Go语言的内置容器(1) 数组

数组的定义

```
func main() {
   var arr1 [5]int          //仍然是相反的和一般的语言
   arr2 := [3]int{1, 3, 5} //可以直接打印
   arr3 := [...]int{2, 4, 6, 8, 10}//编译器辅助来确定数组的长度，而不是由
程序员给出
   fmt.Println(arr1, arr2, arr3)

}
```

数组的遍历

```
//数组的遍历 一般方法
for i := 0; i < len(arr3); i++ {
   fmt.Println(arr3[i])
}
//go语言特有的方法
/*
这里i为角标 v为其中的值
*/
for i, v := range arr3 {
    fmt.Println(i, v)
}

//若不要下标只需要值
//下划线省略变量
for _,v:=range arr3{
    fmt.Println(v)
}
```

在这个过程中我们又发现go这一门语言，真的是极简，C语言和C++中冗余的操作，在Go语言这里又进一步变化了

※数组作为函数参数

注意：这里是直接对数组进行拷贝，在函数中进行操作，并不会改变原本的值

```
//函数的定义
func printArray(arr [5]int) {  //注意这里的长度是需要连长度都传入进去的
   fmt.Println(arr)
}

//main函数中的内容
//但是不同的是，Go语言的数组传参的时候，不是引用传递，而是作为数值进行传递
printArray(arr3)
```

但是我们可以使用指针进行操作

```
//函数的定义
func printArray(arr *[5]int) {  //注意这里的长度是需要连长度都传入进去的
   arr[0]=100 //这里发现，这里调用不需要进行*进行调用
   for i,v:=range arr{
       fmt.Println(i,v)
    }
}

//main函数中的内容
//但是不同的是，Go语言的数组传参的时候，不是引用传递，而是作为数值进行传递
printArray(&arr3)
```

在go语言中一般不使用数组，一般使用切片，在下一节中会讲述。