package main  
  
import "fmt"  
  
type Account struct{  
AccountNumber int  
 FirstName string  
 LastName string  
}  
func (account Account) GetFullName() string{  
 return account.FirstName + " " + account.LastName  
}  
  
func (account Account) ChangeFirstName(firstName string) {  
 account.FirstName = firstName  
}  
  
func main(){  
 *//:= kullanimi variable'i hem create eder hemde initialize eder* account:=Account{1,"Okyanus","Kuce"}  
 account.ChangeFirstName("Cihangir")  
 fmt.Println(account.GetFullName())  
}

* Class’lar yerine go’da type’lar kullanılıyor.Type’in içerisin de herhangi bir method yada başka bir yapı tanımlanamıyor
* GetFullName methodu string tipinde veri döndürecek ve Account type’ina bağlanmış halde olacak.Yani Account type’i ile çalışacak
* ChangeFirstName methodu da dışarıdan gelen firstName değişkeni ile FirstName field’ini değiştirmeye çalışacak. Fakat burada çok önemli bir nokta var. Burada ki GetFullName() ve ChangeFirstName methodları Account objesinin sadece birer değer kopyasına sahip olurlar ve bu değerler iki method içinde sadece kopyalanmış olur .Main methodu içerisinde account.ChangeFirstName çalıştırıldığında bir altındaki GetFullName methodu çalıştırıldığında sonuç Okyanus Küce olacaktır. Çünkü iki methodda farklı değer kopyalarına sahiptirler.ChangeFirstName methodu ile değiştirilen FirstName sadece o method için geçerlidir.Bunları aynı objeyi göstermelerini istiyorsak pointer kullanmalıyız.

func (account \*Account) GetFullName() string{  
 return account.FirstName + " " + account.LastName  
}  
  
func (account \*Account) ChangeFirstName(firstName string) {  
 account.FirstName = firstName  
}

* Görüldüğü üzere \*Account tipinde tanımladığımız için artık bir pointer nesnesi haline gelecek