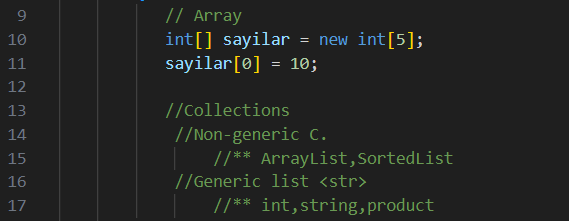
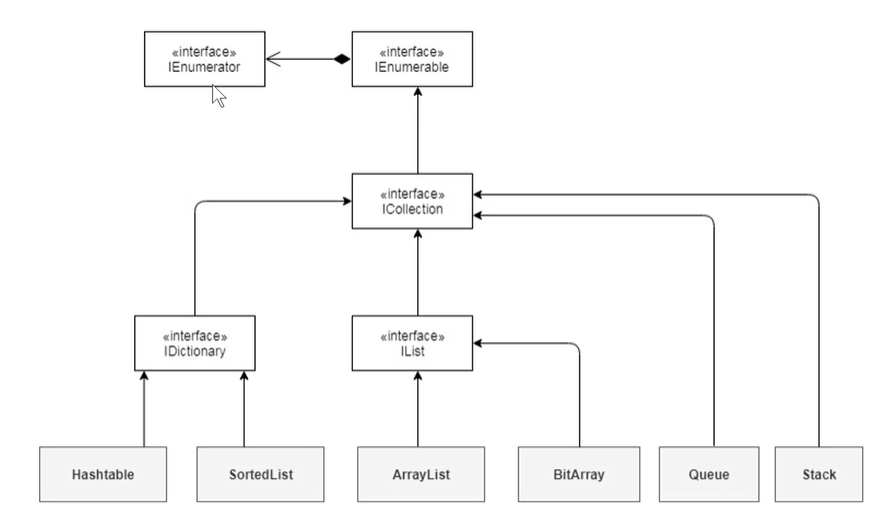
**C# Collections**

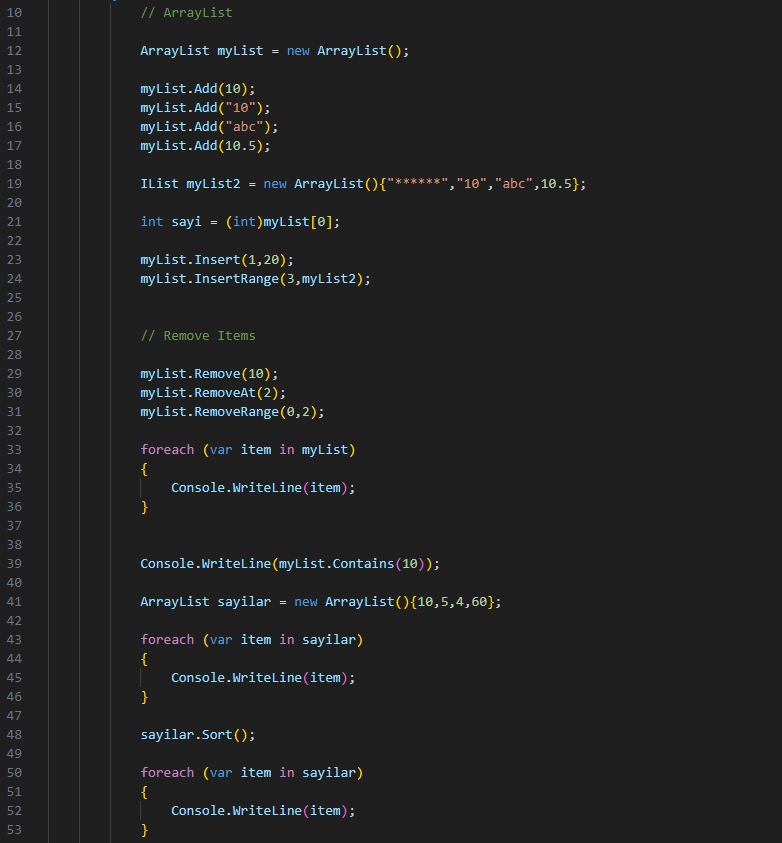


* Collections’ların normal arrayler ile genel farkı arraylar da dizi tanımlaması yaparken boyut belirlemek gerekir ama Collectionslarda böyle bir şart yoktur.



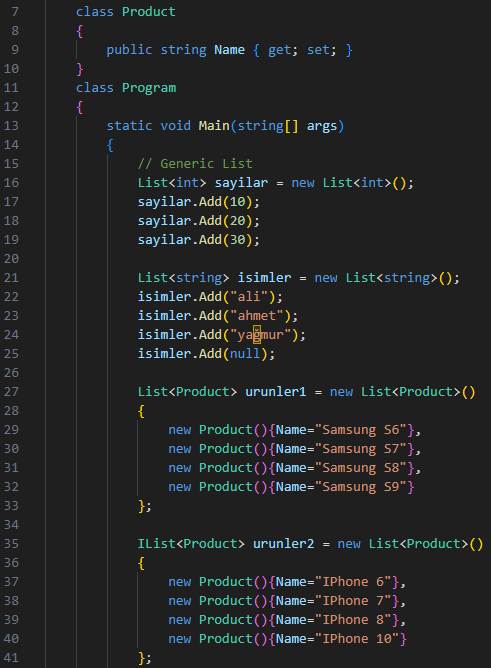
* Genel şema olarak böyle bir yapısı vardır.
* Dictinary (Sözlük) yapısı içerisinde bir key-value durumları vardır. Genelde veri yapılarında bu yapı kullanılır. Örnek bir isim:bilal, okul: hitit üniversitesi gibi düşünülebilir.
* Ilist ise genel olarak listeler olarak geçer ve dizilere kıyasla istenilen eleman eklenip, çıkarılabilir. Belirli bir boyutu şart koşmaz.

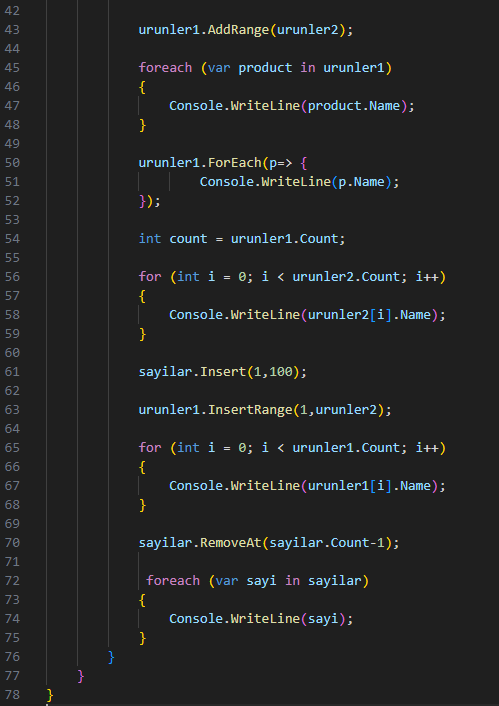
**ArrayList**



* ArrayList tanımlaması 12. Satırda yapılmıştır.
* .Add ile eleman eklenmektedir.
* 19.Satırdaki gibi IList de tanımlanır lakin ArrayList özelliği kadar özelliği yoktur. Örnek Insert çalışırken InstertRange çalışmaz.
* 21. Satırdaki gibi eleman bir değişkene atanabilir.
* 23,24. Satırlardaki Intsert ile (index,eleman) istenen index’e eleman eklenebilir. (1,2,3) instert(1,6) => (1,6,2,3). İnsertRange ile liste eklenebilir.
* 29-31. Satırlarda Remove direk 10 elemanını siler, RemoveAt ise 2. İndexteki elemanı siler. RemoveRange ile 0. İndexten başla iki tane eleman sil.
* 33. Foreach ile elemanları yazdırma işlemi yapılır.
* 39. Satır ile .Contains ile listede 10 varmı kontrolü yapar true false değeri dönderir.
* 41. Satırda int listesi tanımlandı, 43 ile ekrana yazdırdık, 48. Satır ile sıraladık daha sonra 50. Satırla tekrar ekrana yazdırdık.

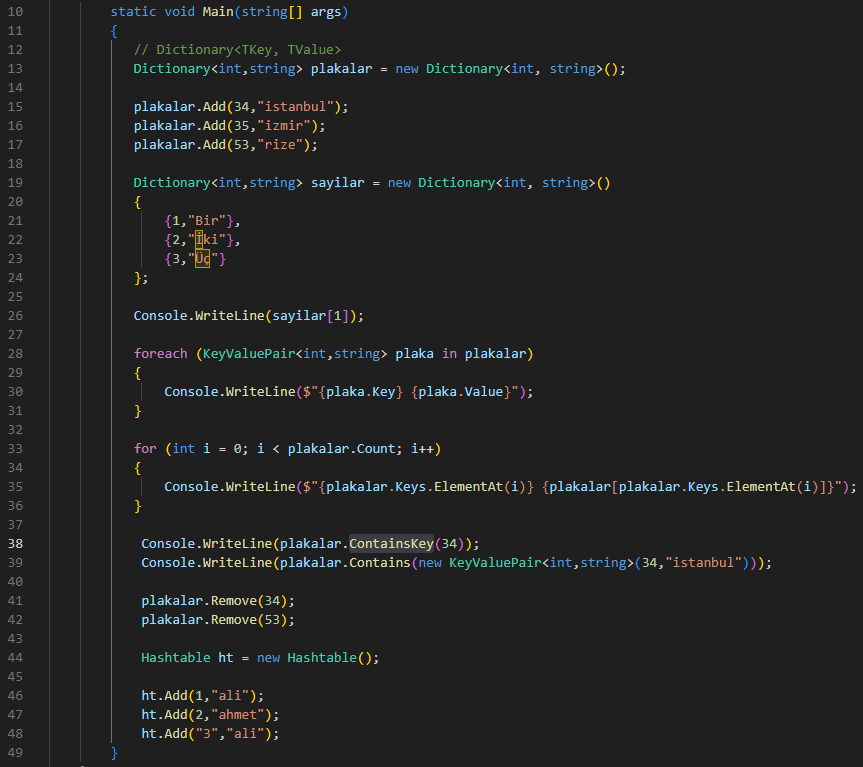
**Generic List<T>**





* 16,21,27. Satırlardaki gibi generic bir liste tanımlandığında tipi belirtilmelidir.
* Genel olaral List kullanılır, IList de kullanılabilir lakin Arraylistte olduğu gibi IList, List kadar kapsamlı değildir.
* 7. Satırda Product classı oluşturulduğundan 27 ve 35. Satırlarda Product nesnesi oluşturuldu.
* 43. Satırda AddRange ile ürünler 2 listesi ürünler 1’e eklendi.
* 50. Satırda kullanılan ForEach, forech döngüsünün farklı kullanımıdır. Urunler1 listesinin elemanlarını p’ye atar sonrada p’yi yazdırır. (p değişkendir, istediğinizi yazabilirsiniz.)
* 54. Satırda normalde dizi boyutu için lenght fonksiyonu kullanılırdı, generic listelerde count kullanılmaktadır.
* Insert kullanımları arraylist ile aynıdır, belirli indexe eleman ekleme işlemi yapar.

**Dictionary<TKey, TValue>**



* Listelerde bizler değerler giriyorduk ve bunların da bir index numarası oluyordu. Dictionary(Sözlük) yapısında ise indexleri biz kendimiz ayarlayabiliyoruz. Örnek 34: İstanbul şeklinde.
* Dictionary yapısı 13. Satırda gösterilmiştir. İnt key(index), string Value karşılık gelmektedir.
* 15-17. Satırlarda eleman ekleme işlemleri yapılmıştır.
* 19. Satırda farklı bir kullanımla gösterilmiştir.
* 26. Satırda [] yine key bilgisi girilmesi gerekiyor. Yani plakalar sözlüğü için İstanbul değerini getirmek istiyorsanız 34 yazılmalıdır.
* 28. Satırda Foreach döngüsü kullanımı verilmiştir. KeyValuePair<int,string> yerine var da yazılabilir. 30. Satırda ise plakanın altında sonuçta iki değer var. Ondan plaka.key diye çağırıyoruz.
* 33. Satırda for döngüsü ile kullanımı gösterilmiştir. Plakalar.Keys kodu bize sadece key listesini getirmektedir. .ElementAt ile biz gelen listedeki elemanları tek tek geziyoruz. 35. Satırın sağ tarafında plakalar[] içerisinde Plakalar.Keys.ElementAt yazmasının sebebi key bilgileri gelmekte. Örnek plakalar[34].
* 38,39. Satırlarda ContainsKey ile değer sorgulaması yapılmaktadır. Böyle bir değer varmı yokmu gibisinden true false değeri döner.
* Remove içerisine key bilgisi girilerek silme işlemi yapılır. Örnek 34 değeri girildiğinden İstanbul silincek.
* Hashtable yapısı içerisinde ise nesne yapısı gibidir. Örnek index int ver sonra string ver istenilen şekilde verilebilir. Lakin Hashtable yapısı dictionary yapısına kıyasla çok daha yavaştır.