1. **问题描述**
2. **题目描述**

在Internet的域名系统中，以树型结构实现域名的搜索。即输入某站点的域名，在域名系统的树型结构中进行搜索，直至域名全部匹配成功或匹配失败；若成功则给出该站点的IP地址166.111.9.2

1. **基本要求**

首先实现一个反映域名结构的树，例如清华大学站点www.tsinghua.edu.cn在该树从根到叶子的各层结点就应该是root、cn、edu、tsinghua、www。叶子结点www另外有一个数据域、存放清华大学站点的IP地址166.111.9.2。

1. **测试数据**

可以取常用到的著名站点的域名和IP 地址为例构建域名结构的树，一般应有３０个左右的站点域名。当输入www.tsinghua.edu.cn时，输出为“166.111.9.2”；而输入www.tsinghuo.edu.cn时，输出应为“找不到服务器或发生DNS错误”

1. **需求分析**

1.本程序用来模拟因特网域名的查询，需要用文件存储域名及IP地址数据作为数据库以供查询。

2.本程序还实现因特网域名数据添加，扩充数据库。

1. **单元设计**
2. 抽象数据类型：

**typedef struct CSNode**

**{**

**char data[50]; //存储域名**

**char ip[20]; //存储IP**

**struct CSNode \*firstchild, \*nextsibling;**

**}CSNode, \*CSTree;**

**typedef struct Pointer**

**{**

**CSNode \* point[30]; //辅助指针数组**

**int last; //记录数组最后一个元素**

**}Pointer;**

1. 包含模块
   1. 实现因特网域名存储模块：树的孩子-兄弟二叉链表
2. **源代码**

#include <iostream>

#include <stdlib.h>

#include <string.h>

#include <stdio.h>

#include <windows.h>

#include <conio.h>

using namespace std;

typedef struct CSNode

{

char data[50];

char ip[20];

struct CSNode \*firstchild, \*nextsibling;

}CSNode, \*CSTree;

typedef struct Pointer

{

CSNode \* point[30];

int last;

}Pointer;

CSTree T;

Pointer P;

void WriteToFile(CSTree T, FILE \*fp);

void ReadFromFile(CSTree &T, FILE \*fp);

void CreateTree(CSTree &T);

void SearchTree(CSTree T);

void CreateTree(CSTree &T)

{

char data[50] = {'\0'};

char ip[20];

while(1)

{

memset(P.point,'\0',sizeof(P));

system("cls");

cout<<"\n\t\*\*\*\*\*\*| tip: enter 'N' to stop ! |\*\*\*\*\*\*"<<endl;

cout<<"\n\t\* put in the domain name: ";

cin>>data;

if( !strcmp(data,"N") )

{

break;

}

cout<<"\n\t\* put in IP :";

cin>>ip;

int i = 0, j = 0, k = 0;

char transient[20] = {'\0'};

while(data[i] != '\0')

{

transient[j++] = data[i++];

if(data[i] == '.' || data[i] == '\0')

{

CSNode \*p = new CSNode;

strcpy(p->data, transient);

strcpy(p->ip,"NULL");

p->firstchild = p->nextsibling = NULL;

P.point[k] = p;

P.last = k;

k++;

for(int m = 0; m<20; m++)

transient[m] = '\0';

j = 0;

i++;

}

}

CSNode \*t = T, \*q;

while( P.last >= 0 )

{

CSNode \*p = P.point[P.last];

if( (t->firstchild) == NULL)

{

t->firstchild = p;

t = p;

}

else

{

t = t->firstchild;

while(t != NULL && strcmp(t->data , p->data) != 0)

{

q = t;

t = t->nextsibling;

}

if( t==NULL )

{

q->nextsibling = p;

t = p;

}

}

P.last--;

}

strcpy(t->ip, ip);

system("cls");

cout<<"\n\n\tLoading ¡¤ ¡¤ ¡¤ ¡¤ ¡¤ ¡¤"<<endl;

Sleep(1000);

}

FILE \*fp = fopen("data.txt", "w");

WriteToFile(T->firstchild, fp);

fclose(fp);

}

void SearchTree(CSTree T)

{

char data[50] = {'\0'};

FILE \*fp = fopen("data.txt","r");

ReadFromFile(T->firstchild,fp);

fclose(fp);

while(1)

{

memset(P.point,'\0',sizeof(P));

system("cls");

cout<<"\n\t\*\*\*\*\*\*| tip: enter 'N' to stop ! |\*\*\*\*\*\*"<<endl;

cout<<"\n\t\* put in the domain name: ";

cin>>data;

if( !strcmp(data,"N") )

{

break;

}

int i = 0, j = 0, k = 0;

char transient[20] = {'\0'};

while(data[i] != '\0')

{

transient[j++] = data[i++];

if(data[i] == '.' || data[i] == '\0')

{

CSNode \*p = new CSNode;

strcpy(p->data, transient);

strcpy(p->ip,"NULL");

p->firstchild = p->nextsibling = NULL;

P.point[k] = p;

P.last = k;

k++;

for(int m = 0; m<20; m++)

transient[m] = '\0';

j = 0;

i++;

}

}

CSNode \*t = T, \*q;

while( P.last >= 0 )

{

CSNode \*p = P.point[P.last];

t = t->firstchild;

if( !t && strcmp(t->data, p->data) != 0)

t = t->nextsibling;

if(t == NULL)

{

cout<<"\n\tDNS wrong !"<<endl;

break;

}

P.last--;

}

if(t != NULL)

cout<<"\n\tThe IP is: "<<t->ip<<endl;

cout<<"\n\n\tPress any key to continue... "<<endl;

getch();

}

}

void WriteToFile( CSTree T, FILE \*fp)

{

char Ltag = '0', Rtag = '0';

if(T)

{

fputs(T->data,fp);

fputs("\t",fp);

fputs(T->ip,fp);

fputs("\t",fp);

if(T->firstchild != NULL)

{

Ltag = '1';

}

fwrite(&Ltag, sizeof(char), 1, fp);

fputs("\t",fp);

if(T->nextsibling != NULL)

{

Rtag = '1';

}

fwrite(&Rtag, sizeof(char), 1, fp);

fputs("\n",fp);

WriteToFile(T->firstchild, fp);

WriteToFile(T->nextsibling, fp);

}

}

void ReadFromFile( CSTree &T, FILE \*fp)

{

CSTree p;

char LG='0',RG='0';

p = new CSNode;

p->firstchild = p->nextsibling = NULL;

fscanf(fp,"%s\t%s\t%c\t%c",p->data,p->ip,&LG,&RG);

T = p;

if(LG=='1')

ReadFromFile(T->firstchild,fp);

if(RG=='1')

ReadFromFile(T->nextsibling,fp);

}

void Welecome()

{

cout<<"\n\n\n"

<<"\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n"

<<"\t\t\* \*\n"

<<"\t\t\* \*\n"

<<"\t\t\* \*\n"

<<"\t\t\* Welecome to use \*\n"

<<"\t\t\* \*\n"

<<"\t\t\* the domain name inquiry system \*\n"

<<"\t\t\* \*\n"

<<"\t\t\* (^ \_ ^) \*\n"

<<"\t\t\* \*\n"

<<"\t\t\* \*\n"

<<"\t\t\* \*\n"

<<"\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n"

<<endl;

Sleep(2000);

}

void Menu()

{

system("cls");

cout<<"\n\n\n"

<<"\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n"

<<"\t\t\* \*\n"

<<"\t\t\* \* Internet Searching Programing \* \*\n"

<<"\t\t\* \*\n"

<<"\t\t\* ^ \_ ^ \*\n"

<<"\t\t\* \*\n"

<<"\t\t\* \*\n"

<<"\t\t\* [1] Original domain name information input \*\n"

<<"\t\t\* \*\n"

<<"\t\t\* [2] Internet domain name searching \*\n"

<<"\t\t\* \*\n"

<<"\t\t\* [3] exit \*\n"

<<"\t\t\* \*\n"

<<"\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n"

<<endl;

}

int main()

{

int n;

T = new CSNode;

T->firstchild = T->nextsibling = NULL;

Welecome();

while(1)

{

Menu();

cin>>n;

system("cls");

switch(n)

{

case 1:

{

CreateTree(T);

break;

}

case 2:

{

SearchTree(T);

break;

}

case 3:

{

exit(0);

}

}

}

return 0;

}

1. **使用说明**
2. 选择查询，输入域名查询IP地址。
3. 选择添加，输入域名及IP地址。
4. **改进分析**
5. 可以添加域名修改IP地址修改功能。
6. 可以添加该网址的信息介绍功能。