NESNE YÖNELİMLİ PROGRAMLAMA(Object Oriented Programming/OOP)

Öğr. Gör. Celil ÖZTÜRK
Marmara Üniversitesi
Teknik Bilimler Meslek Yüksekokulu

İçerik

- ✓İyi Bir Yazılımda Olması Gerekenler
- ✓Java Anatomisi
- ✓Veri Tipleri
- ✓Değişkenler
- ✓Tip Dönüşümü(Type-Casting)
- ✓Operatörler, Operatör Önceliği

Yazılım Geliştirme

- Tasarımlar şu yollardan geçerek evrimleşir ve iyileşir:
  - Tasarımın gözden geçirilmesiyle,
  - Kodun yazılmasıyla kazanılan deneyimle * ! *,
  - Kodun revize edilmesiyle kazanılan deneyimle * ! *.

Tasarımdan beklenen karakteristik özellikler

- En az seviyede karmaşıklık
- Kolay bakım yapılabilirlik
- Gevşek bağlılık (loose coupling)
- Genişletilebilirlik
- Yeniden kullanılabilirlik
- Taşınabilirlik
- Yalınlık
- Katmanlaşma
- Standart teknikler

Yazılım Geliştirme

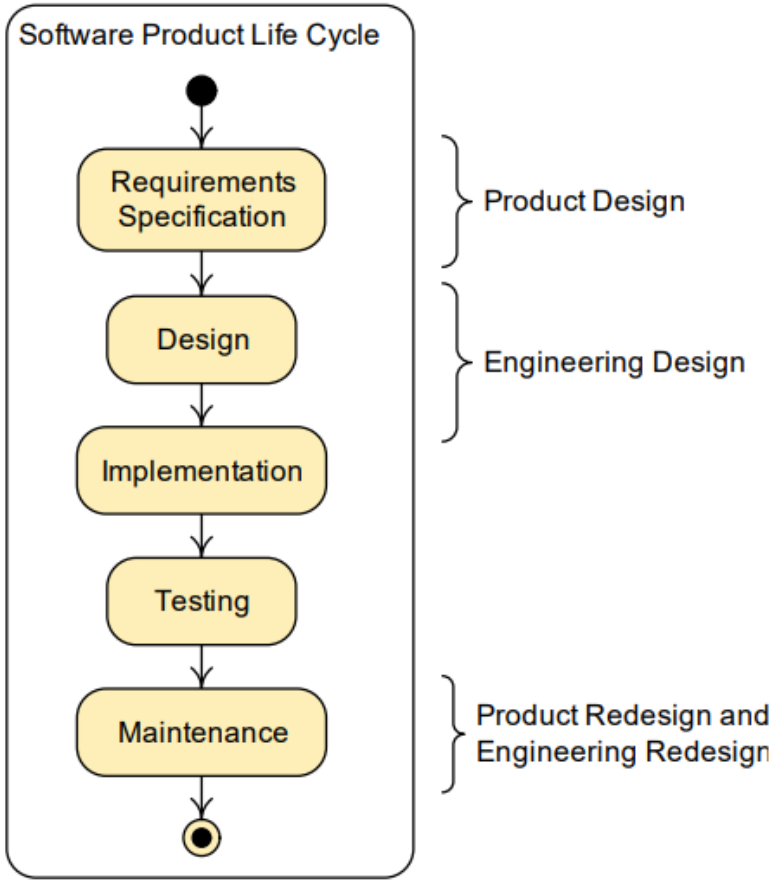
- Okunaklı Kod Yazmak
  - Anlaşılabilirlik (Comprehensibility)
  - Gözden geçirilebilirlik (Reviewability)
  - Hata oranı (Error rate)
  - Hata ayıklama (Debugging)
  - Değiştirilebilirlik (Modifiability)
  - Geliştirme zamanı (Development time) – yukarıdakilerin tümünün sonucu olarak
  - Dışsal kalite (External quality) – yukarıdakilerin tümünün sonucu olarak

YAZILIM GELİŞTİRME

- Bilgi sahibi olunması gereken başlıklar,
- Yazılım tasarımı,Use Case Diyagramları bkz.
- Code Refactor(Kodun düzenlenmesi, kod işlevselliğini değiştirmeden, kod kalitesinin artırılması).
- Reusability(Yeniden Kullanılabilirlik !)-Modülerlik, Esneklik, Tutarlılık,

Düşük Karmaşıklık, Genişletilebilirlik→ Değişen gereksinimlere uyum sağlama,

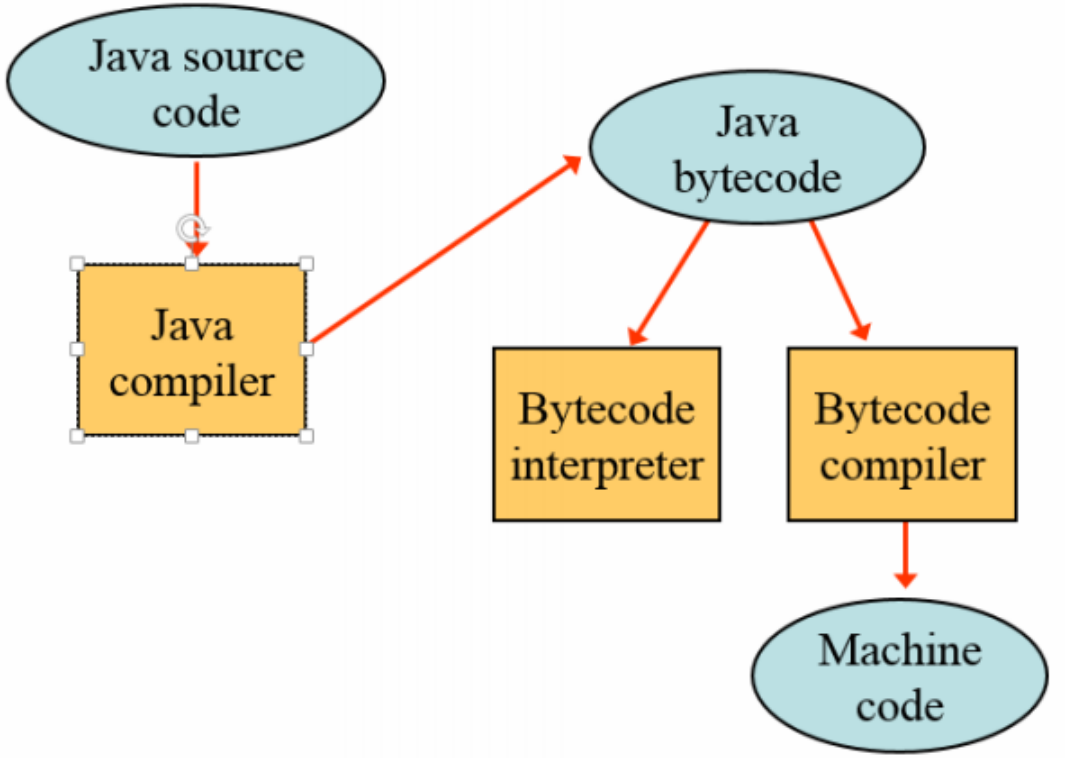
- Tasarım prensipleri bilinmeli, özellikle S-O-L-I-D Prensipleri ! ***

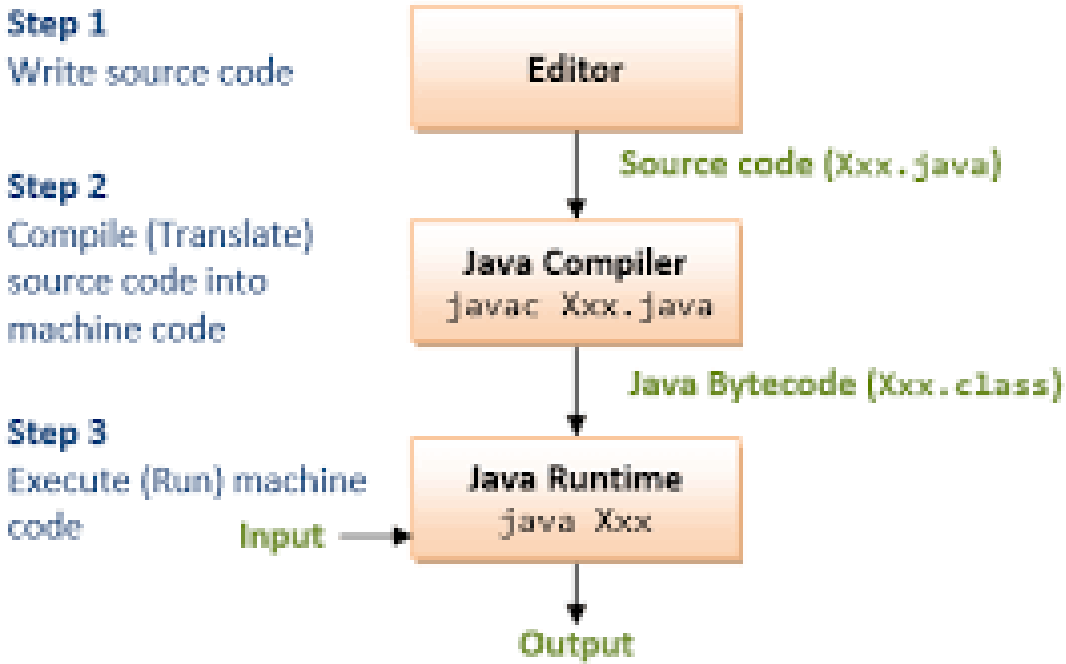


Yazılım Geliştirme

- İyi bir yazılım tasarımı için, soyutlama ve modelleme
- Soyutlama, problemin anlaşılmasını ve çözümünü kolaylaştırmak üzere nesnelerin, olayların veya durumların bazı özelliklerinin bilinçli ve kasıtlı olarak görmezden gelinmesidir.
- Soyutlama, bir problemin basitleştirilmesidir.
- Model ile bir yazılımın görselleştirilmesidir.
- Bir model bir hedefi, modelin parçaları hedefin parçalarına ve modelin parçaları arasındaki ilişkiler hedefin parçaları arasındaki ilişkilere karşılık gelecek şekilde temsil eder.

Java Program Geliştirme





Java Anatomisi

- Package Import(s)
- Comments(açıklamalar)
- Declaration(bildirim)
- Fields(alanlar)
- Constructors(yapıcılar)
- Methods(metotlar)

Bir Java Dosyası 1/2

- 1. Opsiyonel olarak package ifadesi (dosyanın hangi paketin altına eklenmesi istendiğini belirten ifade),
- 2. Gerekli sınıfların eklenmesi için kullanılan import satırları,
- 3. Daha sonra sınıf ismi gelir,
- 4. Sınıf ismini takiben kalıtım (inheritance) ve arayüz (interface) ifadeleri gelir,
- 5. Eğer kaynak kod içerisinde birden fazla sınıf (class) veya arayüz (interface) tanımlanmışsa sadece bir tanesi public olarak tanımlanabilir. Buna ek olarak, kaynak kod dosyasının ismi public olarak tanımlanan sınıf ismi ile aynı olmak zorundadır.

Bir Java Dosyası 2/2

Örnek:

```
package fatih.edu.ceng217;
import java.util.*;
import fatih.edu.ceng217.lecturenotes.*;
import netscape.javascript.JSObject;
import netscape.javascript.JSException;

public class SplayTree implements TreeType,
TreeConstants
    { ...

    }// SplayTree
```

Paket ismi global olarak tek olmak zorundadır. Yani, aynı isimde başka bir paket olamaz. Eğer herhangi bir paket ismi belirtilmezse, kaynak dosyamız içinde bulunulan dizin içerisinde isimsiz ve yazılım geliştirme ortamı tarafından varsayılan olarak belirlenmiş bir paketin altına kaydedilir.

Karakter Dizileri

- Bir karakter dizisi, çift tırnak işareti arasına alınarak bir karakter değişkeni (string literal) olarak gösterilebilir:
- Örnekler:

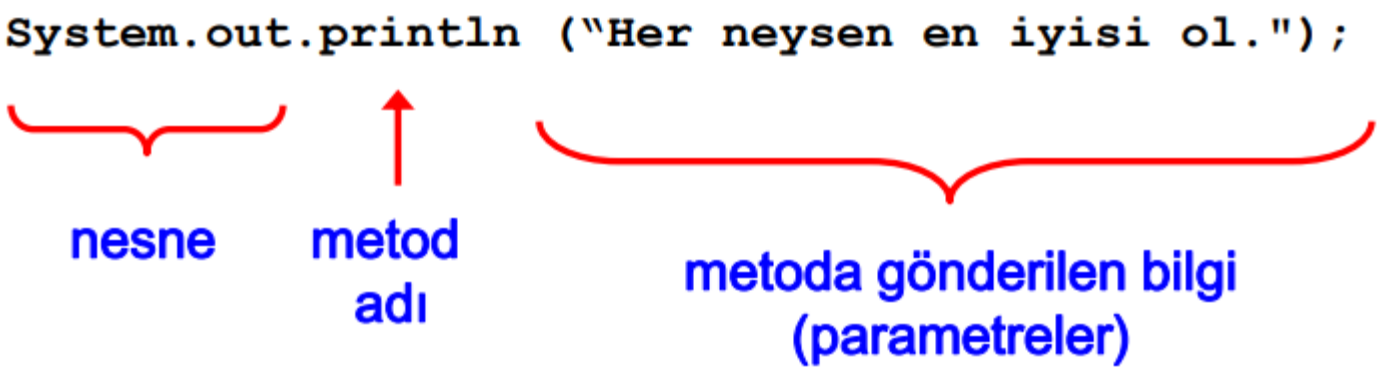
  “Bu bir karakter degiskenidir.«

  "123 Main Street"

  "X“
- Java’da her karakter dizisi String sınıfı tarafından tanımlanmış bir nesnedir.
- Her karakter değişkeni bir String nesnesini gösterir.

Println Metodu

- bir karakter dizisini yazdırmak için println metodu çağrıldı.
- System.out nesnesi çıktının gönderileceği hedefi (ekran) belirtir.
- System.out.println(parametre)



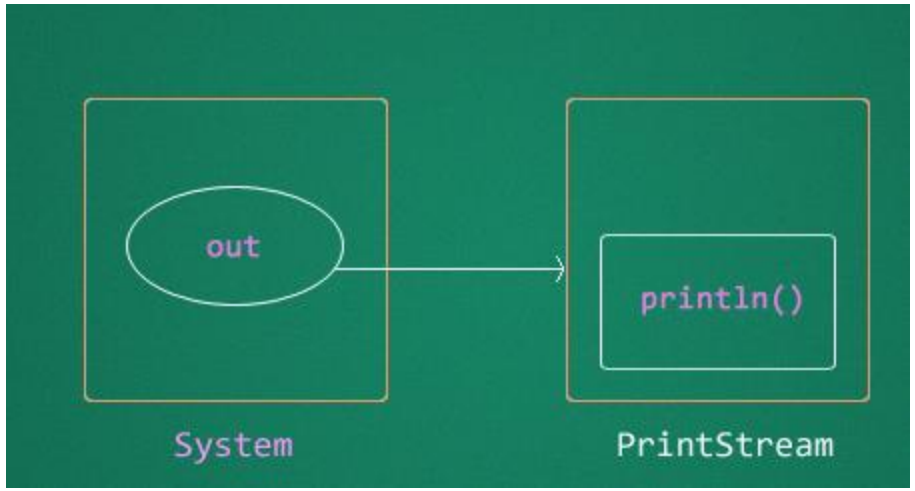
Print Metodu

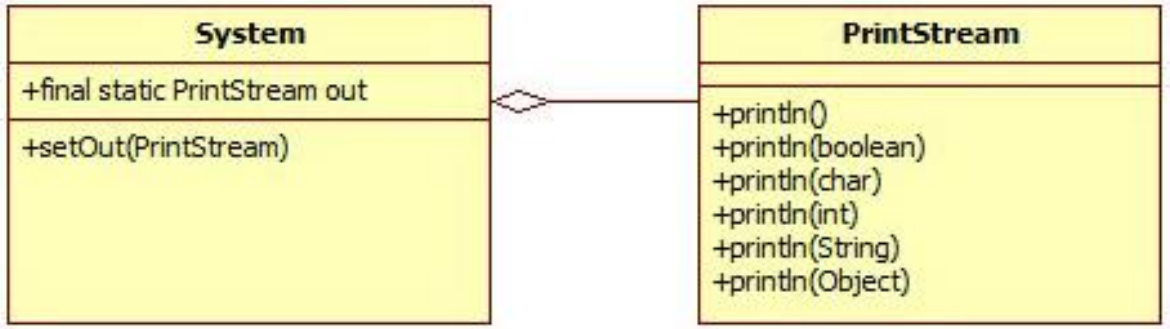
- System.out nesnesi başka bir servis de sağlar.
- print metodu da sonraki satıra geçmek dışında println metodu ile benzerdir.
- Bu yüzden bir print ifadesinden sonra yazdırılan her şey aynı satırda görünmektedir.

System ve PrintStream Sınıfları

```
public final class System {
    static PrintStream out;
    static PrintStream err;
    static InputStream in;
    ...
}
```

```
public class PrintStream extends FilterOutputStream {
    //out object is inherited from FilterOutputStream class
    public void println() {
    ...
}
```





https://javapapers.com/core-java/system-out-println/

String Birleştirme 1/2

- String birleştirme operatörü (+), bir stringi diğerinin sonuna eklemek için kullanılır.
- “Fistik ezmesi " + “ve jole"
- (+) operatörü, bir sayıyı bir string’e eklemek için de kullanılır.
- Bir karakter değişkeni, bir programda iki satıra bölünemez.

String Birleştirme 2/2

- + operatörünün uyguladığı işlem, üzerinde çalıştığı bilginin tipine bağlıdır.
- Eğer iki işlenen bilgiden en az biri ya da her ikiside string ise, «string birleştirme» işlemini,
- Eğer iki işlenen bilgide sayı ise, «toplama» işlemini yapar.
- + operatörü soldan sağa doğru işlem yapar, fakat parantezler kullanılarak sıralama değiştirilebilir.

Kaçış Dizileri(Escape Sequences) 1/2

- Şu satır derleyicinin kafasını karıştıracaktır, çünkü ikinci tırnak işaretini string'in sonu olarak yorumlayacaktır
- System.out.println ("Sana "Merhaba" dedim.");
- Bir komut dizisi özel bir karakteri temsil eden karakterler serisidir.
- Bir komut dizisi \ işareti ile başlar.
- System.out.println ("Sana \"Merhaba\" dedim.");

Kaçış Dizileri(Escape Sequences) 2/2

Komut Dizisi	Anlamı
\b	backspace
\t	tab
\n	yeni satır
\r	satır başı
\"	çift tırnak
\'	tek tırnak
\\	backslash

Tab, 8 karakter boşluk bırakır.

Veri Tipleri 1/2

- İlkel Veri Tipleri(Primitive)

  - ✓Tamsayılar : Byte,Short, İnteger, Long
  - ✓Ondalıklı Sayılar: Float, Double
  - ✓Karakterler: Char
  - ✓Mantıksal Tipler(true-false): Boolean

- Referans/Nesne Veri Tipleri : Sınıf(Class) nesneleri, Dizi(Array)
- String ? ☺

Veri Tipleri 2/2

- ✓ Diğer bir ifade, değişken bellekte verinin girilebileceği bir alanı ifade eder.
- ✓ İlkel veri tipleri ile değişkenler oluşturulduklarında veri tiplerine göre bellekte yer ayrılır.

Primitive Type Keyword FacingIssuesO

Type	Size in bytes	Range	Default Value
byte	1 byte	-128 to 127	0
short	2 bytes	-32,768 to 32,767	0
int	4 bytes	-2,147,483,648 to 2,147,483, 647	0
long	8 bytes	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	0
float	4 bytes	approximately ±3.40282347E+38F (6-7 significant decimal digits) Java implements IEEE 754 standard	0.0f
double	8 bytes	approximately ±1.79769313486231570E+308 (15 significant decimal digits)	0.0d
char	2 bytes	0 to 65,536 (unsigned)	'\u0000'
boolean	Not precisely defined*	true or false	false

Karakter Kümeleri

- Bir char değişkeni tek bir karakter depolar.
- Karakter değerler tek tırnaklar ile sınırlandırılır: 'a' 'X' '7' '$' ',' '\n‘ Örnek bildirimler:
- char enIyi = 'A’;
- char noktaliVirgul = ';’, bosluk = ' ‘;
- Sadece bir karakter tutan ilkel karakter değişkeni ile birçok karakterler tutabilen String nesnesi arasındaki farka dikkat ediniz

Karakter Kümeleri

- Bir karakter kümesi, her birinin kendine has bir sayı karşılığı olan sıralı karakterler listesidir.
- Java'daki bir char değişkeni Unicode karakter kümesindeki herhangi bir karakteri tutabilir.
- Unicode karakter kümesi, 65,536 benzersiz karakter sağlayarak her bir karakter için 16 bit kullanır.
- Unicode karakter kümesi, birçok dünya dilinden karakterler ve semboller içeren uluslararası bir karakter kümesidir.

Karakter Kümeleri

- ASCII karakter kümesi Unicode’dan daha eski ve küçüktür, fakat hala yaygındır.
- ASCII karakter kümesi, Unicode karakter kümesinin bir alt kümesidir ve şunları içerir: 15.10.2019 KARAKTER KÜMELERİ Büyük harfler Küçük harfler Noktalama Rakam Özel semboller Kontrol karakterleri A, B, C, … a, b, c, … nokta ,noktalı virgül, … 0, 1, 2, … &, |, \, … Satır başı, tab, ...

Değişkenler 1/2

- Program içerisinde verilerin depolanması amacıyla değişkenler kullanılmaktadır.
- Erişim Belirteci + Değişken tipi + Değişken adı = Değişken değeri ;

```
public int sayi = 5;

private boolean durum = true;

protected double ortalama = 35.12;
```

Değişkenler 2/2

- Java type-safe bir dildir. Her değişkenin bir tipi olmak zorundadır.
- Java statik tip bir dildir.(Her değişken, değişmez bir tipe sahiptir.)
- Java derlenen(compiled) dillerde değişkenler tanımlanarak kullanılmak zorundadır. Yorumlanan(interpreted) dillerde ise değişkenler doğrudan kullanılabilir.(Bkz. Compiled vs İnterpreted Languages)

Statik & Dinamik Tipli Diller

- Statik Tipli dillerde her değişkenin, değişmez bir tipi olmak zorundadır.
- Statik Tipli dillerde yapılan işlemlerden önce Tip Kontrolü yapıldığı için, operandların tiplerinin belirlenmesi gerekir. **Tip kontrolleri compile (derleme) esnasında yapılır**. Java, C#, C++,Ada, Pascal …
- Dinamik Tipli dillerde, değerler değişmez bir tipe sahiptir.
- Değişkenler ve ifadelerin belirli bir tipe sahip olma zorunlulukları yoktur. **Değişkenlerin ve ifadelerin tipleri run-time (çalışma) esnasında belirlenir.** PHP, Python, Javascript, Ruby …

Type Casting (Tip Dönüşümleri) 1/3

- Tip dönüşümü, bir veri tipindeki değişkeni bir başka veri tipine dönüştürme işlemidir.
- Küçük veri tipleri, büyük veri tiplerine kolaylıkla dönüştürülebilir. Ancak büyük veri tipleri, küçük veri tiplerine dönüştürülürken veri kaybı yaşanabilir.
- Byte < Short < Int <Long < float(virgüllü) <double(virgüllü).

Not: Kayar noktalı(virgüllü/ondalıklı) sayı tipindeki değerlerin varsayılan değeri double'dır. Bir değişkene float değeri vermek için ''f'' son eki kullanılmalıdır.

float altinOran = 1.618f;

Type Casting (Tip Dönüşümleri) 2/3

- byte byteTipiDegeri = 10;
- short shortTipDegeri =byteTipiDegeri; //Byte tipi, short tipine dönüştü.
- int intTipDegeri = shortTipDegeri; …… //short tipi, int tipine geldi.

- int x;
- float pi = 3.14F;
- x = (int)pi;
- System.out.println(pi); // Çıktı → 3.14
- System.out.println(x); // Çıktı → 3

```
int x=5;
int y=2;
double z;
z = (double) x/y;
System.out.println(z); // 2.5
```

Type Casting (Tip Dönüşümleri) 3/3

Int -> String dönüştürme…

- int sayi = 13;
- String s1 =String.valueOf(sayi); //1. Kullanım
- String s1= Integer.toString(sayi); //2. Kullanım

String -> int dönüştürme…

- String s1= "13" ;
- int sayi1 = Integer.valueOf(s1);
- İnt sayi2 = Integer.parseInt(s1);

Operatörler

Aritmetik Operatörler

- Integer A değişkeninin 10, B değişkenin 20 değerini aldığını varsayalım.

Operator	Description	Example
+	Toplama: Operatörün iki tarafındaki değeri birbiriyle toplar.	A + B 30 değerini verecektir.
–	Çıkartma: : Operatörün iki tarafındaki değeribirbirinden çıkartır.	A – B -10 değerini verecektir
*	Çarpma: Operatörün iki tarafındaki değeri birbiriyleçarpar	A * B 200 değerini verecektir
/	Bölme: Operatörün iki tarafındaki değeri birbirineböler	B / A 2 değerini verecektir.
%	Mod alma: Operatörün solundaki değişkenin;sağındakine göre modunu alıp kalanı verir.	B % A 0 değerini verecektir.
++	Arttırma: Değişkenin değerini 1 arttırır	B++ 21 değerini verecektir.
—	Azaltma: Değişkenin değerini 1 azaltacaktır.	B– 19 değerini verecektir

Operatörler

İlişkisel Operatörler

- Integer A değişkeninin 10, B değişkenin 20 değerini aldığını varsayalım.

Operator	Description	Example
==	İki değişkenin değerini kontrol eder,eğer eşitse true değerini döndürür.	(A==B) False değerini döner.
!=	İki değişkenin değerini kontrol eder,eğer eşit değilse true değerini döndürür.	(A != B) True değer döner.
>	Operatörün solundaki değişkenin, sağındakinden büyük olup olmadığını kontrol eder. Eğer büyükse true değer döner.	(A > B) False değer döner.
<	Operatörün solundaki değişkenin, sağındakinden küçük olup olmadığını kontrol eder. Eğer küçükse true değer döner.	(A < B) true değer döner
>=	Operatörün solundaki değişkenin, sağındakine eşit veya büyük olup olmadığını kontrol eder. Eğer öyleyse true değer döner.	(A >= B) False değer döner.
<=	Operatörün solundaki değişkenin, sağındakine eşit veya küçük olup olmadığını kontrol eder. Eğer öyleyse true değer döner.	(A <= B) true değer döner

Operatörler

Mantıksal Operatörler

- Aşağıdaki tablo mantıksal operatörleri göstermektedir. boolean A'nın **true** , B'nin **false** değerlerini aldığını varsayalım.

Operator	Description	Example
&&	Mantıksal AND operatörü. Eğer iki taraftaki değişken de false değilse sonuç true döner	(A && B) false değer döner..
||	Mantıksal OR operatörü. Eğer operatörün iki tarafındaki değişkenlerden birisi true ise sonuç true döner.	(A || B) true değer döner.
!	Mantıksal NOT operatörü. Operasyonun mantıksal cevabını terse çevirir. Eğer true değer dönüyorsa, sonucu false değere çevirir.	!(A && B) true değer döner..

Operatörler

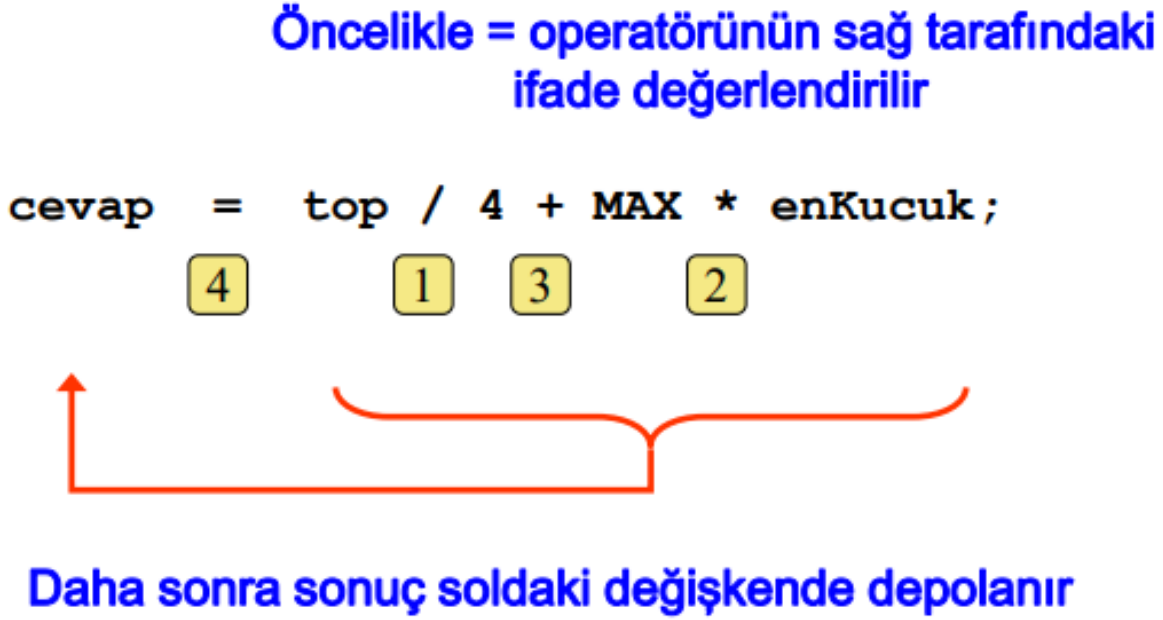
Atama Operatörler

Operatör	Örnek	Denkliği
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y

Operator	Description	Example
=	Basit atama operatörüdür. Operatörün sağındaki değişkenin değeri solundaki(lere)'ne atanır.	C = A + B; A+B nin değerini C'ye atayacaktır.
+=	Operatörün sağındaki değeri solundaki değerle toplayıp tekrar solundaki değere atar.	C += A ; C = C + A değerine eşittir.
-=	Operatörün sağındaki değeri solundaki değerden çıkartıp tekrar solundaki değere atar.	C -= A; C = C – A değerine eşittir.
*=	Operatörün sağındaki değeri solundaki değerle çarpıp tekrar solundaki değere atar.	C *= A; C = C * A değerine eşittir.
/=	Operatörün sağındaki değeri solundaki değere bölüp tekrar solundaki değere atar.	C /= A; C = C / A değerine eşittir.
%=	Operatörün sağındaki değerin solundakine göre modunu alıp tekrar solundaki değere atar.	C %= A; C = C % A değerine eşittir.

Atama Operatörü

- Atama operatörü aritmetik operatörlere göre daha düşük bir önceliğe sahiptir.



ATAMA OPERATÖRÜ

İlk önce, say'ın var olan değerine bir eklenir

```
say  =  say + 1;
```

Sonra sonuç eski değeri silinerek say içinde depolanır

Artırma ve Azaltma Operatörleri

- Artırma ve azaltma operatörleri sadece bir işlenen kullanırlar.
- Artırma operatörü (++) işlenenine bir ekler.

  say++;

  ifadesi, işlevsel olarak

  say = say + 1; ifadesine eşdeğerdir.

- Azaltma operatörü (- -) işleneninden bir çıkarır.

Artırma ve Azaltma Operatörleri

- Artırma ve azaltma operatörleri sonek, count++ ya da önek ++count halinde de uygulanabilirler:
- Daha büyük bir ifadenin parçası olarak kullanıldıklarında, bu iki durumun farklı sonuçları olabilir.
- Anlaşılmazlıklarından dolayı artırma ve azaltma operatörleri dikkatli kullanılmalıdır.

sayi++ ifadesinde önce sayi değeri kullanılır, daha sonra artırım yapılır.

++sayi ifadesinde ise, önce sayi değeri artırılır, ve sonra kullanılır.

Artırma ve Azaltma Operatörleri

- Sık sık bir değişken üzerinde bir işlem gerçekleştirir ve sonucu tekrar o değişkende depolanır. Java bu işlemi kolaylaştırmak için atama operatörleri sağlar.
- Örnek:

```
num += count;
```

ifadesi

```
num = num + count;
```

ifadesi ile eşdeğerdir.

Koşul Operatörü (? :) (Ternary)

- booleanExpression ? expression1 : expression2
- İfade1 ? İfade2 : ifade3
- İfade1 doğru ise ifade2, değil ise ifade3 çalışır.
- Örnek;

```
int x=10;
int y=4;
int buyukSayi = (x<=y) ? y : x;
System.out.println(max);
```

Operatör Önceliği 1/4

- Operatörler karmaşık iafadeler içinde kullanılabilirler.
- sonuc = toplam + sayi / max - konum;
- Operatörler hangi sırada değerlendirileceklerini belirleyen iyi tanımlı bir öncelik sırasına sahiptirler.
- Çarpma, bölme ve kalan; toplama, çıkarma ve string birleştirmeden önceliklidir.
- Eşit öncelikli aritmetik operatörler soldan sağa doğru değerlendirilir, fakat değerlendirme sırasını değiştirmek için parantezler kullanılabilir.

Operatör Önceliği 2/4

```
int a=5, b=3;
int sonuc=a*3+b;
System.out.println(sonuc); //??
```

Operator Kategorisi	Operatörler
Primary	x.y f(x) a[x] x++ x--
Unary	+ - ! ~ ++x --x (T)x
Çarpımsal (Multiplicative)	* / %
Toplamsal (Additive)	+ -
Kayma (Shift)	<< >>
İlişkisel (Relational)	< > <= >= is as

Operatör Önceliği 3/4

- not > 50 && not < 100 !!

Java operatörlerinin öncelik sırası Üst öncelikten alt önceliğe doğru sıralıdır	
Operator Kategorisi	**Operatörler**
Eşitlik (Equality)	== !=
Mantıksal VE (Logical AND)	&
Mantıksal XOR (Logical XOR)	^
Mantıksal VEYA (Logical OR)	|
Koşullu VE (Conditional AND)	&&
Koşullu VEYA (Conditional OR)	||
Koşullu (Conditional, ternary)	?:
Atama (Assignment)	= *= /= %= += -= <<= >>= &= ^= |=

Operatör Önceliği 4/4

Bu ifadeler için değerlendirme sırası nedir?

```
a + b + c + d + e
  1   2   3   4
```

```
a + b * c - d / e
  3   1   4   2
```

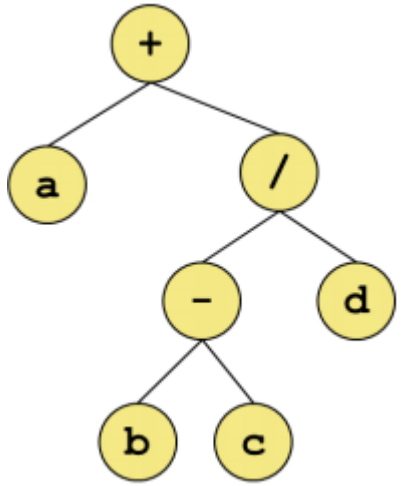
```
a / (b + c) - d % e
  2    1    4   3
```

```
a / (b * (c + (d - e)))
  4    3    2    1
```

İfade Ağaçları

- Belirli bir ifadenin değerlendirilmesi ifade ağacı kullanılarak gösterilebilir.
- Ağacın alt katmanlarındaki operatörler ifade için daha yüksek önceliğe sahiptirler.

```
a + (b - c) / d
```



Sabitler (Final Kelimesi)

- Java programlama dilinde bir değişkenin sabit olarak tanımlanması için final anahtar kelimesi kullanılır.
- Sabit olarak tanımlanan değişkenler, bir kere değer ataması yapıldıktan sonra değiştirilemezler.

```
final double pi=3.14;
final float altinOran=1.618f;
```

Değişken İsimlendirme Kuralları

- Boşluk karakteri değişken isimlendirmelerinde kullanılamaz.
- Case sensitive özellikli programlama dillerinde değişken isimlendirilmesinde büyük, küçük harf duyarlılığı olduğu için isimlendirilirken dikkat edilmelidir. (sonuc != Sonuc)
- Türkçe karakter kullanmaktan kaçınılmalıdır.
- Değişken isimleri rakam ve özel karakter ile başlamaz.
- Programlama dilindeki kullanılan anahtar kelimeler/komutlar değişken ismi olarak kullanılamaz.

Kaynaklar

- https://www.oracle.com/tr/sun/
- https://bidb.itu.edu.tr/seyir-defteri/blog/2019/02/05/object-oriented-programming
- Java ve Java Teknolojileri, *Tevfik KIZILÖREN* – Kodlab Yayınları
- https://medium.com/@kplnosmn94/jvm-jre-ve-jdk-nedir-6cfee2727812
- Yrd. Doç. Dr. Deniz KILINÇ-Celal Bayar Üniversitesi / Nesneye Yönelik Programlama Ders Notları
- Yrd. Doç. Dr. Erbil Akbay –Marmara Üniversitesi / Object Oriented Programming 1-2 Ders notları
- https://kodcu.com/2011/07/statik-tipli-diller-ile-dinamik-tipli-diller-arasindaki-farklar/
- http://javayaz.com/?page_id=84
- http://mail.baskent.edu.tr/~tkaracay/etudio/ders/prg/java/ch06/privilagesOfOperators.pdf
- Doç. Dr. Deniz KILINÇ – Celal Bayar Üniversitesi / Yazılım Mimarisi ve Tasarımı Ders Notları
- Dr. Öğr. Gör. Emin BORANDAĞ – Celal Bayar Üniversitesi / Yazılım Yapımı Ders Notları