输入例句：S1="计算语言学课程有意思" ；

定义：最大词长MaxLen = 5；S2= " "；分隔符 = “/”；

假设存在词表：…，计算语言学，课程，意思，…；

最大逆向匹配分词算法过程如下：

（1）S2=""；S1不为空，从S1右边取出候选子串W="课程有意思"；

（2）查词表，W不在词表中，将W最左边一个字去掉，得到W="程有意思"；

（3）查词表，W不在词表中，将W最左边一个字去掉，得到W="有意思"；

（4）查词表，W不在词表中，将W最左边一个字去掉，得到W="意思"

（5）查词表，“意思”在词表中，将W加入到S2中，S2=" 意思/"，并将W从S1中去掉，此时S1="计算语言学课程有"；

（6）S1不为空，于是从S1左边取出候选子串W="言学课程有"；

（7）查词表，W不在词表中，将W最左边一个字去掉，得到W="学课程有"；

（8）查词表，W不在词表中，将W最左边一个字去掉，得到W="课程有"；

（9）查词表，W不在词表中，将W最左边一个字去掉，得到W="程有"；

（10）查词表，W不在词表中，将W最左边一个字去掉，得到W="有"，这W是单字，将W加入到S2中，S2=“ /有 /意思”，并将W从S1中去掉，此时S1="计算语言学课程"；

（11）S1不为空，于是从S1左边取出候选子串W="语言学课程"；

（12）查词表，W不在词表中，将W最左边一个字去掉，得到W="言学课程"；

（13）查词表，W不在词表中，将W最左边一个字去掉，得到W="学课程"；

（14）查词表，W不在词表中，将W最左边一个字去掉，得到W="课程"；

（15）查词表，“意思”在词表中，将W加入到S2中，S2=“ 课程/ 有/ 意思/”，并将W从S1中去掉，此时S1="计算语言学"；

（16）S1不为空，于是从S1左边取出候选子串W="计算语言学"；

（17）查词表，“计算语言学”在词表中，将W加入到S2中，S2=“计算语言学/ 课程/ 有/ 意思/”，并将W从S1中去掉，此时S1=""；

（18）S1为空，输出S2作为分词结果，分词过程结束。

[IMG_256](http://www.cnblogs.com/csudanli/p/javascript:void(0);" \o "复制代码)

// 分词.cpp : 定义控制台应用程序的入口点。//

#include "stdafx.h"

#include<iostream>

#include<fstream>

#include<set>

#include<string>

#include<sstream>

#include<algorithm>using namespace std;

int main()

{

set<string> dicSet; //建立字典的集合，用来搜索，看某个词是否是在字典集中

ifstream dic("D:\\file\\word\_freq\_list.txt"); //打开字典

ifstream test("D:\\file\\pku\_test.txt"); //打开要分词的文本

ofstream out("D:\\file\\word\_output.txt"); //这个用来储存分词后的文本

string dicLine,testLine; //dicLine用来读取字典的一行，testLine用来读取要分词的文本的一行

if (!dic||!test||!out) //测试文件是否打开 {

cerr << "Open file failed!";

return 0;

}

while (getline(dic, dicLine)) //读取字典的一行 {

string word = "";

istringstream iss(dicLine);//将字典的一行放到istringstream对象中

for (int i = 0;i <= 1;++i) //将这一行中间的词或字放到dicSet中去，因为我们只需要这个词或字是否在字典中 {

iss >> word;

if (1 == i)

dicSet.insert(word);

}

}

while (getline(test, testLine))//读取测试文本的一行 {

size\_t len = 0; //记录产生的分词的总长度

size\_t pos = 0; //记录分词的长度

int i = testLine.size() - 10; //用来获取字字符串的偏移量

while(true)

{

string word;

size\_t j = 0;

if (i < 0) //如果i小于0，则读取的字符串的长度是小于10的 {

word = testLine.substr(0, 10 +i);

}

else

word = testLine.substr(i, 10); //读取这一行从i开始的长度为10的子字字符串

for (;j < word.size();j+=2)

{

string character = word.substr(j); //求得这个子字符串的字符串

if (character.size()==2||(find(dicSet.begin(), dicSet.end(), character) != dicSet.end()))

{ //如果word是字典中的词，或者word只有一个字，则应该把word作为一个分词

out << character << "/";

pos = character.size();

len += pos; //记录这一行的分词的总长度

i = i - pos; //让i的位置变小，向前缩进

break; //跳出这个循环，寻找这一行的下一个分词 }

}

if (len == testLine.size()) //如果这一行的分词长度等于这行字符串的长度，则这一行分词结束

{ //跳出这个循环，进行下一行的分词

out << endl;

break;

}

}

}

dic.close();

test.close();

out.close();

return 0;

}