# Swift OOP Eğitimi

## Collections

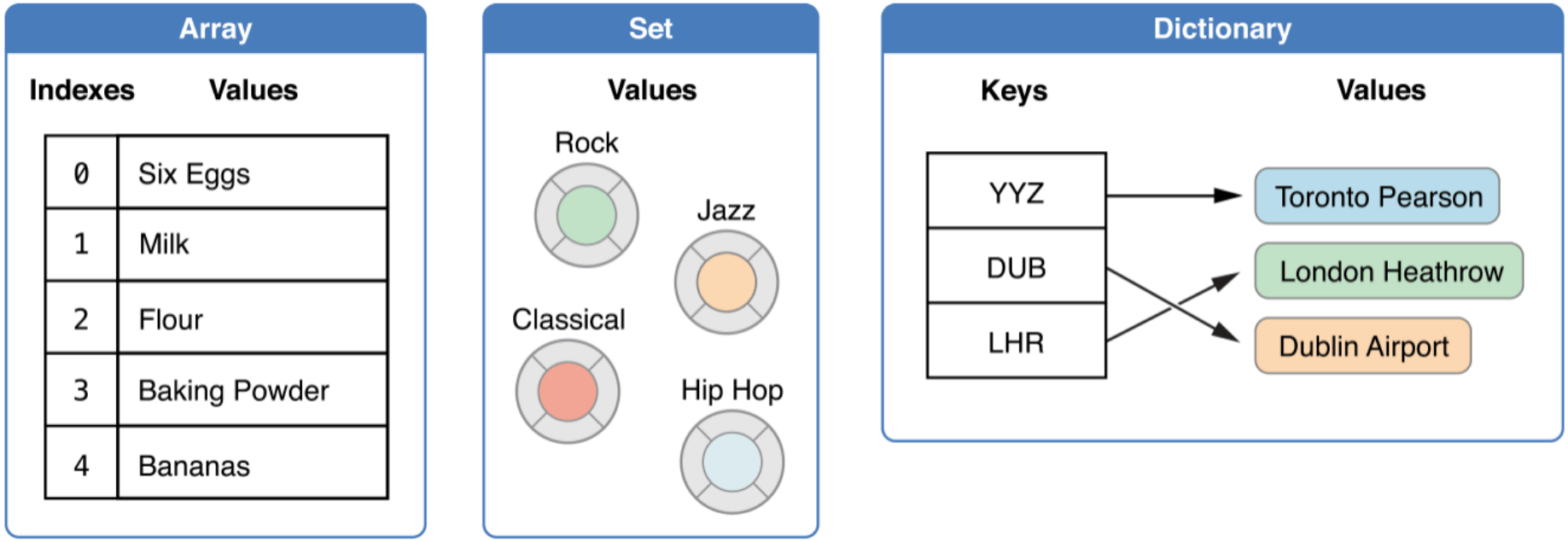
Kasım ADALAN

Elektronik ve Haberleşme Mühendisi
Freelance Software Developer

# Eğitim İçeriği

1. Array
2. Set
3. Dictionary

# Collection Types



# Array

# Array [ ]

- Aynı türde verileri bir arada tutar.
- İndeks numaraları 0 dan başlar.
- Array tanımlarken türü belirtilmelidir.

```
var meyveler:[String] = ["Çilek","Muz","Elma","Kivi","Kiraz"]
```

**meyveler**

| İndeks | Değer |
|---|---|
| 0 | Çilek |
| 1 | Muz |
| 2 | Elma |
| 3 | Kivi |
| 4 | Kiraz |

# Array Tanımlama Yöntemleri

```swift
var dizi1 = [Int]() //Boş bir dizi

var dizi2 = [1, 2, 3] //Baştan değer verilmiş

var dizi3:[Float] = [10.0, 20.0, 30.0] //Baştan değer verilmiş

var dizi4 = [Int](repeating: 0, count: 3)
//Varsayılan değerleri otomatik veren yapı
//3 Adet 0 değeri oluşur.
```

# Array Verilerine Erişim

```
var meyveler:[String] = ["Çilek","Muz","Elma","Kivi","Kiraz"]

var str1 = meyveler[2] // Elma

print(meyveler[4]) //Kiraz
```

**meyveler**

| İndeks | Değer |
|---|---|
| 0 | Çilek |
| 1 | Muz |
| 2 | Elma |
| 3 | Kivi |
| 4 | Kiraz |

# Array Veri Ekleme

```swift
var meyveler:[String] = ["Çilek","Muz","Elma","Kivi","Kiraz"]

meyveler.append("Karpuz")//Var olan verilerin sonuna ekleme

meyveler += ["Mandalina"]//Var olan verilerin sonuna ekleme

//Mevcut yerin üzerine veri yazma, Elma yerine Ananas
meyveler[2] = "Ananas"

print(meyveler)
```

```
["Çilek", "Muz", "Elma", "Kivi", "Kiraz"]

["Çilek", "Muz", "Elma", "Kivi", "Kiraz", "Karpuz"]

["Çilek", "Muz", "Elma", "Kivi", "Kiraz", "Karpuz", "Mandalina"]


"Ananas"

"["Çilek", "Muz", "Ananas", "Kivi", "Kiraz", "Karpuz", "Mandalina"]
```

# Array İşlemleri

```swift
var meyveler:[String] = ["Çilek","Muz","Elma","Kivi","Kiraz"]

meyveler.append("Karpuz")//Var olan verilerin sonuna ekleme

meyveler += ["Mandalina"]//Var olan verilerin sonuna ekleme

//Mevcut yerin üzerine veri yazma, Elma yerine Ananas
meyveler[2] = "Ananas"

print(meyveler)

//Belirtilen indekse veri yerleştirilir var olan
//veriler birer yana kayar
meyveler.insert("Portakal", at: 3)

meyveler.isEmpty //Boş Dolu Kontrol
meyveler.count //Arrayin eleman sayısı
meyveler.first //Arrayin ilk elemanı
meyveler.last //Arrayin son elemanı

//Belirtilen verinin array içinde olup olmadığının kontrolü
meyveler.contains("Kiraz")
//Sayısal veya metinsel olarak sıralamada en büyük değer
meyveler.max()
//Sayısal veya metinsel olarak sıralamada en küçük değer
meyveler.min()

meyveler.reverse() //Dizi içindeki verileri tersine çevirir
meyveler.sort() //Sayısal veya metinsel olarak küçükten
    büyüğe doğru sıralama yapar.

meyveler.remove(at: 2) //Belirtilen indeksteki veri silinir.
print(meyveler)

meyveler.removeAll() //Arrayi sıfırlamak için kullanılır.
```

```
["Çilek", "Muz", "Elma", "Kivi", "Kiraz"]

["Çilek", "Muz", "Elma", "Kivi", "Kiraz", "Karpuz"]

["Çilek", "Muz", "Elma", "Kivi", "Kiraz", "Karpuz", "Mandalina"]


"Ananas"

"["Çilek", "Muz", "Ananas", "Kivi", "Kiraz", "Karpuz", "Mandalina"]\n"



["Çilek", "Muz", "Ananas", "Portakal", "Kivi", "Kiraz", "Karpuz", "Mandalina"]

false
8
"Çilek"
"Mandalina"


true

"Portakal"

"Ananas"

["Mandalina", "Karpuz", "Kiraz", "Kivi", "Portakal", "Ananas", "Muz", "Çilek"]
["Ananas", "Çilek", "Karpuz", "Kiraz", "Kivi", "Mandalina", "Muz", "Portakal"]


"Karpuz"
"["Ananas", "Çilek", "Kiraz", "Kivi", "Mandalina", "Muz", "Portakal"]\n"

[]
```

# Array Filtreleme

# Array Filtreleme

```
var sayilar = [1,2,3,4,5,6,7,8,9,10]

var sonuc1 = sayilar.filter({$0>7})
print(sonuc1)//[8, 9, 10]

var sonuc2 = sayilar.filter({$0<7})
print(sonuc2)//[1, 2, 3, 4, 5, 6]

var sonuc3 = sayilar.filter({$0>3 && $0<7})
print(sonuc3)//[4, 5, 6]
```

# Array Iterating – Döngüler ile Veri Çekme

```swift
var meyveler:[String] = ["Çilek","Muz","Elma","Kivi","Kiraz"]

for meyve in meyveler {
    print("Sonuç : \(meyve)")
}
```

```
Sonuç : Çilek
Sonuç : Muz
Sonuç : Elma
Sonuç : Kivi
Sonuç : Kiraz
```

```swift
for (index, meyve) in meyveler.enumerated() {
    print("Sonuç \(index) : \(meyve)")
}//indeks ve içeriği aynı anda alma işlemi
```

```
Sonuç 0 : Çilek
Sonuç 1 : Muz
Sonuç 2 : Elma
Sonuç 3 : Kivi
Sonuç 4 : Kiraz
```

# Nesne Tabanlı - Array

# Örnek

```
class Ogrenci{
    var no:Int?
    var ad:String?
    var sinif:String?

    init(no:Int,ad:String,sinif:String) {
        self.no = no
        self.ad = ad
        self.sinif = sinif
    }
}

var o1 = Ogrenci(no: 100, ad: "Ahmet", sinif: "11F")
var o2 = Ogrenci(no: 90,  ad: "Zeynep", sinif: "10R")
var o3 = Ogrenci(no: 130, ad: "Ceyda", sinif: "12A")
var o4 = Ogrenci(no: 150, ad: "Mehmet",sinif: "9Z")
var o5 = Ogrenci(no: 110, ad: "Yasin", sinif: "11F")

var ogrenciListesi = [Ogrenci]()

ogrenciListesi.append(o1)
ogrenciListesi.append(o2)
ogrenciListesi.append(o3)
ogrenciListesi.append(o4)
ogrenciListesi.append(o5)

for ogrenci in ogrenciListesi {
    print("****************")
    print("Öğrenci No    : \(ogrenci.no!)")
    print("Öğrenci Ad    : \(ogrenci.ad!)")
    print("Öğrenci Sınıf : \(ogrenci.sinif!)")
}
```

# Array sort()

```
print("Sayısal Büyükten Küçüğe")
let siralamaArray1 = kisilerArray.sorted(by: {$0.kisiNo > $1.kisiNo} )
```

```
print("Sayısal Küçükten Büyüğe")
let siralamaArray2 = kisilerArray.sorted(by: {$0.kisiNo < $1.kisiNo} )
```

```
print("Harf Küçükten Büyüğe")
let siralamaArray3 = kisilerArray.sorted(by: {$0.kisiAd < $1.kisiAd} )
```

# Örnek

```
class Kisiler {
    var kisiNo:Int?
    var kisiAd:String?

    init(kisiNo:Int,kisiAd:String) {
        self.kisiAd = kisiAd
        self.kisiNo = kisiNo
    }
}

let kisi1 = Kisiler(kisiNo: 1,kisiAd: "Ahmet")
let kisi2 = Kisiler(kisiNo: 2,kisiAd: "Zeynep")
let kisi3 = Kisiler(kisiNo: 3,kisiAd: "Berna")

var kisilerArray = [Kisiler]()

kisilerArray.append(kisi1)
kisilerArray.append(kisi2)
kisilerArray.append(kisi3)
```

```
Önce
1 - Ahmet
2 - Zeynep
3 - Berna
Sayısal Büyükten Küçüğe
3 - Berna
2 - Zeynep
1 - Ahmet
Sayısal Küçükten Büyüğe
1 - Ahmet
2 - Zeynep
3 - Berna
Harf Küçükten Büyüğe
1 - Ahmet
3 - Berna
2 - Zeynep
```

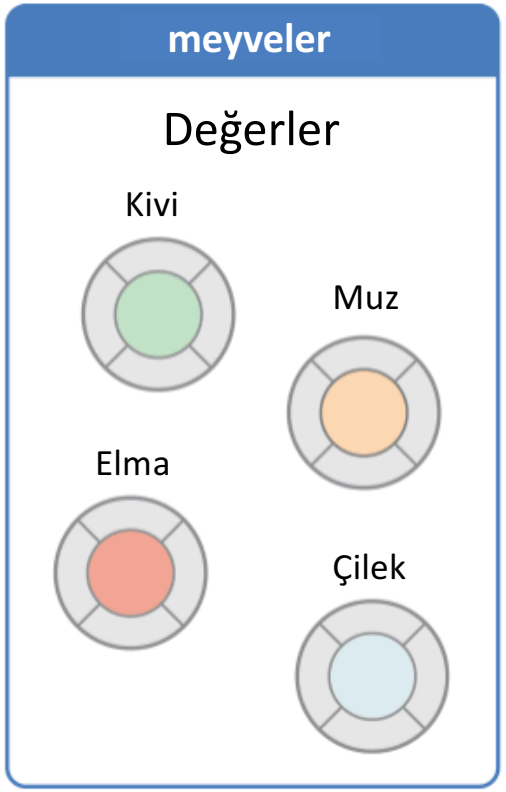
```
print("Önce")
for k in kisilerArray {
    print("\(k.kisiNo) - \(k.kisiAd)")
}

print("Sayısal Büyükten Küçüğe")
let siralamaArray1 = kisilerArray.sorted(by: {$0.kisiNo > $1.kisiNo} )

for k in siralamaArray1 {
    print("\(k.kisiNo) - \(k.kisiAd)")
}

print("Sayısal Küçükten Büyüğe")
let siralamaArray2 = kisilerArray.sorted(by: {$0.kisiNo < $1.kisiNo} )

for k in siralamaArray2 {
    print("\(k.kisiNo) - \(k.kisiAd)")
}

print("Harf Küçükten Büyüğe")
let siralamaArray3 = kisilerArray.sorted(by: {$0.kisiAd < $1.kisiAd} )

for k in siralamaArray3 {
    print("\(k.kisiNo) - \(k.kisiAd)")
}
```

# Set

# Set

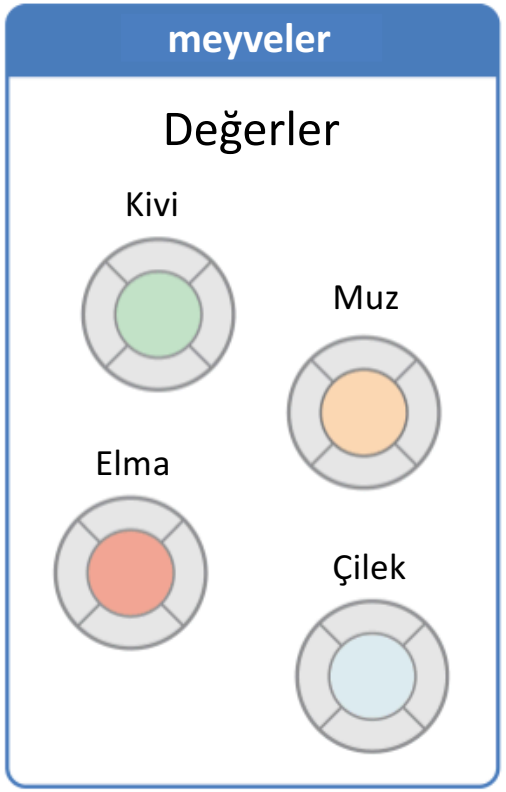
- Array ile aynı özelliklere sahiptir.
- İçine eklenen veriler düzensiz rasgele yerleştirilir.
- İndeks değerlerinin takibi zordur.

```
let meyveler: Set = ["Çilek","Muz","Elma","Kivi"]
```



# Set Tanımlama Yöntemleri

```
var sayilar = Set <Int>()
var meyveler: Set = ["Çilek","Muz","Elma","Kivi"]
var sehirler: Set<String> = ["Bursa", "İstanbul", "Ankara"]
```



# Set İşlemleri

```
var sayilar = Set <Int>()

sayilar.insert(10)//veri ekleme
sayilar.insert(20)
sayilar.insert(30) [20, 30, 10]

sayilar.isEmpty//Boş Dolu Kontrol
sayilar.contains(10) //Veri içeriyor mu kontrol
sayilar.sorted() //Verileri sıralama işlemli
sayilar.remove(30) //İndeks numarasına göre değil verinin içeriğine göre silme işlemi
                   [20, 10]
sayilar.removeAll()//Bütün içeriği sil
```

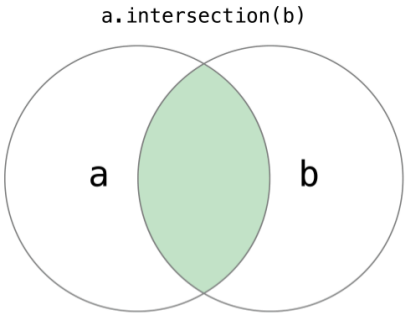
# Set Verilerin Alınması

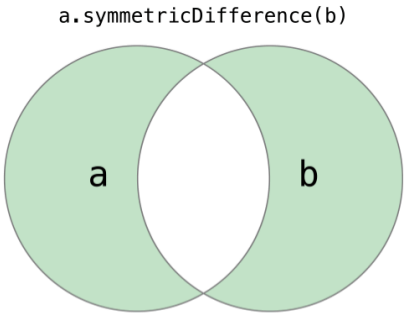
```
var sayilar = Set <Int>()

sayilar.insert(10)//veri ekleme
sayilar.insert(20)
sayilar.insert(30)

for s in sayilar {
    print(s)
}
```

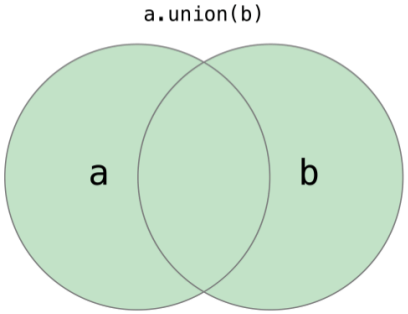
```
20
30
10
```

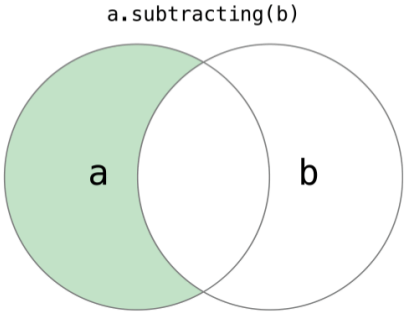
# Set Yapısına Özgü Metodlar

```
let tekler: Set = [1, 3, 5, 7, 9]
let ciftler: Set = [0, 2, 4, 6, 8]
let asal: Set = [2, 3, 5, 7]

tekler.union(ciftler).sorted()
// [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
tekler.intersection(ciftler).sorted()
// []
tekler.subtracting(asal).sorted()
// [1, 9]
tekler.symmetricDifference(asal).sorted()
// [1, 2, 9]
```



# Nesne Tabanlı - Set

# Nesne Tabanlı - Set

- Set yapı itibari ile içine insert edilen verileri rasgele sıralamaktadır.
- Bu rasgele sıralama ınt,string içeren set gibi ifadelerde kolaylıkla yapılabiliyor.
- Fakat set içine nesne yerleştirildiğinde nesne içindeki hangi değişkene göre bu rasgele sıralamayı yapacağını belirtmemiz gerekiyor.
- Bundan dolayı protocol olarak **Equatable,Hashable** kullanılır.
- Örn : Öğrencinin nosuna göre mi ? adına göre mi ? sınıfına göre mi ? sıralama yapılacak

```
var o1 = Ogrenci(no: 100, ad: "Ahmet", sinif: "11F")
var o2 = Ogrenci(no: 90,  ad: "Zeynep", sinif: "10R")
var o3 = Ogrenci(no: 130, ad: "Ceyda", sinif: "12A")
var o4 = Ogrenci(no: 150, ad: "Mehmet",sinif: "9Z")
var o5 = Ogrenci(no: 110, ad: "Yasin", sinif: "11F")
```

# Örnek

```
class Ogrenci: Equatable,Hashable{
    var no:Int?
    var ad:String?
    var sinif:String?

    init(no:Int,ad:String,sinif:String) {
        self.no = no
        self.ad = ad
        self.sinif = sinif
    }

    var hashValue: Int {//Hashable için gerekli metod
        get {
            return no.hashValue
            //öğrenci no suna göre
        }
    }

    static func == (lhs: Ogrenci, rhs: Ogrenci) -> Bool {
       return lhs.no == rhs.no////öğrenci no suna göre
    }//Equatable için gerekli metod
}
```

```
//NOT : SET içinde nesne kullanılacak ise nesnenin ait olduğu sınıf
//protocol olarak Equatable,Hashable özelliğine sahip olmalıdır.

var o1 = Ogrenci(no: 100, ad: "Ahmet", sinif: "11F")
var o2 = Ogrenci(no: 90,  ad: "Zeynep", sinif: "10R")
var o3 = Ogrenci(no: 130, ad: "Ceyda", sinif: "12A")
var o4 = Ogrenci(no: 150, ad: "Mehmet",sinif: "9Z")
var o5 = Ogrenci(no: 110, ad: "Yasin", sinif: "11F")

var ogrenciListesi = Set <Ogrenci>()//SET

ogrenciListesi.insert(o1)
ogrenciListesi.insert(o2)
ogrenciListesi.insert(o3)
ogrenciListesi.insert(o4)
ogrenciListesi.insert(o5)

for ogrenci in ogrenciListesi {
    print("****************")
    print("Öğrenci No    : \(ogrenci.no!)")
    print("Öğrenci Ad    : \(ogrenci.ad!)")
    print("Öğrenci Sınıf : \(ogrenci.sinif!)")
}
```

# Dictionary

# Dictionary

- Boş dictionary [:] temsil edilir.
- Javadaki hash map yapısının aynısıdır.
- Key ve value ilişkisi vardır.

```
var dic1 = [Int:String]()

var dic2 = [3.14:"Pi",2.71:"e"]

var dic3:[Int:String] = [1:"Bir",2:"İki",3:"Üç"]
```

# Veri Ekleme

```swift
var iller = [Int:String]()

iller[16] = "BURSA"
iller[34] = "İSTANBUL"
iller[6] = "ANKARA"
```

**Çıktı :**

```
[34: "İSTANBUL", 16: "BURSA", 6: "ANKARA"]
```

# Veri Güncelleme

```
var iller = [16:"Bursa",34:"İstanbul",6:"Ankara"]

iller[35] = "İzmir" // Veri ekleme
iller[10] = "Balıkesir" // Veri ekleme

iller[16] = "Yeni Bursa"//Veri güncelleme
iller.updateValue("Yeni İzmir", forKey: 35)//Veri güncelleme

print(iller)//[34: "İstanbul", 16: "Yeni Bursa", 35: "Yeni İzmir", 10: "Balıkesir", 6: "Ankara"]
```

# Veri Okuma

```
var iller = [16:"Bursa",34:"İstanbul",6:"Ankara"]

print(iller[34]!)//istanbul
```

# Döngü ilişkisi – Veri Çekme

```swift
var iller = [16:"Bursa",34:"İstanbul",6:"Ankara"]

//Verileri döngü ile alma
for (anahtar,deger) in iller {
    print("Anahtar : \(anahtar) ,İl : \(deger)")
}
```

```
Anahtar : 34 ,İl : İstanbul
Anahtar : 6 ,İl : Ankara
Anahtar : 16 ,İl : Bursa
```

# Veri Silme

```
var iller = [16:"Bursa",34:"İstanbul",6:"Ankara"]

iller.removeValue(forKey: 6)

//[16:"Bursa",34:"İstanbul"]
```

# Dictionary İşlemleri

```
iller.isEmpty //Boş Dolu Kontrol
iller.count //Dictionary eleman sayısı
iller.first //Dictionary ilk elemanı [16:"Yeni Bursa"]

var tersDic = iller.reversed() //Dictionary içindeki verileri tersine çevirir
print(tersDic)
```

# İki Diziden Dictionary Oluşturma

```
var dersler = ["Kimya","Matematik","Edebiyat"]

var notlar = [50,80,70]

var dersNotlari = Dictionary(uniqueKeysWithValues: zip(notlar, dersler))
//[70: "Edebiyat", 50: "Kimya", 80: "Matematik"]
```

# Dictionary'i Diziye Dönüştürme

```
var urunfiyatlari:[Double:String] = [15.99:"Kitap",59.99:"T-shirt",239.99:"Saat"]

var fiyatlar = [Double](urunfiyatlari.keys)
var urunler = [String](urunfiyatlari.values)

print(fiyatlar)//[59.99, 15.99, 239.99]
print(urunler)//["T-shirt", "Kitap", "Saat"]
```

# Filtreleme

```swift
var okul:[Int:String] = [154:"Ahmet",67:"Mehmet",871:"Zeynep",45:"Ahmet"]

//$0 dictionary temsil eder.

//KEY FİLTRELEME
//key'i 70'ten büyük olanları filtreler
var sonucDic1 = okul.filter({ $0.key > 70})
print(sonucDic1)//[154: "Ahmet", 871: "Zeynep"]

//VALUE FİLTRELEME
//value Ahmet'e  eşit olanları filtreler
var sonucDic2 = okul.filter({ $0.value == "Ahmet"})
print(sonucDic2)//[45: "Ahmet", 154: "Ahmet"]

//VALUE VE KEY FİLTRELEME
//value Ahmet'e  eşit olanları filtreler
var sonucDic3 = okul.filter({ $0.value == "Ahmet" && $0.key > 70})
print(sonucDic3)//[154: "Ahmet"]
```

# Nesne Tabanlı - Dictionary

# Örnek

```
class Ogrenci{
    var no:Int?
    var ad:String?
    var sinif:String?

    init(no:Int,ad:String,sinif:String) {
        self.no = no
        self.ad = ad
        self.sinif = sinif
    }
}
```

```
var o1 = Ogrenci(no: 100, ad: "Ahmet", sinif: "11F")
var o2 = Ogrenci(no: 90,  ad: "Zeynep", sinif: "10R")
var o3 = Ogrenci(no: 130, ad: "Ceyda", sinif: "12A")
var o4 = Ogrenci(no: 150, ad: "Mehmet",sinif: "9Z")
var o5 = Ogrenci(no: 110, ad: "Yasin", sinif: "11F")

var ogrenciListesi = [Int:Ogrenci]()//DICTIONARY

ogrenciListesi[o1.no!] = o1 //veri ekleme
ogrenciListesi[o2.no!] = o2
ogrenciListesi[o3.no!] = o3
ogrenciListesi[o4.no!] = o4
ogrenciListesi[o5.no!] = o5

for ( ogrenciNo,nesne) in ogrenciListesi {
    print("****************")
    print("Öğrenci No    : \(ogrenciNo)")
    print("Öğrenci Ad    : \(nesne.ad!)")
    print("Öğrenci Sınıf : \(nesne.sinif!)")
}
```

# Teşekkürler...



kasım-adalan


kasimadalan@gmail.com


kasimadalan