数据结构实验报告

评分

满分——5分

学号：2015111833 姓名：邓金红 专业：计算机科学与技术

知识范畴：字符串 完成日期：2017年4月14日

实验题目：基于改进KMP算法的字符文件子串查找

实验内容及要求：

从键盘输入字符文件名以及子串，用改进KMP算法在字符文件中实现子串查找。要求程序输出子串的改进nextval数组元素值以及子串在文件中成功匹配的次数（查找失败输出成功匹配次数为0）。

实验目的：掌握子串查找的KMP算法。

数据结构设计简要描述：

输入数据为文件中的字符串，子字符串从键盘输入，存入字符数组中。Nextval数组存入一位数组中。

算法设计简要描述：

朴素的字符串匹配算法，在每次匹配失败的时候，指向父字符串的指针必须回溯。为达到不用回溯的效果，经过研究发现，子字符串本身具有一定的特点，也就是可以生成next数组，next数组优化后为nextval数组。实验的要求需要输出在文件中成功匹配的次数，所以常规的nextval数组求法并不能实现，需求得子字符串的’\0’位置的元素的nextval数组值，由此，文件指针扫描文件时，不会因为一次成功匹配而指针回溯。

输入/输出设计简要描述：

输入：文件名 字符串类型

输出：next数组，int类型 数组的值用空格隔开 + 成功匹配的次数

输入输出有文字提示

编程语言说明：

使用Visual C++编程。 主要代码采用C语言实现 ；动态存储分配采用C++的new和delete操作符实现；输入与输出采用C++的cin和cout流；程序注释采用C/C++规范。

主要函数说明：

void get\_nextval(char \*t, int nextval[]) //得到nextval数组

int getLength(char \*a) //得到一个字符串的长度

int cnt\_kmp(FILE \*fp, char t[], int nextval[], int length)// 累计子字符串在文件中出现的次数，指针不需要回溯

int getLength(char \*a) //得到一个字符串的长度

程序测试简要报告：

说明：数据存在文件中，文件名为:input.txt，就要测试差别在于文件中的内容—父字符串

和子字符串的不同。

***测试样例一：***

文件中：abcdefg

输入: cd

输出：nextval数组(包括最后\0位置的值):

-1 0 1

子字符串在文件中成功匹配的次数：1

***结论***

程序输出结果与期望输出结果相符。

***测试样例二：***

文件中：aaaaaaaaa

输入: aa

输出：nextval数组(包括最后\0位置的值):

-1 -1 1

子字符串在文件中成功匹配的次数：8

***结论***

程序输出结果与期望输出结果相符。

***测试样例二：***

文件中：abaabcacabaabcac

输入: abaabcac

输出：nextval数组(包括最后\0位置的值):

-1 0 -1 1 0 2 -1 1 0

子字符串在文件中成功匹配的次数：2

***结论***

程序输出结果与期望输出结果相符。

源程序代码：

#include <iostream>

#include <stdio.h>

#include <stdlib.h>

using namespace std;

/\*

测试样例：

abaabcacabaabcac

abaabcac

2

aaaaaaaaa

aa

8

abcdefg

cd

\*/

/\*\*

得到next数组

\*\*/

void get\_next(char \*t, int next[])

{

int j, k;

next[0] = k = -1;

j = 0;

while(t[j])

{

if(k == -1 || t[j] == t[k])

next[++j] = ++k;

else

k = next[k];

}

}

/\*

得到nextval数组

\*/

void get\_nextval(char \*t, int nextval[])

{

int j, k;

nextval[0] = k = -1;

j = 0;

while(t[j])

{

if(k == -1 || t[j] == t[k])

{

if(t[++k] == t[++j])

nextval[j] = nextval[k];

else

nextval[j] = k;

}

else

k = nextval[k];

}

}

int getLength(char \*a) //得到一个字符串的长度

{

int length = 0;

while(a[length])

{

length++;

}

return length;

}

/\*\*

累计子字符串在文件中出现的次数，指针不需要回溯

\*\*/

int cnt\_kmp(FILE \*fp, char t[], int nextval[], int length)

{

int k0 = nextval[length];

int c = 0;

int ch = getc(fp);

int j = 0;

while(1)

{

while(ch != -1 && (j == -1 || t[j]))

{

if(j == -1 || ch == t[j])

{

ch = fgetc(fp);j++;

}else{

j = nextval[j];

}

}

if(!t[j])

{

j = k0; c++;

}

if(ch == -1)break;

}

return c;

}

int main()

{

char filename[100], t[100];

int nextval[100];

int length;

cout << "请输入文件名:";

cin >> filename;

FILE \*fp;

while(1)

{

fp = fopen(filename, "r");

if(!fp)

{ cout << "不存在此文件，请重新输入:";

cin >> filename;

}

else

break;

}

cout << "请输入子串:";

cin >> t;

length = getLength(t);

get\_nextval(t, nextval);

cout << "next数组(包括最后\\0位置的值):" << endl;

for(int i = 0; i <= length; i++)

cout << nextval[i] << " ";

cout << endl;

cout << "子字符串在文件中成功匹配的次数：" << cnt\_kmp(fp, t, nextval, length) << endl;

return 0;

}