天津大学

数据结构上机实验报告

题目:根据关键字进行字符串拷贝

学生姓名 王雨朦

学生学号 2016229082

学院名称 国际工程师学院

专业 计算机

时间 2016/10/20

目　　录

第一章　需求分析

1.1 原题表述

将源字符串拷贝到目的字符串：

如果指定关键字，则以该关键字结束(不包括关键字本身)
如果拷贝失败，则得到空串。

具体要求：实现如下函数原型 SafeStrcpy2KeyWord()，并在代码中调用该函数实现上述功能。该函数的实现要考虑各种可能的参数取值，以确保程序不出现崩溃。

```
int SafeStrcpy2KeyWord(char* pDestBuffer,    //拷贝的目的地地址

                char* pSourceString, //拷贝的源地址

                int nDestBufferSize, //拷贝的目的地缓冲区长度

                char* szKeyWord);    //指定关键字符串
```

返回值：所拷贝的字符串长度。如果拷贝失败，则返回 0。

1.2 解决方案

1) 将字符串用 char 型数组存储；
2) 先输入字符串，再输入 key 字符串；
3) 将 key 和字符串进行匹配，返回需要复制的字符串的长度；
4) 输出复制的字符串。

第二章　概要设计

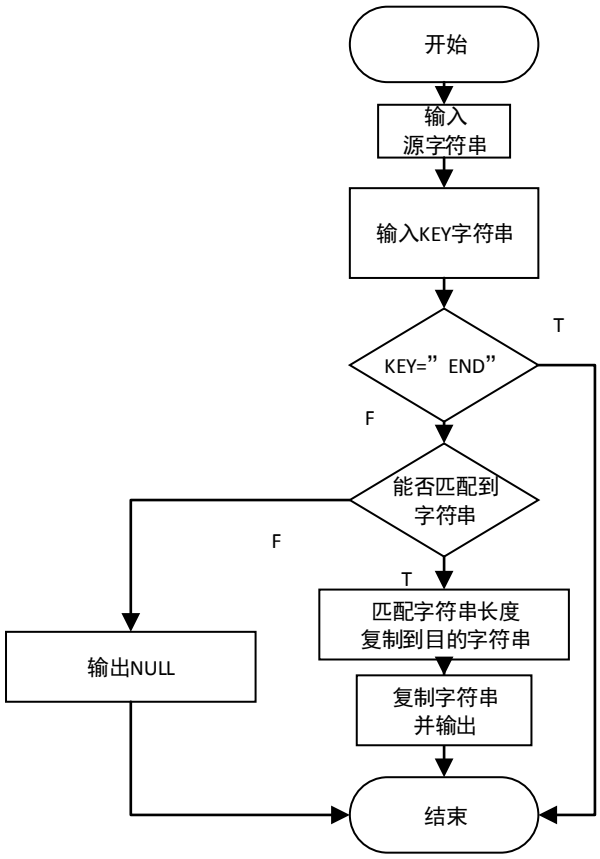
2.1 抽象数据类型

字符串:

```
#define MAXSTRLEN 256
#define MAXKEYLEN 16
char *str=new char [MAXSTRLEN];
char *cps= new char[MAXSTRLEN];
char *key= new char[MAXKEYLEN];

int strlen( char *s)
int SafeStrcpy2KeyWord(char* pDestBuffer,   //拷贝的目的地地址
                char* pSourceString, //拷贝的源地址
                int &nDestBufferSize,    //拷贝的目的地缓冲区长度
                char* szKeyWord){    //指定关键字符串
void display(char* cps,int len)
```

2.2 算法



第三章　详细设计

3.1 程序代码

```
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

using namespace std;

#define MAXSTRLEN 256
#define MAXKEYLEN 16

int strlen( char *s){
    int len=0;
    while(1){
        if(s[len]!='\0') len++;
        else return len;
    }
}

int SafeStrcpy2KeyWord(char* pDestBuffer,   //拷贝的目的地地址
               char* pSourceString, //拷贝的源地址
               int &nDestBufferSize,    //拷贝的目的地缓冲区长度
               char* szKeyWord){    //指定关键字符串
    int i=0, j=0;
    int len1=strlen(pSourceString),
        len2=strlen(szKeyWord);
    while (i<len1 && j<len2){
        if(pSourceString[i]==szKeyWord[j]) {++i; ++j;}
        else {i=i-j+1; j=0;}//i 从初始位置后移，j 复位为 0
    }
    if(j>=len2){
        nDestBufferSize=i-len2;
        //copy 子串
        for(int k=0; k<nDestBufferSize;k++){
            pDestBuffer[k]=pSourceString[k];
        }
        return nDestBufferSize;//计算子串长度
    }
    else return 0;
```

```
}

void display(char* cps,int len){
    for(int i=0;i<len;i++){
        cout<<cps[i];
    }
    cout<<endl;
}

int main(){
    char *str=new char[MAXSTRLEN];
    char *cps= new char[MAXSTRLEN];
    char *key= new char[MAXKEYLEN];

    printf("Please input a string:\n");
    scanf("%s",str);
   // printf("%c",str[2]);

    printf("Please input key words:\n");
    while(cin>>key && !(key[0]=='E' &&key[1]=='N' &&key[2]=='D' )){
         int len=0;
        //len=Index(str,key);
        //printf("len  %d",len);
        SafeStrcpy2KeyWord(cps,str,len,key);
        cout<<len<<" ";
        if(len!=0) display(cps,len);
        else cout<<"NULL"<<endl;

    }

}
```

第四章　调试分析

4.1 调试过程

Bug 名称	无法返回匹配的长度
Bug 描述	要求的函数返回的匹配长度始终为 0
Bug 原因	函数参数传值的时候没有用引用传递，导致参数值没变
Bug 解决方案	在函数参数类型后面加‘&’
Bug 总结	引用传递运用不够熟练

Bug 名称	返回长度时返回的是一段地址
Bug 描述	应该返回的长度却返回了长度的地址
Bug 原因	至今没找到
Bug 解决方案	重新梳理逻辑顺序，在得到匹配长度后立即进行匹配
Bug 总结	基础知识薄弱

Bug 名称	循环中初始值不唯一
Bug 描述	当匹配值越来越大时，再测试匹配值小的字符串，也会返回最大匹配值
Bug 原因	循环中初始值不够明确
Bug 解决方案	将初始值写在循环内部
Bug 总结	进行代码编写时，思路不够清晰

第五章　测试结果

5.1 测试过程

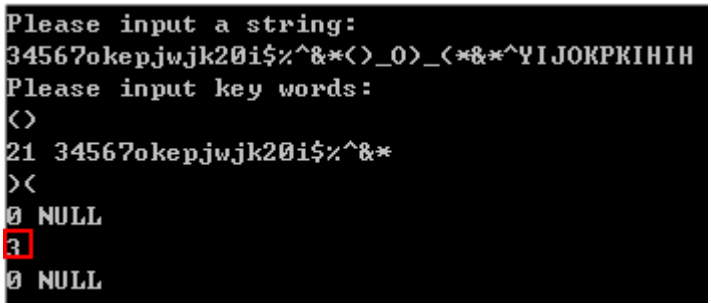
测试编号	1
测试对象	正常的匹配功能
测试输入参数	源字符串： 34567okepjwjk20i$%^&*()_O)_(*&*^YIJOKPKIHIH Key 串：（）
测试步骤	先输入一串字符串，再输入可以匹配的 KEY 字符串， 查看是否能正确匹配。
测试预期结果	21 34567okepjwjk20i$%^&*
测试输出结果	21 34567okepjwjk20i$%^&*
测试分析	程序正确

```
Please input a string:
34567okepjwjk20i$%^&*()_O)_(*&*^YIJOKPKIHIH
Please input key words:
()
21 34567okepjwjk20i$%^&*
```

测试编号	2
测试对象	不匹配串的输出
测试输入参数	源字符串： 34567okepjwjk20i$%^&*()_O)_(*&*^YIJOKPKIHIH Key 串：）（
测试步骤	先输入一串字符串，再输入不可以匹配的 KEY 字符串， 查看是否能正确匹配。
测试预期结果	0 NULL
测试输出结果	0 NULL
测试分析	程序正确

```
Please input a string:
34567okepjwjk20i$%^&*()_O)_(*&*^YIJOKPKIHIH
Please input key words:
()
21 34567okepjwjk20i$%^&*
)(
0 NULL
```

测试编号	3
测试对象	可以匹配的字符串，但长度为 0
测试输入参数	源字符串： 34567okepjwjk20i$%^&*()_O)_(*&*^YIJOKPKIHIH Key 串：3
测试步骤	先输入一串字符串，再输入可以匹配但是长度为 0 的 KEY 字符串，查看输出结果
测试预期结果	0 NULL
测试输出结果	0 NULL
测试分析	程序正确



测试编号	4
测试对象	END 是否退出
测试输入参数	源字符串： 34567okepjwjk20i$%^&*()_O)_(*&*^YIJOKPKIHIH Key 串：END
测试步骤	先输入一串字符串，再输入 KEY 字符串“END”
测试预期结果	程序退出
测试输出结果	程序退出
测试分析	程序正确