

Ejercicio U4_B4B_E1:

En los bordes de la matriz las celdas como mucho tendrán 5 vecinos, así que en la búsqueda puedo descartar la primera y última fila y la primera y última columna, o sea, los bordes de la matriz original.

Para cada caso

```
Leer datos y meterlos en variable tabla
alMenos6=0// un contador de celdas con 6 o más minas vecinas
Para cada fila i de segunda a penultima
    Para cada columna j de segunda a penultima
        if(tablero[i][j]==**) continue
        //la celda está vacía y cuento vecinos que son mina
        minas=0 //un contador de vecinos de i,j que son mina
        //me despreocupo de indexar fuera de rango
        Recorrer 8 vecinos mirando cuantos son mina
        if (minas>=6)
            alMenos6++

println(alMenos6)
```

Ejercicio U4_B4B_E2:

¡ojo!. Las coordenadas de los casos de prueba para árboles son col,fil no son *fil,col*

pseudocódigo:

el camping va a ser una matriz de int con el siguiente significado cada int

```
0->sol
1->sombra
9->árbol
```

mientras hay caso:

```
crear camping(matriz). Automáticamente todo es sol
colocar árboles
crear sombras
contar sombras
imprimir total sombras
```

Variantes: al mismo tiempo que se colocan los árboles se van poniendo las sombras.

mientras hay caso:

```
crear camping(matriz). Automáticamente todo es sol
para cada árbol leído
    colocar árboles
    crear sombras
finpara
contar sombras
imprimir total sombras
```

También simultáneamente a la creación de sombras se pueden ir contando dichas sombras.

mientras hay caso:

```
crear camping(matriz). Automáticamente todo es sol
para cada árbol leído
    colocar árboles
    crear sombras y actualizar total de sombras
finpara
contar sombras
imprimir total sombras
```

