```
import java.util.Scanner;

public class TracciaC {

	/*
	 * Si scriva un metodo verificaVettore che riceve in ingresso due
vettori di
	 * interi v e w, e restituisce un valore booleano. In particolare
	 * restituisce true se per ogni elemento di v esiste in w un elemento
con lo
	 * stesso valore assoluto ma di segno opposto, false altrimenti. Ad
esempio,
	 * nel caso in cui v= [ 20, -9, 4, -8], restituisce true se w = [ 1, 9,
-20,
	 * 4, -4, 0 ,8 ], false se w = [ 1, 9, 4, 0 ,8 ].
	 *
	 *
	 */

	public static boolean verificaVettore(int[] v, int w[]) {

		for (int i = 0; i <= v.length; i++) {
			boolean trovato = false;
			for (int j = 0; j < w.length && !trovato; j++)
				if (v[i] == -w[j])
					trovato = true;
			if (!trovato)
				return false;
		}
		return true;
	}

	/*
	 * 1. un metodo creaArray che riceve in ingresso una matrice M ed un
intero
	 * p e restituisce in output un array contenente tutti gli elementi di M
	 * strettamente maggiori di p presenti sulle colonne pari
	 */
	public static int[] creaArray(int[][] m, int p) {
		int ncolPari = m[0].length / 2;
		if (m[0].length % 2 == 1)
			ncolPari += 1;
		int[] temp = new int[m.length * ncolPari];
		int k = 0;
		for (int j = 0; j < m[0].length; j += 2)
			for (int i = 0; i < m.length; i++)

				if (m[i][j] > p) {
					temp[k] = m[i][j];
					k++;
				}

		int[] ris = new int[k];
		System.arraycopy(temp, 0, ris, 0, k);
		return ris;
	}

	/*
	 * 2. un metodo estraiCroce che riceve in ingresso una matrice quadrata
M di
	 * dimensione dispari e restituisce in output una matrice contenente gli
	 * elementi di M presenti sulla riga e sulla colonna centrali escluso
	 * l'elemento centrale.;
	 */
	public static int[][] estraiCroce(int[][] m) {
		if (m.length % 2 != 1) {
			System.exit(1);// dimensione non dispari
		}
```

```
		if (m.length != m[0].length) {
			System.exit(2);// matrice non quadrata
		}
		int[][] ris = new int[m.length - 1][2];
		int offset = 0;
		int med = m.length / 2;
		for (int i = 0; i < ris.length; i++) {
			if (i == m.length / 2)
				offset = 1;
			ris[i][0] = m[med][i + offset];
			ris[i][1] = m[i + offset][med];
		}
		return ris;
	}

	/*
	 * 3. un metodo verificaMatrice che riceve in ingresso una matrice M e
	 * restituisce in output un booleano. In particolare il metodo
restituisce
	 * true se ogni colonna di M non contiene elementi duplicati, false
	 * altrimenti;
	 */
	public static boolean verificaMatrice(int[][] m) {
		for (int j = 0; j < m[0].length; j++) {
			for (int i = 0; i < m.length; i++) {

				for (int k = i + 1; k < m.length; k++) {
				    if (m[i][j] == m[k][j])
					return false;
				}
			}
		}
		return true;
	}

	/*
	 * 4. un metodo main che legge una matrice quadrata M di numeri interi e
	 * dimensione dispari e provvede, invocando opportunamente i metodi
	 * proposti, a realizzare i compiti di cui ai punti (1), (2) e (3)
	 */
	public static void main(String[] args) {
		Scanner console = new Scanner(System.in);
		int dim = 0;
		do {
			System.out.println("inserire dim matrice");
			dim = console.nextInt();
			if (dim <= 0) {
				System.out.println("la dim deve essere
positiva!!");
			} else if (dim % 2 == 0) {
				System.out.println("la dim deve essere
dispari!!");
			}
		} while (dim <= 0 || dim % 2 == 0);

		System.out.println("inserire valore di p");
		int p = console.nextInt();
		int[][] M = new int[dim][dim];
		leggiMatrice(M);
		int[] v = creaArray(M, p);
		System.out.println("risultato invocazione creaArray");
		stampaVettore(v);
		int[][] d = estraiCroce(M);
		System.out.println("risultato invocazione estrai croce");
		stampaMatrice(d);
		boolean esito = verificaMatrice(M);
		System.out.println("risultato invocazione verificaMatrice: " +
esito);
```

```
	}

	public static void stampaVettore(int[] v) {
		for (int i = 0; i < v.length; i++)

			System.out.print(v[i] + " ");

		System.out.println();

	}

	public static void leggiMatrice(int[][] m) {
		Scanner console = new Scanner(System.in);
		for (int i = 0; i < m.length; i++) {
			for (int j = 0; j < m[0].length; j++) {
				System.out.print("m[" + i + "," + j + "]=");
				m[i][j] = console.nextInt();

			}
		}
	}

	public static void leggiVettore(int[] v) {
		Scanner console = new Scanner(System.in);
		for (int i = 0; i < v.length; i++) {
			System.out.print("v[" + i + "]=");
			v[i] = console.nextInt();

		}
	}

	public static void stampaMatrice(int[][] m) {
		for (int i = 0; i < m.length; i++) {
			for (int j = 0; j < m[0].length; j++) {
				System.out.print(m[i][j] + " ");
			}
			System.out.println();
		}
	}
}
```