Tema 1: Se tiene una matriz precargada con secuencias de enteros distintos de cero en sus filas. Se pide guardar en un arreglo los números de filas de las primeras 4 filas que cumplen con la condición de tener a lo sumo 5 secuencias sin orden ascendente.

Observaciones: _el tamaño arreglo es igual a la cantidad de filas de la matriz, y está inicializado con -1,
_las secuencias están separadas por uno o más ceros, empiezan y terminan con uno o más ceros,
_realizar el programa completo sin realizar métodos de carga e impresión de matriz, ni utilizar estructuras auxiliares.

Una posible solución

```
public class RecuperatorioTema12022 {
	public static final int MAXFILA = 10;
	public static final int MAXCOLUMNA = 30;
	public static final int MAXDES = 5;
	public static final int PRIMFILAS = 4;
	public static void main(String[] args) {
		int [][] matint;
		int [] arrint;
		matint = new int[MAXFILA][MAXCOLUMNA];
		arrint = new int[MAXFILA];
		agregar_filas(matint,arrint);
	}

	public static void agregar_filas(int[][] mat,int[] arr){
		int fila,cant;
		fila = 0;
		cant = 0;
		while ((fila < MAXFILA)&&(cant < PRIMFILAS)){
			if (a_lo_sumo_cant_secuencias_sin_orden(mat[fila])){
				arr[cant] = fila;
				cant++;
			}
			fila++;
		}
	}

	public static boolean a_lo_sumo_cant_secuencias_sin_orden(int[] arr){
		int inicio,fin,cant;
		inicio = 0;
		fin = -1;
		cant = 0;
		while ((cant <= MAXDES)&&(inicio < MAXCOLUMNA)){
			inicio = obtener_inicio_secuencia(arr,fin+1);
			if (inicio < MAXCOLUMNA){
				fin = obtener_fin_secuencia(arr,inicio)-1;
				if (!secuencia_ordenada(arr,inicio,fin)){
					cant++;
				}
			}
		}
		return (cant <= MAXDES);
	}

	public static int obtener_inicio_secuencia(int [] arr, int pos){
		while ((pos < MAXCOLUMNA)&&(arr[pos] == 0)){
			pos++;
		}
		return pos;
	}
```

```
	public static int obtener_fin_secuencia(int [] arr, int pos){
		while ((pos < MAXCOLUMNA)&&(arr[pos] != 0)){
			pos++;
		}
		return pos;
	}

	public static boolean secuencia_ordenada(int[] arr, int inicio, int fin){
		while ((inicio < fin)&&(arr[inicio] < arr[inicio+1])){
			inicio++;
		}
		return (inicio == fin);
	}

}
```

Tema 2: Se tiene una matriz precargada con secuencias de enteros distintos de cero en sus filas. Además se tiene precargado un arreglo con algunos números de filas. Los números de filas no se repiten y están ubicados de forma consecutiva desde el principio del arreglo, luego se completa con -1. De dichas filas se pide eliminar la penúltima y la antepenúltima secuencia que tengan al menos 3 números impares.

Observaciones: _el tamaño arreglo es igual a la cantidad de filas de la matriz,
_las secuencias están separadas por uno o más ceros, empiezan y terminan con uno o más ceros,
_realizar el programa completo sin realizar métodos de carga e impresión de arreglo y matriz, ni utilizar estructuras auxiliares.

Una posible solución

```
public class RecuperatorioTema22022 {
	public static final int MAXFILA = 4;
	public static final int MAXCOLUMNA = 20;
	public static final int MAXCANT = 3;
	public static final int MAXIMP = 3;
	public static void main(String[] args) {
		int [][] matint;
		int [] arrint;
		matint = new int[MAXFILA][MAXCOLUMNA];
		arrint = new int[MAXFILA];
		eliminar_filas(matint,arrint);
	}

	public static void eliminar_filas(int[][] mat,int[] arr){
		int fila,cant;
		fila = 0;
		cant = 0;
		while ((cant < MAXFILA)&&(arr[cant] != -1)){
			fila = arr[cant];
			eliminar_anteultima_antepenultima(mat[fila]);
			cant++;
		}
	}

	public static void eliminar_anteultima_antepenultima(int[] arr){
		int inicio,fin,cant,tam;
		inicio = MAXCOLUMNA;
		fin = MAXCOLUMNA;
		cant = 0;
		while ((cant <= MAXCANT)&&(fin >= 0)){
			fin = obtener_fin_secuencia(arr,inicio-1);
			if (fin >= 0){
				inicio = obtener_inicio_secuencia(arr,fin)+1;
				if (cant_numeros_impares(arr,inicio,fin) >= MAXIMP){
					if (cant > 0){
						tam = fin-inicio+1;
						while (tam>0){
							corrimiento_izq(arr,inicio);
							tam--;
						}
					}
					cant++;
				}
			}
		}
	}
```

```
	public static void corrimiento_izq(int[] arr, int pos){
		while (pos < MAXCOLUMNA-1){
			arr[pos] = arr[pos+1];
			pos++;
		}
	}

	public static int obtener_fin_secuencia(int [] arr, int pos){
		while ((pos >= 0)&&(arr[pos] == 0)){
			pos--;
		}
		return pos;
	}

	public static int obtener_inicio_secuencia(int [] arr, int pos){
		while ((pos >= 0)&&(arr[pos] != 0)){
			pos--;
		}
		return pos;
	}

	public static int cant_numeros_impares(int [] arr, int inicio, int fin){
		cant = 0;
		while (inicio<=fin){
			if (arr[inicio]%2==1){
				cant++;
			}
			inicio++;
		}
		return cant;
	}

}
```