

**T.C.
MİLLÎ EĞİTİM BAKANLIĞI**

BİLİŞİM TEKNOLOJİLERİ

METOTLAR

Ankara, 2017

- Bu bireysel öğrenme materyali, mesleki ve teknik eğitim okul/kurumlarında uygulanan çerçeve öğretim programlarında yer alan kazanımların gerçekleştirilmesine yönelik öğrencilere rehberlik etmek amacıyla hazırlanmıştır.
- Millî Eğitim Bakanlığınca ücretsiz olarak verilmiştir.
- PARA İLE SATILMAZ.

İÇİNDEKİLER

AÇIKLAMALAR	iii
GİRİŞ	1
ÖĞRENME FAALİYETİ-1	2
1. METOTLAR	2
1.1 Metot kavramı	2
1.2 Metot tanımlama	3
1.3 Metotlarla ilgili Önemli Özellikler ve Hatalar	6
1.4 Metotların farklı türlerle ve birden çok parametre ile kullanımı	7
1.5 Özyinelemeli (Recursive) Metotlar	11
1.6 Main() Metodu	12
DEĞERLER ETKİNLİĞİ:-1	14
UYGULAMA	15
ÖLÇME VE DEĞERLENDİRME	16
ÖĞRENME FAALİYETİ-2	19
2. HAZIR METOTLAR	19
2.1. Metinsel (String) Metotları	19
2.1.1. Compare()	20
2.1.2. Concat()	22
2.1.3. Copy()	22
2.1.4. Format()	23
2.1.5. IsNullOrEmpty()	28
2.1.6. CompareTo ()	29
2.1.7. Contains()	30
2.1.8. CopyTo()	31
2.1.9. EndsWith()	31
2.1.10. IndexOf()	32
2.1.11. Insert(int baslangic,string value)	34
2.1.12. LastIndexOf()	35
2.1.13. PadLeft ()	37
2.1.14. PadRight ()	38
2.1.15. Remove ()	40
2.1.16. Replace ()	41
2.1.17. Split ()	43
2.1.18. StartsWith ()	44
2.1.19. Substring ()	45
2.1.20. ToLower ()	45
2.1.21. ToUpper ()	46
2.2. Matematiksel (Math) Metotları	46
2.2.1. Abs()	46
2.2.2. BigMul()	47
2.2.3. Ceiling()	47
2.2.4. DivRem()	48
2.2.5. Max()	49
2.2.6. Min()	49
2.2.7. Pow()	50

2.2.8. Round()	51
2.2.9. Sign()	51
2.2.10. Sqrt()	51
2.2.11. Cos()	53
2.2.12. Sin()	53
2.2.13. Tan()	54
2.2.14. Acos()	54
2.2.15. Asin()	54
2.2.16. Atan()	54
2.3. Tarih/Saat (DateTime) Metotları	54
2.3.1. MinValue	55
2.3.2. MaxValue	55
2.3.3. Today	56
2.3.4. Now	56
2.3.6. DateTime.DaysInMonth()	59
2.3.7. DateTime.IsLeapYear()	59
2.3.8. DateTime.Parse()	59
2.3.9. Subtract()	61
2.3.10. AddDays()	61
2.3.11. AddMonths()	61
2.3.12. AddYears()	62
2.3.13. AddHours()	62
2.3.14. AddMinutes()	62
2.3.15. AddSeconds()	62
2.3.16. AddMilliseconds()	62
DEĞERLER ETKİNLİĞİ:-2	Hata! Yer işareti tanımlanmamış.
Aşağıdaki uygulama faaliyeti adımlarını iş güvenliği önlemlerini alarak uygulayınız.	
FAALİYETİ	65
ÖLÇME VE DEĞERLENDİRME	66
MODÜL DEĞERLENDİRME	68
CEVAP ANAHTARLARI	69
KAYNAKÇA	70

AÇIKLAMALAR

ALAN	Bilişim Teknolojileri
DAL	Alan Ortak
MODÜLÜN ADI	Metotlar
MODÜLÜN SÜRESİ	40/35
MODÜLÜN AMACI	Bireye/öğrenciye; iş sağlığı ve güvenliği tedbirlerini alarak metotlarla çalışma ile ilgili bilgi ve becerileri kazandırmaktır.
MODÜLÜN ÖĞRENME KAZANIMLARI	1. Algoritmaya uygun metotları oluşturabileceksiniz 2. Problemler için hazır metotları kullanabileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	Ortam: Bilgisayar laboratuvarı. Donanım: Bilgisayar, programlama yazılımı.
ÖLÇME VE DEĞERLENDİRME	Bireysel öğrenme materyali içinde yer alan ve her öğrenme faaliyetinden sonra verilen ölçme araçları ile kendinizi değerlendirebileceksiniz. Öğretmeniniz, bireysel öğrenme materyalinin sonunda, ölçme araçları (uygulamalı faaliyetler, iş ve performans testleri, çoktan seçmeli / doğru-yanlış ve boşluk doldurmalı sorular, vb.) kullanarak kazandığınız bilgi ve becerileri ölçüp değerlendirecektir..

GİRİŞ

Sevgili Öğrencimiz,

Programlama Temelleri dersinin bu bireysel öğrenme materyalinde programlamanın bir diğer temel yapı taşı olan **metotlar** öğreneceksiniz.

Bilgisayar programcılığına giden bu yolda önemli bir adımı da bu bireysel öğrenme materyalini öğrenerek atacaksınız.

Program yazarken belli bir işi yapan kod bloğunu birkaç kez kullanmak gerekebilir. Bunun için aynı kod bloğunu tekrar yazmak yerine bu bir metot olarak hazırlanır ve ihtiyaç duyulan yerde metot ismi ile çağrılarak çalıştırılabilir. Bu durum daha az kod yazma imkânı sağlayıp zaman kazandırdığı gibi olası değişikliklerde de daha az hata yapılmasını sağlar.

Bu bireysel öğrenme materyalinde birçok programlama dilinin temel kavramlarından olan metotlar detayları ile incelenecektir. Metotların tanımları ve kullanımı, parametrelerin özellikleri örneklerle işlenecektir.

ÖĞRENME FAALİYETİ-1

ÖĞRENME KAZANIMI

Algoritmaya uygun metotları oluşturabileceksiniz.

ARAŞTIRMA

- Çeşitli programlama dillerindeki alt program kavramlarını araştırınız.
- Fonksiyonların çalışma mantığı hakkında ön bilgi edininiz.
- Parametre, geri dönüş değeri nedir? Araştırınız.

1. METOTLAR

Programların hazırlanması esnasında aynı işlemi gerçekleştiren program parçalarına programın birçok yerinde ihtiyaç duyulabilir. Bu ihtiyaçlar, metotlar yazılarak giderilir. Metotlar kullanılmazsa programda aynı kodu defalarca yazmak gerekebilir ve program kodlarının okunması zorlaşır. Kaynak kodun gereksiz uzamasına sebep olur. Bunun için programın birçok yerinde ihtiyaç duyulan ve aynı işlemleri yapan program parçaları metotlar olarak hazırlanır.

1.1 Metot Kavramı

Programların herhangi bir yerinde kullanılmak için belirli bir işi yerine getirmek amacıyla tasarlanmış alt programlara **metot** denilir. Metotlar belirli bir işi yapması için geliştirilir.

Bir sefere mahsus yazılan bu kod parçaları programın akışı içinde defalarca çağrılarak kullanılabilir. Örneğin 100 satırlık bir kodu program içinde 100 kere kullanmak gerekirse bu 10.000 satır kod yapar. Fakat bu metot yöntemiyle yapılırsa 100 satırlık kod 100 kere çalışır ve aynı kodu 100 kere yazmaya gerek kalmaz. Bu yöntem kodun daha derli toplu görünmesini sağlar.

Metotların amacı; programın yapısal olmasını sağlamak ve birbiriyle ilgili komutları veya programın bir bölümünü istenen isim altında toplamaktır. Bu şekilde programın okunması kolaylaşır ve yapısal bir görünüm kazanır.

Bir metot, bir veya daha fazla ifade içerebilir. İyi yazılmış bir programda her metot yalnızca tek bir görev yürütür.

Metotlar tek başına çalışabilen yapılar değildir. Ancak ana program içinden çağrılarak çalıştırılır.

1.2 Metot Tanımlama

Her metodun bir ismi vardır ve program içinde metot çağrılırken bu isim kullanılır. Bu ismi metodun yaptığı işe göre vermek, kullanmak istenildiği zaman ona daha rahat erişmeyi sağlar.

Bir metodun iş yapabilmesi için kendi çağırılan metottan aldığı bilgilere **parametre**, kendisini çağırılan fonksiyona döndürdüğü değere de **metot geri dönüş değeri** (return value) denir.

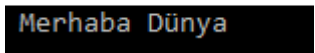
➤ Tanımlanması

```
erişim dönüş-tipi isim(parametre-listesi)
{
    //      metodun gövdesi;
}
```

- **Erişim:** Bir metoda programın diğer bölümlerinin nasıl erişebileceğini belirleyen bir erişim niteleyicisidir. Bunun kullanımı isteğe bağlıdır. Herhangi bir erişim belirteci kullanılmazsa varsayılan olarak sınıfa özel (**private**) belirlenir. Private olarak kullanıldığında yalnızca metodun yazıldığı sınıf içinden çağrılabilmesini öngörür. Programın içinde bulunan diğer kodlar içinden de bu metot çağrılabilir isteniyorsa erişim belirteci **public** olarak belirtilmelidir. Nesne yönelimli programlama dillerinde metotlar, tanımlandıkları sınıf adı ile birlikte çağrılırken metot, programın ana metodu (Main()) içinden çağrılacaksa **static** olarak tanımlanır ve sınıf adını yazmaya gerek kalmadan çağrılır.
- **Dönüş-tipi:** Metodun çalıştırdıktan sonra programda çağrıldığı noktaya döndürdüğü verinin tipinin belirlendiği kısımdır. Metot bir değer döndürmeyecekse dönüş-tipi **void** olarak belirtilmelidir. Değer döndürecekse değişken tanımlamada olduğu gibi (int, string, bool vb.) tanımlanmalı ve kesinlikle bir **return** satırı olmalıdır.
- **İsim:** Metodunun isminin belirtildiği kısımdır. Metoda isim verirken yapacağı iş ile alakalı bir isim vermek hem metodun ne işe yaradığıyla ilgili bilgi verecektir hem de daha sonra aynı programı kodlayacak kimselere yol gösterecektir. Metoda isim verirken aynı değişken isimler tanımlanırken kullanılan kurallar yine göz önünde bulundurulmalıdır. Geri dönüş tiplerinin veya parametre-listesinin farklı olması durumunda aynı isme sahip birden fazla metot olabilir.
- **Parametre-Listesi:** Virgül (,) ile ayrılmış tip ve tanımlayıcı çiftlerden oluşan bir listedir. Parametreler, metot çağrıldığında metodun kullanması için gönderilen bilgilerdir. Bu bilgiler metot içinde kullanılır. Metot hiç parametre kullanmayacaksa parametre listesi de boş olur.

UYGULAMA FAALİYETİ

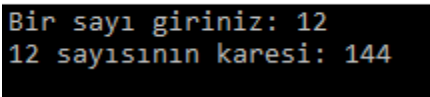
MT_1



Fotoğraf 1.1: Parametresiz ve geri dönüş türü void olan metot tanımlama

Bu örnekte önce Main() metot çalışmış, içinde MerhabaDunyaYazdir() metodunu görmüş ve bu metodu program içinde arayarak bulmuş, MerhabaDunyaYazdir() metodunu çalıştırmış ve bu metodun içindeki kodları ekrana yansıtmıştır.

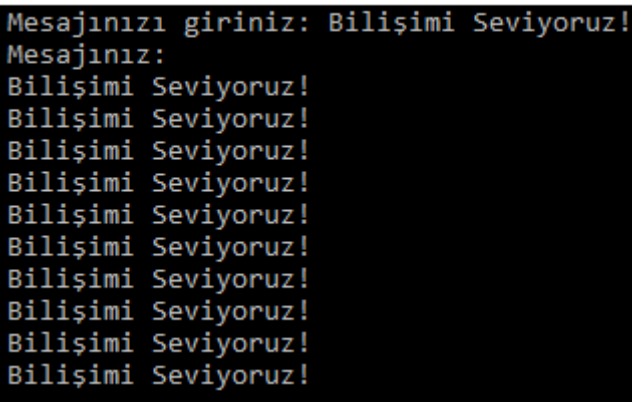
MT_2



Fotoğraf 1.2: Geri dönüş değeri ve parametresi olan metot tanımlama

Metot geri dönüşlü bir metot olduğu için int türünde tanımlanmış ve return satırıyla bir değer döndürmüştür. Önce Main() metodu çalışmış, ekrana ileti(mesaj) yazdırılmış, değer okunmuş ve alınan değer KaresiniAl() metoduna parametre olarak gönderilmiştir. KaresiniAl() metoduna gelen sayı burada **karesi** değişkenine çarpılarak aktarılmıştır. Sonuç olarak da **karesi** değişkeni **return** komutuyla ana metoda değer döndürmüş ve ekrana yazdırılmıştır.

MT_3



UYGULAMA FAALİYETİ

Fotoğraf 1.3: Geri dönüş türü void olan parametrelili metot uygulaması

1.3 Metotlarla İlgili Önemli Özellikler ve Hatalar

Metotlarla ilgili bilinmesi gereken bazı önemli özellikler şunlardır:

- Metotlara isim verilirken aynı değişkenlere isim verirken uyulan kurallara uyulmalıdır. **Main()** ismi, programın çalışmasını başlatan ana metodun ismi olduğu için bu isim metot ismi olarak verilemez.
- Aynı isme sahip farklı geri dönüş tiplerine veya farklı parametre-listesine sahip metotlar oluşturulabilir. Fakat bu yöntem pek de tavsiye edilen bir yöntem değildir. Bu şekilde aynı isme sahip farklı metotlar oluştururken çok dikkatli olunmalıdır.
- Metotlar çağrılırken başlangıçta belirlenen parametre sayısından ne az ne de çok sayıda parametre girilmelidir. Metot 2 parametre ile işlem yapıyorsa bu metoda 1 veya 3 adet parametre gönderilemez. Aksi takdirde hata mesajı alınır.
- Metotların geri dönüş değerleri vardır. Geri dönüş değeri olmayacak olan metotlarda geri dönüş tipi **void** olarak belirtilir ve **return** anahtar kelimesinin bu türdeki metotlarda kullanımına izin verilmez.
- Geri dönüş türü void olan metotlar, herhangi bir değişken içine atanamaz.
- Metotların geri dönüş değerleri herhangi bir veri türünde olabilir. Metot içindeki bir değer return anahtar sözcüğüyle metodun çağrıldığı yere geri döndürülür. Burada metodun geri dönüş tipine uyumlu bir değişken içine atanmalıdır. Aksi takdirde tür uyumsuzluğundan dolayı hata mesajı alınır. Yani int türünde tanımlanmış olan sonuç değişkeni içine float türünde tanımlanmış bir metodun geri dönüşü atanmaya çalışılırsa hata mesajı alınır.
- Bir metot parametre almadan da tanımlanabilir. Bu şekilde tanımlanan bir metoda parametre gönderilmez. Parametre-listesi parantezleri boş bırakılır.
- Metotlar tanımlanırken oluşturulan parametre-listesindeki tüm parametreler virgül (,) ile birbirinden ayrılmalıdır. Tek bir tür yazıp virgülle değişken isimleri ayrılamaz.
- Parametre-listesinde tanımlanan değişkenlerin isimleri, metot içinde tanımlanacak başka bir değişkende tekrar kullanılamaz.
- Bir metot içinde başka bir metot tanımlanamaz. Ancak başka bir metot çağrılabilir.
- Metotların içinde tanımlanan hiçbir değişken metot dışında kullanılamaz, hepsi

UYGULAMA FAALİYETİ

geçersiz olur.

1.4 Metotların Farklı Türlerle ve Birden Çok Parametre İle Kullanımı

Parametrenin tanımını ve kullanımı daha önce metotların tanımlanması sırasında parametre listeleri oluşturulurken anlatıldı.

Parametre-listeleri, tek bir türde verileri içeren bir liste olabileceği gibi farklı türlerde de veriler içerebilen listelerdir.

Parametreler veri türünde olabileceği gibi nesneler de parametre olarak bir metoda gönderilebilir.

Her bir parametre aralarına virgül kullanılarak birbirinden ayrılır. Aynı veri türüne sahip parametrelerin her biri için değişken isimlerinden önce ayrı ayrı veri türleri de yazılmak zorundadır.

MP_1

MP_2

MP_3

1.5 Özyinelemeli (Recursive) Metotlar

Bir metodun kendi kendini çağırmasına **yinelenme** (recursion), kendi kendini çağıran metotlara da **yinelenen** veya **özyineli** (recursive) **metotlar** denir.

Bir metodun kendi kendini çağırması, zaman zaman da olsa program yazarken ihtiyaç duyulan bir olaydır. Yinelenen metotlar tasarlanırken çok dikkatli olunmalıdır. Aksi takdirde sonsuz bir döngü içine girilebilir. Bu döngünün bir şekilde sonlandırılması gerekir.

MÖ_1

UYGULAMA FAALİYETİ

file:///c:/users/imsiyat/doc

```
0! = 1
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
7! = 5040
8! = 40320
9! = 362880
10! = 3628800
```

Fotoğraf 1.4: Yineleme metodunun sonlandırılması

1.6 Main() Metodu

Metotlar bireysel öğrenme materyalinin başından bu yana örneklerde hep Main() isminde bir metot içinde ana programların yazımı gerçekleştirildi.

Aslında Main() metodu şimdiye kadar yazılan veya kullanılan metotlardan pek de farkı olmayan bir metot türüdür. **Tek ve en önemli farkı Main() metodunun ana programın başlamasını sağlayan nokta olmasıdır.** İşte bu yüzden Main() metodu diğer metotlara göre daha özeldir.

Şimdiye kadar Main() metodu kullanılırken herhangi bir geri dönüş değeri kullanılmadı. Geri dönüş hep **void** olarak belirlendi. Ancak bazı durumlarda Main() metodunun **void** dışında **int** türünde bir geri dönüş değeri de kullanılır. Bu geri dönüş değeri, aslında kullanıcıların işine yarayacak bir geri dönüş değeri değildir. Bu değer, genellikle (bütün programların üzerinde çalıştığı) işletim sisteminin yazılan programın nasıl sonlandırıldığıyla ilgili bilgi almasını sağlayacak bir değerdir.

Eğer dönüş değeri;

- Sıfır (0) ise program normal bir şekilde,
- Sıfırdan farklı ise programın bir hata sebebiyle **sonlandırılmış olduğunu belirtir.**

Main() metodunun geri dönüş değerinin olmasının yanı sıra bazı durumlarda da parametre alması mümkündür.

Programların **komut satırından** aldıkları parametrelere **argüman** adı verilir. Programlar komut satırından çalıştırıldığında program isminden sonra gelen bilgiler o programın argümanlarıdır.

MM_1

C:\Windows\system32\cmd.exe

```
C:\Users\imsiyat\Documents\Visual Studio 2012\Projects\mainfonk\mainfonk\bin\Debug>mainfonk.exe 85 24
Girilen sayıların toplamı: 109
C:\Users\imsiyat\Documents\Visual Studio 2012\Projects\mainfonk\mainfonk\bin\Debug>
```

UYGULAMA FAALİYETİ

UF_1

UF_2

UF_3

UF_4

UF_5

UF_6

UF_7

MDY_1

ÖĞRENME FAALİYETİ-2

ÖĞRENME KAZANIMI

Problemler için hazır metotları kullanabileceksiniz.

ARAŞTIRMA

- String değişken türleri nasıl tanımlanır? Araştırınız.
- Sayılarla kullanılabilen değişken türleri araştırınız.
- Tarih/Zaman ifadeleri ile birlikte kullanılan değişken türlerini araştırınız.

2. HAZIR METOTLAR

Programlama dili kütüphaneleri içinde önceden tanımlanmış ve programcıların işlerini kolaylaştıran birtakım hazır metotlar vardır.

2.1. Metinsel (String) Metotları

Programlama dili içindeki string sınıfı altında bulunan ve metinsel (String) ifadelerle ilgili bir takım işlemleri daha kolay yapabilmek için bir takım hazır metotlar vardır.

Metinsel metotlardan sık kullanılanlar şunlardır:

- String sınıfı ile çağrılan metotlar; Compare, Concat, Copy, Format, IsNullOrEmpty.
- String ifade ile birlikte çağrılan metotlar; CompareTo, Contains, CopyTo, EndsWith, IndexOf, Insert, LastIndexOf, PadLeft, PadRight, Remove, Replace, Split, StartsWith, Substring, ToLower, ToUpper.

2.1.1. Compare()

Parametre olarak verilen iki string ifadeyi karşılaştırır ve geriye int türünde bir veri döndürür. Dönüş değeri sıfır (0) ise iki metin birbirine eşittir. Aksi takdirde parametre olarak verilen metinleri ilk harflerinden itibaren tek tek karşılaştırır ve farklılığın olduğu ilk harflerin alfabetedeki sıralarına göre -1 veya 1 sayı değerlerini döndürür.

➤ Kullanımı

```
int donusDegeri=String.Compare (metin1,metin2) ;
```

Aşağıdaki tabloda metinlerin karşılaştırma durumları ve geri dönüş değerleri verilmektedir.

Durum	Dönüş Değeri
metin1>metin2	1
metin1=metin2	0
metin1<metin2	-1

Burada metinlerin büyüktür/küçüktür karşılaştırmaları, harflerin alfabetik sırasıyla ilgilidir.

HMS1_1

Girilecek Değerler		Geri Dönüş Değeri
metin1	metin2	
Defne	defne	
Defne	DEFNE	
Defne	Deffne	
Defne	Defne	
Defne	defene	

Metin karşılaştırmalarında büyük/küçük harfe dikkat edilsin istenmezse Compare() metodunun bir başka kullanımı olan Compare(metin1,metin2,boolean) formu kullanılmalıdır.

➤ Kullanımı

```
bool buyukKucuk=true;  
int donusDegeri=String.Compare(metin1,metin2,buyukKucuk);
```


Bu kullanımda **bool** türündeki değişkenin değeri **true** ise **Compare()** metodu büyük/küçük harfe bakmasızın iki kelimeyi karşılaştırır. **False** değeri gönderilirse bu durumda karşılaştırma işlemini büyük/küçük harf biçimde gerçekleştirir.

HMS1_2

file:///c:/users/imsiyat/documents/visual studio 2012/Projects/S

```
duyarlilik = true olduğunda dönüş değeri: 0  
duyarlilik = false olduğunda dönüş değeri: 1
```

Fotoğraf 2.1: Compare metodu büyük/küçük harf duyarlılığı

Bu programdaki metin değerlerini değiştirerek farklı büyük küçük harf duyarlılıklarını incelenmelidir.

2.1.2. Concat()

Parametre olarak verilen nesneleri string türünde birbirine peşi sıra ekler ve geriye string türünde bir değer döndüren string metodudur.

➤ Kullanımı

```
string donenMetin=String.Concat (parametre-listesi);
```

HMS2_1

2.1.3. Copy()

Parametre olarak verilen string türündeki metnin bir kopyasını almaya yarayan string metodudur.

➤ Kullanımı

```
string kopyaMetin=String.Copy (metin);
```

1.2.4. Format()

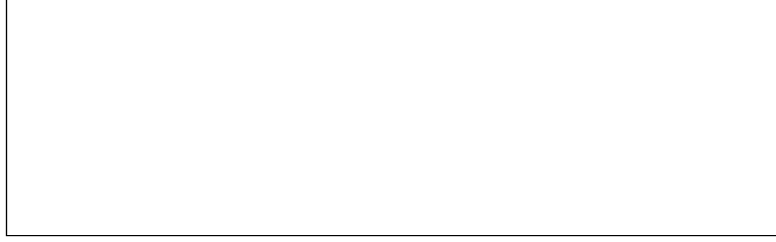
Programlama esnasında bazı ifadelerin belirli bir biçim içinde yazılması istenirse **String.Format()** metodu kullanılır.

Bu metot geriye **string** türünde bir veri döndürür.

Örneğin **metinsel ifadelerin** belirli bir biçim içinde ekranda yazılmasını istiyorsak aşağıdaki gibi yazılmalıdır.

12

```
String.Format("{0,5}",degisken);
```



Şekil 2.1: String.Format() metodu kullanımı

Yukarıdaki şekil incelendiğinde String.Format() metodunun kullanımında küme parantezleri ({ }) içindeki ilk değer, **degisken** isimli değişken içindeki değeri referans gösterir. İkinci değer ise değişkenin içeriğinin ekranda kaç karakterlik alan kaplayacağını (Bu değer 5 ise ekranda 6 karakterlik, -7 ise 8 karakterlik yer kaplar.) belirtir. Bu değer pozitif olması değişken değerinin ayrılan alanının sağına hizalı olacağını, negatif olması ise soluna hizalı olacağını belirler.

HMS3_1

file:///c:/users/imsiyat/documents/visual studio 2

Sıra No	Adınız	Soyad
1	Sevcan	Kudret
2	Murat	İmsiyat
3	Özden	Durmaz

Fotoğraf 2.2: String.Format() metodu ile metin biçimleme

Örneğin **int türündeki sayısal ifadelerin** belirli bir biçim içinde ekranda yazılması istenirse

```
String.Format("{0:00000}", 15); // "00015"
```

ifadesiyle 15 sayısı ekrana başına 3 adet 0 eklenerek toplamda 5 karakter olarak yazılır.

```
String.Format("{0:00000}", -15); // "-00015"
```

ifadesiyle -15 sayısı ekrana başına 3 adet 0 eklenerek toplamda 5 karakter olarak yazılır.

```
String.Format("{0,5}", 15); // " 15"
```

ifadesiyle 15 sayısı ekrana başına 3 adet boşluk eklenerek toplamda 5 karakterlik bir alana sağa hizalı olarak yazılır.

```
String.Format("{0,-5}", 15); // "15  "
```

ifadesiyle 15 sayısı ekrana başına 3 adet boşluk eklenerek toplamda 5 karakterlik bir alana sola hizalı olarak yazılır.

```
String.Format("{0,5:000}", 15); // " 015"
```

ifadesiyle 15 sayısı ekrana başına bir adet 0 ve iki adet boşluk eklenerek toplamda 5 karakterlik bir alana sağa hizalı olarak yazılır.

```
String.Format("{0,-5:000}", 15); // "015  "
```

ifadesiyle 15 sayısı ekrana başına bir adet 0 ve iki adet boşluk eklenerek toplamda 5 karakterlik bir alana sola hizalı olarak yazılır.

String.Format() metodunun sıfır ve negatif sayılar için özel formları vardır. Sayılar biçimlenirken kullanılan noktalı virgül (;) ile format üç bölüme ayrılır. Buradaki ilk bölüm sayının değerini, ikinci bölüm negatif sayıların biçimini,

üçüncü bölüm ise sıfırın ekrana nasıl yazılacağını biçimini belirlemeye yardımcı olur.

```
String.Format("{0:#;eksi #}", 15); // "15"  
String.Format("{0:#;eksi #}", -15); // "eksi 15"  
String.Format("{0:#;eksi #;Sıfır}", 0); // "Sıfır"
```

Sayılar isteğe bağlı biçimlendirilmek istenirse (*örneğin bir telefon numarası; alan kodu ve telefon numarası ayrı ayrı yazılsın istenirse*) biçimlendirme işleminde diyez (#) işareti ile format belirlenir.

```
String.Format("{0:### ## ##}", 1234567); // 123 45 67  
String.Format("{0:(#) ###-##-##}", 12345678); // (1) 234-56-78
```

Örneğin **double türündeki sayısal ifadelerin** belirli bir biçim içinde ekranda yazılması istenirse

Ondalıkli sayılarda virgülden (programlamada nokta) sonra kaç basamak görünsün istenirse köşeli parantezler ({ }) içindeki biçimleme kısmında noktadan sonra o kadar sıfır (0) koymak gerekir.

```
String.Format("{0:0.00}", 123.4567); // "123.46"  
String.Format("{0:0.00}", 123.4); // "123.40"  
String.Format("{0:0.00}", 123); // "123.00"
```

Ondalıkli sayının en fazla kaç basamağının ekranda çıkması istenirse bu sefer sıfır yerine o kadar sayıda diyez (#) işareti kullanmak gerekir.

```
String.Format("{0:0.##}", 123.4567); // "123.46"  
String.Format("{0:0.##}", 123.4); // "123.4"  
String.Format("{0:0.##}", 123); // "123"
```

Ondalıkli sayılarda virgülden önce kaç basamak görüntülemek istenirse biçimlendirme yaparken noktadan önce kaç basamak istenirse o kadar sıfır (0) kullanmak gerekir.

```
String.Format("{0:000.0}", 123.4567); // "123.5"  
String.Format("{0:000.0}", 23.4567); // "023.5"  
String.Format("{0:00.00}", 3.4567); // "03.46"  
String.Format("{0:00.00}", -3.4567); // "-03.46"
```

Sayıların görüntülenmesinde bin ayracı kullanılmak istenirse

```
String.Format("{0:0,0.0}", 12345.678); // "12,345.7"  
String.Format("{0:0,0.00}", 12345.678); // "12,345.68"  
String.Format("{0:0,0}", 12345.678); // "12,346"
```

0 ile 1 arasındaki ondalıklı sayıların gösterimi iki şekilde olur. Birincisinde sayının tam kısmı 0 ve noktadan sonra ondalıklı kısmı gelir (*0.123 şeklinde*). Bir diğer gösterim şeklinde ise sayının tam kısmı yazılmaz sadece nokta ve sonrasındaki ondalıklı kısım yazılır (*.123 şeklinde*).

Bu durumdaki sayıların gösterimi şu şekilde gerçekleştirilir:

```
String.Format("{0:0.0}", 0.0); // "0.0"  
String.Format("{0:0.#}", 0.0); // "0"  
String.Format("{0:#.0}", 0.0); // ".0"  
String.Format("{0:#.}", 0.0); // ""
```

Bütün bu formların dışında sayıların aşağıdaki gibi istenen metinler ile birlikte yazmak da mümkündür.

```
String.Format("{0:sonuç 0.0}", 12.3); // "sonuç 12.3"  
String.Format("{0:x0x.yy0yy}", 12.3); // "x12x.yy3yy"
```

Örneğin **tarih/saat türündeki** ifadelerin belirli bir biçim içinde ekranda yazılmasını istenirse

Tarih/Zaman ifadelerini belirtmek için önceden belirlenmiş bazı anahtar harfler vardır. Bunlar:

y: Yıl, **M:** Ay

h: 12'lik sistemde saat

s: Saniye

d: Gün

H: 24'lük sistemde saat

f: Salise

m: Dakika

z: Zaman dilimi

HMS3_2

file:///c:/users/imsiyat/documents/visual studio 2012/Projects/

```
Tarih:19.03.2017 18:05:07
-----
Yıl gösterimleri:      17 17 2017 2017
Ay gösterimleri:      3 03 Mar Mart
Gün gösterimleri:     19 19 Paz Pazar
Saat gösterimleri:    6 06 18 18
Dakika gösterimleri:  5 05
Saniye gösterimleri:  7 07
Salise gösterimleri:  1 12 123 1230
Zaman dilimi gösterimleri: +2 +02 +02:00
```

Fotoğraf 2-3: Tarih/Zaman gösterim biçimleri -1

Tarih/Zaman gösterimlerinin biçimlendirilmesinde kullanılan bir diğer yol da daha önceden tanımlanmış belirteçler ile değerlerin ekrana yazdırılmasıdır. Bu değerler aşağıdaki tabloda verilmiştir.

Belirteç	Tarih/Zaman Özelliği	Gösterim Örneği
t	Kısa zaman gösterimi	h:mm
d	Kısa tarih gösterimi	M/d/yyyy
T	Uzun zaman gösterimi	h:mm:ss
D	Uzun tarih gösterimi	dddd, MMMM dd, yyyy
f	D ve t' nin birleşimi	dddd, MMMM dd, yyyy h:mm
F	D ve T'nin birleşimi	dddd, MMMM dd, yyyy h:mm:ss
g	d ve t'nin birleşimi	M/d/yyyy h:mm
G	d ve T'nin birleşimi	M/d/yyyy h:mm:ss
m, M	Ay ve gün gösterimi	MMMM dd
y, Y	Yıl ve ay gösterimi	MMMM, yyyy

HMS3_3

file:///c:/users/imsiyat/documents/visual studio 2012/Projects/StringKarsilastirma/StringKa

```
Tarih:19.03.2017 18:05:07
-----
Kısa Zaman Gösterimi: 18:05
Kısa Tarih Gösterimi: 19.03.2017
Uzun Zaman Gösterimi: 18:05:07
Uzun Tarih Gösterimi: 19 Mart 2017 Pazar
Uzun Tarih ve Kısa Zaman Birleşimi: 19 Mart 2017 Pazar 18:05
Full Tarih ve Zaman Gösterimi: 19 Mart 2017 Pazar 18:05:07
Kısa Tarih ve Kısa Zaman Birleşimi: 19.03.2017 18:05
Kısa Tarih ve Uzun Zaman Birleşimi: 19.03.2017 18:05:07
Ay ve Gün Gösterimi: 19 Mart
Ay ve Gün Gösterimi: 19 Mart
Yıl ve Ay Gösterimi: Mart 2017
Yıl ve Ay Gösterimi: Mart 2017
```

2.1.5. IsNullOrEmpty()

Parametre olarak verilen string türündeki değişkenin içeriğinin boş olup olmadığını kontrol eden metottur. Değişkenin içeriği boşsa geriye bool türünde **true** değeri döndürür. Değişkene herhangi bir değer ataması yapılmışsa geriye **false** değerini döndürür.

➤ Kullanımı

```
string metin="";  
bool sonuc=String.IsNullOrEmpty(metin);
```

HMS4_1

Bu kısımdan sonra anlatılacak olan metinsel metotlar doğrudan string sınıfı üzerinden değil string değişken üzerinden çağrılacak olan metotlardır.

2.1.6. CompareTo ()

Çağırdığı string ifade ile parametre olarak verilen string ifadeyi karşılaştırır ve iki ifade de birbirine eşitse geriye int türünde sıfır (0) değerini döndürür. Aksi takdirde metinleri ilk harflerinden itibaren tek tek karşılaştırır ve farklılığın olduğu ilk harflerin alfabetik sıralarına göre -1 veya 1 sayı değerlerini döndürür.

Kullanımı ve çalışma prensibi daha önce anlatılan String.Compare() metoduyla hemen hemen aynıdır.

➤ Kullanımı

```
int donusDegeri=metin1.CompareTo(metin2);
```

Aşağıdaki tabloda metinlerin karşılaştırma durumları ve geri dönüş değerleri verilmiştir.

Durum	Dönüş Değeri
metin1>metin2	1
metin1=metin2	0
metin1<metin2	-1

Metinlerin büyüktür/küçüktür karşılaştırmaları harflerin alfabetik sırasıyla ilgilidir.

2.1.7. Contains()

Birlikte çağrıldığı metinsel ifade içinde parametre olarak verilen char türündeki karakteri veya yine parametre olarak verilen string türündeki metinsel ifadeyi arar ve geriye bool türünde bir değer döndürür.

➤ Kullanımı

Metinsel ifade içinde karakter arama

```
char karakter= ' ';  
bool donusDegeri=metin1.CompareTo(karakter);
```

Metinsel ifade içinde string arama

```
string aranan= "";  
bool donusDegeri=metin1.CompareTo(aranan);
```

HMS5_1

HMS5_2

2.1.8. CopyTo()

Bu metot kaynakBaslangicIndexi (int türünde), hedefDizisi (char dizisi türünde), hedefBaslangicIndexi (int türünde) ve miktar (int türünde) olmak üzere dört parametre alır.

Birlikte çağrıldığı metinsel ifadenin;

- Parametre olarak verilen int türündeki kaynak başlangıç indeksinden itibaren,
- Parametre olarak verilen char[] dizisinin içine,
- Parametre olarak verilen hedef başlangıç indeksinden itibaren,
- Parametre olarak verilen sayıda karakteri kopyalamaya yarayan metottur.

➤ Kullanımı

```
string metin1= "";  
char[] hedefDizisi={,,,,,} ;  
int kayBasInd, hedBasInd, adet;  
metin1.CopyTo(kayBasInd,hedefDizisi,hedBasInd,adet);
```

2.1.9. EndsWith()

Birlikte çağrıldığı metinsel ifade parametre olarak verilen string türündeki ifade ile bitip bitmediğini kontrol eden metottur. Geriye bool türünde bir değer döndürür. Metin parametre olarak verilen ifade ile bitiyorsa geriye **true** değerini döndürür. Metin parametre olarak verilen ifade ile bitmiyorsa geriye **false** değerini döndürür.

➤ Kullanımı

```
metin.EndsWith(ifade);
```

HMS6_1

2.1.10. IndexOf()

Bu metodun birden fazla kullanım şekli vardır.

2.1.10.1. IndexOf(char)

Birlikte çağrıldığı metinsel ifade içinde parametre olarak verilen karakteri arar ve geriye bu karakterin metin içinde **ilk** bulunduğu karakter sırasını döndürür. Metnin ilk karakterinin indeks numarasının sıfır (0) olduğu unutulmamalıdır.

Aranan karakter kelime içinde bulunamazsa geriye -1 değeri döndürür.

Bu metot büyük/küçük harf duyarlı olduğu için aranan karakterin büyük/küçük olma durumlarına dikkat edilmelidir.

➤ Kullanımı

```
int indeks=metin.IndexOf(char);
```

Örnek: IndexOf(char) metodunun kullanımı

```
string metin = "Bilişim teknolojileri";  
Console.WriteLine(metin.IndexOf('T')); // -1  
Console.WriteLine(metin.IndexOf('e')); // 9  
Console.WriteLine(metin.IndexOf('i')); // 1  
Console.WriteLine(metin.IndexOf('z')); // -1
```

2.1.10.2. IndexOf(string)

Birlikte çağrıldığı metinsel ifade içinde parametre olarak verilen string ifadeyi arar ve geriye bu ifadenin metin içinde ilk bulunduğu karakter sırasını döndürür.

Aranan ifade metin içinde bulunamazsa geriye -1 değeri döndürür.

Bu metot büyük/küçük harf duyarlı olduğu için aranan ifadenin büyük/küçük olma durumlarına dikkat edilmelidir.

➤ **Kullanımı**

```
int indeks=metin.IndexOf(string);
```

Örnek: IndexOf(string) metodunun kullanımı

```
string metin = "Bilişim teknolojileri";  
Console.WriteLine(metin.IndexOf("bilisim")); // -1  
Console.WriteLine(metin.IndexOf("Bilişim")); // 0  
Console.WriteLine(metin.IndexOf("loji")); // 13  
Console.WriteLine(metin.IndexOf("Bilisim")); // -1
```

2.1.10.3. IndexOf(char deger, int baslangic)

Birlikte çağırıldığı metinsel ifade içinde parametre olarak verilen karakteri yine parametre olarak verilen başlangıç indeksinden başlayarak arar. Geriye bu ifadenin metin içinde başlangıç indeksinden sonra ilk bulunduğu karakter sırasını döndürür.

Aranan ifade metin içinde bulunamazsa geriye -1 değeri döndürür.

Bu metot büyük/küçük harf duyarlı olduğu için aranan ifadenin büyük/küçük olma durumlarına dikkat edilmelidir.

➤ Kullanımı

```
int indeks=metin.IndexOf(char deger,int baslangic);
```

Örnek: IndexOf(char deger, int baslangic) metodunun kullanımı

```
string metin = "Bilişim teknolojileri";  
Console.WriteLine(metin.IndexOf('T',3)); // -1  
Console.WriteLine(metin.IndexOf('e',10)); // 18  
Console.WriteLine(metin.IndexOf('i',4)); // 5  
Console.WriteLine(metin.IndexOf('o',18)); // -1
```

2.10.1.4. IndexOf(string deger, int baslangic)

Birlikte çağırıldığı metinsel ifade içinde parametre olarak verilen metinsel ifadeyi yine parametre olarak verilen başlangıç indeksinden başlayarak arar. Geriye bu ifadenin metin içinde başlangıç indeksinden sonra ilk bulunduğu karakter sırasını döndürür.

Aranan ifade metin içerisinde bulunamazsa geriye -1 değeri döndürür.

Bu metot büyük/küçük harf duyarlı olduğu için aranan ifadenin büyük/küçük olma durumlarına dikkat edilmelidir.

➤ Kullanımı

```
int indeks=metin.IndexOf(string deger,int baslangic);
```

Örnek: IndexOf(string deger, int baslangic) metodunun kullanımı

```
string metin = "Bilişim teknolojileri";  
Console.WriteLine(metin.IndexOf("bilişim",0)); // -1  
Console.WriteLine(metin.IndexOf("Bilişim",1)); // -1  
Console.WriteLine(metin.IndexOf("loji",3)); // 13  
Console.WriteLine(metin.IndexOf("il",2)); // 16
```

2.1.11. Insert(int baslangic, string value)

Parametre olarak verilen **int** türündeki başlangıç indeksinden başlayarak yine parametre olarak verilen metinsel ifadeyi çağırıldığı metnin içine eklemeye yarayan metottur. Geriye **string** türünde metinsel bir ifade döndürür.

➤ Kullanımı

```
string yeniMetin=metin.Insert(int baslangic, string eklenecek);
```

Örnek: Insert(int baslangic, string deger) metodunun kullanımı

```
string metin = "Elektrik teknolojileri";
string yeniMetin=metin.Insert(9, "ve Elektronik ");
Console.Write(yeniMetin);
```

2.1.12. LastIndexOf()

Bu metodun da IndexOf metodu gibi birden fazla kullanım şekli vardır.

2.1.12.1. LastIndexOf(char)

Birlikte çağırıldığı metinsel ifade içinde parametre olarak verilen karakteri arar ve geriye bu karakterin metin içinde **son** bulunduğu karakter sırasını döndürür. Metnin ilk karakterinin indeks numarasının sıfır (0) olduğu unutulmamalıdır.

Aranan karakter kelime içinde bulunamazsa geriye -1 değeri döndürür.

Bu metot büyük/küçük harf duyarlı olduğu için aranan karakterin büyük/küçük olma durumlarına dikkat edilmelidir.

➤ Kullanımı

```
int indeks=metin.LastIndexOf(char);
```

Örnek: LastIndexOf(char) metodunun kullanımı

```
string metin = "Bilişim teknolojileri";
Console.WriteLine(metin.LastIndexOf('T')); // -1
Console.WriteLine(metin.LastIndexOf('e')); // 18
Console.WriteLine(metin.LastIndexOf('i')); // 20
Console.WriteLine(metin.LastIndexOf('z')); // -1
```

2.1.12.2. LastIndexOf(string)

Birlikte çağırıldığı metinsel ifade içinde parametre olarak verilen string ifadeyi arar ve geriye bu ifadenin metin içinde son bulunduğu karakter sırasını döndürür.

Aranan ifade metin içinde bulunamazsa geriye -1 değeri döndürür.

Bu metot büyük/küçük harf duyarlı olduğu için aranan ifadenin büyük/küçük olma durumlarına dikkat edilmelidir.

➤ Kullanımı

```
int indeks=metin.LastIndexOf(string);
```

Örnek: LastIndexOf(string) metodunun kullanımı

```
string metin = "Bilişim teknolojileri";
```

```

Console.WriteLine(metin.LastIndexOf("bilisim")); // -1
Console.WriteLine(metin.LastIndexOf("Bilişim")); // 0
Console.WriteLine(metin.LastIndexOf("il")); // 16
Console.WriteLine(metin.LastIndexOf("Bilisim")); // -1

```

2.1.12.3. LastIndexOf (char deger, int baslangic)

Birlikte çağırıldığı metinsel ifade içinde parametre olarak verilen karakteri yine parametre olarak verilen başlangıç indeksinden başlayarak arar ve geriye bu ifadenin metin içinde başlangıç indeksinden sonra son bulunduğu karakter sırasını döndürür.

Aranan ifade metin içinde bulunamazsa geriye -1 değeri döndürür.

Bu metot büyük/küçük harf duyarlı olduğu için aranan ifadenin büyük/küçük olma durumlarına dikkat edilmelidir.

➤ Kullanımı

```
int indeks=metin.LastIndexOf(char deger,int baslangic);
```

Örnek: LastIndexOf(char deger, int baslangic) metodunun kullanımı

```

string metin = "Bilişim teknolojileri";
Console.WriteLine(metin.LastIndexOf('T',3)); // -1
Console.WriteLine(metin.LastIndexOf('e',10)); // 9
Console.WriteLine(metin.LastIndexOf('i',4)); // 13
Console.WriteLine(metin.LastIndexOf('o',18)); // 14

```

2.1.12.4. LastIndexOf(string deger, int baslangic)

Birlikte çağırıldığı metinsel ifade içinde parametre olarak verilen metinsel ifadeyi yine parametre olarak verilen başlangıç indeksinden başlayarak arar ve geriye bu ifadenin metin içinde başlangıç indeksinden sonra son bulunduğu karakter sırasını döndürür.

Aranan ifade metin içinde bulunamazsa geriye -1 değeri döndürür.

Bu metot büyük/küçük harf duyarlı olduğu için aranan ifadenin büyük/küçük olma durumlarına dikkat edilmelidir.

➤ Kullanımı

```
int indeks=metin.LastIndexOf(string deger,int baslangic);
```

Örnek: LastIndexOf(string deger, int baslangic) metodunun kullanımı

```

string metin = "Bilişim teknolojileri";
Console.WriteLine(metin.LastIndexOf("bilisim",0)); // -1

```

```
Console.WriteLine (metin.LastIndexOf ("Bilişim", 1)); // -1
Console.WriteLine (metin.LastIndexOf ("im", 15)); // 5
Console.WriteLine (metin.LastIndexOf ("il", 2)); // 1
```

2.1.13. PadLeft ()

PadLeft metodunun iki farklı kullanımı vardır.

2.1.13.1. PadLeft(int deger)

PadLeft() metodunun bu kullanımında birlikte çağrıldığı metne parametre olarak verilen değer kadar karakterlik bir alan ayırır ve metni sağa hizalanmış yeni bir metinsel ifade geriye döndürür.

Hizalamanın görülebilmesi için parametre olarak verilen değer metnin karakter uzunluğundan fazla olduğuna emin olunmalıdır.

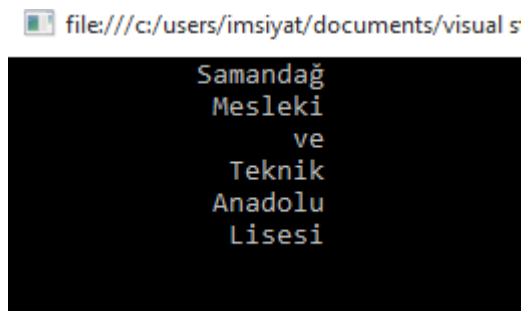
➤ Kullanımı

```
string yeniMetin=metin.PadLeft(int deger);
```

Örnek: PadLeft(int deger) metodunun kullanımı

```
string metin = "Samandağ";
string metin2 = "Mesleki";
string metin3 = "ve";
string metin4 = "Teknik";
string metin5 = "Anadolu";
string metin6 = "Lisesi";
Console.WriteLine (metin.PadLeft (20));
Console.WriteLine (metin2.PadLeft (20));
Console.WriteLine (metin3.PadLeft (20));
Console.WriteLine (metin4.PadLeft (20));
Console.WriteLine (metin5.PadLeft (20));
Console.WriteLine (metin6.PadLeft (20));
```

Yukarıdaki kod parçası çalıştırıldığında aşağıdaki gibi bir ekran görüntüsüyle karşılaşılır.



Fotoğraf 2.5: PadLeft(int deger) metodunun kullanımı

2.1.13.2. PadLeft(int deger, char karakter)

PadLeft() metodunun bu kullanımında da bir önceki kullanımda olduğu gibi birlikte çağrıldığı metne parametre olarak verilen değer kadar karakterlik bir alan ayırır ve metni sağa hizalanmış yeni bir metinsel ifade geriye döndürür. Ancak bir önceki kullanımda metin hizalanırken metnin sol tarafı boşluk karakteriyle doldurulur. Bu kullanımda parametre olarak verilen karakter bu doldurma işlemi için kullanılır.

Hizalamanın görülebilmesi için parametre olarak verilen değer metnin karakter uzunluğundan fazla olduğuna emin olunmalıdır.

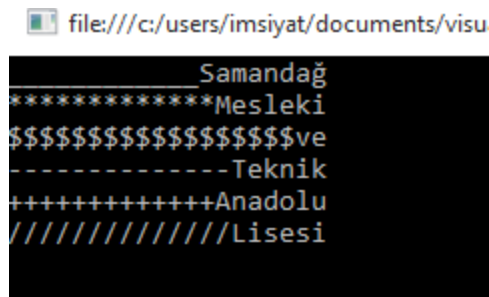
➤ Kullanımı

```
string yeniMetin=metin.PadLeft(int deger,char karakter);
```

Örnek: PadLeft(int deger, char karakter) metodunun kullanımı

```
string metin = "Samandağ";  
string metin2 = "Teknik";  
string metin3 = "ve";  
string metin4 = "Endüstri";  
string metin5 = "Meslek";  
string metin6 = "Lisesi";  
Console.WriteLine(metin.PadLeft(20, '_'));  
Console.WriteLine(metin2.PadLeft(20, '*'));  
Console.WriteLine(metin3.PadLeft(20, '$'));  
Console.WriteLine(metin4.PadLeft(20, '-'));  
Console.WriteLine(metin5.PadLeft(20, '+'));  
Console.WriteLine(metin6.PadLeft(20, '/'));
```

Yukarıdaki kod parçası çalıştırıldığında aşağıdaki gibi bir ekran görüntüsüyle karşılaşılır.



Fotoğraf 2.6: PadLeft(int deger, char karakter) metodunun kullanımı

Yukarıdaki fotoğrafta da görüleceği üzere hizalama işleminde boşlukların doldurulması çeşitlik karakterler ile gerçekleştirilmiştir.

2.1.14. PadRight()

PadRight metodunun iki farklı kullanımı vardır.

2.1.14.1. PadRight (int deger)

PadRight() metodunun bu kullanımında birlikte çağrıldığı metne parametre olarak verilen değer kadar karakterlik bir alan ayırır ve metni sola hizalanmış yeni bir metinsel ifade geriye döndürür.

PadLeft() metodunun kullanımıyla tamamen aynı olan bu metot ile metinsel ifade bu kez sola hizalı olarak yeni bir metinsel ifade geriye döndürür.

Hizalamanın görülebilmesi için parametre olarak verilen değer metnin karakter uzunluğundan fazla olduğuna emin olunmalıdır.

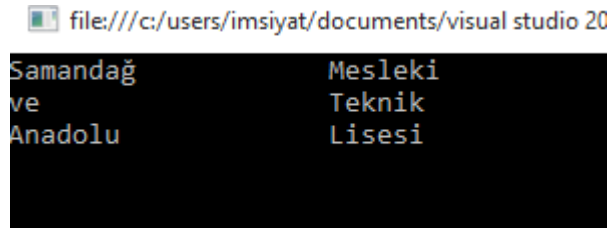
➤ Kullanımı

```
string yeniMetin=metin.PadRight(int deger);
```

Örnek: PadRight(int deger) metodunun kullanımı

```
string metin = "Samandağ";  
string metin2 = "Mesleki";  
string metin3 = "ve";  
string metin4 = "Teknik";  
string metin5 = "Anadolu";  
string metin6 = "Lisesi";  
Console.WriteLine(metin.PadRight(20) + metin2.PadRight(20));  
Console.WriteLine(metin3.PadRight(20) + metin4.PadRight(20));  
Console.WriteLine(metin5.PadRight(20) + metin6.PadRight(20));
```

Yukarıdaki kod parçası çalıştırıldığında aşağıdaki gibi bir ekran görüntüsüyle karşılaşılır.



Fotoğraf 2.7: PadRight(int deger) metodunun kullanımı

2.1.14.2. PadRight(int deger, char karakter)

PadRight() metodunun bu kullanımında da bir önceki kullanımda olduğu gibi birlikte çağrıldığı metne parametre olarak verilen değer kadar karakterlik bir alan ayırır ve metni sola hizalanmış yeni bir metinsel ifade geriye döndürür.

Ancak bir önceki kullanımda metin hizalanırken metnin sağ tarafı boşluk karakteriyle doldurulur. Bu kullanımda parametre olarak verilen karakter bu doldurma işlemi için de kullanılır.

Hizalamanın görülebilmesi için parametre olarak verilen değer metnin karakter uzunluğundan fazla olduğuna emin olunmalıdır.

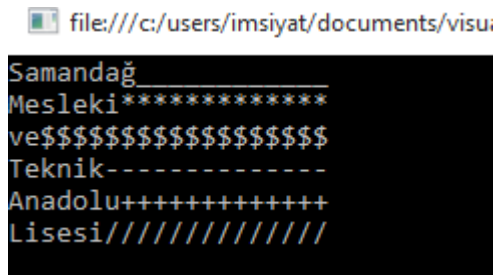
➤ Kullanımı

```
string yeniMetin=metin.PadRight(int deger,char karakter);
```

Örnek: PadRight(int deger, char karakter) metodunun kullanımı

```
string metin = "Samandağ";  
string metin2 = "Mesleki";  
string metin3 = "ve";  
string metin4 = "Teknik";  
string metin5 = "Anadolu";  
string metin6 = "Lisesi";  
Console.WriteLine(metin.PadRight(20, ' '));  
Console.WriteLine(metin2.PadRight(20, '*'));  
Console.WriteLine(metin3.PadRight(20, '$'));  
Console.WriteLine(metin4.PadRight(20, '-'));  
Console.WriteLine(metin5.PadRight(20, '+'));  
Console.WriteLine(metin6.PadRight(20, '/'));
```

Yukarıdaki kod parçası çalıştırıldığında aşağıdaki gibi bir ekran görüntüsüyle karşılaşılır.



Fotoğraf 2.8: PadRight(int deger, char karakter) metodunun kullanımı

Yukarıdaki fotoğrafta da görüleceği üzere hizalama işleminde boşlukların doldurulması çeşitli karakterler ile gerçekleştirilmiştir.

2.1.15. Remove ()

Remove metodunun iki farklı kullanımı vardır.

2.1.15.1. Remove (int deger)

Birlikte çağrıldığı metnin parametre olarak verilen değerinin bulunduğu indeks değerinden itibaren sonuna kadar olan kısmını siler. Silinme işleminden arta kalan metni geriye döndürür.

➤ Kullanımı

```
string yeniMetin=metin.Remove(int deger);
```

Örnek: Remove(int deger) metodunun kullanımı

```
string metin = "Samandağ";  
Console.WriteLine(metin.Remove(3));  
Console.WriteLine(metin.Remove(2));  
Console.WriteLine(metin.Remove(5));
```

2.1.15.2. Remove (int deger, int adet)

Birlikte çağrıldığı metnin parametre olarak verilen değerinin bulunduğu indeks değerinden itibaren yine parametre olarak verilen adet kadar olan kısmını siler. Silinme işleminden arta kalan metni geriye döndürür.


➤ Kullanımı

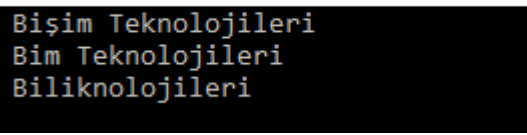
```
string yeniMetin=metin.PadRight(int deger,int adet);
```

Örnek: Remove(int deger, int adet) metodunun kullanımı

```
string metin = "Bilişim Teknolojileri";  
Console.WriteLine(metin.Remove(3, 2));  
Console.WriteLine(metin.Remove(2, 4));  
Console.WriteLine(metin.Remove(5, 6));
```

Yukarıdaki kod parçası çalıştırıldığında aşağıdaki gibi bir ekran görüntüsüyle karşılaşılır.

 file:///c:/users/imsiyat/documents/visual stu



```
Bişim Teknolojileri  
Bim Teknolojileri  
Biliknolojileri
```

Fotoğraf 2.9: Remove(int deger, int adet) metodu ile metin kırpma işlemi

2.1.16. Replace ()

Replace metodunun iki farklı kullanımı vardır.

2.1.16.1. Replace (char eski, char yeni)

Birlikte çağrıldığı metin içinde ilk parametredeki karakterleri ikinci parametredeki karakter değerleriyle değiştiren metottur. Geriye değiştirme işleminin gerçekleştirildiği string türünde bir ifade döndürür.

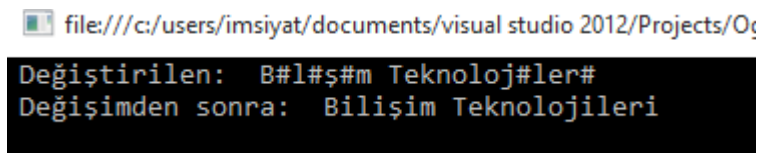
➤ Kullanımı

```
string yeniMetin=metin.Replace(char eski,char yeni);
```

Örnek: Replace(int deger, int adet) metodunun kullanımı

```
string metin = "Bilişim Teknolojileri";  
Console.WriteLine("Değiştirilen: "+metin.Replace('i', '#'));  
Console.WriteLine("Değişimden sonra: "+metin);
```

Yukarıdaki kod parçası çalıştırıldığında aşağıdaki gibi bir ekran görüntüsüyle karşılaşılır.



file:///c:/users/imsiyat/documents/visual studio 2012/Projects/Oğ
Değiştirilen: B#l#ş#m Teknoloj#ler#
Değişimden sonra: Bilişim Teknolojileri

Fotoğraf 2.10: Replace(char eski, char yeni) metodu ile karakter değiştirme

2.1.16.2. Replace (string eski, string yeni)

Replace metodunun bir önceki kullanımından farkı parametre olarak bu kez char türünde karakterlerin yeni string türünde metinsel ifade almasıdır.

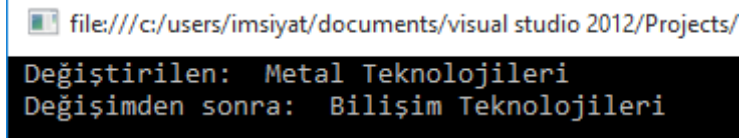
➤ Kullanımı

```
string yeniMetin=metin.Replace(string eski, string yeni);
```

Örnek: Replace(int deger, int adet) metodunun kullanımı

```
string metin = "Bilişim Teknolojileri";  
Console.WriteLine("Değiştirilen: "+metin.Replace("Bilişim",  
"Metal"));  
Console.WriteLine("Değişimden sonra: "+metin);
```

Yukarıdaki kod parçası çalıştırıldığında aşağıdaki gibi bir ekran görüntüsüyle karşılaşılır.



Fotoğraf 2.11: Replace(string eski, string yeni) metodu ile metin değiştirme

2.1.17. Split ()

Split () metodu, çağrıldığı metni istenilen karakterden itibaren parçalara bölmek için kullanılan bir metottur. İstenilen karakter mevcut metin ifadesi içerisinde yer alıyorsa Split () metodu metni karakterlerden öncesi ve sonrası şeklinde parçalara ayırır ve bu parçaları string türünde bir dizi içerisinde saklar. Geriye de bu string[] türündeki diziyi döndürür.

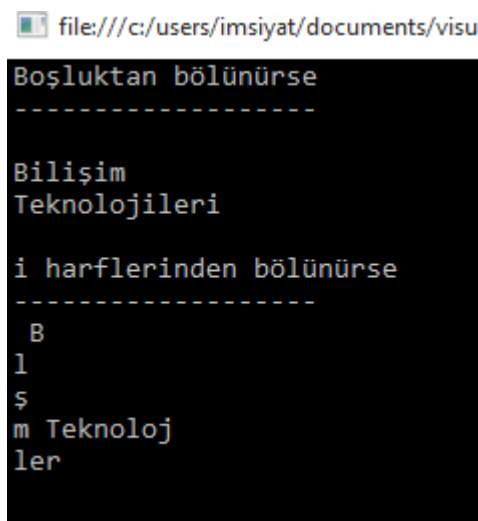
➤ Kullanımı

```
string[] dizi=metin.Split(char karakter);
```

Örnek: Split(char karakter) metodunun kullanımı

```
string metin = "Bilişim Teknolojileri";  
Console.WriteLine("Boşluktan bölünürse");  
Console.WriteLine("-----");  
foreach(string harf in metin.Split(' '))  
    Console.WriteLine(harf);  
Console.WriteLine(" ");  
Console.WriteLine("i harflerinden bölünürse");  
Console.WriteLine("-----");  
foreach (string harf in metin.Split('i'))  
    Console.WriteLine(harf);
```

Yukarıdaki kod parçası çalıştırıldığında aşağıdaki gibi bir ekran görüntüsüyle karşılaşılır.



Fotoğraf 2.12: Split(char karakter) metodu ile metin parçalama

2.1.18. StartsWith ()

Birlikte çağrıldığı metinsel ifade parametre olarak verilen string türündeki ifade ile başlayıp başlamadığını kontrol eden metottur. Geriye bool türünde bir değer döndürür. Metin parametre olarak verilen ifade ile başlıyorsa geriye **true** değerini döndürür. Metin parametre olarak verilen ifade ile başlamıyorsa geriye **false** değerini döndürür.

➤ Kullanımı

```
metin.StartsWith(ifade);
```

Örnek: Klavyeden girilen kullanıcı adının rakamla başlayıp başlamadığını kontrol eden, kullanıcı adı rakam ile başlıyorsa uyarı mesajı veren bir metot tanımlayınız.

```
static void Main(string[] args)
{
    Console.Write("Kullanıcı adı belirleyiniz: ");
    string kAdi= Console.ReadLine();
    if (KullaniciAdiKontrol(kAdi))
        Console.WriteLine("Kullanıcı adı tanımınız başarılı");
    else
        Console.WriteLine("Kullanıcı adı sayı ile başlayamaz");
}

static bool KullaniciAdiKontrol(string kAdi)
{
    if (kAdi.StartsWith("1"))
        return false;
    else if (kAdi.StartsWith("2"))
        return false;
    else if (kAdi.StartsWith("3"))
        return false;
    else if (kAdi.StartsWith("4"))
        return false;
    else if (kAdi.StartsWith("5"))
        return false;
    else if (kAdi.StartsWith("6"))
        return false;
    else if (kAdi.StartsWith("7"))
        return false;
    else if (kAdi.StartsWith("8"))
        return false;
    else if (kAdi.StartsWith("9"))
        return false;
    else if (kAdi.StartsWith("0"))
        return false;
    else
        return true;
}
```

Yukarıdaki uygulama çeşitli kelime girişleri ile denenerek pekiştirilmelidir.

2.1.19. Substring ()

Substring() metodunun iki kullanımı vardır.

2.1.19.1. Substring (int indeks)

Birlikte çağrıldığı metni parametre olarak verilen indeks değerinden itibaren kesen ve arta kalan metni geriye string türünde döndüren metottur.

➤ Kullanımı

```
string yeniMetin=metin.Substring(int indeks);
```

Örnek: Substring(int indeks) metodunun kullanımı

```
string metin = "Bilişim Teknolojileri";  
Console.WriteLine(metin.Substring(3)); //işim Teknolojileri  
Console.WriteLine(metin.Substring(8)); //Teknolojileri  
Console.WriteLine(metin.Substring(14)); //ojileri
```

2.1.19.2. Substring (int indeks, int uzunluk)

Substring metodunun bu kullanımında ilk parametre indeks değerini, ikinci parametre ise kaç karakter uzunluğunda bir metnin kesileceğini belirtir.

➤ Kullanımı

```
string yeniMetin=metin.Substring(int indeks,int uzunluk);
```

Örnek: Substring (int indeks, int uzunluk) metodunun kullanımı

```
string metin = "Bilişim Teknolojileri";  
Console.WriteLine(metin.Substring(3,4)); //işim  
Console.WriteLine(metin.Substring(8,3)); //Tek  
Console.WriteLine(metin.Substring(14,1)); //o
```

2.1.20. ToLower ()

Birlikte çağrıldığı metnin tüm karakterlerini küçük harfe dönüştürerek yeni bir metin geriye döndürür.

➤ Kullanımı

```
string yeniMetin=metin.ToLower();
```

Örnek: ToLower() metodunun kullanımı

```
string metin = "Bilişim Teknolojileri";
```

```
Console.WriteLine(metin.ToLower()); //bilisim teknolojileri
string metin2= "sAMaDağ mesLEKİ ve tEKNİK AnaDOLU LİSESİ";
Console.WriteLine(metin2.ToLower());
```

2.1.21. ToUpper ()

ToLower() metodunun tam tersi şeklinde çalışır ve birlikte çağrıldığı metnin tüm karakterlerini büyük harfe dönüştürerek yeni bir metin geriye döndürür.

➤ Kullanımı

```
string yeniMetin=metin.ToUpper();
```

Örnek: ToUpper() metodunun kullanımı

```
string metin = "Bilişim Teknolojileri";
Console.WriteLine(metin.ToUpper()); //BİLİŞİM TEKNOLOJİLERİ
string metin2= " sAMaDağ mesLEKİ ve tEKNİK AnaDOLU LİSESİ";
Console.WriteLine(metin2.ToUpper());
```

2.2. Matematiksel (Math) Metotları

Programlama dili içindeki Math sınıfı altında bulunan ve matematiksel bazı işlem ve fonksiyonları daha kolay yapabilmek için bir takım hazır metotlar vardır.

Matematiksel metotlardan sık kullanılanlar şunlardır:

- Abs	- BigMul	- Ceiling	- DivRem
- Max	- Min	- Pow	- Round
- Sign	- Sqrt	- Cos	- Sin
- Tan	- Acos	- Asin	- Atan

2.2.1. Abs()

Abs() metodu parametre olarak verilen sayının mutlak değerini veren metottur. Parametre olarak farklı sayı türlerinde değerler alabilir ve aldığı değerin türünde bir değer geri döndürür.

➤ Kullanımı

```
int mutlakDeger=Math.Abs(int sayi);
decimal mutlakDeger=Math.Abs(decimal sayi);
double mutlakDeger=Math.Abs(double sayi);
float mutlakDeger=Math.Abs(float sayi);
long mutlakDeger=Math.Abs(long sayi);
short mutlakDeger=Math.Abs(short sayi);
sbyte mutlakDeger=Math.Abs(sbyte sayi);
```

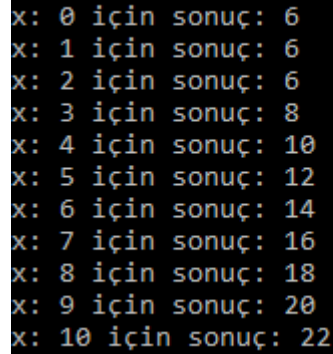
MTM_1

```
int sonuc=0;
for (int x = 0; x <= 10; x++)
{
    sonuc = Math.Abs(x - 2) + 2 + Math.Abs(2 + x);
    Console.WriteLine("x: {0} için sonuç: {1}", x, sonuc);
}
```

Yukarıdaki kod parçası çalıştırıldığında aşağıdaki gibi bir ekran çıktısı ile karşılaşılır.

file:///c:/users/imsiyat/documents/\

2.2.2. BigMul()



```
x: 0 için sonuç: 6
x: 1 için sonuç: 6
x: 2 için sonuç: 6
x: 3 için sonuç: 8
x: 4 için sonuç: 10
x: 5 için sonuç: 12
x: 6 için sonuç: 14
x: 7 için sonuç: 16
x: 8 için sonuç: 18
x: 9 için sonuç: 20
x: 10 için sonuç: 22
```

Fotoğraf 2.13: Math.Abs() metodunun kullanımı

Parametre olarak verilen iki **int** türündeki sayının çarpımını **long** türünde veren metottur.

➤ Kullanımı

```
long sonuc=Math.BigMul(int a, int b);
```

Örnek: Math.BigMul() metodunun kullanımı

```
long sonuc=Math.BigMul(2,4);           // 8
sonuc=Math.BigMul(43,2);               // 86
sonuc=Math.BigMul(9,80);               // 720
sonuc=Math.BigMul(100,10);            // 1000
```

2.2.3. Ceiling()

Parametre olarak verilen **double** türündeki ondalıklı sayıdan büyük en küçük tam sayının değerini veren metottur.

➤ Kullanımı


```
decimal sonuc=Math.Ceiling(decimal sayi1);  
double sonuc2=Math.Ceiling(double sayi2);
```

Örnek: Math.Ceiling() metodunun kullanımı

```
double sayi= 2.00;  
double sonuc=Math.Ceiling(sayi);           // 2  
sayi= 2.01;  
sonuc=Math.Ceiling(sayi);                 // 3  
sayi= 2.50;  
sonuc=Math.Ceiling(sayi);                 // 3  
sayi= 2.99;  
sonuc=Math.Ceiling(sayi);                 // 3
```

2.2.4. DivRem()

Parametre olarak verilen ilk iki sayının bölme işlemini yapan, geriye bölme işleminin sonucunu döndüren ve 3. parametre olarak verilen değişkene de bölme işleminin kalanını aktaran metottur.

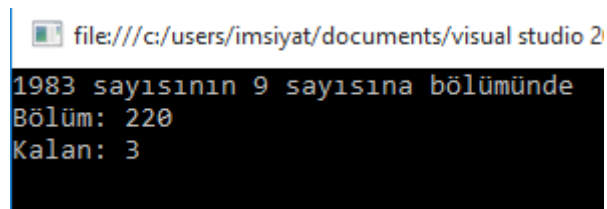
➤ Kullanımı

```
int bolum=Math.DivRem(int bolunen, int bolen,out int kalan);  
long bolum=Math.DivRem(long bolunen,long bolen,out long kalan);
```

Örnek: Math.DivRem() metodu ile bölme işlemi

```
int bolunen = 1981;  
int bolen = 9;  
int kalan = 0;  
int bolum = Math.DivRem(bolunen, bolen, out kalan);  
Console.WriteLine("{0} sayısının {1} sayısına bölümünde",  
bolunen, bolen);  
Console.WriteLine("Bölüm: " +bolum);  
Console.WriteLine("Kalan: " +kalan);
```

Yukarıdaki kod parçası çalıştırıldığında aşağıdaki gibi bir ekran çıktısı ile karşılaşılır.



Fotoğraf 2.14: Math.DivRem() ile bölme işlemi

2.2.5. Max()

Parametre olarak verilen iki sayıdan büyük olanı geriye döndüren metottur. Bütün sayı türleri tarafından desteklenen bir metot çeşididir.

➤ Kullanımı

```
byte maksimum=Math.Max(byte sayi1,byte sayi2);
decimal maksimum=Math.Max(decimal sayi1,decimal sayi2);
double maksimum=Math.Max(double sayi1,double sayi2);
float maksimum=Math.Max(float sayi1,float sayi2);
int maksimum=Math.Max(int sayi1,int sayi2);
long maksimum=Math.Max(long sayi1,long sayi2);
sbyte maksimum=Math.Max(sbyte sayi1,sbyte sayi2);
short maksimum=Math.Max(short sayi1,short sayi2);
unit maksimum=Math.Max(unit sayi1,unit sayi2);
ulong maksimum=Math.Max(ulong sayi1,ulong sayi2);
ushort maksimum=Math.Max(ushort sayi1,ushort sayi2);
```

Örnek: Math.Max() metodu ile klavyeden girilen sayılardan büyüğünü bulma

```
Console.Write("1. sayıyı giriniz:");
int sayi1 = Convert.ToInt32(Console.ReadLine());
Console.Write("2. sayıyı giriniz:");
int sayi2 = Convert.ToInt32(Console.ReadLine());
int maksimum = Math.Max(sayi1, sayi2);
Console.WriteLine(sayi1 + " ve " + sayi2 + " sayılarından en
büyüğü " + maksimum + " sayıdır.");
```

Yukarıdaki kod parçası çalıştırıldığında aşağıdaki gibi bir ekran çıktısı ile karşılaşılır.

 file:///c:/users/imsiyat/documents/visual studio 2012/Projects/Stri

2.2.6. Min()

```
1. sayıyı giriniz:87
2. sayıyı giriniz:63
87 ve 63 sayılarından en büyüğü 87 sayıdır.
```

Fotoğraf 2.15: Math.Max() ile en büyük sayıyı bölme

Parametre olarak verilen iki sayıdan küçük olanı geriye döndüren metottur. Bütün sayı türleri tarafından desteklenen bir metot çeşididir.

➤ Kullanımı

```
byte minimum=Math.Min(byte sayi1,byte sayi2);
decimal minimum=Math.Min(decimal sayi1,decimal sayi2);
double minimum=Math.Min(double sayi1,double sayi2);
float minimum=Math.Min(float sayi1,float sayi2);
int minimum=Math.Min(int sayi1,int sayi2);
```

```

long minimum=Math.Min(long sayi1,long sayi2);
sbyte minimum=Math.Min(sbyte sayi1,sbyte sayi2);
short minimum=Math.Min(short sayi1,short sayi2);
unit minimum=Math.Min(unit sayi1,unit sayi2);
ulong minimum=Math.Min(ulong sayi1,ulong sayi2);
ushort minimum=Math.Min(ushort sayi1,ushort sayi2);

```

Örnek: Math.Min() metodu ile klavyeden girilen sayılardan küçüğünü bulma

```

Console.Write("1. sayıyı giriniz:");
int sayi1 = Convert.ToInt32(Console.ReadLine());
Console.Write("2. sayıyı giriniz:");
int sayi2 = Convert.ToInt32(Console.ReadLine());
int minimum = Math.Min(sayi1, sayi2);
Console.WriteLine(sayi1 + " ve " + sayi2 + " sayılarından en
küçüğü " + minimum + " sayıdır.");

```

Yukarıdaki kod parçası çalıştırıldığında aşağıdaki gibi bir ekran çıktısı ile karşılaşılır.

file:///c:/users/imsiyat/documents/visual studio 2012/Projects/StringKarsilastirm

```

1. sayıyı giriniz:68
2. sayıyı giriniz:78
68 ve 78 sayılarından en küçüğü 68 sayıdır.

```

Fotoğraf 2.16: Math.Min() ile en büyük sayıyı bölme

2.2.7. Pow()

Parametre olarak verilen ilk sayının yine parametre olarak verilen ikinci sayı kadar üssünü hesaplayan metottur.

➤ Kullanımı

```
double usluSayi=Math.Pow(double x, double y);
```

$$x^y = \text{Math.Pow}(x, y);$$

Şekil 2.2: Math.Pow() kullanımı

Örnek: Math.Pow() metodunun kullanımı

```

double usluSayi=Math.Pow(3,3);           // 27
double usluSayi=Math.Pow(2,16);          // 65536
double usluSayi=Math.Pow(12,0);           // 1
double usluSayi=Math.Pow(5,-2);           // 0.04
double usluSayi=Math.Pow(-10,-2);         // 0.01

```

2.2.8. Round()

Parametre olarak verilen sayıyı en yakın tam sayıya yuvarlayan metottur.

➤ Kullanımı

```
double yuvarlanmis=Math.Round(double sayi);
```

Örnek: Math.Round() metodunun kullanımı

```
double yuvarlanmis=Math.Round(3.14);           // 3
double yuvarlanmis=Math.Round(3.499);          // 3
double yuvarlanmis=Math.Round(3.5);             // 4
double yuvarlanmis=Math.Round(3.9999);          // 4
```

2.2.9. Sign()

Parametre olarak verilen sayının işaretini verir. Sayı pozitif ise 1, negatif ise -1, sayı sıfıra eşitse de geriye 0 değerini döndüren metottur.

➤ Kullanımı

```
int isaret=Math.Sign(int sayi);
```

Örnek: Math.Sign() metodunun kullanımı

```
int isaret=Math.Sign(26638);                    // 1
int isaret=Math.Sign(-26638);                   // -1
int isaret=Math.Sign(0);                        // 0
```

2.2.10. Sqrt()

Parametre olarak verilen **double** türündeki sayının karekök değerini **double** türünde geriye döndüren metottur.

➤ Kullanımı

```
double karekok=Math.Sqrt(double sayi);
```

Örnek: Klavyeden iki kenar uzunluğu girilen dik üçgenin hipotenüsünün uzunluğunu hesaplayan programın kodunu yazınız.

```
Console.Write("1. kenar uzunluğunu giriniz: ");
double kenar1= Convert.ToDouble(Console.ReadLine());
Console.Write("2. kenar uzunluğunu giriniz: ");
double kenar2 = Convert.ToDouble(Console.ReadLine());
```

```
double kenarlarinKareToplami = Math.Pow(kenar1, 2) +
Math.Pow(kenar2, 2);
double kenar3= Math.Sqrt(kenarlarinKareToplami);
Console.WriteLine("Hipotenüsün uzunluğu: "+kenar3);
```

Uyarı: Trigonometrik fonksiyonlarda açı değerleri radyan cinsinden verilmelidir.

Derece	0°	30°	45°	60°	90°	180°	270°	360°
Radyan	0	$\frac{\pi}{6}$	$\frac{\pi}{4}$	$\frac{\pi}{3}$	$\frac{\pi}{2}$	π	$\frac{3\pi}{2}$	2π

Trigonometrik metotlara geçilmeden önce derece cinsinden verilen açı değerini radyana dönüştüren ve radyan değeri verilen açının derece cinsinden değerini veren basit metotlar yazılmalı ve bundan sonraki örneklerde de bu metottan faydalanılmalıdır.

Örnek: Derece cinsinden açı değerini radyan cinsinden açı değerine dönüştüren metodun yazımı

```
static double radyanaDonustur(double derece)
{
    double radyan = 0;
    double piSayisi = Math.PI;
    radyan = piSayisi * (derece / 180);
    return radyan;
}
```

Yukarıdaki şekilde yazılan metot aşağıdaki gibi çağrılarak programların içinde kullanılabilir.

```
double radyanDegeri=radyanaDonustur(double aci)
```

Örnek: Radyan cinsinden verilen açı değerini derece cinsinden açı değerine dönüştüren metodun yazımı

```
static double aciyaDonustur(double radyan)
{
    double derece = 0;
    double piSayisi = Math.PI;
    derece = (radyan / piSayisi) * 180;
    return derece;
}
```

Yukarıdaki şekilde yazılan metot aşağıdaki gibi çağrılarak programların içinde kullanılabilir.

```
double dereceDegeri=aciyaDonustur(double radyan)
```

Matematiksel metotların içinde yer alan ve trigonometrik işlemlerde sıkça kullanılan diğer metotlar şunlardır:

2.2.11. Cos()

Parametre olarak verilen radyan açı değerinin kosinüs değerini veren metottur.

➤ Kullanımı

```
double kosinus=Math.Cos(double aci);
```

MTM_11

2.2.12. Sin()

Parametre olarak verilen radyan açı değerinin sinüs değerini veren metottur.

➤ Kullanımı

```
double sinus=Math.Sin(double aci);
```

MTM_12

2.2.13. Tan()

Parametre olarak verilen radyan açı değerinin tanjant değerini veren metottur.

➤ Kullanımı

```
double tanjant=Math.Sin(double aci);
```

MTM_13

2.2.14. Acos()

Parametre olarak verilen kosinüs değerinin radyan açı değerini veren metottur.

➤ Kullanımı

```
double kosinusAcisi=Math.Acos(double kosinus);
```

2.2.15. Asin()

Parametre olarak verilen sinüs değerinin radyan açı değerini veren metottur.

➤ Kullanımı

```
double sinusAcisi=Math.Asin(double sinus);
```

2.2.16. Atan()

Parametre olarak verilen tanjant değerinin radyan açı değerini veren metottur.

➤ Kullanımı

```
double tanjantAcisi=Math.Atan(double tanjant);
```

2.3. Tarih/Saat (DateTime) Metotları

Programlama dili içinde tarih ve zamanlar ile ilgili işlemler yaparken birtakım işleri daha kolay yapabilmek için önceden tanımlanmış Tarih/Zaman metotları kullanılır.

Tarih/Zaman (DateTime) metotlarından sık kullanılanlar şunlardır:

- DateTime sınıfı ile çağrılan metotlar: Compare, DaysInMonth, IsLeapYear, Parse.
- DateTime türünde bir ifade ile birlikte çağrılan metotlar: Subtract, AddDays, AddMonths, AddYears, AddHours, AddMinutes, AddSeconds, AddMilliseconds.

Tarih/Zaman metotlarının ayrıntılarını incelemeden önce DateTime sınıfı altında yer alan ve sıkça kullanılan **üyeleri** tanımak gerekir.

2.3.1. MinValue

Bu özellik sayesinde DateTime yapısı ile kullanılabilen en küçük tarih-saat bilgisine erişilebilir.

➤ Kullanımı

```
DateTime enKucuk=DateTime.MinValue;
```

Bu sabit özellik çağrıldığında geriye veri türü DateTime olan “01.01.0001 00:00:00” değerini döndürür. Bu değer değiştirilemeyen **Salt okunur** bir veridir.

2.3.2. MaxValue

Bu özellik sayesinde DateTime yapısı ile kullanılabilen en büyük tarih-saat bilgisine erişilebilir.

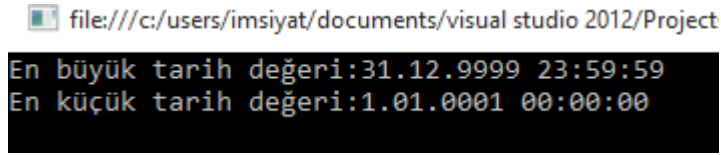
➤ Kullanımı

```
DateTime enBuyuk=DateTime.MaxValue;
```

Bu sabit özellik çağrıldığında geriye veri türü DateTime olan “31.12.9999 23:59:59” değerini döndürür. Bu değer de aynı MinValue gibi değiştirilemeyen **Salt okunur** bir veridir.

TSM_2

Yukarıdaki kodlar çalıştırıldığında aşağıdaki gibi bir ekran görüntüsüyle karşılaşılır.



Fotoğraf 2.17: Sistemdeki en büyük ve en küçük tarihler

2.3.3. Today

Bu özellik sayesinde DateTime yapısı ile birlikte kullanılır ve bugünün tarihini DateTime türünde geri döndürür.

➤ Kullanımı

```
DateTime bugün=DateTime.Today;
```

Programın çalıştırıldığı sistemin tarih değerini gösterir.

2.3.4. Now

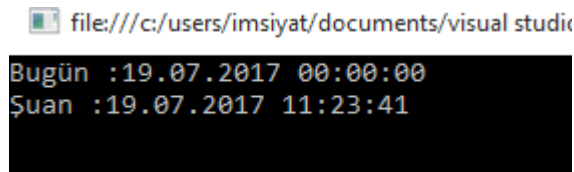
Bu özellik de DateTime yapısı ile birlikte kullanılır ve çağrıldığı anın hem tarih hem de saat bilgisini DateTime türünde geri döndürür.

➤ Kullanımı

```
DateTime simdi=DateTime.Now;
```

TSM_5

Yukarıdaki kodlar çalıştırıldığında aşağıdaki gibi bir ekran görüntüsüyle karşılaşılır.



Fotoğraf 2.18: Today ve now kullanımı

DateTime sınıfından türetilmiş bir nesne ile çalışırken sıkça kullanılacak özellikler aşağıdaki tabloda açıklamalarıyla birlikte verilmiştir.

Özellik Adı	Açıklama	Geri Dönüş Türü
Date	Nesneye ilişkin saat dışındaki bilgiyi verir.	DateTime
Month	Nesnenin ay bilgisini verir.	int
Day	Nesnenin gün bilgisini verir.	int
Year	Nesnenin yıl bilgisini verir.	int
DayOfWeek	Haftanın günü, nesnenin haftanın kaçınıcı günü olduğu bilgisini verir.	DayOfWeek
DayOfYear	Nesnenin yılın kaçınıcı günü olduğu bilgisini verir.	int
TimeOfDay	Nesneye ait saat bilgisini verir.	TimeSpan
Hour	Nesnenin saat bilgisini verir.	int
Minute	Nesnenin dakika bilgisini verir.	int
Second	Nesnenin saniye bilgisini verir.	int
Millisecond	Nesnenin milisaniye bilgisini verir.	int

Örnek: DateTime sınıfı ile kullanılabilen özelliklerin kullanımı ve ekrana yazdırılması

```

DateTime zaman = DateTime.Now;
DateTime tarih = zaman.Date;
int ay=zaman.Month;
int gun=zaman.Day;
int yıl=zaman.Year;
DayOfWeek haftaninGunu=zaman.DayOfWeek;
int yilinKacinciGunu=zaman.DayOfYear;
TimeSpan sure=zaman.TimeOfDay;
int saat=zaman.Hour;
int dakika=zaman.Minute;
int saniye=zaman.Second;
int salise = zaman.Millisecond;

Console.WriteLine("Şuandaki Zaman      : {0}", zaman);
Console.WriteLine("Tarih Bilgisi       : {0}", tarih);
Console.WriteLine("Ay Bilgisi         : {0}", ay);
Console.WriteLine("Gün Bilgisi        : {0}", gun);
Console.WriteLine("Yıl Bilgisi        : {0}", yıl);
Console.WriteLine("Haftanın Günü      : {0}", haftaninGunu);
Console.WriteLine("Yılın Kaçınıcı Günü : {0}",
yilinKacinciGunu);
Console.WriteLine("Süre                : {0}", sure);
Console.WriteLine("Saat                : {0}", saat);
Console.WriteLine("Dakika             : {0}", dakika);
Console.WriteLine("Saniye             : {0}", saniye);
Console.WriteLine("Salise            : {0}", salise);

```

Yukarıdaki kodlar çalıştırıldığında aşağıdaki gibi bir ekran görüntüsüyle karşılaşılır.

```
file:///c:/users/imsiyat/documents/visual studio 2012/Projects/StringK
Şuandaki Zaman      : 19.07.2017 11:24:12
Tarih Bilgisi       : 19.07.2017 00:00:00
Ay Bilgisi          : 7
Gün Bilgisi         : 19
Yıl Bilgisi         : 2017
Haftanın Günü       : Wednesday
Yılın Kaçınıcı Günü : 200
Süre                : 11:24:12.0088904
Saat                : 11
Dakika              : 24
Saniye              : 12
Salise              : 8
```

Fotoğraf 2.19: DateTime nesnesinin özelliklerinin kullanımı

DateTime sınıfından türetilen nesnelerle birlikte kullanılan özelliklerden sonra DateTime sınıfı altında yer alan metotlar incelenecektir.

2.3.5. DateTime.Compare()

Bu metot DateTime türünde parametre olarak verilen iki değeri karşılaştırır ve geriye int türünde bir değer döndürür.

Karşılaştırmada;

- 1. parametredeki tarih değeri, 2. parametredeki tarih değerinden daha eski bir tarih değeri ise sonuç -1,
- 2. parametredeki tarih değeri, 1. parametredeki tarih değerinden daha eski bir tarih değeri ise sonuç 1,
- 1. parametredeki tarih değeri ile 2. parametredeki tarih değeri birbirine eşitse sonuç 0 olarak geriye döner.

➤ Kullanımı

```
int sonuc=DateTime.Compare(DateTime tarih1, DateTime tarih2);
```

DT_1

2.3.6. DateTime.DaysInMonth()

Bu metot **int** türünde parametresi verilen yıl ve ay bilgilerine denk gelen ayın kaç günden oluştuğunu geriye **int** türünde bir değer olarak döndürür.

➤ Kullanımı

```
int gunSayisi=DateTime.Compare(int yil, int ay);
```

```
int gunSayisi = DateTime.DaysInMonth(2008,2);           // 29
int gunSayisi = DateTime.DaysInMonth(2009,8);           // 31
int gunSayisi = DateTime.DaysInMonth(2010,4);           // 30
int gunSayisi = DateTime.DaysInMonth(2011,2);           // 28
```

2.3.7. DateTime.IsLeapYear()

Bu metot **int** türünde parametre olarak verilen yılın **artık yıl** (şubat ayının 29 günden, yılın toplam 366 günden oluştuğu) olup olmadığını kontrol eder. Geriye **bool** türünde bir değer döndürür. Yıl artık yıl ise geriye **true**, değilse **false** değeri döndürür.

➤ Kullanımı

```
bool artikYilMi=DateTime.IsLeapYear(int yil);
```

```
bool artikYilMi=DateTime.IsLeapYear(2008);             // true
bool artikYilMi=DateTime.IsLeapYear(2010);             // false
bool artikYilMi=DateTime.IsLeapYear(2012);             // true
bool artikYilMi=DateTime.IsLeapYear(1996);             // true
```

2.3.8. DateTime.Parse()

Bu metot **string** türünde parametre olarak verilen metni DateTime türündeki tarih ve saat bilgisine dönüştürür.

➤ Kullanımı

```
DateTime tarih=DateTime.Parse(string metin);
```

Örnek: DateTime.Parse() ile metinlerin tarihe dönüştürülmesi

```
string metin1 = "14.08.1990";
string metin2 = "14/08/1990";
string metin3 = "14/Ağustos/1990";
```

```


string metin4 = "14.Şubat.1990 14:53";
string metin5 = "14/08/1990 14:53:05";
string metin6 = "08/1990";
string metin7 = "Ağustos/1990";

DateTime tarih1 = DateTime.Parse(metin1);
DateTime tarih2 = DateTime.Parse(metin2);
DateTime tarih3 = DateTime.Parse(metin3);
DateTime tarih4 = DateTime.Parse(metin4);
DateTime tarih5 = DateTime.Parse(metin5);
DateTime tarih6 = DateTime.Parse(metin6);
DateTime tarih7 = DateTime.Parse(metin7);

Console.WriteLine("{0} ==> : {1}" , metin1, tarih1);
Console.WriteLine("{0} ==> : {1}" , metin2, tarih2);
Console.WriteLine("{0} ==> : {1}" , metin3, tarih3);
Console.WriteLine("{0} ==> : {1}" , metin4, tarih4);
Console.WriteLine("{0} ==> : {1}" , metin5, tarih5);
Console.WriteLine("{0} ==> : {1}" , metin6, tarih6);
Console.WriteLine("{0} ==> : {1}" , metin7, tarih7);

```

Yukarıdaki kod parçası çalıştırıldığında aşağıdaki gibi bir ekran çıktısı ile karşılaşılır.

 file:///c:/users/imsiyat/documents/visual studio 2012/Projects/StringKarsi

```

14.08.1990 ==> : 14.08.1990 00:00:00
14/08/1990 ==> : 14.08.1990 00:00:00
14/Ağustos/1990 ==> : 14.08.1990 00:00:00
14.Şubat.1990 14:53 ==> : 14.02.1990 14:53:00
14/08/1990 14:53:05 ==> : 14.08.1990 14:53:05
08/1990 ==> : 1.08.1990 00:00:00
Ağustos/1990 ==> : 1.08.1990 00:00:00

```

Fotoğraf 2.20: DateTime.Parse() ile metni tarihe dönüştürme

Yukarıdaki örnekte çeşitli biçimlerde metin olarak verilen tarih/saat bilgilerinin nasıl sonuçlar verdiği incelenmelidir.

Bu kısımdan sonra anlatılacak olan tarih/zaman metotları doğrudan DateTime sınıfı üzerinden değil DateTime sınıfından türetilmiş nesneler üzerinden çağrılacak olan metotlardır.

DateTime sınıfından türetilen nesnelerle birlikte kullanılan metotlar şunlardır:

2.3.9. Subtract()

Bu metot ile DateTime türünde türetilmiş bir nesnenin değerinden parametre olarak verilen **DateTime** veya **TimeSpan** türündeki değer çıkartılır ve geriye **TimeSpan** türünde bir değer döndürülür.

➤ Kullanımı

```
TimeSpan yeniTarih=eskiTarih.Subtract(DateTime cikarilanTarih);
```

TimeSpan türünde dönüş yapılan değer üzerinden;

- **TotalDays** özelliği ile toplam gün sayısını,
- **TotalHours** özelliği ile toplam saati,
- **TotalMinutes** özelliği ile toplam dakikayı,
- **TotalSeconds** özelliği ile toplam saniyeyi
- **TotalMilliseconds** özelliği ile toplam saliseyi görebiliriz.

TSM_9

2.3.10. AddDays()

Bu metot ile DateTime türünde türetilmiş bir nesneye parametre olarak verilen **double** türündeki değer kadar gün eklenir ve geriye **DateTime** türünde bir değer döndürülür.

➤ Kullanımı

```
DateTime yeniTarih=eskiTarih.AddDays(double gunSayisi);
```

2.3.11. AddMonths()

Bu metot ile DateTime türünde türetilmiş bir nesneye parametre olarak verilen **double** türündeki değer kadar ay eklenir ve geriye **DateTime** türünde yeni bir tarih döndürülür.

➤ **Kullanımı**

```
DateTime yeniTarih=eskiTarih.AddDays(double ay);
```

2.3.12. AddYears()

Bu metot ile DateTime türünde türetilmiş bir nesneye parametre olarak verilen **double** türündeki değer kadar yıl eklenir ve geriye **DateTime** türünde yeni bir tarih döndürülür.

➤ **Kullanımı**

```
DateTime yeniTarih=eskiTarih.AddDays(double yil);
```

2.3.13. AddHours()

Bu metot ile DateTime türünde türetilmiş bir nesneye parametre olarak verilen **double** türündeki değer kadar saat eklenir ve geriye **DateTime** türünde yeni bir tarih döndürülür.

➤ **Kullanımı**

```
DateTime yeniTarih=eskiTarih.AddDays(double saat);
```

2.3.14. AddMinutes()

Bu metot ile DateTime türünde türetilmiş bir nesneye parametre olarak verilen **double** türündeki değer kadar dakika eklenir ve geriye **DateTime** türünde yeni bir tarih döndürülür.

➤ **Kullanımı**

```
DateTime yeniTarih=eskiTarih.AddDays(double dakika);
```

2.3.15. AddSeconds()

Bu metot ile DateTime türünde türetilmiş bir nesneye parametre olarak verilen **double** türündeki değer kadar saniye eklenir ve geriye **DateTime** türünde yeni bir tarih döndürülür.

➤ **Kullanımı**

```
DateTime yeniTarih=eskiTarih.AddDays(double saniye);
```

2.3.16. AddMilliseconds()

Bu metot ile DateTime türünde türetilmiş bir nesneye parametre olarak verilen **double** türündeki değer kadar saniye eklenir ve geriye **DateTime** türünde yeni bir tarih döndürülür.

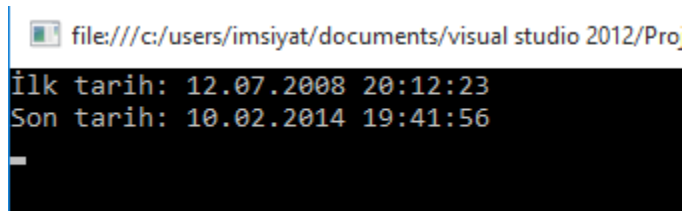
➤ **Kullanımı**

```
DateTime yeniTarih=eskiTarih.AddDays(double saniye);
```

Örnek: 12/07/2008 20:12:23:33 tarihinden 5 yıl, 6 ay, 28 gün, 23 saat, 29 dakika, 33 saniye ve 43 salise sonrasının tarihi nedir?

```
DateTime ilkTarih = new DateTime(2008, 07, 12, 20, 12, 23, 33);  
DateTime bitisTarihi = ilkTarih.AddYears(5);  
bitisTarihi = bitisTarihi.AddMonths(6);  
bitisTarihi = bitisTarihi.AddDays(28);  
bitisTarihi = bitisTarihi.AddHours(23);  
bitisTarihi = bitisTarihi.AddMinutes(29);  
bitisTarihi = bitisTarihi.AddSeconds(33);  
bitisTarihi = bitisTarihi.AddMilliseconds(43);  
Console.WriteLine("İlk tarih: "+ ilkTarih);  
Console.WriteLine("Son tarih: "+ bitisTarihi);
```

Yukarıdaki kodlar çalıştırıldığında aşağıdaki gibi bir ekran görüntüsüyle karşılaşılır.



Fotoğraf 2.21: Ekleme metotlarının toplu kullanımı

UYGULAMA FAALİYETİ

UF2_1

UF2_2

UF2_3

UF2_4

ÖLÇME VE DEĞERLENDİRME

Bu faaliyet kapsamında kazandığınız bilgileri aşağıdaki soruları cevaplayarak belirleyiniz.

GD

ÖD2_1

ÖD2_2

ÖD2_3

ÖD2_4

ÖD2_5

ÖD2_6

ÖD2_7

ÖD2_8

ÖD2_9

ÖD2_10

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru “Modül Değerlendirme”ye geçiniz.

MODÜL DEĞERLENDİRME

MDY2_2

KAYNAKÇA

MODÜL DEĞERLENDİRME

MDY2_2