# 数据结构和算法

# 第2章

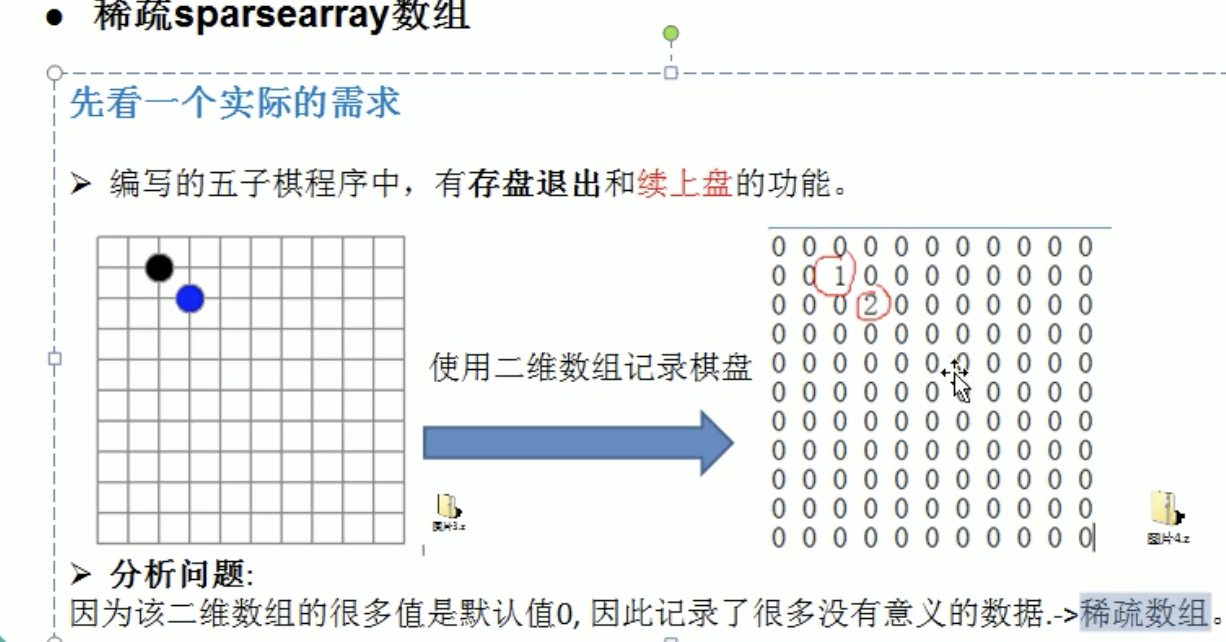
线性结构和非线性结构

线性结构：

1. 数据元素之间存在一对一的线性关系
2. 顺序存储结构和链式存储结构，顺序存储的线程表称为顺序表，顺序表的存储元素是连续的
3. 链式存储的线性表为链表，链表中的存储元素不一定是连续的，元素节点中存放数据元素及响铃的元素的地址信息
4. 线性表常见：数组、对列、链表和栈

非线性结构：二维数组，多为数组，广义表，树结构，图结构

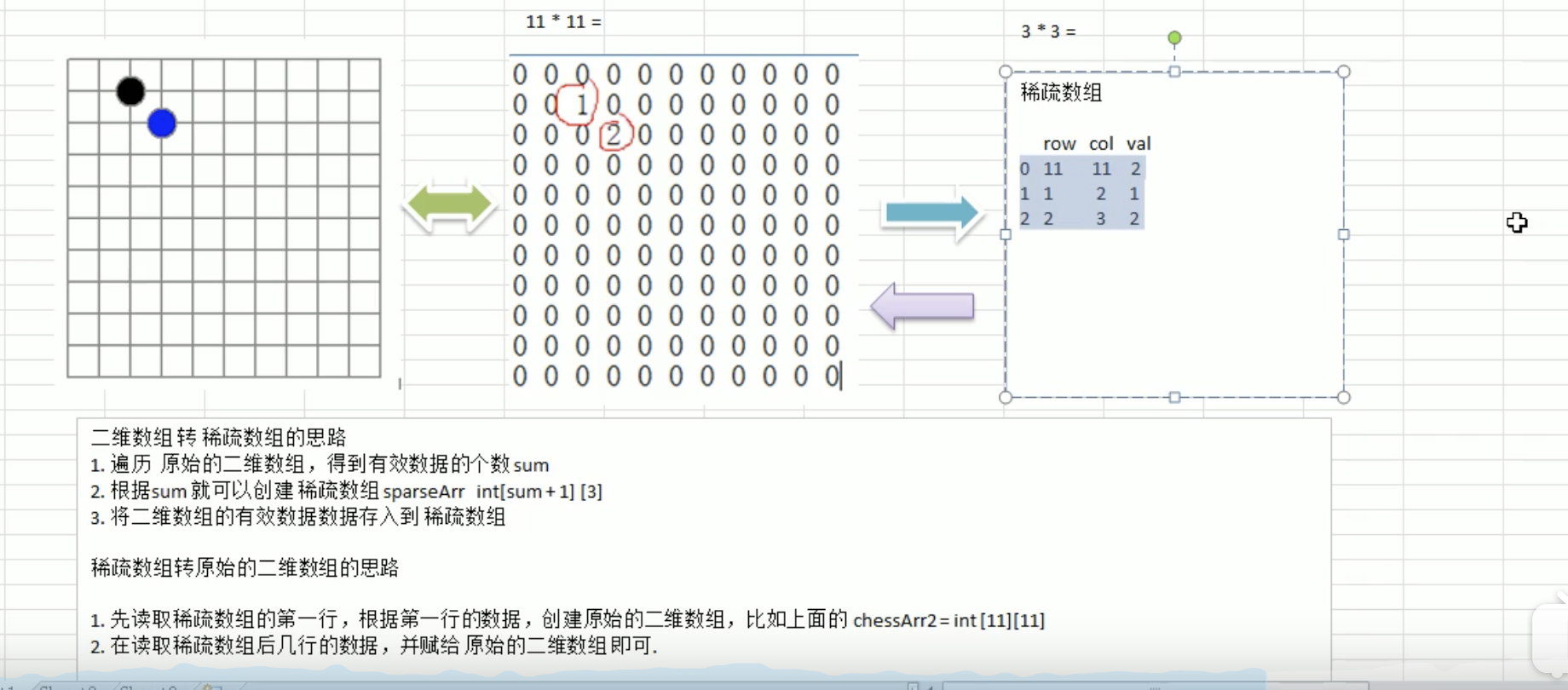
## 2.1 稀疏数组



当一个数组中大部分为0，或者为同一个值的数组时，可以使用稀疏数组来保存该数组

稀疏数组的处理方法是：

1. 记录数组一共有几行几列，有多少个不同的值
2. 把具有不同的值的元素的行列及值记录在一个小规模的数组中，从而缩小程序的规模



二维数组转稀疏数组：

1. 遍历 原始的二维数组，得到有效数据的个数sum
2. 根据sum就可以创建稀疏数组spaseArr int[sum+1][3]
3. 将二维数组的有效数据存入到稀疏数组

稀疏数组转原始的二维数组的思路：

1. 先读取稀疏数组的第一行，根据第一行的数据，创建原始的二维数组，比如：
2. 在读取稀疏数组后几行的数据。并赋给原始二维数组即可

稀疏数组的简单使用

|  |
| --- |
| package com.datastructure.sparsearray;  public class SparseArray {    public static void main(String[] args) {  // 创建一个原始的二维数组11\*11  // 0: 没有棋子，1 表示黑子 2表示篮子   int chessArr1[][] = new int[11][11];   chessArr1[1][2] = 1;  chessArr1[2][3] = 2;   System.*out*.println("原始的二维数组");  for (int[] row : chessArr1) {  for (int data : row) {  System.*out*.printf("%d\t", data);  }  System.*out*.println();  }   // 将二维数组 转稀疏数组的思路  // 1.先遍历二位数组 得到非0数据的个数   int sum = 0;  for (int i = 0; i < 11; i++) {   for (int j = 0; j < 11; j++) {  if (chessArr1[i][j] != 0) {  sum++;  }  }  }  System.*out*.println("sum=" + sum);   // 创建稀疏数组  int sparseArray[][] = new int[sum + 1][3];  // 给稀疏数组赋值  sparseArray[0][0] = 11;  sparseArray[0][1] = 11;  sparseArray[0][2] = sum;   int count = 0;  for (int i = 0; i < 11; i++) {  for (int j = 0; j < 11; j++) {  if (chessArr1[i][j] != 0) {  count++;  sparseArray[count][0] = i;  sparseArray[count][1] = j;  sparseArray[count][2] = chessArr1[i][j];  }  }  }   System.*out*.println();   System.*out*.println("得到的稀疏数组为");   for (int i = 0; i < sparseArray.length; i++) {  System.*out*.printf("%d\t%d\t%d\t\n", sparseArray[i][0], sparseArray[i][1], sparseArray[i][2]);  }   // 将稀疏数组恢复成二维数组   System.*out*.println("将稀疏数组恢复原始数组");  int chessArr2[][] = new int[sparseArray[0][0]][sparseArray[0][1]];   for (int i = 1; i < sparseArray.length; i++) {  chessArr2[sparseArray[i][0]][sparseArray[i][1]] = sparseArray[i][2];  }   for (int[] row : chessArr2) {  for (int data : row) {  System.*out*.printf("%d\t", data);  }  System.*out*.println();  }   }   } |

## 2.1 对列

