

1. Código en Go para la función findPeak

```
1  package main
2
3  import "fmt"
4
5  // FindPeak finds the peak element in the array
6  func findPeak(arr []int) int {
7      n := len(arr)
8
9      // first or last element is peak element
10     if n == 1 {
11         return 0
12     }
13     if arr[0] >= arr[1] {
14         return 0
15     }
16     if arr[n-1] >= arr[n-2] {
17         return n - 1
18     }
19
20     // check for every other element
21     for i := 1; i < n-1; i++ {
22         // check if the neighbors are smaller
23         if arr[i] >= arr[i-1] && arr[i] >= arr[i+1] {
24             return i
25         }
26     }
27     return -1 // indicating no peak found
28 }
29
30 func main() {
31     arr := []int{1, 3, 20, 4, 1, 0}
32     fmt.Println("Index of a peak point is", findPeak(arr))
33 }
34
```

2. Explicación del código en Go

Explicación de cada línea:

```
package main
import "fmt"
```

Definimos el paquete principal ‘main’ y luego importamos el paquete ‘fmt’, que necesitaremos para imprimir los resultados en la consola.

```
func findPeak(arr []int) int {
    n := len(arr)
```

Definimos una función llamada ‘findPeak’ que toma una matriz de enteros como entrada y devuelve un entero (el índice con el número más alto). Calculamos la longitud de la matriz de entrada ‘arr’ y la almacenamos en la variable ‘n’.

```

1     if n == 1 {
2         return 0
3     }
4
5

```

Si la longitud de la matriz es 1, significa que hay solo un elemento en la matriz, y ese elemento es el pico. Entonces, devolvemos 0 como el índice del pico.

```

1     if arr[0] >= arr[1] {
2         return 0
3     }
4
5

```

Comprobamos si el primer elemento de la matriz es mayor o igual que el segundo. Si es así, entonces el primer elemento es el pico y devolvemos 0 como el índice del pico.

```

10    if arr[n-1] >= arr[n-2] {
11        return n - 1
12    }
13
14

```

Similar al paso anterior, aquí comprobamos si el último elemento de la matriz es mayor o igual que el segundo desde el final. Si es así, entonces el último elemento es el pico y devolvemos su índice.

```

15        for i := 1; i < n-1; i++ {
16
17
18
19
20

```

Este bucle itera sobre los elementos de la matriz, excepto el primero y el último.

```

21            if arr[i] >= arr[i-1] && arr[i] >= arr[i+1] {
22                return i
23            }
24        return -1
25    }
26
27
28
29
30

```

Dentro del bucle, comprobamos si el elemento actual es mayor o igual que sus vecinos. Si es así, entonces el elemento actual es el pico y devolvemos su índice. Si no se encuentra ningún pico, devolvemos -1 para indicar que no se encontró ningún pico en la matriz.

```

31    func main() {
32        arr := []int{1, 3, 20, 4, 1, 0}
33        fmt.Println("Index of a peak point is", findPeak(arr))
34    }
35
36
37
38
39
40

```

Ahora definimos la función 'main', que es la entrada principal del programa. Creamos una matriz de enteros llamada 'arr' con valores dados. Llamamos a la función 'findPeak' con la matriz 'arr' y mostramos el índice del punto de pico en la consola.