如果只有一个循环变量，那么 range 仅返回元素的下标。

另外，range 也可作用于数组指针，其效果与作用于数组没有区别，仍然返回数组元素的下标和元素值。

2. 作用于切片

range 作用于切片时，与数组类似，返回元素的下标和元素值。举例如下：

```
func RangeSlice() {
    s := []int{1, 2, 3}
    for i, v := range s {
            fmt.Printf("index: %d value:%d\n", i, v)
    }
}
```

函数输出：

```
index: 0 value:1
index: 1 value:2
index: 2 value:3
```

同样，如果只有一个循环变量，那么 range 仅返回元素的下标。但 range 不可作用于切片指针。

3. 作用于 string

range 作用于 string 时，仍然返回元素的下标和元素值，但由于 string 底层使用 Unicode 编码存储字符，字符可能占用 1～4 个字节，所以下标可能是不连续的，并且元素值是该字符对应的 Unicode 编码的首个字节的值。

对于纯 ASCII 码中的字母来说，每个字符的 Unicode 编码仍占用 1 个字节：

```
func RangeString() {
    s := "Hello"
    for i, v := range s {
            fmt.Printf("index: %d, value: %c\n", i, v)
    }
}
```

函数输出：

```
index: 0, value: H
```