《数据结构》第五次作业

回溯法与树的遍历

# 折半查找：

随机生成100个单调增的整数，编程实现折半查找程序，并输出每次查找区间、比较元素的位置和比较结果。若未找到，则输出“查找失败”，否则输出元素所在位置。

折半查找函数代码：

int bisearch(int a[], int t, int left, int right, int \*p){

int m;

if(left < 0 || right > N - 1 || left > right) return -1;

(\*p)++;

m = (left + right)/2;

if(t == a[m]){

return m;

}

else if(t < a[m]) return bisearch(a, t, left, m - 1, p);

else return bisearch(a, t, m + 1, right, p);

return -1;

}

测试程序代码

#pragma GCC optimize(1)

#pragma GCC optimize(2)

#pragma GCC optimize(3."Ofast","inline")

#include<bits/stdc++.h>

#include<regex>

using namespace std;

#define N 100

int bisearch(int a[], int t, int left, int right, int \*p){

int m;

if(left < 0 || right > N - 1 || left > right) return -1;

(\*p)++;

m = (left + right)/2;

if(t == a[m]){

return m;

}

else if(t < a[m]) return bisearch(a, t, left, m - 1, p);

else return bisearch(a, t, m + 1, right, p);

return -1;

}

int main(){

int n[N],temp;

int i,j,k;

srand(time(0));

for(i=0;i<N;i++){

temp = rand()%300+1;

for(j=0;j<i;j++){

if(temp < n[j]){

for(k=i-1;k>=j;k--){

n[k+1] = n[k];

}

break;

}

}

n[j] = temp;

}

for(i=0;i<N-1;i++){

cout<<i<<":"<<n[i]<<", ";

}

cout<<i<<":"<<n[i]<<"\n";

int a = rand()%300+1;

cout<<"\na:"<<a<<"\n";

int cnt = 0;

int index = bisearch(n, a, 0, 99, &cnt);

if(index == -1){

cout<<"Not in the array"<<endl;

}

else cout<<"Result index: "<<index<<" "<<"find times: "<<cnt<<endl;

return 0;

}

运行结果（示例）：

0:1, 1:7, 2:10, 3:11, 4:14, 5:16, 6:19, 7:20, 8:20, 9:20, 10:22, 11:24, 12:24, 13:37, 14:41, 15:46, 16:52, 17:55, 18:57, 19:57, 20:60, 21:60, 22:74, 23:76, 24:81, 25:85, 26:90, 27:91, 28:93, 29:93, 30:93, 31:94, 32:98, 33:99, 34:100, 35:100, 36:108, 37:111, 38:113, 39:113, 40:117, 41:124, 42:126, 43:132, 44:136, 45:145, 46:153, 47:154, 48:161, 49:167, 50:168, 51:170, 52:173, 53:175, 54:176, 55:177, 56:178, 57:180, 58:183, 59:187, 60:193, 61:196, 62:199, 63:199, 64:201, 65:202, 66:204, 67:210, 68:216, 69:225, 70:225, 71:229, 72:233, 73:235, 74:241, 75:246, 76:248, 77:249, 78:252, 79:252, 80:252, 81:256, 82:257, 83:258, 84:258, 85:261, 86:262, 87:263, 88:279, 89:280, 90:282, 91:283, 92:286, 93:287, 94:289, 95:289, 96:291, 97:295, 98:298, 99:299

a:24

Result index: 11 find times: 3

# 八皇后问题

编程实现八皇后问题：在8×8的国际棋盘上放置八个皇后，要求这八个皇后相互不能攻击(同行，同列，斜线)。

程序代码

#pragma GCC optimize(1)

#pragma GCC optimize(2)

#pragma GCC optimize(3,"Ofast","inline")

#include<bits/stdc++.h>

#include<regex>

using namespace std;

#define N 8

int cnt = 0;

bool check(bool queen[N][N], int i, int j){

int k;

for(k=0;k<i;k++){

if(queen[k][j] == 1) return(0);

}

for(k=1;i-k>=0&&j-k>=0;k++){

if(queen[i-k][j-k] == 1) return(0);

}

for(k=1;i-k>=0&&j+k<N;k++){

if(queen[i-k][j+k] == 1) return(0);

}

return(1);

}

void print(bool queen[N][N]){

int i,j;

for(i=0;i<N;i++){

for(j=0;j<N;j++){

if(queen[i][j]) cout<<"Q";

else cout<<"□";

}

cout<<"\n";

}

cout<<"\n";

}

int solve(bool queen[N][N], int i){

// begin from ith row

int j;

if(i == N){

print(queen);

cnt++;

}

else{

for(j=0;j<N;j++){

if(check(queen, i, j)){

queen[i][j] = 1;

solve(queen, i + 1);

queen[i][j] = 0;

}

}

}

}

int main()

{

ios::sync\_with\_stdio(false);

bool queen[N][N] = {0};

solve(queen, 0);

cout<<cnt;

return 0;

}

运行结果

Q□□□□□□□

□□□□Q□□□

□□□□□□□Q

□□□□□Q□□

□□Q□□□□□

□□□□□□Q□

□Q□□□□□□

□□□Q□□□□

Q□□□□□□□

□□□□□Q□□

□□□□□□□Q

□□Q□□□□□

□□□□□□Q□

□□□Q□□□□

□Q□□□□□□

□□□□Q□□□

Q□□□□□□□

□□□□□□Q□

□□□Q□□□□

□□□□□Q□□

□□□□□□□Q

□Q□□□□□□

□□□□Q□□□

□□Q□□□□□

Q□□□□□□□

□□□□□□Q□

□□□□Q□□□

□□□□□□□Q

□Q□□□□□□

□□□Q□□□□

□□□□□Q□□

□□Q□□□□□

□Q□□□□□□

□□□Q□□□□

□□□□□Q□□

□□□□□□□Q

□□Q□□□□□

Q□□□□□□□

□□□□□□Q□

□□□□Q□□□

□Q□□□□□□

□□□□Q□□□

□□□□□□Q□

Q□□□□□□□

□□Q□□□□□

□□□□□□□Q

□□□□□Q□□

□□□Q□□□□

□Q□□□□□□

□□□□Q□□□

□□□□□□Q□

□□□Q□□□□

Q□□□□□□□

□□□□□□□Q

□□□□□Q□□

□□Q□□□□□

□Q□□□□□□

□□□□□Q□□

Q□□□□□□□

□□□□□□Q□

□□□Q□□□□

□□□□□□□Q

□□Q□□□□□

□□□□Q□□□

□Q□□□□□□

□□□□□Q□□

□□□□□□□Q

□□Q□□□□□

Q□□□□□□□

□□□Q□□□□

□□□□□□Q□

□□□□Q□□□

□Q□□□□□□

□□□□□□Q□

□□Q□□□□□

□□□□□Q□□

□□□□□□□Q

□□□□Q□□□

Q□□□□□□□

□□□Q□□□□

□Q□□□□□□

□□□□□□Q□

□□□□Q□□□

□□□□□□□Q

Q□□□□□□□

□□□Q□□□□

□□□□□Q□□

□□Q□□□□□

□Q□□□□□□

□□□□□□□Q

□□□□□Q□□

Q□□□□□□□

□□Q□□□□□

□□□□Q□□□

□□□□□□Q□

□□□Q□□□□

□□Q□□□□□

Q□□□□□□□

□□□□□□Q□

□□□□Q□□□

□□□□□□□Q

□Q□□□□□□

□□□Q□□□□

□□□□□Q□□

□□Q□□□□□

□□□□Q□□□

□Q□□□□□□

□□□□□□□Q

Q□□□□□□□

□□□□□□Q□

□□□Q□□□□

□□□□□Q□□

□□Q□□□□□

□□□□Q□□□

□Q□□□□□□

□□□□□□□Q

□□□□□Q□□

□□□Q□□□□

□□□□□□Q□

Q□□□□□□□

□□Q□□□□□

□□□□Q□□□

□□□□□□Q□

Q□□□□□□□

□□□Q□□□□

□Q□□□□□□

□□□□□□□Q

□□□□□Q□□

□□Q□□□□□

□□□□Q□□□

□□□□□□□Q

□□□Q□□□□

Q□□□□□□□

□□□□□□Q□

□Q□□□□□□

□□□□□Q□□

□□Q□□□□□

□□□□□Q□□

□Q□□□□□□

□□□□Q□□□

□□□□□□□Q

Q□□□□□□□

□□□□□□Q□

□□□Q□□□□

□□Q□□□□□

□□□□□Q□□

□Q□□□□□□

□□□□□□Q□

Q□□□□□□□

□□□Q□□□□

□□□□□□□Q

□□□□Q□□□

□□Q□□□□□

□□□□□Q□□

□Q□□□□□□

□□□□□□Q□

□□□□Q□□□

Q□□□□□□□

□□□□□□□Q

□□□Q□□□□

□□Q□□□□□

□□□□□Q□□

□□□Q□□□□

Q□□□□□□□

□□□□□□□Q

□□□□Q□□□

□□□□□□Q□

□Q□□□□□□

□□Q□□□□□

□□□□□Q□□

□□□Q□□□□

□Q□□□□□□

□□□□□□□Q

□□□□Q□□□

□□□□□□Q□

Q□□□□□□□

□□Q□□□□□

□□□□□Q□□

□□□□□□□Q

Q□□□□□□□

□□□Q□□□□

□□□□□□Q□

□□□□Q□□□

□Q□□□□□□

□□Q□□□□□

□□□□□Q□□

□□□□□□□Q

Q□□□□□□□

□□□□Q□□□

□□□□□□Q□

□Q□□□□□□

□□□Q□□□□

□□Q□□□□□

□□□□□Q□□

□□□□□□□Q

□Q□□□□□□

□□□Q□□□□

Q□□□□□□□

□□□□□□Q□

□□□□Q□□□

□□Q□□□□□

□□□□□□Q□

□Q□□□□□□

□□□□□□□Q

□□□□Q□□□

Q□□□□□□□

□□□Q□□□□

□□□□□Q□□

□□Q□□□□□

□□□□□□Q□

□Q□□□□□□

□□□□□□□Q

□□□□□Q□□

□□□Q□□□□

Q□□□□□□□

□□□□Q□□□

□□Q□□□□□

□□□□□□□Q

□□□Q□□□□

□□□□□□Q□

Q□□□□□□□

□□□□□Q□□

□Q□□□□□□

□□□□Q□□□

□□□Q□□□□

Q□□□□□□□

□□□□Q□□□

□□□□□□□Q

□Q□□□□□□

□□□□□□Q□

□□Q□□□□□

□□□□□Q□□

□□□Q□□□□

Q□□□□□□□

□□□□Q□□□

□□□□□□□Q

□□□□□Q□□

□□Q□□□□□

□□□□□□Q□

□Q□□□□□□

□□□Q□□□□

□Q□□□□□□

□□□□Q□□□

□□□□□□□Q

□□□□□Q□□

Q□□□□□□□

□□Q□□□□□

□□□□□□Q□

□□□Q□□□□

□Q□□□□□□

□□□□□□Q□

□□Q□□□□□

□□□□□Q□□

□□□□□□□Q

Q□□□□□□□

□□□□Q□□□

□□□Q□□□□

□Q□□□□□□

□□□□□□Q□

□□Q□□□□□

□□□□□Q□□

□□□□□□□Q

□□□□Q□□□

Q□□□□□□□

□□□Q□□□□

□Q□□□□□□

□□□□□□Q□

□□□□Q□□□

Q□□□□□□□

□□□□□□□Q

□□□□□Q□□

□□Q□□□□□

□□□Q□□□□

□Q□□□□□□

□□□□□□□Q

□□□□Q□□□

□□□□□□Q□

Q□□□□□□□

□□Q□□□□□

□□□□□Q□□

□□□Q□□□□

□Q□□□□□□

□□□□□□□Q

□□□□□Q□□

Q□□□□□□□

□□Q□□□□□

□□□□Q□□□

□□□□□□Q□

□□□Q□□□□

□□□□□Q□□

Q□□□□□□□

□□□□Q□□□

□Q□□□□□□

□□□□□□□Q

□□Q□□□□□

□□□□□□Q□

□□□Q□□□□

□□□□□Q□□

□□□□□□□Q

□Q□□□□□□

□□□□□□Q□

Q□□□□□□□

□□Q□□□□□

□□□□Q□□□

□□□Q□□□□

□□□□□Q□□

□□□□□□□Q

□□Q□□□□□

Q□□□□□□□

□□□□□□Q□

□□□□Q□□□

□Q□□□□□□

□□□Q□□□□

□□□□□□Q□

Q□□□□□□□

□□□□□□□Q

□□□□Q□□□

□Q□□□□□□

□□□□□Q□□

□□Q□□□□□

□□□Q□□□□

□□□□□□Q□

□□Q□□□□□

□□□□□□□Q

□Q□□□□□□

□□□□Q□□□

Q□□□□□□□

□□□□□Q□□

□□□Q□□□□

□□□□□□Q□

□□□□Q□□□

□Q□□□□□□

□□□□□Q□□

Q□□□□□□□

□□Q□□□□□

□□□□□□□Q

□□□Q□□□□

□□□□□□Q□

□□□□Q□□□

□□Q□□□□□

Q□□□□□□□

□□□□□Q□□

□□□□□□□Q

□Q□□□□□□

□□□Q□□□□

□□□□□□□Q

Q□□□□□□□

□□Q□□□□□

□□□□□Q□□

□Q□□□□□□

□□□□□□Q□

□□□□Q□□□

□□□Q□□□□

□□□□□□□Q

Q□□□□□□□

□□□□Q□□□

□□□□□□Q□

□Q□□□□□□

□□□□□Q□□

□□Q□□□□□

□□□Q□□□□

□□□□□□□Q

□□□□Q□□□

□□Q□□□□□

Q□□□□□□□

□□□□□□Q□

□Q□□□□□□

□□□□□Q□□

□□□□Q□□□

Q□□□□□□□

□□□Q□□□□

□□□□□Q□□

□□□□□□□Q

□Q□□□□□□

□□□□□□Q□

□□Q□□□□□

□□□□Q□□□

Q□□□□□□□

□□□□□□□Q

□□□Q□□□□

□Q□□□□□□

□□□□□□Q□

□□Q□□□□□

□□□□□Q□□

□□□□Q□□□

Q□□□□□□□

□□□□□□□Q

□□□□□Q□□

□□Q□□□□□

□□□□□□Q□

□Q□□□□□□

□□□Q□□□□

□□□□Q□□□

□Q□□□□□□

□□□Q□□□□

□□□□□Q□□

□□□□□□□Q

□□Q□□□□□

Q□□□□□□□

□□□□□□Q□

□□□□Q□□□

□Q□□□□□□

□□□Q□□□□

□□□□□□Q□

□□Q□□□□□

□□□□□□□Q

□□□□□Q□□

Q□□□□□□□

□□□□Q□□□

□Q□□□□□□

□□□□□Q□□

Q□□□□□□□

□□□□□□Q□

□□□Q□□□□

□□□□□□□Q

□□Q□□□□□

□□□□Q□□□

□Q□□□□□□

□□□□□□□Q

Q□□□□□□□

□□□Q□□□□

□□□□□□Q□

□□Q□□□□□

□□□□□Q□□

□□□□Q□□□

□□Q□□□□□

Q□□□□□□□

□□□□□Q□□

□□□□□□□Q

□Q□□□□□□

□□□Q□□□□

□□□□□□Q□

□□□□Q□□□

□□Q□□□□□

Q□□□□□□□

□□□□□□Q□

□Q□□□□□□

□□□□□□□Q

□□□□□Q□□

□□□Q□□□□

□□□□Q□□□

□□Q□□□□□

□□□□□□□Q

□□□Q□□□□

□□□□□□Q□

Q□□□□□□□

□□□□□Q□□

□Q□□□□□□

□□□□Q□□□

□□□□□□Q□

Q□□□□□□□

□□Q□□□□□

□□□□□□□Q

□□□□□Q□□

□□□Q□□□□

□Q□□□□□□

□□□□Q□□□

□□□□□□Q□

Q□□□□□□□

□□□Q□□□□

□Q□□□□□□

□□□□□□□Q

□□□□□Q□□

□□Q□□□□□

□□□□Q□□□

□□□□□□Q□

□Q□□□□□□

□□□Q□□□□

□□□□□□□Q

Q□□□□□□□

□□Q□□□□□

□□□□□Q□□

□□□□Q□□□

□□□□□□Q□

□Q□□□□□□

□□□□□Q□□

□□Q□□□□□

Q□□□□□□□

□□□Q□□□□

□□□□□□□Q

□□□□Q□□□

□□□□□□Q□

□Q□□□□□□

□□□□□Q□□

□□Q□□□□□

Q□□□□□□□

□□□□□□□Q

□□□Q□□□□

□□□□Q□□□

□□□□□□Q□

□□□Q□□□□

Q□□□□□□□

□□Q□□□□□

□□□□□□□Q

□□□□□Q□□

□Q□□□□□□

□□□□Q□□□

□□□□□□□Q

□□□Q□□□□

Q□□□□□□□

□□Q□□□□□

□□□□□Q□□

□Q□□□□□□

□□□□□□Q□

□□□□Q□□□

□□□□□□□Q

□□□Q□□□□

Q□□□□□□□

□□□□□□Q□

□Q□□□□□□

□□□□□Q□□

□□Q□□□□□

□□□□□Q□□

Q□□□□□□□

□□□□Q□□□

□Q□□□□□□

□□□□□□□Q

□□Q□□□□□

□□□□□□Q□

□□□Q□□□□

□□□□□Q□□

□Q□□□□□□

□□□□□□Q□

Q□□□□□□□

□□Q□□□□□

□□□□Q□□□

□□□□□□□Q

□□□Q□□□□

□□□□□Q□□

□Q□□□□□□

□□□□□□Q□

Q□□□□□□□

□□□Q□□□□

□□□□□□□Q

□□□□Q□□□

□□Q□□□□□

□□□□□Q□□

□□Q□□□□□

Q□□□□□□□

□□□□□□Q□

□□□□Q□□□

□□□□□□□Q

□Q□□□□□□

□□□Q□□□□

□□□□□Q□□

□□Q□□□□□

Q□□□□□□□

□□□□□□□Q

□□□Q□□□□

□Q□□□□□□

□□□□□□Q□

□□□□Q□□□

□□□□□Q□□

□□Q□□□□□

Q□□□□□□□

□□□□□□□Q

□□□□Q□□□

□Q□□□□□□

□□□Q□□□□

□□□□□□Q□

□□□□□Q□□

□□Q□□□□□

□□□□Q□□□

□□□□□□Q□

Q□□□□□□□

□□□Q□□□□

□Q□□□□□□

□□□□□□□Q

□□□□□Q□□

□□Q□□□□□

□□□□Q□□□

□□□□□□□Q

Q□□□□□□□

□□□Q□□□□

□Q□□□□□□

□□□□□□Q□

□□□□□Q□□

□□Q□□□□□

□□□□□□Q□

□Q□□□□□□

□□□Q□□□□

□□□□□□□Q

Q□□□□□□□

□□□□Q□□□

□□□□□Q□□

□□Q□□□□□

□□□□□□Q□

□Q□□□□□□

□□□□□□□Q

□□□□Q□□□

Q□□□□□□□

□□□Q□□□□

□□□□□Q□□

□□Q□□□□□

□□□□□□Q□

□□□Q□□□□

Q□□□□□□□

□□□□□□□Q

□Q□□□□□□

□□□□Q□□□

□□□□□Q□□

□□□Q□□□□

Q□□□□□□□

□□□□Q□□□

□□□□□□□Q

□Q□□□□□□

□□□□□□Q□

□□Q□□□□□

□□□□□Q□□

□□□Q□□□□

□Q□□□□□□

□□□□□□□Q

□□□□Q□□□

□□□□□□Q□

Q□□□□□□□

□□Q□□□□□

□□□□□Q□□

□□□Q□□□□

□□□□□□Q□

Q□□□□□□□

□□Q□□□□□

□□□□Q□□□

□Q□□□□□□

□□□□□□□Q

□□□□□Q□□

□□□Q□□□□

□□□□□□Q□

Q□□□□□□□

□□□□□□□Q

□Q□□□□□□

□□□□Q□□□

□□Q□□□□□

□□□□□Q□□

□□□□□□□Q

□Q□□□□□□

□□□Q□□□□

Q□□□□□□□

□□□□□□Q□

□□□□Q□□□

□□Q□□□□□

□□□□□□Q□

Q□□□□□□□

□□Q□□□□□

□□□□□□□Q

□□□□□Q□□

□□□Q□□□□

□Q□□□□□□

□□□□Q□□□

□□□□□□Q□

□Q□□□□□□

□□□Q□□□□

Q□□□□□□□

□□□□□□□Q

□□□□Q□□□

□□Q□□□□□

□□□□□Q□□

□□□□□□Q□

□Q□□□□□□

□□□□□Q□□

□□Q□□□□□

Q□□□□□□□

□□□Q□□□□

□□□□□□□Q

□□□□Q□□□

□□□□□□Q□

□□Q□□□□□

Q□□□□□□□

□□□□□Q□□

□□□□□□□Q

□□□□Q□□□

□Q□□□□□□

□□□Q□□□□

□□□□□□Q□

□□Q□□□□□

□□□□□□□Q

□Q□□□□□□

□□□□Q□□□

Q□□□□□□□

□□□□□Q□□

□□□Q□□□□

□□□□□□Q□

□□□Q□□□□

□Q□□□□□□

□□□□Q□□□

□□□□□□□Q

Q□□□□□□□

□□Q□□□□□

□□□□□Q□□

□□□□□□Q□

□□□Q□□□□

□Q□□□□□□

□□□□□□□Q

□□□□□Q□□

Q□□□□□□□

□□Q□□□□□

□□□□Q□□□

□□□□□□Q□

□□□□Q□□□

□□Q□□□□□

Q□□□□□□□

□□□□□Q□□

□□□□□□□Q

□Q□□□□□□

□□□Q□□□□

□□□□□□□Q

□Q□□□□□□

□□□Q□□□□

Q□□□□□□□

□□□□□□Q□

□□□□Q□□□

□□Q□□□□□

□□□□□Q□□

□□□□□□□Q

□Q□□□□□□

□□□□Q□□□

□□Q□□□□□

Q□□□□□□□

□□□□□□Q□

□□□Q□□□□

□□□□□Q□□

□□□□□□□Q

□□Q□□□□□

Q□□□□□□□

□□□□□Q□□

□Q□□□□□□

□□□□Q□□□

□□□□□□Q□

□□□Q□□□□

□□□□□□□Q

□□□Q□□□□

Q□□□□□□□

□□Q□□□□□

□□□□□Q□□

□Q□□□□□□

□□□□□□Q□

□□□□Q□□□

92

# 整数拆分

将正整数n表示成一系列正整数之和，  
即： n = n1+n2+…+nk,  
其中：n1≥n2 ≥…≥nk≥1 (可以不满足)  
并给出给定正整数所有的分法的。

程序代码：

#pragma GCC optimize(1)

#pragma GCC optimize(2)

#pragma GCC optimize(3,"Ofast","inline")

#include<bits/stdc++.h>

#include<regex>

using namespace std;

void print(int n, std::vector<int>& a){

std::vector<int>::iterator it = a.begin();

cout<<n<<"="<<\*(it++);

while(it != a.end()){

cout<<"+"<<\*(it++);

}

cout<<"\n";

}

void divide(int n, std::vector<int>& a, int sum){

if(sum == n) print(n, a);

else{

int i = n-sum;

if(!a.empty()) i = min(i, a.back());

for(;i>0;i--){

a.push\_back(i);

sum += i;

divide(n, a, sum);

a.pop\_back();

sum -= i;

}

}

}

int main()

{

ios::sync\_with\_stdio(false);

int n;

cout<<"please enter an integer: ";

cin>>n;

std::vector<int> a;

divide(n, a, 0);

return 0;

}

整数5到10的拆分结果

please enter an integer: 5

5=5

5=4+1

5=3+2

5=3+1+1

5=2+2+1

5=2+1+1+1

5=1+1+1+1+1

please enter an integer: 6

6=6

6=5+1

6=4+2

6=4+1+1

6=3+3

6=3+2+1

6=3+1+1+1

6=2+2+2

6=2+2+1+1

6=2+1+1+1+1

6=1+1+1+1+1+1

please enter an integer: 7

7=7

7=6+1

7=5+2

7=5+1+1

7=4+3

7=4+2+1

7=4+1+1+1

7=3+3+1

7=3+2+2

7=3+2+1+1

7=3+1+1+1+1

7=2+2+2+1

7=2+2+1+1+1

7=2+1+1+1+1+1

7=1+1+1+1+1+1+1

please enter an integer: 8

8=8

8=7+1

8=6+2

8=6+1+1

8=5+3

8=5+2+1

8=5+1+1+1

8=4+4

8=4+3+1

8=4+2+2

8=4+2+1+1

8=4+1+1+1+1

8=3+3+2

8=3+3+1+1

8=3+2+2+1

8=3+2+1+1+1

8=3+1+1+1+1+1

8=2+2+2+2

8=2+2+2+1+1

8=2+2+1+1+1+1

8=2+1+1+1+1+1+1

8=1+1+1+1+1+1+1+1

please enter an integer: 9

9=9

9=8+1

9=7+2

9=7+1+1

9=6+3

9=6+2+1

9=6+1+1+1

9=5+4

9=5+3+1

9=5+2+2

9=5+2+1+1

9=5+1+1+1+1

9=4+4+1

9=4+3+2

9=4+3+1+1

9=4+2+2+1

9=4+2+1+1+1

9=4+1+1+1+1+1

9=3+3+3

9=3+3+2+1

9=3+3+1+1+1

9=3+2+2+2

9=3+2+2+1+1

9=3+2+1+1+1+1

9=3+1+1+1+1+1+1

9=2+2+2+2+1

9=2+2+2+1+1+1

9=2+2+1+1+1+1+1

9=2+1+1+1+1+1+1+1

9=1+1+1+1+1+1+1+1+1

please enter an integer: 10

10=10

10=9+1

10=8+2

10=8+1+1

10=7+3

10=7+2+1

10=7+1+1+1

10=6+4

10=6+3+1

10=6+2+2

10=6+2+1+1

10=6+1+1+1+1

10=5+5

10=5+4+1

10=5+3+2

10=5+3+1+1

10=5+2+2+1

10=5+2+1+1+1

10=5+1+1+1+1+1

10=4+4+2

10=4+4+1+1

10=4+3+3

10=4+3+2+1

10=4+3+1+1+1

10=4+2+2+2

10=4+2+2+1+1

10=4+2+1+1+1+1

10=4+1+1+1+1+1+1

10=3+3+3+1

10=3+3+2+2

10=3+3+2+1+1

10=3+3+1+1+1+1

10=3+2+2+2+1

10=3+2+2+1+1+1

10=3+2+1+1+1+1+1

10=3+1+1+1+1+1+1+1

10=2+2+2+2+2

10=2+2+2+2+1+1

10=2+2+2+1+1+1+1

10=2+2+1+1+1+1+1+1

10=2+1+1+1+1+1+1+1+1

10=1+1+1+1+1+1+1+1+1+1

(1234+学号)% 10+10的的拆分结果

please enter an integer: 12

12=12

12=11+1

12=10+2

12=10+1+1

12=9+3

12=9+2+1

12=9+1+1+1

12=8+4

12=8+3+1

12=8+2+2

12=8+2+1+1

12=8+1+1+1+1

12=7+5

12=7+4+1

12=7+3+2

12=7+3+1+1

12=7+2+2+1

12=7+2+1+1+1

12=7+1+1+1+1+1

12=6+6

12=6+5+1

12=6+4+2

12=6+4+1+1

12=6+3+3

12=6+3+2+1

12=6+3+1+1+1

12=6+2+2+2

12=6+2+2+1+1

12=6+2+1+1+1+1

12=6+1+1+1+1+1+1

12=5+5+2

12=5+5+1+1

12=5+4+3

12=5+4+2+1

12=5+4+1+1+1

12=5+3+3+1

12=5+3+2+2

12=5+3+2+1+1

12=5+3+1+1+1+1

12=5+2+2+2+1

12=5+2+2+1+1+1

12=5+2+1+1+1+1+1

12=5+1+1+1+1+1+1+1

12=4+4+4

12=4+4+3+1

12=4+4+2+2

12=4+4+2+1+1

12=4+4+1+1+1+1

12=4+3+3+2

12=4+3+3+1+1

12=4+3+2+2+1

12=4+3+2+1+1+1

12=4+3+1+1+1+1+1

12=4+2+2+2+2

12=4+2+2+2+1+1

12=4+2+2+1+1+1+1

12=4+2+1+1+1+1+1+1

12=4+1+1+1+1+1+1+1+1

12=3+3+3+3

12=3+3+3+2+1

12=3+3+3+1+1+1

12=3+3+2+2+2

12=3+3+2+2+1+1

12=3+3+2+1+1+1+1

12=3+3+1+1+1+1+1+1

12=3+2+2+2+2+1

12=3+2+2+2+1+1+1

12=3+2+2+1+1+1+1+1

12=3+2+1+1+1+1+1+1+1

12=3+1+1+1+1+1+1+1+1+1

12=2+2+2+2+2+2

12=2+2+2+2+2+1+1

12=2+2+2+2+1+1+1+1

12=2+2+2+1+1+1+1+1+1

12=2+2+1+1+1+1+1+1+1+1

12=2+1+1+1+1+1+1+1+1+1+1

12=1+1+1+1+1+1+1+1+1+1+1+1

列出不同拆分数与整数N之间的关系：N=1~100

1: 1

2: 2

3: 3

4: 5

5: 7

6: 11

7: 15

8: 22

9: 30

10: 42

11: 56

12: 77

13: 101

14: 135

15: 176

16: 231

17: 297

18: 385

19: 490

20: 627

21: 792

22: 1002

23: 1255

24: 1575

25: 1958

26: 2436

27: 3010

28: 3718

29: 4565

30: 5604

31: 6842

32: 8349

33: 10143

34: 12310

35: 14883

36: 17977

37: 21637

38: 26015

39: 31185

40: 37338

41: 44583

42: 53174

43: 63261

44: 75175

45: 89134

46: 105558

47: 124754

48: 147273

49: 173525

50: 204226

51: 239943

52: 281589

53: 329931

54: 386155

55: 451276

56: 526823

57: 614154

58: 715220

59: 831820

60: 966467

61: 1121505

62: 1300156

63: 1505499

64: 1741630

65: 2012558

66: 2323520

67: 2679689

68: 3087735

69: 3554345

70: 4087968

71: 4697205

72: 5392783

73: 6185689

74: 7089500

75: 8118264

76: 9289091

77: 10619863

78: 12132164

79: 13848650

80: 15796476

81: 18004327

82: 20506255

83: 23338469

84: 26543660

85: 30167357

86: 34262962

87: 38887673

88: 44108109

89: 49995925

90: 56634173

91: 64112359

92: 72533807

93: 82010177

94: 92669720

95: 104651419

96: 118114304

97: 133230930

98: 150198136

99: 169229875

100: 190569292

# 马踏棋盘问题（选作）

在国际棋盘上一只马从某格出发，踏遍其他所有格，要求每个格经过且只经过一次，最后回到出发点。

要求：如果N表示学号的最后两位，开始的坐标为：N/8行，N%8列

程序代码：

#pragma GCC optimize(1)

#pragma GCC optimize(2)

#pragma GCC optimize(3,"Ofast","inline")

#include<bits/stdc++.h>

#include<regex>

using namespace std;

#define N 8

int step[8][2];

int x,y;

int board[N][N];

int next\_step(int xnow, int ynow){

int rt = 0;

int xnext, ynext;

for(int i=0;i<8;i++){

xnext = xnow + step[i][0];

ynext = ynow + step[i][1];

if(xnext >= 0 && xnext < N && ynext >= 0 && ynext < N && board[xnext][ynext] == 0){

if(!(xnext == x && ynext == y)){

rt++;

}

}

}

return rt;

}

bool compare(const std::pair<int, int>& a, const std::pair<int, int>& b){

return next\_step(a.first, a.second) < next\_step(b.first, b.second);

}

bool solve(int xnow, int ynow, int cnt){

if(cnt == N\*N) return 1;

else{

std:vector<pair<int, int>> v;

int xnext, ynext;

for(int i=0;i<8;i++){

xnext = xnow + step[i][0];

ynext = ynow + step[i][1];

if(xnext >= 0 && xnext < N && ynext >= 0 && ynext < N && board[xnext][ynext] == 0){

if(!(cnt + 1 < N\*N && xnext == x && ynext == y)){

v.push\_back({xnext, ynext});

}

}

}

std::sort(v.begin(),v.end(),compare);

for(std::vector<pair<int, int>>::iterator it=v.begin(); it != v.end(); it++){

board[it->first][it->second] = cnt + 1;

if(solve(it->first, it->second, cnt + 1)) return 1;

board[it->first][it->second] = 0;

}

return 0;

}

}

int main(){

ios::sync\_with\_stdio(false);

step[0][0] = 1; step[0][1] = 2;

step[1][0] = 1; step[1][1] = -2;

step[2][0] = -1; step[2][1] = 2;

step[3][0] = -1; step[3][1] = -2;

step[4][0] = 2; step[4][1] = 1;

step[5][0] = 2; step[5][1] = -1;

step[6][0] = -2; step[6][1] = 1;

step[7][0] = -2; step[7][1] = -1;

cout<<"enter x: ";

cin>>x;

cout<<"enter y: ";

cin>>y;

cout<<"\n";

if(solve(x, y, 0)){

for(int i=0;i<N;i++){

for(int j=0;j<N;j++){

cout<<setw(3)<<board[i][j];

}

cout<<"\n";

}

}

else cout<<"no path";

return 0;

}

运行结果：

enter x: 7

enter y: 2

51 4 59 26 53 6 45 28

58 25 52 5 60 27 40 7

3 50 61 54 39 44 29 46

24 57 38 43 62 55 8 41

17 2 49 56 37 42 47 30

20 23 18 63 48 33 12 9

1 16 21 36 11 14 31 34

22 19 64 15 32 35 10 13

--------------------------------

Process exited after 454.4 seconds with return value 0