## 冒泡排序：

1. 实现代码

#include<stdio.h>

int arr[1000];

void bubbleSort(const int n) {

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

for (int j = 0; j < n - i - 1; j++) {

if (arr[j] > arr[j + 1]) {

const int temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

int main() {

int n;

printf("Input the number of elements:");

scanf("%d", &n);

printf("Input the elements:\n");

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

scanf("%d", &arr[i]);

}

bubbleSort(n);

printf("After sort: ");

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

printf("%d ", arr[i]);

}

}

1. 运行截图

文本

描述已自动生成

## 快速排序：

1. 实现代码

#include<stdio.h>

int arr[1000];

void quickSort(const int start, const int end) {

if (start >= end) return;

int i = start, j = end;

const int pivot = arr[i];

while (i < j) {

while (i < j && arr[j] > pivot) j--;

if (i < j) {

arr[i] = arr[j];

i++;

}

while (i < j && arr[i] <= pivot) i++;

if (i < j) {

arr[j] = arr[i];

j--;

}

}

arr[i] = pivot;

quickSort(start, i - 1);

quickSort(i + 1, end);

}

int main() {

int n;

printf("Input the number of elements:");

scanf("%d", &n);

printf("Input the elements:\n");

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

scanf("%d", &arr[i]);

}

quickSort(0, n - 1);

printf("After sort: ");

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

printf("%d ", arr[i]);

}

}

1. 运行截图

文本

描述已自动生成

## 奇数阶魔方阵：

1. 实现代码

#include<stdio.h>

int square[1000][1000];

void magicSquare(const int n) {

int i = 0, j = n / 2;

for (int num = 1; num <= n \* n; num++) {

square[i][j] = num;

i--;

j++;

if (i < 0) i = n - 1;

if (j == n) j = 0;

if (square[i][j] != 0) {

i++;

j--;

if (i == n) i = 0;

if (j < 0) j = n - 1;

i++;

}

}

}

int main() {

int n;

printf("Input n:");

scanf("%d", &n);

magicSquare(n);

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

for (int j = 0; j < n; j++)

printf("%4d", square[i][j]);

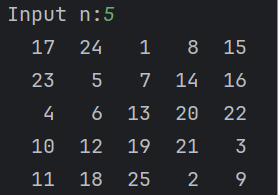
printf("\n");

}

return 0;

}

1. 运行截图



## 递归八皇后：

1. 实现代码

#include<stdio.h>

int board[10][10];

void eightQueens(int row) {

if (row == 8) {

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

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

printf("%d ", board[i][j]);

}

printf("\n");

}

printf("\n");

return;

}

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

int valid = 1;

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

if (board[j][i] == 1) {

valid = 0;

break;

}

}

for (int j = 1; j <= row; j++) {

if (i - j >= 0 && board[row - j][i - j] == 1) {

valid = 0;

break;

}

if (i + j < 8 && board[row - j][i + j] == 1) {

valid = 0;

break;

}

}

if (valid) {

board[row][i] = 1;

eightQueens(row + 1);

board[row][i] = 0;

}

}

}

int main() {

eightQueens(0);

return 0;

}

1. 运行截图

蓝色的键盘

低可信度描述已自动生成 ……

## 循环八皇后：

1. 实现代码

#include<stdio.h>

int board[10][10];

int check(){

int diagonal1[15] = {0}, diagonal2[15] = {0}, column[8] = {0};

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

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

if (board[i][j] == 1) {

if (column[j] == 1 || diagonal1[j - i + 7] == 1 || diagonal2[i + j] == 1) {

return 0;

}

column[j] = 1;

diagonal1[j - i + 7] = 1;

diagonal2[i + j] = 1;

}

}

}

return 1;

}

void eightQueens() {

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

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

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

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

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

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

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

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

board[0][i] = 1;

board[1][j] = 1;

board[2][k] = 1;

board[3][l] = 1;

board[4][m] = 1;

board[5][n] = 1;

board[6][o] = 1;

board[7][p] = 1;

if (check()) {

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

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

printf("%d ", board[q][r]);

}

printf("\n");

}

printf("\n");

}

board[0][i] = 0;

board[1][j] = 0;

board[2][k] = 0;

board[3][l] = 0;

board[4][m] = 0;

board[5][n] = 0;

board[6][o] = 0;

board[7][p] = 0;

}

}

}

}

}

}

}

}

}

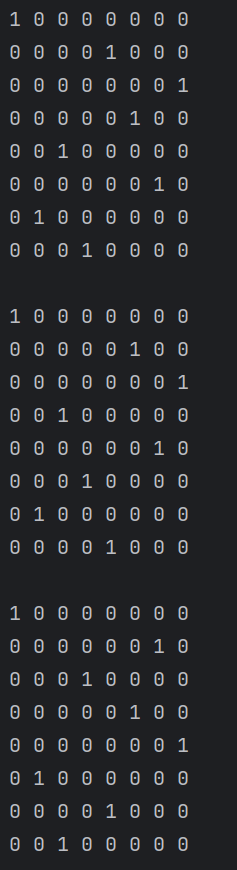
int main() {

eightQueens();

return 0;

}

1. 运行截图

 ……