# 递归

# 回溯

也叫试探法。是一种选优搜索法，按照选优条件搜索，当搜索到某一步，发现原先选择并不优或达不到目标，就退回重新选择。

## 一般步骤

1、针对问题，定义解空间（ 这时候解空间是一个集合，且包含我们要找的最优解）

2、组织解空间，确定易于搜索的解空间结构，通常组织成树结构 或 图结构

3、深度优先搜索解空间，搜索过程中用剪枝函数避免无效搜索

回溯法求解问题时，一般是一边建树，一边遍历该树；且采用非递归方法。

## 八皇后问题

8x8的国际象棋棋盘上放置8个皇后，使得任何一个皇后都无法直接吃掉其他的皇后。任意2个皇后都不能处于同一个 横线，纵线，斜线上。

分析

任意2个皇后不能同一行，也就是每个皇后占据一行，通用的，每个皇后也要占据一列

一个斜线上也只有一个皇后

## 其它案例

### 迷宫问题

### 求一个序列中和为特定值的组合

题目要求：求一个序列元素之和等于固定值的特定组合。

代码：

#include <iostream>

#include <vector>

#include <string>

using namespace std;

/\*

求一个序列中元素之和等于固定值的特定组合

\*/

void Combination\_helper(vector<int>& vec,int begin,int target,int& cur,vector<int>& path)

{

if(begin >=vec.size())

return ;

cur += vec[begin];

path.push\_back(vec[begin]);

if(cur == target)

{

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

cout<<path[i]<<endl;

cout<<"=========="<<endl;

}

Combination\_helper(vec,begin+1,target,cur,path);

path.pop\_back();

cur -= vec[begin];

int j;

for(j=begin+1;j<vec.size();)

{

if(vec[j] == vec[begin])

j++;

else

break;

}

Combination\_helper(vec,j,target,cur,path);

}

void Combination(vector<int>& vec,int target)

{

vector<int> path;

int cur = 0;

if(vec.size() == 0)

return ;

Combination\_helper(vec,0,target,cur,path);

}

int main()

{

vector<int> path;

int array[]={1,1,2,5,6,7,10};

int cur =0;

vector<int> vec(array,array+sizeof(array)/sizeof(int));

Combination(vec,8);

return 0;

}

### 求括号正确的组合方式

题目要求：

代码：

#include <iostream>

#include <string>

#include <vector>

using namespace std;

/\*

对于几对括号，有多少种正确的组合方式

思路：使用递归的方法，只不过在递归的时候

时刻需要保证左边的括号比右边的括号多

只有在左括号比右括号多的情况下才有可能保证整个序列为合法的括号匹配

\*/

void helper(vector<char>& str,int l,int r)

{

if(l == 0 && r == 0)

{

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

{

cout<<str[i];

}

cout<<endl;

}

if(l>0)

{

str.push\_back('(');

helper(str,l-1,r);

str.pop\_back();

}

if(r>0 && l<r)

{

str.push\_back(')');

helper(str,l,r-1);

str.pop\_back();

}

}

void GenerateParenthess(int n)

{

if(n<=0)

return ;

vector<char> tmp;

helper(tmp,n,n);

}

int main()

{

GenerateParenthess(3);

return 0;

}

### 求九宫格中字符和数字的组合

题目要求：在九宫格手机键盘上，每个数字都对应着几个字符给出一个数字的字符串，找出所有对应的字符串的组合，而且每个数字只能有一个字符进行对应。

代码：

#include <iostream>

#include <string>

#include <vector>

using namespace std;

/\*

题目：在九宫格手机键盘上，每个数字都对应着几个字符

给出一个数字的字符串，找出所有对应的字符串的组合，每个

数字找一个字符进行对应

\*/

void helper(string& str,int begin,vector<string>& hash,vector<char>& vec)

{

int i;

if(begin > str.length())

return;

if(begin == str.length())

{

for(i=0;i<vec.size();i++)

cout<<vec[i];

cout<<endl;

return;

}

for(i=0;i<hash[str[begin]-'0'].length();i++)

{

vec.push\_back(hash[str[begin]-'0'][i]);

helper(str,begin+1,hash,vec);

vec.pop\_back();

}

}

void Combination(string& str,vector<string>& hash)

{

if(str.length()==0)

return ;

vector<char> vec;

helper(str,0,hash,vec);

}

void LetterCom(string& str)

{

int i;

vector<string> hash(10); //这里假设有10个数字

hash[0]=" ";

hash[1]="-";

hash[2]="abc";

hash[3]="def";

hash[4]="ghi";

hash[5]="jkl";

hash[6]="mno";

hash[7]="pqrs";

hash[8]="tuv";

hash[9]="wxyz";

Combination(str,hash);

return ;

}

int main()

{

string str("23");

LetterCom(str);

return 0;

}