习题19

1.1.生成组合

1.1.1.Johnson-Trotter算法

1.1.1.1.算法描述

每个元素一个标记位置，从右往左进行移动，直到每个都移动过为止

1.1.1.2.算法伪代码

Johnson\_Trotter

//Johnson\_Trotter算法

//输入：字符串str

//输出：所有的排列组合

while(执行move\_str.length()次){

//move\_str=最后一个字符

//从右往左移动move\_str并输出

}

1.1.1.3.算法实现

void John\_Trotter(char\* str){

int len=strlen(str);

char tmp;

while(len--){

for(int i=strlen(str)-1;i>0;i--){

tmp=str[i];

str[i]=str[i-1];

str[i-1]=tmp;

printf("%s\n",str);

}

}

}

1.1.1.4.测试用例

int main()

{

char str[100];

scanf("%s",str);

cout<<endl;

John\_Trotter(str);

system("pause");

return 0;

}

1.2.生成子集

1.2.1.算法描述

二进制反射格雷码（binary reflected Gray code）是19世纪70年代，埃米尔鲍勃特（Emile Baudot）把这种编码用于电报中。我们可以通过二进制反射格雷码进行生成子集。

1.2.2.算法伪代码

ERGC(n)

//递归生成n位的二进制反射格雷码

//输入：一个正整数n

//输出：所有长度为n的格雷码位串列表

if n=1 表L包含位串0和1

else 调用BRGC(n-1)生成长度为n-1的位串列表L1

把表L1倒序后复制给表L2

把0加到L1的每位串前面

把1加到L1的每位串前面

把表L2添加到表L1后面得到表L

1.2.3.算法实现（C++）

//递归

vector<string> BRGC(int n){

if(n==1) {

vector<string> vec;

vec.push\_back("0");

vec.push\_back("1");

return vec;/\*\*如果n为1，返回0,1集合\*/

}else{

vector<string> L=BRGC(n-1);/\*\*递归\*/

vector<string> L1=L;/\*\*拷贝一份1\*/

reverse(L1.begin(),L1.end());/\*\*将L1逆序\*/

for(int i=0;i<L.size();i++)

L[i]="0"+L[i];/\*\*L的前面加0\*/

for(int i=0;i<L1.size();i++)

L1[i]="1"+L1[i];/\*\*L1的前面加1\*/

L.insert(L.end(),L1.begin(),L1.end());/\*\*L1加入L尾部\*/

return L;

}

}

//非递归

vector<string> BRGC\_not\_recursion(int n){

vector<string> L;

bool start=true;

while(n--){

if(start) {

L.push\_back("0");

L.push\_back("1");

start=false;

}else{

vector<string> L2=L;

reverse(L2.begin(),L2.end());

for(int i=0;i<L.size();i++) L[i]="0"+L[i];

for(int i=0;i<L2.size();i++) L2[i]="1"+L2[i];

L.insert(L.end(),L2.begin(),L2.end());

}

}

return L;

}

1.2.4.测试用例

int main()

{

int n;

cin>>n;

vector<string> vec=BRGC(n);

for(int i=0;i<vec.size();i++) cout<<vec[i]<<endl;

system("pause");

return 0;

}