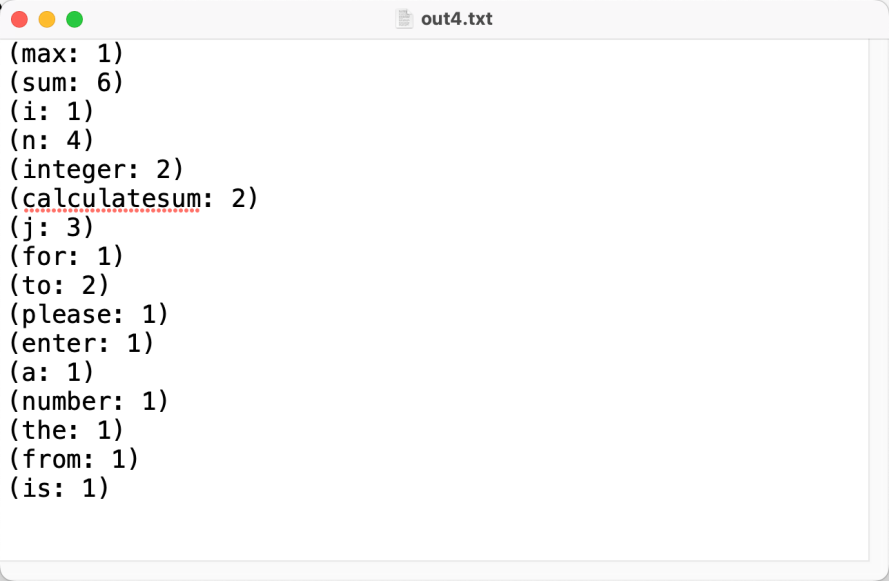
**运行结果：**



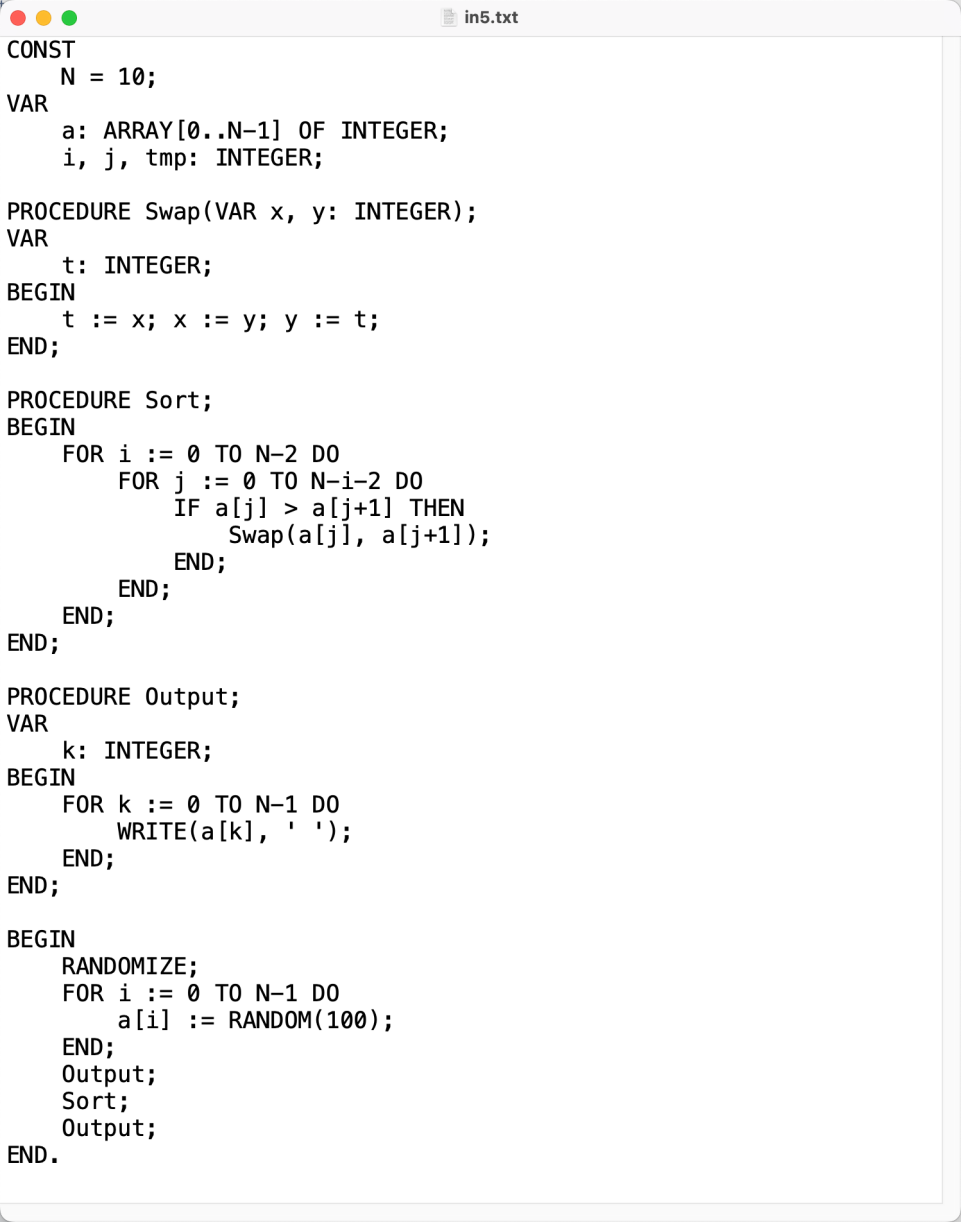
**测试用例5：**



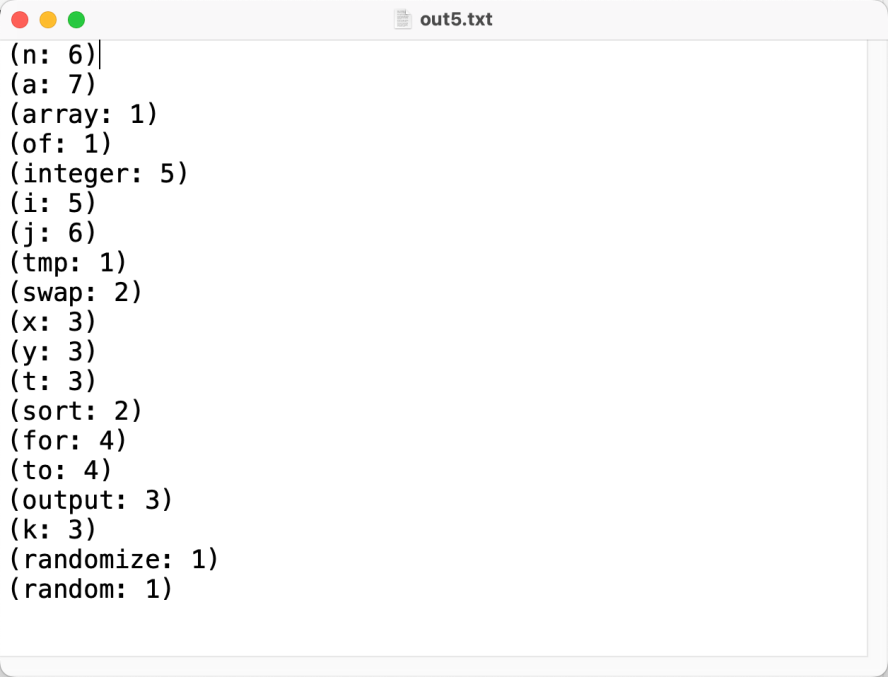
**运行结果：**



**测试用例6：**



**运行结果：**



**五、收获与体会**

每个同学完成该实验后的收获和体会。

徐驰：这次实验让我更加深入地了解了编译原理中的词法分析，之前只是了解了大概的原理和流程，但通过本次实验的实践，我更加直观地感受到了词法分析器的重要性和实现的难度。同时，也意识到了编写高效的词法分析器需要对语言的语法和结构有深入的理解和掌握。

陈世锦：本次实验让我更加重视代码的规范和可读性，词法分析器的实现需要涉及到大量的字符串处理和正则表达式匹配，如果代码不规范、不清晰，就会导致代码难以维护和调试。因此，我认为在编写代码的同时，也要注重代码的可读性和可维护性，这对于长期的软件开发来说非常重要。

李子浩：通过本次实验，我深刻地认识到了编译原理的重要性和应用价值。编译器是现代计算机系统中不可或缺的一部分，它们负责将高级语言翻译成机器语言，使得程序员可以更加高效地编写程序。同时，也意识到编译器的开发需要多个领域的知识和技能，包括计算机体系结构、算法和数据结构、语言学等多个方面。

邵启承：本次实验让我更加深入地了解了 PL/0 语言的语法和结构，学习了如何编写词法分析器并将其应用到实际的程序中。同时，也意识到编写高效的词法分析器需要对字符串处理和正则表达式等方面有深入的了解和掌握。我认为这些知识和技能对于我的未来发展和工作都有着重要的意义，我会继续学习和探索相关的领域。

**2023.3.14**