**ТЪРСЕНЕ С ВРЪЩАНЕ НАЗАД**

Съществуват задачи, чието решение е свързано с търсене на един или всички възможни варианти за постигане на поставената цел (последния случай е известен като търсене с пълно изчерпване на множеството от решения). За решаване на такива задачи често се използва метод “търсене с връщане назад” или “backtracking”. При този методът обикновено се тръгва от едно частно решение, като на всяка следваща стъпка се прави опит текущото частно решение да се разшири. Ако на една от следващите стъпки се окаже невъзможно да се продължи, преди да се стигне до окончателното решение, извършва се връщане назад (с една или няколко стъпки) и се прави опит да се продължи по друг възможен начин. Прилаганият подход на “пробите и грешките” може да се разглежда като процес на търсене, в който се изгражда дърво от частичните решения, при това някои клонове на дървото се отсичат като ненужни.

Търсенето с връщане назад се реализира чрез използване на механизма на рекурсията. Предвижването напред се осъществява чрез вложени извиквания на рекурсивна процедура или функция. Връщането назад се получава при завършване на работата на тази процедура или функция за всяко рекурсивно обръщение.

По този метод се решават задачи в областта на изкуствения интелект: шахматни задачи, движение в лабиринт и т.н.

**Задача 2. Задача за 8-те царици**

Задачата за осемте царици е добре известен пример за използването на метода на пробите и грешките при алгоритмите с обратно търсене (backtracking). Тя е предизвикала интереса дори и на К. Ф. Гаус, който не е успял да даде пълно решение. Това е така, защото за тези задачи не съществува аналитично решение и само наличието на съвременна изчислителна техника може да даде пълното решение в допустим интервал от време.

Условието на задачата за осемте царици е: Върху шахматна дъска с размери 8x8 да се разположат 8 царици по такъв начин, че една царица да не застрашава друга. Царицата е фигура, която застрашава всички фигури, разположени по вертикал, хоризонтал и двата диагонала.

Естественият подход за намиране на решение на задачата е рекурсивен: ако сме разположили удачно **i** царици, проблемът се свежда до намиране на задоволително положение на **i+1**-вата царица. Задачата е решена, когато **i=N.** Остава да изведем намереното решение.  
  
 Известно е, че царицата заплашва всяка фигура в реда, колоната и диагонала, в които се намира, т.е. всеки ред и всяка колона може да съдържа само една царица. Цариците можем да представим с номерата на съответните колони(**i**). Тогава изборът на позиция за **i**-та царица се ограничава в **i**-та колона и се свежда до определяне на номера **j** на реда, в който тя може да се разположи в тази колона:

i

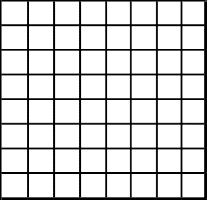
j



Полуформалното описание на алгоритъма за разполагане на **i**-та царица е:  
  


В основата на алгоритъма стои идеята за проверка на бито поле. За целта се използват три помощни масива **a, b**, и **c**. Предполага се, че е известно на кой вертикал се намира поредната царица (например в масив се записва само номера на хоризонтала, а вертикалите (индексите на масива) се запълват последователно от 1 до 8) и затова, той въобще не се проверява. Масивът **а** с размерност 8 ще има стойност ***true*** на съответния ***i***-ти елемент, ако на ***i***-тия хоризонтал има вече царица. Масивът ***b*** ще показва наличието или отсъствието на царица по вторичния диагонал (в червено) или успоредните на него, за които е характерно, че сумите от двата индекса (реда и стълба) са равни (напр. 1+8 = 2+7 = 3+6 и т.н.). Масивът **с** ще съдържа информация за наличието или отсъствието на царици по главния диагонал (в синьо) или успоредните на него, за които е характерно, че разликата между индексите е константа (напр. 1-1=0, 2-2=0 и т.н.). Полето ***i,j*** е свободно, ако едновременно е изпълнено: ***a[j]=true*,** ***b[i+j]=true*** и ***c[i-j]=true***.

i



j

Така можем да дефинираме три булеви масива, които да съхраняват нужната ни информация:

***a[j]=1***, ако в j-ти ред няма царица, и 0, ако има;  
 ***b[i+j]=1***, ако в този диагонал няма царица, и 0, ако има;  
 ***c[i-j]= 1***, ако в този диагонал няма царица, и 0, ако има;

Важно обобщение на приложената програма е, че се намира не само едно, а всички решения на задачата. Резултатът, който се получава в случая, извежда осем числа, които показват номера на реда, в който е разположена съответната царица в поредния вертикал.

//8 Queens Problem

#include <iostream>

using namespace std;

void print();

void true\_false(int i, int j, int k);

void place(int i);

const int N=8;

int a[N]={1,1,1,1,1,1,1,1};

int b[2\*N-1]={1,1,1,1,1,1,1,1,1,1,1,1,1,1,1};

int c[2\*N-1]={1,1,1,1,1,1,1,1,1,1,1,1,1,1,1};

int pos[N]={0,0,0,0,0,0,0,0};

void main()

{

place(1);

system("pause");

}

void print()

{

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

cout<< " " <<pos[m];

cout<<'\n';

}

void true\_false(int i, int j, int k)

{

a[j-1]=k;

b[i+j-2]=k;

c[i-j+7]=k;

}

void place(int i)

{

int j;

for (j=1; j<=N;j++)

{

if ((a[j-1]==1) && (b[i+j-2]==1) && (c[i-j+7]==1))

{

pos[i-1]=j;

true\_false(i,j,0);

if (i<N)

place(i+1);

else

print();

true\_false(i,j,1);

}

}

}