Go - Array

Explanation

> This Go program demonstrates the initialization and manipulation of arrays. Below are the key points explained:
>
> Array Initialization:
>
> var a1 = [3]int{100, 200, 399}: Declares and initializes an array a1 of length 3 with specified values. a2 := [4]int{4, 5, 6, 7}: Short-hand declaration and initialization of an array a2 of length 4. var a3 = [...]int{11, 22, 33}: Uses ... to let the compiler determine the length of the array a3. a4 := [...]int{11, 12, 13, 14}: Similar to a3, but with a different set of values. var a5 = [2]string{"Go", "C#"}: Declares and initializes an array a5 of strings. Default Initialization:
>
> a6 := [4]int{}: Declares an array a6 of length 4 with default values (0). a7 := [4]int{0, 1}: Partially initializes an array a7 of length 4. a8 := [4]int{0, 1, 2, 3}: Fully initializes an array a8 of length 4. Array Manipulation:
>
> prices := [3]int{10000, 20000, 30000}: Declares and initializes an array prices of length 3. prices[2] = 25000: Modifies the third element of the prices array. Printing Arrays and Elements:
>
> Uses fmt.Println to print entire arrays and specific elements. fmt.Println(len(prices)): Prints the length of the prices array. This program provides a comprehensive overview of array operations in Go, including declaration, initialization, modification, and accessing elements.

```
package main

import "fmt"

func main() {

    // Initializing an array of 3 integers
    var a1 = [3]int{100, 200, 399}
    // Short-hand declaration and initialization of an array of 4 integers
```

```
    a2 := [4]int{4, 5, 6, 7}

    // Using [...] to let the compiler determine the length of the array
    var a3 = [...]int{11, 22, 33}
    a4 := [...]int{11, 12, 13, 14}

    // Initializing an array of 2 strings
    var a5 = [2]string{"Go", "C#"}

    // Initializing an array of 3 integers
    prices := [3]int{10000, 20000, 30000}

    // Initializing an array of 4 integers with default values (0)
    a6 := [4]int{}           // Not initialized, so it will be filled with default
0 values [0 0 0 0]
    // Partially initializing an array of 4 integers
    a7 := [4]int{0, 1}       // Partially initialized
    // Fully initializing an array of 4 integers
    a8 := [4]int{0, 1, 2, 3} // Fully initialized

    // Printing arrays
    fmt.Println()
    fmt.Println(a1) // Print array a1
    fmt.Println(a2) // Print array a2

    fmt.Println()
    fmt.Println(a3) // Print array a3
    fmt.Println(a4) // Print array a4

    fmt.Println()
    fmt.Println(a5) // Print array a5

    fmt.Println()
    fmt.Println(prices) // Print array prices
    // Accessing and printing specific elements of the array
    fmt.Println(prices[0]) // Print the first element of prices
    fmt.Println(prices[2]) // Print the third element of prices

    // Modifying an element of the array
    prices[2] = 25000 // Change the third element of prices to 25000
    fmt.Println()
    fmt.Println(prices) // Print the modified array prices

    // Printing more arrays
    fmt.Println()
    fmt.Println(a6) // Print array a6
    fmt.Println(a7) // Print array a7
    fmt.Println(a8) // Print array a8

    // Printing the length of the array
    fmt.Println()
    fmt.Println(len(prices)) // Print the length of the array prices
}
```