if eğer koşul varsa kullanılır.

1 Köşeli parantez koymazsan

1 if in koşulu sağlanırsa aşağıdaki tüm işlemleri yapar

2 sağlanmazsa if in altındaki ilk işlem dikkate alınmaz, if in koşulu sağlanmamasına rağmen 2. İşlem pc yapar.

2 Köşeli parantez koyarsan koşul sağlanmıyorsa içeri girmeden çıkar gider.

Köşeli parantez koysan ve koşul sağlansa bile

Else koymazsan

aşağadaki if lere de uğrar. Hem yukarıdaki hemde aşağıdaki iflerin koşulunun sağlandığı ancak tek sonuç almak istediğimiz durumlarda 2 sonuca maruz kalırız (işlemi uzun sürer)

---------------------------------------------------------------------------------------------------------------------------------------

if else değilse ifadesi geçiyorsa kullanılır. İf in koşulu sağlanıyorsa ve else( değilse) ifadesi olduğu için aşağadaki if lere gitmez işlemi kısa sürer. Köşeli parantez koymadan else nin olduğu if in altına birden fazla komut giremezsin

if (saat <= 2)   
 System.*out*.println("5 €");  
 int toplam =a+b;  
   
 else if (saat <= 5) { yukarıda köşeli parantez olmazsa else burada hata veriyor.  
 System.*out*.println("10 €");  
} else  
 System.*out*.println("15 €");

----------------------------------------------------------------------------------------------------------------------------------------

Nestead if ***böyle olduğunda buda varsa*** dediğin de kullanılır *else ye gerek yok* ***else değilse****.*

Switch case Çoklu seçenekler de kullanılır. İki tane seçenek varsa if daha mantıklı.

kontrol baştaysa While de şart olduğu sürece yap sürekli yap. Koşul sağlandıkça sonsuz kadar yapar. Koşul sağlanmazsa. İçine girmeden çıkar gider. int sayac=1;   
 while(sayac <= 5)   
 sayac++;

Kontrol sondaysa Do while de işlem bir kez mutlaka do nun içine girer koşul sağlandıkça devam eder. Sağlanmıyorsa çıkar gider içine mutlaka bir kez gireceği için bazı komutları dışarıya yazmamıza gerek yok içine girer. Aşağıda komut sağlanıyorsa işlemi yapar

Farkları while veya do while de **ne zaman biteceği, kaç kere döneceği belli olmayan durumlarda kullanılır** kullanıcı 0 girene kadar. devam et dersek.

For da kaç tane döneceği belli ise artış miktarı da sabit ise kullanılır.

Sayısal ritmik ifadeler için for kullanılmalı. 11 den başlayıp tek sayıları yazdırmak için 3 5 7 veya 2 4 6

Loop in loop anahtar ifade 2 değişkende belli değilse kullanılır.

Array ın loop tan farkı lenght e kadar döngü yap dememiz ve dizi olup 1 variable name in içine çok sayıda değer alması

**Gün 18 Array değişken tanımlama**

int[] dizi2=new int[5];

int [] dizi= new int[] {2,3,4,5};

int dizi1 [] = new int[] {2,3,4,5};

int [] dizi3={11,33,44,5555,};*Kutu içinde belirtmedik [] tane eleman olsun diye. Eleman sayısı kadar uzunlukta dizi tanımlanabilir.*

String [] si = {"$12", "$23", "$10", "$2", "$5", "$2"}; si.length = eleman sayısı 6 yazdırarak buldum yanlışlık yok. 6 tane oda var oda numaraları 0 dan başlayıp 5 e kadar gidiyor.

System.*out*.println("si[2] = " + si[2]); *// si[2] = $10*

String [] [] a= {{"$12,$22,5$"},{"€9" , "€40" , "$1" , "$2"}, {"€12"**}**};boyutlu dizi. Eleman sayısı yani lenght i3 düşün yazdırarak buldum yanlışlık yok

0index ,1.indx, 2indx. - 0 1 2 3 - 0

Lengh 3 tane; yani[3]gibi düşün {parantez içi bize leng verir} lengh 1 2 3 diye başlar.

a[2].lengh=4 a[3].lengh=1 [ ] parantezin içi index i verir. index 0 dan başlar

System.*out*.println(a[1][0]); €9

int [][] dizi={{9},{11},{25},{3,74},{7,8,5,}}; bu bir araydır. Bu array ın içindeki her bir elemanda arraydır (*0 ,1 ,2, 3, 4 )çünkü bunların her biri diziler halinde sıralanmıştır. Bu grupların içindekiler birer birer düşünüldüğünde Ve ya işleme alındıklarında integerdir. 0 , 1 ,2, 3, 4*

int [][] mat=new int[4][5]; *//hafızada 20 tane ama 4 satır 5 sutun şeklinde yer ayırır*

**Array da kullanılan komutlar**

cumle.length(); string kullanımı

dizi1.length; array kullanımı

for(int i=0;i<dizi1.length;i++) setlerde ise setin in elemanlarında get olmadığı için fori yerine foreach kullanılır.

System.*out*.println("dizi1 (int) = " + dizi1[i]);

**Yazdırma Şekilleri**

Arrays to string

Foreach : sadece yazdıracaksan işlem yapmayacaksan daha avantajlı gezinmek zor.

Fori : işlem yapacaksan daha avantajlı

**soru 2**

int[] dizi=new int[5];  
  
 for(int i=0;i<dizi.length;i++)  
 {  
 dizi[i] = (int)(Math.*random*()\*10)+1; *//random paranteze alınmalı. double olmaması için.* }  
*sort gün 19 da*

*--------------------------------------------------------------------------------------------------------------------------------------------------------------------------*

Gün 27 28 setler temel özellikleri

*Setler Array list in bir gelişmiş hali artısı içine aynı elemanı almaması Veriler in* ***tekrar etmemesi.***

*verilerin* ***sayısı belli değil, artabilir azalabilir***

***eklenme sırasına*** *göre,* ***her zaman sıralı*** *veya* ***özel sıralı*** *olması gerektiği durumlarda SET LERDE DİZİDİR.*

*setlerde elemana direk ulaşılmıyor elemanı direkt getir yok . bu işlem için set in bir üstünü kullanacağız Map*

set in elemanlarında get olmadığı için fori yerine foreach kullanılır.

Set<Integer>hs1= new HashSet<>(Arrays.*asList*(1,2,3,4,5)); *böylede eleman atanabiliyor. hs1Add((1,2,3) ve hsAdd(4,5); ile aynı int [] dizi= {1,2,3}; gibi başlangıçta değer atadığımız tarzda değer atadık*

***Sete eleman atama set eleman***

***Sete başlangıçta***

Set<Integer>hs1=new HashSet<>();

Set s=new HashSet(); *// şeklin de de set tanımlanabilir. türünü belirtmezsen integer double diye içine herşeyi atarsın"*

HashSet<Integer> hs1=new HashSet<Integer>(); *// sadece ınteger alabilen bir set .*

Set<Integer>hs2= new HashSet<>(); *ÖNERİLEN sadece ınteger alabilen bir set . en pratik tanımla ma şekli budur setler hashSet linkedSet ve TreeSet in atası olduğu için başta set diye tanımlayıp eşitlikten sonra hangi set türü olacaksa tanımlanabilir*

Set hsObject=new HashSet(); *// Bu tanımlamada her türlü değişken atanabilir.  
 bütün tipleri tanımlayan tipin adı : OBJECT-> NESNE -> HERŞEYİN ATASI  
 //bu tanımlama diğerlerine göre daha yavaş çalışır ve içine başka şeylerde atıldığından güvenli değildir. veriler tutarsız ve kontrol dışı olabilir. ÇÜNKÜ*

hsObject.add("12");  
 hsObject.add(12); *// kullanıcı string de Integerde girebilir.*

HashSet<Integer> fark= new HashSet<>(a); ***// Bir Seti clone lama veya bir seti başka sete assign etme***

*yeni tanımlanan HashSet i a ya eşitledik. yeni HashSet ten b de kileri çıkaracağız*

***SETLERİN KOMUTLARI***

HashSet<Integer>birlesik =new HashSet<>();  
birlesik.addAll(a); setleri birleştirme

birlesik.addAll(b);

HashSet<Integer> fark= new HashSet<>(a); *// yeni tanımlanan HashSet i a ya eşitledik. yeni HashSet ten b de kileri çıkaracağız*fark.removeAll(b); YALIN A YI BULACAĞIZ FARKI BULACAĞIZ

HashSet<Integer> kesisim =new HashSet<>(a); *// aya eşit oluyor. kesisimBul*kesisim.retainAll(b);

*clear: Set içerisindeki bütün elemanları siler.*

*contains (Object o): Bir eleman dizi içinde mi kontrol eder.*

*remove (Object o): Bir elemanı siler.*

*size: O ana kadar kaç eleman eklendiğini döndürür*

*isEmpty(); map boşmu dolumu*

public static int [] convertToArray(TreeSet<Integer>set) *// set i diziye çeviren metod* {int [] yeniDizi= new int[set.size()]; ***set i array e dönüştürme 1. Yol***   
  
 int i=0; iteratör de kullanılabilirmiş  
 for (Integer sayi: set) set in elemanlarında get olmadığı için fori yerine foreach kullanıldı  
 {  
 yeniDizi[i++]=sayi;

*1 int sayi;  
 2 int [] dizi= new int[100] 100 kez int sayı tanımlamak zor olacağı için dizi yi öğrendik oda yetmedi.  
 3 int [][] mat= new int [3][2] iki boyutlu veri sakladık.*

*. bunu nerede tutabilirim yukarıda 5 elemanlı diye tanımlıyoruz ama 6 . ve ya daha fazla data girmek gerekebiliyor.*

*4 ARRAYLİST*

*örnek: bir site yaptığında username lerin aynı olmasını istemediğimiz için setler kullanılır.****dizilerde sabit olduğu için lenght.******Setlerde ise******büyüyüp küçülebilen değişebilen yerlerde ise size().*** *size yi metod şeklinde yapmışlar çünkü sabit değil büyüyüp kısalabiliyor. O Anda metod gidip işleme girip sayıp geliyor. her zaman değişebildiği için.*

***İteratörler setlerin başlangıçta yazılma şekline göre farklılık gösterebiliyor.***

Gün 31 | Object Oriented Programming

*1. Yöntem aynı dosyada var olan en üstteki class ın altında main ve main altında class açılıp new denilip içine attribute fields eklenip mainden call ettik*

#### Mainde class name+ object name = new class name yazıp accesing verdik.

#### 2. yöntem de en üstte var olan class ı yani daha dosyayı ilk açarken oluşturmuş olduğumuz class ın içine fields attribute oluşturup mainden call ettik ve kullandık

#### Mainde class name+ object name = new class name yazıp accesing verdik.

#### 3. Yöntemde 3. Class ı 2 . class a tanıttık ve ardın dan 2. Class taki ismi ile 1. Class ta yani main class erişime açtık 2. Dosyada erişime açılmış hali ile .

public class EvEx1 {

public static void **main**(String[] args) {

evStudent ogrenci1= new evStudent();

*//1. yöntem*

evSchool okulu =new evSchool();

//2. Yöntem

ogrenci1.schoolInfo=okulu; *// oluşturulup dolduruldu öğrenciye atandı*

ogrenci1.schoolInfo= new evSchool();

#### }

#### }

public class evStudent {

evSchool schoolInfo; okulu class ını öğrenci class ı na assign etmiş olabilirsin ama gerekmezse kullanmayacağın için

#### okul class ının özellikleri mainde student class ının özellikleri içinde görünmeyecek. Hafızayı daha iyi

#### } kullanabilmek için gerektiği anda new diyerek obje olarak dünyaya getirmelisin.

Public class evSchool {

#### }

#### Gün 32 | Multiple Objects,Attributes and Access attribute different class

#### 1 Aynı dosyada metod çağırma

#### 2 Aynı dosyada farklı class içinde metod oluşturma ve bu metod u mainden çağırıp kullanma

#### 3 Farklı dosyadaki bir metod a ulaşmak 2. Yöntem ile aynı

#### Yazdırma yöntemi

public class evEx1 {

public static void main(String[] args) {

Personn kisi1= new Personn(); // person classından bir obje oluşturuldu

*BilgiYazdır*(kisi1); // parametreleri aklımda tutmam gerekiyor

kisi1.BilgiYazdır(); // object Name + . basınca görünecek

}

public static void BilgiYazdır(Personn kisi)  
{

System.*out*.println("kisi metod = " + kisi.name);  
System.*out*.println("kisi metod= " + kisi.surname);

}

#### }

class Personn

{ String name;

String surname;

String BilgiYazdır()

#### {

System.*out*.println("kisi class = " + surname); *// class ın kendi değerleri gelecek zaten*System.*out*.println("kisi class = " + age);

#### }

#### }*// new demeden bir class ı kullanamaz sın ancak new olayını class ın içinde de yapabilirmişiz.* ---------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Gün 33 | Static vs. Non-Static Method, Recap

**Metodların basında static varsa main metod da (ve belki heryerde emin ol) direkt kullanılabiliyor yoksa bulunduğu dosyanın main metod da veya başka bir dosyanın main metod unda new denilip oluşturulup öyle kullanılabiliyor. ?**

***Başka dosyadaki classlar***

*1 public metod metod un tasarım halinden çıkarıp new diyerek call edip doğumun gerçekleşmesi ile kullanabiliyoruz  
2 public* ***staic*** *metod oluşturuldu mainde new demeden kullanabildik .*

*Kendi dosyası içinde kullanımı da başka dosyalar içindeki kullanımları ile aynı*

*Eğer metod static değilse dosya ismi + variable name + new dosya ismi*

*Eğer metod staticse new demeden kullanılabiliyor*

public class evJavaStaticAndNonStaticMethodss {

public static void main(String[] args) {

int sayı=345;  
String strRakam=String.*valueOf*(sayı);  
int intRakam=Integer.*parseInt*(strRakam);

evUtility util =new evUtility();

metod static olmadığı için new demeden kullanılmıyor

strRakam=util.*getString*(sayı);

strRakam= evUtility.*getString*(sayı);

*}*

***Başka dosyadaki classlar***

public class evUtility {

public String getString(int valueDeğer)  
{  
 return String.*valueOf*(valueDeğer); *// kendine gelen int değerleri hep stringe çevirip geriye vercek*}

public static String getString(int valueDeğer)  
{  
  
 return String.*valueOf*(valueDeğer).trim(); *// kendine gelen int değerleri hep stringe çevirip geriye vercek*}

**}**

Map lerde add yerine putt var

#### Gün 18 | Array (1d Array), Array with loop teach Arrays.toString, for each | Part 1 - 06/17/20

#### Array (1d Array), Array with loop

cumle.length(); string method da kullanımı

dizi1.length; array kullanımı

**int [] dizi= new int[] {2,3,4,5};** *// arraylar böylede tanımlanabilir*

int[] dizi1=new int[5]; *//boyutu 1 olan 5 elemanlı öğren iyice, 5 adet indexli 0,1,2,3,4 beş elemanlı yer hafızada ayrıldı.*

*Değerler girilmeden önce hafızada yer ayrıldı ancak 5 tane değer girilebilir.*

int [] dizi1={11,22,33,44,5555,};

*0 ,1 ,2, 3, 4 5 indexli değerleri. İndex numaraları 0,1,2,3,4, olan 5 elemanlı array*

int[] dizi2={5,3,76,9,6,0,-34,}; *// Kutu içinde belirtmedik [] tane eleman olsun diye. Eleman sayısı kadar uzunlukta dizi tanımlandı. Ve hafızada* ***biz yazarken o kadar yer ayrılır****. {Ayrılan yerlerin içindeki değerler parantez içine yazıldı}*

String[] dizi3=new String[5];*String tipinde 5 elelaman lı içi boş dizi tanımlandı./scaner e bunu tanıtırsak aşağıdan 5 tane sayı girebiliriz*

String[] si = {"$12", "$23", "$10", "$2", "$5", "$2"}; tek boyutlu string length 6 index 5

1. index 1.index 2.index:bu indexlerin içindekilerde grup grup düşünüldüğünde arraydır

String [] [] a= {{"$12,$22,5$"},{"€9" , "€40" , "$1" , "$2"}, {"€12"}}; 2 boyutlu dizi. Eleman sayısı 3 düşün

0index ,1.indx, 2indx. - 0 1 2 3 - 0

Lengh 3 tane; yani[3]gibi düşün {parantez içi bize leng verir} lengh 1 2 3 diye başlar.

a[2].lengh=4 a[3].lengh=1 [ ] parantezin içi index i verir. index 0 dan başlar

System.*out*.println(a[1][0]); €9

int [][] dizi={{9},{11},{25},{3,74},{7,8,5,}}; bu bir araydır. Bu array ın içindeki her bir elemanda arraydır (*0 ,1 ,2, 3, 4 )çünkü bunların her biri diziler halinde sıralanmıştır. Bu grupların içindekiler birer birer düşünüldüğünde Veya işleme alındıklarında integerdir. 0 , 1 ,2, 3, 4*

int [][] mat=new int[4][5]; *//hafızada 20 tane ama 4 satır 5 sutun şeklinde yer ayırır*boolean[] dizi4=new boolean[4]; *// boolean (true, false) tipinde default değerli dizi*double[] dizi5=new double[5]; *// double cinsinden 5 elemanlı bir dizi*

String[] dizi3=new String[5];*String tipinde 5 elelaman lı içi boş dizi tanımlandı./scaner e bunu tanıtırsak aşağıdan 5 tane sayı girebiliriz*

***Eleman sayısı kadar Scanner den giriş yapma. Array ve Scanner***

Scanner sc=new Scanner(System.*in*); ***array scanner***

String[] dizi3=new String[5]; ***scanner array***  
for (int i = 0; i <dizi3.length ; i++) {  
 dizi3[i]=sc.nextLine(); dizi3 []=sc.nextLine(); forinin üstüne yazılmıyor next te next line da  
}  
System.*out*.println(Arrays.*toString*(dizi3));

Yukarıdakinden farklı olarak **Scannerden girilecek kadar eleman sayısı ile doldurma** for loop ile

Scanner sc=new Scanner(System.*in*);  
System.*out*.println(" KAÇ ADET ELEMANLI SAYI OLUŞTURMAK İSTİYORSANIZ GİRNİZ");  
int[ ] dizi3=new int [sc.nextInt()];  
  
for (int i = 0; i <dizi3.length ; i++) {  
 dizi3[i]=sc.nextInt();  
}  
System.*out*.println(Arrays.*toString*(dizi3));

**String** ve Scannerden Eleman girme in Array

Scanner sc=new Scanner(System.*in*);  
System.*out*.println(" KAÇ ADET ELEMANLI SAYI OLUŞTURMAK İSTİYORSANIZ GİRİNİZ");  
String[] dizi3=new String[sc.nextInt()]; eleman sayısı gireceğimiz için integer olmalı tane  
  
for (int i = 0; i <dizi3.length ; i++) {  **next** yazmalısın nextline yazarsan scanner den yukarıda girmek istediğin ilk değer ide  
 dizi3[i]=sc.next(); hesaba katıyor. 3 adet elemanlı bir array oluşturmak için yukarıda 3 yazdığında ilk girilenine dikkate   
} alıp aşağıda 2 tane sayı girebiliyorsun. Ama next yapınca yukarıda 3 yazınca aşağıda da 3 sayı girebiliyorsun.   
System.*out*.println(Arrays.*toString*(dizi3));

#### teach Arrays.toString,

içinde tek değerin olduğu değişkenlerde değişken değer bir komut ile yazdırabiliyorduk. Dizilerde değişkenin içindeki değerlerin tamamını yazdırmamak için değişkenin hangi elemanının değerini yazdırmak istediğimizi bildirmemiz gerekiyor.

Bir vagonun adresini gösterip koltuklarda oturan yolcuların ismini öğrenemeyiz.

**1. Yöntem.**

for(int i=0;i<dizi.length;i++) {   
 System.*out*.println("dizi[i] = " + dizi[i]); dizi[i] diye yazdırırsan dizinin i. Ci elemanının değerini yazar.

System.*out*.println( i); *//0,1,2,3,4 diye yazar eğer dizi [i] ninelemanları 3 4 5 8 gibi karışıksa birebir onları sırası ile yazar.*

}**Ekrana yazdırma 2.Yöntem**

Tamamını yazdırmak istersek kullanılması gereken yöntem: for döngüsüne gerek yok.  
 System.*out*.println(Arrays.*toString*(dizi)); 1d integer i de yazar *[1, 2, 3, 4, 5, 6, 7]bu şekilde String olarak yazar. Virgülsüz istersen loop ile yap.*

System.*out*.println("dizi2[] = " + dizi2[1][1]);

#### for each

bu yöntemde sırası ile yazdıracağından emin olamayız. Dedi hoca ama mentorler sırası ile de yazar dedi

*// Ekrana yazdırma 3.Yöntem : Dizi Array ının içindeki elemanları değer adını vererek döngüye gönderiyor.*

for (int deger: dizi ) { *//dizini içindeki elemanları değer e ata ve yazdır.*

System.*out*.println("deger = " + deger);

} *Tanıtılmış olan değişken in ismi yazdırılmak yerine asign edilen değişkenin adı normal bir şekilde for each ile yazdırılabilir. Bu yöntem uzun uzun for döngüsünü yazmaktan kurtarır. Başka bir değişkene assign etmemizin sebebi elimizdeki değişkenin array olması bu şekilde tek tek yazdıramayacağımız için.*   
 for(int eleman : dizi){  
 System.*out*.println("eleman = " + eleman);

**Forloop ve ForEach Mantığı**

Dakika 12den sonra for ve for each yöntemleri deneme 2 çözümlerinde geçiyor.

Foreach değerleri sırası ile getirir.

Foreach ta indexleri kontrol edemeyiz

İndex ile uğraşmayacaksak foreach daha avantajlı

Sadece yazdırma işlemi yapacaksak for each daha kullanışlı. İşlem yapacaksak fori daha avantajlı.

For lopta indexlere dokunabilir üzerinde işlem yapabiliriz. Lenght ine kadar değil de belirli bir elemanı yazdır 3 ,4,5 diyebiliriz. Ancak for eachta break ile veya başka bir şey ile bölmezsek hepsini yazar.

int[] deneme1={1,2,3,4,5};  
 *//deneme 1 array dir. cvp nin array özelliği yok. bir integerdir deneme 1 in her bir* for(int cvp :deneme1){ *//elemanı integerdir. integer olan cvpnin içine atılıp konsolda yazılabilir.* System.*out*.println(cvp);  
 if(cvp==3);  
 }

int[][] disArray={{ 1, 2, 3, 4, 5}, { 56, 23}};  
 *//x[0][0] x[0][1] x[1][0] x[1][1]  
 //x[0] x[1]  
  
 forloop aşağıdaki 2 tane for each in veya 3 boyutlu for each in yaptığı işlemi yapıyor.* for(int [] icArray: disArray){ *//int d2 yi tanımlayıp integer olarak bıraksaydık array olduğunu belirtmek için* for(int değerInt:icArray) *//[] kutu koymasaydık hata verecekti. çünkü dis Array deki array olan 2 elemanını* System.*out*.println(değerInt);*//dizi halinde icArray e atamazdı iç array da değerleri integer olarak alıp* } *// okuyup yazdırabilmek için bir for integer e ihtiyaç duyar.* int[][][] boyut3Array={ { { 1, 2, 3, 4, 5},{ 56, 23} } };  
  
 for(int[][]boyut2Array: boyut3Array){  
 for(int[]boyut1Array: boyut2Array)  
 for(int normalİnteger: boyut1Array)  
 System.*out*.println(normalİnteger);  
 }  
 }  
**}**

String [][] nums= {{"$12" , "$22" , "5"}, {"€9" , "€40" , "$1","$2" }, {"€12"}};  
int total=0;  
   
for(String[] satir :nums) *// nums iki boyutlu array olduğu için tek boyutlu String satır array açıldı ve* for(String sutun : satir) { *//onun da başka normal bir string açarak ona attık. String diye başta belirterek* if (sutun.contains("$"))

**For each in içine caunter atarak döngü sırası değiştirilebilir.**

Soru= *Verilen String arrayı ters çevir, ters arrayi yazdır*Scanner scan = new Scanner(System.*in*);  
 String myStr = scan.nextLine();  
 String[] useThisArray = myStr.split(" "); burada önce string olarak değer konsoldan alınmış split edilerek array e çevrilmiş

String [] tersArray= new String[useThisArray.length];*// oluşturulan yeni array in uzunluğu use this array kadar olsun.* int j=useThisArray.length-1;  
for(String str: useThisArray) *//string olarak da değer atanabiliyor sadece int değil. demek başka değerlerde atanabilir.* tersArray[j--]=str; System.*out*.println(Arrays.*toString*(tersArray));

UsethisArray içindeki değerleri tek tek yazabileceği bir değişkene ataması gerekiyor. Çünkü usethis array bir dizi ve dizi halinde yazdıramayız. biz tek tek elemanları görebileceğimiz ve tek tek yazdırabileceğimiz bir yere **ihtiyacımız var. Str** ye atacak ancak biz tersten yazmasını istiyoruz. Str değerleri alıp ters olarak başka bir yere atması gerekiyor ki biz ters şeklinde yazdırabilelim. Each döngüsünün tersinde hareket eden bir şeye ihtiyacımız var bunu da bir caunter ile ve caunter in başlangıç değerini yüksek ten aşağıya doğru hareket ettirecek bir kurguya ihtiyaç var

**Farklı Bir Yazdırma Tarzı**

TahirHocaSoru { *[1,5] yazdırın.*   
 int[] nums={1,2,3,4,5}; *// [1,5] yazdırın. 5 elemanlı yer hafızada ayrıldı* int[]soru=new int[2]; soru[0]=nums[0];*//genel bir tanım her dizide [0] vardır  
 // nums[4] olsaydı spesifik bir tanım olacak tı her dizide yok* soru[1]=nums[nums.length-1];*// genel bir tanım* System.*out*.println(Arrays.*toString*(soru));

#### Gün 22Java Methods: Create, call,void,return | Part 1 - 06/23/20

**Konuyu çalıştıktan sonra tekrarda öğrenilenler**

1 public static void main(String[] args) {  
 int [] Array = new int[5];

*addElements(Array); bunu burada yazıp bir de aşağıda aynı şekilde yazıp so deyip çıktı alamaya çalışınca metod içindeki   
 // işlemi istemediğimiz halde 2 kez yaptı* System.*out*.println("addElements(Array) = " + *addElements*(Array));  
}

*2 // retun metod da metod içinde retun yazdıktan sonra so deyip çıktı alamadım so yu mainde alabildim java kursun* *da* *retundan sonra so yazınca çıkıyor*

3 metod a parametre yazmak için mainde variable tanımlamana gerek yok metod u paramete alacak şekilde yazıp metod etiketini mainden call edince metod etiketinin(parantezine parametre yazabilirsin) replit sorularında örneği var

KONUYU ANLAMAYA YARDIMCI ÖRNEK SORU

*/\*Ismi randomNum olan bir method oluşturun.  
  
Parametre olarak int max almalı.  
Bu method, 0 ile max arasında random bir değer döndürmelidir.  
  
Random numarayı döndürünüz.  
  
Note: Eğer bir class oluşturmaya ihtiyaç duyuyorsanız oluşturabilirsiniz..\*/*

public static void main(String[] args) {  
 Scanner sc= new Scanner(System.*in*);  
 System.*out*.println("Random sayı üretilebilecek max sayıyı giriniz");  
 int random=sc.nextInt();  
 System.*out*.println("random = " + *randomNum*(random));  
 System.*out*.println("random = " + random);  
  
 *//burada tanımlanan c değişkeni ile işlem yapmak istersen yazdırma kısmında aşağıdaki metodun etiketini ve etiketin içindeki parantez kısmına c değişkenini yazman gerekiyor  
// aşağıdaki metod kısmındaki işlemi yukarıdakine göre yapıyor sebebi ne ?ilk önce okumaya yukarıdan başlıyor diye olabilir dene   
 // yoksa önce o işlemi gerçekleştirip sonra c yi deneme amaçlı tanımladığım için mi?* int c=5;  
  
 System.*out*.println("randomNum(c) = " + *randomNum*(c));  
  
 }  
 public static int randomNum(int d)   
   
 { *// int d=5; bunu burada kabul etmedi zaten d değişkeni tanımlı dedi metodun parametre kısmında tanımlı.* int ran=(int)(Math.*random*()\*d+1);  
 return ran;  
 }  
}

EvJavaMetod2   
public static void main(String[] args) {

**call,**

*( İçi boşta olabilir)  
ismeMerhabaYaz*(ad"ismet");*// ismeMerhabaYaz (String ad) String olarak tanıtılan ad değişkeninin içine gönderdi ismeMerhabaYaz*(ad"ayşe"); *// bu vermiş olduğumuz parametreler gidiyor kim karşılıyorsa onun içinde yazılmış olan fonk. Yapıyor*

**}**

**Create,**

**p**ublic static void ismeMerhabaYaz(String ad) *// ben bur da parantez içinde ad yazınca yukarıda çağırmış olduğum yerde parantez içine otomatik olarak kendisi yazdı*

Aşağıda oluşturduğumuz metod un etiketi ile mainde onu cal ettik ve main den girilecek bir bilginin temiz bir şekilde sadece fonksiyonda görünmesini

ve mainden girilen tek bilgi ile metod kısmında kodunu yazmış olduğumuz bilginin yeni halini işleyebildik.

**Return**

anahtar ifade **aşağıdaki metod parametre olarak parantez içine ne alıyorsa türü ile birlikte onu yazacağım (String int)yazacağım** . **main e hangi işlem return ediliyorsa return kısmına onu yazacağım(yapılan işlem max min, ortalama)** **void yerinede return edileceğin türünü ( string int double )yazacağım türünü yazacağım**

JavaMethod6 {*//void geriye bir şey göndermeyen yani return olmayan*

public static void main(String[] args) {  
 *// Kullanıcıdan 2 sayı isteyiniz, toplamını bir method da yaptırınız, sonucu mainde yazdırınız.* Scanner oku=new Scanner(System.*in*);  
  
 System.*out*.print("1.Sayı =");  
 int sayi1=oku.nextInt();  
 *// gönderilenler bunlar. Yukarıdaki tanımlanmış değişkenler yazılmak zorunda ama aşağıda*  
 System.*out*.print("2.Sayı ="); istediğini yaz void in yanındaki kısımdan yakalıyor.  
 int sayi2=oku.nextInt();

*// return yapınca sonuc açıldı*  
 int sonuc = *toplamYaz*(sayi1,sayi2); return etiiğimiz için main kısmında bu açıldı aşağıda yapılan işlemi buraya yazdırabildik

Bir değişkene atayıp sonucu direk değişkenin ismi ile yazdırabiliyoruz   
 System.*out*.println("sonuc = " + sonuc); sonuç= istediğimiz isim ile yeni oluşturacağımız değişkene atayabiliriz   
   
 int enb = *enBuyuk*(sayi1,sayi2); *// int enb= Math.max(3,4); bunun gibi en büyükte Math.max gibi bir metod .math max ı* System.*out*.println("enb = " + enb); *bir değişkene atadığımız gibi enb yide atayabiliyoruz*

*//int max=Math.max(3,4); gibi bir şey*

}

*/String olsaydı //void //(sayı1,sayı2); parametre sayısı aynı olmak zorunda*

public static int toplamYaz(int s1, int s2) */yukarıdaki isim ile aynı olmak zorunda değil. Etiket ten yakalıyor*

*//void geriye bir şey göndermeyen yani return olmayan // yukarıdaki 1. aşağıdaki 1 ile, 2. de 2. ile eşleşiyor* {  
 int toplam= s1+s2;  
 *//System.out.println(toplam); normalde burada yazıyordu*  
 return toplam; burada yazacağımız değişken değeri ismi ile yukarıdaki aynı olmak zorunda değil nasıl istersen öyle gösterirsin burada ve mainde  
 }

#### Gün 23 Multiple Parameters, Method Overloading, Varargs, Return Örnek 2-06/24/2020

**VARARGS …** variable argumants  **bir dizidir dizi olduğu için array şeklinde out print alınır.**

parametre parantezinde her zaman sonda olmalı ve bir kere kullanılabilir aynı metod içinde

**varargs örneklerine bak**

*parametre sayısın ve tipi farklı olduğunu durumlarda aynı metod adı kullanılabiliyor.*

int sonuc= *toplam*(sayi1,sayi2);  
int sonuc1=*toplam*(sayi1,sayi2,sayi3); *parametre sayısından*

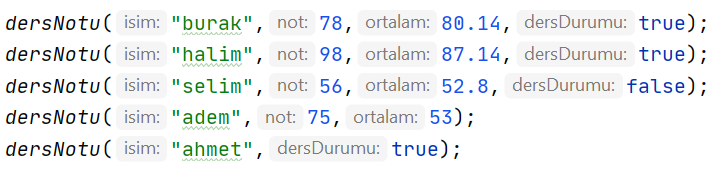
*parametrelerin tip sırası farklı olduğunda aynı isim kullanılabiliyor.*

***KullaniciBilgiYaz***(ad"Ahmet",yas34); aynı isimde 1 den fazla kullandığın için **Overlooad**

*KullaniciBilgiYaz*(yas34, ad"Ahmet");

**MetodOverloading2** {

Public static void main(String[] args)

{  
  }

*Oluşturulan parametrelerin varible type lerini ve variable name yazdıktan sonra mainde etiket ile call edince parantezin  
içine her girdiğim varible değer. Kendiliğin den ismi ile çıkacak sırasıyla*

**Overloading** aynı isimde birden fazla kez metod kullanıldığı için overload  
 public static void dersNotu(String isim, int not,double ortalam, boolean dersDurumu) **multiple parameters**  
 {  
 System.*out*.println("isim " + isim+"sınav notu"+not+"ortalaması"+ortalam+"dersiGeçebildimi"+dersDurumu);  
 }  
 public static void dersNotu(String isim, int not,double ortalam)  
 {  
 System.*out*.println("isim " + isim+"sınav notu"+not+"ortalaması"+ortalam);  
 }  
 public static void dersNotu(String isim, boolean dersDurumu)  
 {  
 System.*out*.println("isim " + isim+"sınav notu"+dersDurumu);

*tekmiçiftmi*(a); bir metod etiketi ile başka bir metod dan ağırılabilir.  
*tekmiçiftmi*(b);  
 }  
} **Eğer mainde Değişken tanımlanmışsa.**

}

1 Aşağıda create ettiğin metod u mainden call edince en yukarıda tanıtılmış olan değişkenlerden hangisi ile işlem yapmak istiyorsan onu parantez içine parametre olarak yazacaksın .

**2** Aşağıdaki metod un etiketi ve mainde çağırmış olduğun metod un etiketi ile eşleşecek ve parantez içindeki parametreler ilgili metod a otomatik olarak gidecek. İşlem burada gerçekleşmiş olacak.

**3** Metod da ki parametre ismi ile yukarıda tanıtılmış olan parametre isminin aynı olmasına gerek yok .  
**RETURN** Return yapmayınca main de so deyip yazdıramazsın

evJavaClass1 {  
 public static void main(String[] args) {  
  
 System.*out*.println(" merhaba dünya1"); *// 100 satırlık programa sürekli bunu yazmak yerine aşağıda bir kere tanıtır*  
 *merhabaYaz*();*// buradan call ederim .değiştirmek istediğimde aşağıdan bir kere değiştiririm hepsi değişir.* int enb=Math.*max*(3,4);*/ mat bir classtır. Max Mat ın bir fonksiyonu max tan döneni enb ye atıyor. Enb ye atmazsak bir işe yaramaz* int enbTurkce=*enBuyukBul*(a3, :b:4); *// rakamları yazmak yetiyor a yı be yi kendi yazıyor.  
 // tanımladığımız fonksiyonda return olan burada çıkıyor. Çağırdığımız için*String adsoyad= *AdSoyadBirleştir*("Ali", "Demir"); *// string ve adsoyad= yazılmalı returnden sonra.*

System.*out*.println("adsoyad = " + adsoyad); *Ad soyad birleştiri return ettimiz için bir değişken türü ve ismi istiyor.*   
  
 }  
 public static String AdSoyadBirleştir(String ad, String soyad)  
 {  
 return ad+" "+soyad; *// return ettiğimiz için istediğimiz değeri ve string olarak tanıttığımız . için mainde ilgili kısımdan buna*

*ulaşabildim retur etmeseydik sadece buradan konsola gönderebilirdik main e gitmezdi  
 main e ad soyadbirleştir ile alakalı bölüme bunu return*

*//System.out.println(ad+" "+soyad); bu işlem void olduğu için sadece yukarıda call ettiğimiz burada tanımladığımız  
 //için sadece konsolda yazar. main e ad soyadbirleştir ile alakalı bölüme bunu return edemeyiz* }

public static void merhabaYaz()  
 {  
 System.*out*.println(" Merhaba Dünya");  
 }  
}

Printline ve Math maxda birer metod; bu isimlerin altına tanıtılmışlar arka planda çalışıyor bize gösterilmiyorlar,

çağırdığımızda Yazdığımız değerleri otomatik olarak kendi türlerine atan ve işlemleri gerçekleştiren metodlar.

evJavaMetod2 {  
 public static void main(String[] args) {  
 */\* int sayı1=5;  
 int sayı2=7;  
 int toplam=sayı1+sayı2; bu normalde böyle yazılırdı. ama 100 satır içinde çok karışık olurdu o yüzden fonk. yaptık  
 System.out.println("toplam = " + toplam);  
 \*/*int sayı1=5; böyle 100 metodluk işlem olsa main karmaşadan kurtulur parametreleri yazar işlemi yaparsın   
 int sayı2=7;

//returnsuz hali  
 *toplamBul*(sayı1,sayı2); Bunu bu şekilde buraya tanımlayıp aşağıdan işlemin sonucunu alırsın

ama burada sonuç almak istersen alamazsın ve mainde so deyip yazdıramazsın  
   
 int sayı3=7; *// bana yukarıdaki toplamın değeri lazım aşağıdan almak için return yapmak gerekli  
 // return dan sonra durum*int sonuc= *toplamBul*(sayı1,sayı2); *// Math.max(3,4);bir yere atamazsak başına eşit koyup tanıtmazsak bir işe yaramaz öylece durur return da aynı böyle*

*sonucu metod dan alabiliyorsak return a negerek var ? bazen sonucu değişkene atayıp onunla tekrar işlem yapmamız gerekebilir.*

}   
 public static int toplamBul(int a,int b)  
 {  
 int toplam=a+b; *// burda parantez içinde işlemler gerçekleştiği için parantezin değerleri tanıtılmalı a, b* System.*out*.println("toplam = " + toplam); *// her iki tarafa da yazabilirsin sana kalmış.* return toplam;  
 }  
}

#### Gün 23 Mentoring TreeMap ve Metodlar da kullanımı | 06/07/2020 izledim 26.06.2020 de yapılmış ders

Aşağıda tanımladığımız metod u etiket ismi ile Main den çağırdığımızda etiket in parantez içine yapmak istediğimiz işlemi parametre olarak yazarak (örnek yukarıda main in hemen altında tanımlanan değişkenleri veya aşağıdaki haliyle.) main in dışında oluşturduğumuz metod da (fonksiyon) işlem yapabiliriz ve yine burada yazdırabiliriz.

İstersek yazdırma işlemini main kısmında da yapabiliriz.

MentorinTahir5 {*// bende 4. class ama*public static void main(String[] args) **{**

call  
 *teaching*(name1"teacher", name2 "student"); silme işlemi metod da yaptık burada sadece call ettik ve parametre kısmına aşağıda gerçekleşen işlemlerin aynısının yapılacağı yeni değişkenler tanımladık

String DeğiskeneAtanmısHali= *Yontem2geriReturnİleYazdırma*(name1"hoca", name2 "talebelere");  
  
 System.*out*.println("DeğiskeneAtanmısHali = " + DeğiskeneAtanmısHali); *aşağıdaki fonksiyonun etiketini yazmaya gerek yok  
 //çümkü yukarda yeni etiket verdik* System.*out*.println(*geriReturnİleYazdırma*("oğretmen","oğrencilere"));  
  **}**

metod olduğu için ve her yerde kullanma ihtiyacı duyabileceğimiz için hem burada hem de aşağıda sıra ile girmek zorunda değiliz   
 public static void teaching(String name1, String name2)  
 {  
 System.*out*.println(name1 +" teaches Java to"+ name2);  
 }

public static String geriReturnİleYazdırma(String isim1, String isim2)  
 {  
 return isim1+" ogretiyor JAVA yı "+isim2;  
 }

public static String Yontem2geriReturnİleYazdırma(String isim1, String isim2)  
 {  
 return isim1+" ogretiyor JAVA yı "+isim2;  
 }  
}

teacher teaches Java tostudent

DeğiskeneAtanmısHali = hoca ogretiyor JAVA yı talebelere

oğretmen ogretiyor JAVA yı öğrencilere

METOD KISMINDA

String str=" "; burada bir değişken tanımlayıp return ü öyle kullanmakta fayda var dedi hoca

**METOD UN İŞLEYİŞ SİSTEMİ Nİ ANLAMAK İÇİN AÇIKLAMALARI OKU**

copyReverseString {  
 public static void main(String[] args) {  
  
 String str="Java'yı Seviyorum";  
 String deneme="sdf";

//return ettiğimiz için burası açıldı yazmak zorunda değilsin  
 String ters= *reverseString*(str);

System.*out*.println("ters = " + ters);

Bu ikisi aynı

System.*out*.println("reverseString(str) = " + *reverseString*(str)); reverse string etiketi içinde gerçekleşen işlemlerin. str stringin i ile yazdırılmış hali

Sen aşağıdaki (parametre kısmına parantez içine variable name e ne istersen yazabilirsin.) Orada gerçekleşen işlemler sana bir örnek oluyor. O işlemlerin aynılarını yukarıda tanımlanan değişkenlerin hangisine uygulamak istersen aşağıdaki metod un isminin yanına yukarıdaki varible name i yazarak işlemi gerçekleştirip konsoldan çıktı alabilirsin.

İşlemi yaptırırken yani so deyip çıktı alırken Aşağıdaki metod ismini yazdıktan sonra yukarıdaki hangi değişkeni kullanacaksan onun ismini yazmak zorundasın

Aşağıdaki sonucu sen zaten return ve etiket ile alıyorsun return yazan kısımdaki ctr ismini burada görmeye çalışma içindeki değer zaten reverse string etiketi altında geldi  
  
 System.*out*.println("denemeString(deneme) = " + *denemeString*(deneme));  
 metodların etiketleri içinde aynı işlemler gerçekleştiği için hangi metod isminin yanına yukarıdaki hangi değişkeni yazarsan yaz sonuç fark etmiyor önemli olan metod içinde gerçekleşen işlem  
 System.*out*.println("reverseString(deneme) = " + *reverseString*(deneme));  
 }  
 public static String reverseString (String atr)  
  
 { String ctr="";  
  
 for (int i = atr.length()-1; i >=0 ; i--) {  
 ctr+= atr.charAt(i);  
 }  
 return ctr;  
 }  
 public static String denemeString (String denem )  
  
 { String ctr="";  
  
 for (int i = denem.length()-1; i >=0 ; i--) {  
 ctr+= denem.charAt(i);  
 }  
 return ctr;  
 }

#### Gün 24 Methods ArrayList , Collections.methods,Wrapper Classes,Autoboxing, String.Format,printf 06/25/2020

import java.util.ArrayList; *Scannerde ki gibi bir kütüphane eklendi*public class evArray1 {  
 public static void main(String[] args) { *Array hangi durumlarda kullanılır. boyut sayısının belli olduğu ve değerlerin çok sık değiştirilmediği durumlarda.*  
 int [] dizi=new int[5]; *// daha sonradan boyutu değiştirilmiyor. ram dan o kadar yer ayrıldığı için.  
  
 ArrayList in Array den farkı nedir:*

*BOYUTU dinamiktir. Yani değişken ve değerlerin hızlı değişmesi durumunda daha hızlıdır. Elemanların azaltılıp arttırılması gereken durumlarda kullanılır.*

*< buraya data tipi belirtiledebilir belirtmesek te olur>* ArrayList<Integer> liste=new ArrayList<Integer>();  
 ArrayList<Boolean> booleanliste=new ArrayList<Boolean>();

Arraay list in intellij e tanıtılması ve 1 tane Arraylist in fonksiyonları anlatılacak aşağıda

***Arraylist tanımlaması***:

ArrayList<String> isimler=new ArrayList<>();

**Array list in fonksiyonları**  
  
 ***add******eleman ekleme*** isimler.add("ahmet");

***istenilen index teki elemanın değerini değiştirme.***isimler.**set**(0,"Deniz");  *replace gibi.*

*add elemanı ekliyor. Set ise verdiğin yerdeki elemanın değerini değiştiriyor.*

***belirli bir index e isim ekleme***isimler.**add**(1,"zeynep");

*eklendiği indexten sonrakiler bir satır aşağıya kayar.*

***istenilen index teki elemanı get etme***String eleamanAtindex2=isimler.**get**(2);

***eleman sayısını alma size***isimler.**size**();

*lenght gibi*

*listenin* ***tamamen boş olup olmadığını*** *verir* boolean listeBosmuDegilmi=isimler.**isEmpty**();

***eleman silme***

*sildiği yer boş kalmaz.. arkasındaki ler silinenin yerine gelir* isimler.**remove**("ayşe");

*büyük küçük fark eder 2 tane varsa ilkini siler*

***istenen indexteki elemanı silme*** isimler.**remove**(1);

*index i girerken tırnak içinde girersen index değil de string olarak algılıyor eşleşme olmayınca da silmiyor.*

***İstenen elamanın indexini bulma***int indexofKaya=isimler.**indexOf**("kaya")

*lastIndexOf, toArray aynı dizilerdeki gibi*

***index of***  *index i veriri* ***get*** *değeri verir biri index i verir diğeri verdiğin indexin değerini verir.*

***tüm elemanları boşaltma*** isimler.**clear**();

*Array list in* ***kopyası clone*** *komutu ile yapılabilir*

***Arraylist*** *i array a* ***dönüştürme*** *to array koutu ile array a dönüştürebiliriz..  
array list te array a dönüşebiliyor. komut ile.*

**contains** list1.**contains**("Türkçe")

Yukarıdaki fonksiyonlarda bir listenin kendi içindeki işlemleri ile 2 listenin birbiri arasındaki ilişki ve fonksiyonlar ile yapılabilecekler gösterildi. 1 listenin kullanılışını özette. 2listenin birbiri ile kullanılışları için ise aşağıda örnekler var

**İki listenin birleri ile ilişkileri**

*parantez içindeki listeyi parantez içinde olmayana ekler.* list1.**addAll**(list2);

*list 1 e list 2 yi ekler aynı olanları da ekler. Eklenmesini istemiyorsan ya for döngüsü yada haftaya yeni bir yöntem öğreneceğiz*

list1.**addAll**(2,list2);

list1.**removeAll**(list2);

*list 1 den list 2ile ortak olan elemanları çıkar. Saf list 1 i göster. Sıraları farklı olsa da olur büyük küçük duyarlı*

**COLLECTİON*****Grubu ve Arraylist ile Yapabileceklerimiz***

*array list collection grubunun bir elemanı collection haftaya da görülecek olan diğer arraylar ın ismi Array i sıralatırken* ***Arrays.sort();*** *u kullandığımız gibi array list te collection metodlarını kullanacağız.*  
 Array lis i **sıralama** **Collections.*sort***(numbers); önceki hali*[69, 3, 1, 2]*  
 System.*out*.println("Sırlaı = " + numbers);   
 *numbers= [1, 2, 3, 69]*

*/ max bulur* int max=Collections.*max*(numbers);  
  
 */ min bulur* int min=Collections.*min*(numbers);  
  
 Array list i **reverse** Etme *//Sıralı = [1, 2, 3, 69]*  
 Collections.*reverse*(numbers);  
 System.*out*.println(" tersi = " + numbers); *reverse den sonra halinin = [69, 3, 2, 1]*

*Array list i bir değere göre* ***fill*** *elem* Collections.*fill*(numbers,101);  
 System.*out*.println("numbers fill den sonra = " + numbers);  
 *numbers fill den sonra = [101, 101, 101, 101]  
  
Arraylist de belli bir değere sahip elemanların* ***hepsini*** *yenisi ile replace****All***Collections.*replaceAll*(numbers,101,5);  
 System.*out*.println("replaceAl dan sonra = " + numbers);  
 *//replaceAl dan sonra = [5, 5, 5, 5]*

System.*out*.println("1. isimler = " + isimler); ***// normal şekilde yazdırılabilir.***

*// arraylarda elemanları arttırıp çıkaramıyorduk burada istediğimiz kadar ekleme çıkarma yapabiliyoruz.  
  
 // dizilerde yapılıp bunda yapılamayan bir şey yok . array list genişleyip daraltılabiliyor.  
  
 // programda eleman sayısı sabit ise bunu sınırlandırmak isteyebiliriz. o yüzden array lar tercih edilebiliyor böyle durumlarda  
 // array kullanımı biraz daha kolay.   
  
 // integer de olsa elemanın değerini yazarak index ini alabiliyorsun 2 nolu index teki elemanı istersen get ile alacaksın dedi anlamadım. char atteki ile aynı*

**Methods ArrayList** yukarıdakinden farklı olarak array list e başlangıçta farklı şekilde eleman atanması gösteriliyor . Yukarıdakinde 1 tane liste ile işlem yapılıyordu burada 2 liste ile ve bu 2 listenin bir biri ile ilişkisi inceleniyor

*array da başlangıçta belirlenen eleman sayısı değişmiyordu array list te değişiyor.*

JavaArrayList3{  
 public static void main(String[] args) {  
  
 int [] dizi= new int[] {2,3,4,5}; *// arraylar böylede tanımlanabilir* ArrayList <String>list1=new ArrayList<>() *// ARRAY LİST E BAŞLANGIÇTA ELEMAN ATAMA.* {  *başlangıçta eleman ekleme diğerinin asine daha zor (add yapıp eklemek daha kolay)* {  *parantezler arasında işlem yaptığımız için variable name yazıp eklemeye gerek yok.* add("Almanca");  
 add("İngilizce");  *ekleme yaparken aralara virgül koymaya gerek yok* add("Türkçe");  
 add("Rusca");  
 }  
 };  
 System.*out*.println("list1 = " + list1); so= list1 = [Almanca, İngilizce, Türkçe, Rusça] virgülü kendisi koyuyor.  
  
 ArrayList<String>list2=new ArrayList<>();  
 list2.add("Rusca");  
 list2.add("Türkçe");   
 list2.add("Arapca");  
   
 **ARRAYlist 3 ü ARRAYlist 2 nin içine yazmışım ben 3 eksik değil 4 den devam et.**

*for döngüsü Java nın kendi fonksiyonlarının yapamadıklarını devreye girip manuel olarak yapıyor.*

**String.Format,printf PrintAndFormat** {} evJavaEvPrintAndFormat {   
 public static void main(String[] args) {  
  
 int a=120;  
 double oran=0.25567687;  
 String str;  
  
 System.*out*.println("a = " + a);  
 System.*out*.println("oran = " + oran);  
  
 ***bazen noktadan son ra 2 hane olsun .falan gibi formatlı yazdırmalar , format ve print İLE YAPILIR.***

*Tam sayılar için d  
 ondalık lı sayılar için f devreye girer.  
  
 // birinci birinciye 2. 2 ye gitti* str=String.*format*("a=%d oran=%f", a,oran); *% işaretleri değişken yazılacağını gösteriyor . d ve f format olduğunu gösteriyor bir string in içine düzenli yerleştirildi*

*BİR STRİNG İN İÇİNE YUKARIDA TANIMLANAN DEĞİŞKENLERİ yazdırmayı istediğimiz yere TAM SAYI İÇİN d ondalık lı sayılar için f koyarak çıktısını aldık* System.*out*.println("1str = " + str); *.  
 // so = 1str = a=120 oran=0.25567687*

System.*out*.println("a="+a+" oran="+oran);  
 *// so a=120 oran=0.25567687 \n alt satıra geçmesi için // alt ta farklı değişkenleri bir arada yazdırmak için print f kullanılmalı deniliyor burada duble ve integer birbirlerine benzedikleri içinmi yazdırmada problem çıkmadı yoksa string olarak mı yazdı*

System.*out*.printf("2. a=%d oran=%f pi=%f\n", a,oran, Math.*PI*); *// print F printline değil. Farklı değişkenleri aynı komut ile yazdırma  
 // so2. a=120 oran=0,255677, pi=3,1415933.*

*// girilen rakam kadar sağa dayalı yazdırma d veye f formatının önüne yazdığımız sayı kadar.*

System.*out*.printf("3. a=%10d oran=%f\n", a,oran); *10d 10 hane kullan/ bu şekilde yazarsak 1leri 1 lerin 10 ları onların altına yazdırabiliriz  
//so 3. a= 120 oran=0,255677 sağa dayalı yazdı // düzenli bir şekilde yoksa karmaşık görünebilir.*

System.*out*.printf("4. a=%-10d oran=%f\n", a,oran); *// -10d sola dayalı yazdırma  
//so4. a=120 oran=0,255677 sola dayalı*

System.*out*.printf("5. a=%010d oran=%f\n", a,oran); *// 10d 10 hane kullan 010d ama boş olan yerlere 0 ata  
//5. a=0000000120 oran=0,255677*

*//f float demek değil float a karşılık gelmiyor. burada ondalıklı sayı demek default olarak float a göre davranıyor* System.*out*.printf("6. a=%d oran=%.3f\n", a,oran); *// .3f noktadan sonra 3 hane kullan  
 // 6. a=120 oran=0,256* System.*out*.printf("7. a=%d oran=%.9f\n", a,oran); *// .9f noktadan sonra 9 hane kullan  
 //7. a=120 oran=0,255676870* System.*out*.printf("8. a=%d oran=%6.1f\n", a,oran); *// virgül dahil 6 hane kullan noktadan sonra 1 tane olsun  
 //8. a=120 oran= 0,3*

**Wrapper**

evJAvaWrapperClasses6 {   
 public static void main(String[] args) {  
 *//wrapper kaplamak demek metodları kapladık içinde çok metod var.  
  
 // programda sadece rakama veya harf gibi veya true false gibi basit değerler saklayacaksa hafızada çok yer kaplamasın diye  
 aynı tiplerin basit yani primetive tiplerini oluşturalım.*

*// eğer bu değerlerden fazlası gerekecekse o zaman bunların ilk harfi büyük halleri olan tipleri kullanılsın  
  
  
 //büyük harf olanlar daha fazla yer kaplıyor WRAPPER ediyor. sadece sayı saklayacaksak rakam saklayacaksak küçük halleri daha mantıklı  
 // hep büyük halleri olsaydı 1 tane rakam saklamak için bile kompleks bir yapı kullanacaktık.  
  
 // büyük harfle başlayanlar kapsamlı olanlar. zaten bu değerlerin yanına tıkladığımızda çok fazla fonksiyon seçeneği çıkıyor*int a=5;  
 String strInt=String.*valueOf*(a); *//bu ise basit sadece bir sayı.  
  
 // bu iki tip farki primitive ve non primetive tiplerden kaynaklanıyor* Integer b=6;  
 String strInteger= b.toString();*//yani bu tipin içinde gerekli çokça metod var. Wrapper class büyük harfle başlayan  
 // tüm tipler için geçerli.*

**AutoBoxing** **Unboxing**

b=a;*// boxing denir basit olan tipin, kapsamlı tipe atanmasına denir.   
 // boxing kutulama paketleme* a=b;*//unboxing kapsamlı tipin basit olan tipe atanmasıdır.* double c=3.4;  
 Double d =5.6;  
  
 char e='F';  
 Character f='Y';  
  
 float k=3.4F; *// temelde double kabul etttiği için float konmalı* Float l=5.6F;  
  
 *String in küçük hali yok string in kendisi kompleks bir yapı primitive tipi yok*

#### Gün 26 2D ArrayList ve Metodlar ile kullanımı | Part 1 - 06/29/2020

for (int i = 0; i <matNotlar.size() ; i++) {  
  
 System.*out*.println(matNotlar.get(i));  
}

**Arraylist** ın içindeki elemanlara **loop** ile ulaşırken  
**size** ile listenin içinde sonuna kadar gidilir.

Listenin içinde giderken istediğimiz yerdeki notu almak işlem yapmak veya yazdırmak için   
**get(i** )kullanılır.

**YAZDIRMA SEÇENEĞİ**

Eğer2 liste varsa 1 döngü kullanılıp dış listenin ismi ile döngüde ilerler yip döngü içinde dış döngü ismi ve get i dersek i. satırdaki dataları yan yana bir parantez içinde string olarak verir.

for(int i=0;i< dışlisteİSMİ.size();i++){

System.*out*.println(dışlisteİSMİ.get(i));

}

[90, 80, 70]

(95, 85, 75] STRİNG

*[45, 55, 65, 35]*

**2 bilinmeyenli değişkenlerde Array** **list** in Array List **loop in loop** ile ulaşılır.   
   
1 **ARAYLİS in içindeki Arraylist e ulaşmak için** dıştaki listenin içinde sonuna kadar **size ile gidilir**   
2 **Dıştaki listede ilerlerken** **istediğimiz sıradaki** listenin elemanlarına ulaşmak için **iç loop ta get (i)** **kullanılır**  
3 **Aldığımız** listenin elemanları içinde **ilerlemek** **için ve sonuna kadar gidebilmek için** tekrar **size** kullanılır

*4 size içinde ilerlerken istediğimizi yazdırabilmek için yazdırma kısmında* ***get (i)*** *[get edip aldığımız dersin]* ***get(j)[****get edip alıp] yazdırarak istediğimiz kişinin notunu yazdırırız.*

**YAZDIRMA SEÇENEĞİ 2**

tek döngüde notları bir arada veriri notları teker teker ele almak istersen çift döngü kullanmalısın.for (int i = 0; i <notlarArrlist.size() ; i++) {

System.*out*.println("notlarArrlist get i = " + notlarArrlist.get(i)); STRİNG  
 [90, 80, 70]

(95, 85, 75]

*[45, 55, 65, 35* for (int j = 0; j <notlarArrlist.get(i).size(); j++) { System.*out*.println("i. yaprakdaki j.not = " + notlarArrlist.get(i).get(j));} 90

} 80

70

System.*out*.println(dışlisteİSMİ.get(i).get(j)); içerideki hangi sırada hangi listenin olduğunu bilmiyoruz bu yüzden **dışlisteİSMİ.get(i).** Bize içerideki listeyi verir.

*// lenght te parantez yoktu size de var neden (parantez metod demek ama çalışma prensibini anlamadım düşün).*

*soru 1 // bana öyle bir metod yazınki metoda dersin nosunu göndereceğim bana ortalamasını verecek* public static int dersOrtalamaBul(int dersNo,ArrayList <ArrayList<Integer > > notlarArrlist)  
 { int toplam=0;  
 *//2* for (int i = 0; i <notlarArrlist.get(dersNo).size() ; i++)  
 {   
 toplam+= notlarArrlist.get(dersNo).get(i);  
 }  
*soru 2 //bana öyle bir metod yazınki tüm derslerin ortalamasını bulsun*public static int tumDersOrtBul(ArrayList <ArrayList<Integer > >notlarArrlist)  
{   
 int sum=0;  
 int notMiktarı=0; *// tüm derslerin tüm notlarının sayısı* for (int i = 0; i <notlarArrlist.size() ; i++) {  
 for (int j = 0; j <notlarArrlist.get(i).size() ; j++) {   
 sum+=notlarArrlist.get(i).get(j);  
 notMiktarı++;  
 }  
 }  
*//soru 3 bana öyle bir metod yazınki tüm derslerdeki 1. yazılının ortalamasını bulsun* public static int sınavSıraOrtBul(int sınavSıra,ArrayList <ArrayList<Integer > > notlarArrlist)  
 { int Toplam=0;  
  
 for (int i = 0; i <notlarArrlist.size() ; i++) {  
  
 Toplam+=notlarArrlist.get(i).get(sınavSıra);  
 }

#### Gün 27 | HashSet, LinkedHashSet, TreeSet , Iterator (2 şekildedir) Part 1 - 06/30/20

set in elemanlarında get olmadığı için fori yerine foreach kullanıldı

*Veriler in* ***tekrar etmemesi*** *gerektiği durumlar ve bazı durumlarda* ***eklenme sırasına*** *göre, bazı durumlarda veri setinin* ***her zaman sıralı*** *veya* ***özel sıralı*** *olması gerektiği durumlarda,* ***set*** *isimli verileri tutacak değişkenlere ihtiyaç oldu.  
Array list in bir gelişmiş hali artısı içine aynı elemanları almaması SET LERDE DİZİDİR.*

Set<Integer>hs1=new HashSet<>(); *// buda HashSet tir. Bu şekilde de tanımlanabilir.*

HashSet<Integer>hs=new HashSet<>(); Array list e benzer şekilde tanımlanır.  
boolean eklediMi\_1=hs.add(1); *true (bu işlem hem ekleme yapar hem de hangisinin eklenip hangisinin eklenmediğini*boolean eklediMi\_2=hs.add(2); *görmemizi sağlar. uzun datalarda bu gerekebilir. )* boolean eklediMi\_3=hs.add(3); *// true* boolean eklediMi\_4=hs.add(3); *// Daha önce eklendiği için eklemeyecek sonuç false* boolean eklediMi\_5=hs.add(2); *// Daha önce eklendiği için eklemeyecek sonuç false//işlemin sonucu boolean olduğu için bir değişkene atayabilirsin dedi.* System.*out*.println("hs = " + hs); *// hs = [1, 2, 3]*

**evHasSet2 { Set i Yazdırma ve Elemanları Arasında Dolaşma Kaç Tanedir En hızlısı Hangisidir** 3 tanedir.

1.string olarak yazdırma

2.foreach

3.iteratör  
 *//Arraylist teki gibi get yok . sırada yok çünkü* HashSet<String> hs= new HashSet<>();  *// büyük küçük olması önemli değil hızlı arayıp bulayım dersen Hashset*  
 hs.add("Red");  
 hs.add("Green");  
 hs.add("Blue");  
 hs.add("Red");  
 hs.add("red");  
 hs.add("rED");

*//1. yol //string olarak sadece bakmak görmek istiyorsan bunu yazarsın* System.*out*.println("hs = " + hs);*// hs = [Red, red, rED, Blue, Green]  
 //Has Algoritmasına göre bir algoritma ile saklıyor ve çıktı veriyor hızlı olacak şekilde  
  
 //2.yol Elemanlar arasında gezinmek görmek istersen bunu kullanırsın* for (String s: hs )  
 {  
 System.*out*.print(" s = " + s); *// s = Red s = red s = rED s = Blue s = Green Ln yok  
 // elemanlar arası gezinip yazdırıyor.* }  
 *//3. yol Elemanlar arasında gezinmek görmek istersen bunu kullanırsın EN HIZLISI BU*

***İteratörler setlerin başlangıçta yazılma şekline göre farklılık gösterebiliyor.***

*1.* Iterator gosterge= hs.iterator(); **hs yukarıda tanıtılan set**  
 while (gosterge.hasNext())  
 {  
 System.*out*.println("gosterge = " + gosterge.next());  
 gosterge.remove(); *// o anda gösterdiği elemanı silebiliyor yukarıdakinden 2. Yoldan farkı*

HashSet<String> hs= new HashSet<>();

*Yukarıdaki iteratör başlangıçta HashSet olarak tanıtıldığı için direkt olarak başka değişkene atanmadan yazdırılabildi.*

*}*

Set<Integer>hs=new HashSet<>();

*Aşağıdaki iteratör ün değişkeni yukarıda bu şekilde tanıtıldığı için aşağıda yukarıdaki iteratör den farklı olarak önce bir değişkene atandı sonra yazdırıldı. Direkt so deyince yazmıyor*

*2.* Iterator it= hs.iterator();*// REMOVE YAPACAKSAN ITERATOR  
 // herzaman döngüye ihtiyaç duyar* while (it.hasNext()) *// iteratör ün konumlandığı yere göre SONRASINDA ELEMAN VAR İSE koşul sağlandıkça döner.* {  
 Integer eleman=(Integer) it.next(); *//tipi her zaman obje sen kendi tipini ver. yukarıda hs ye ınteger dediğimiz   
 için Integer istedi // göstergenin gösterdiği elemanı ver (ınteger String) // int de yazılabilir ama artık Integer yazmakta wrapper fayda var* System.*out*.println("it : eleman = " + eleman);  
  
 if(eleman==34){ Sadece ilgili elemanı siler  
 it.remove();*// daha hızlı silme işlemi yapabilir.* }  
 }  
  
 *Iterator tanıtıldı while döngüsünde iterator next e sahip olduğunda yani ilerleyecek birseye geleceğe sahip oldukça devam etsin dendi  
 ve göstergede gelecek olan yazdırılsın dendi.  
 /\*  
 gosterge = Red  
gosterge = red (pc den her bir element için bir alan ayrılır sıra ile içine konur iteratör bu alanların(kutucukların) başına konumlanır.  
gosterge = rED sonraki kutucuğun içinde element varsa yazdırır.  
gosterge = Blue  
gosterge = Green  
 \*/* hs.remove("Red");*// red i siler.* hs.clear();*// set i temizler.*

*soru*

Mainde iki tane set oluştur, eleman ekle.

1Metod kısmında bunları birleştir,

2 setlerden 1 tanesinin diğerinden farkını al,

3 bu iki metod un kesişimlerini al.

4 bunları mainde yazdır.

public class evHashSet3 {  
 public static void main(String[] args) {  
 HashSet<Integer> setA = new HashSet<>();  
 setA.add(1);  
 setA.add(2);  
 setA.add(3);  
 setA.add(4); Türkçe karakter kullanmamaya alış  
 setA.add(5);

HashSet<Integer> setB = new HashSet<>();  
 setB.add(4);  
 setB.add(5);  
 setB.add(6);  
 setB.add(7);

*birleştir*(setA,setB);  
 System.*out*.println("birleşik hali = " +*birleştir*(setA,setB)); *//tekrarları eklemedi so= birleşik hali = [1, 2, 3, 4, 5, 6, 7]*

*farkiBul*(setA,setB);  
 System.*out*.println("Farkı" +*farkiBul*(setA,setB)); *so = Farkı[1, 2, 3] anın içinden b de olan elemanları çıkardı yalın a yı verdi*

*kesisimBul*(setA,setB);  
 System.*out*.println("kesişim = "+ *kesisimBul*(setA,setB)); *// so kesişim = [4, 5]* }  
 public static HashSet<Integer>birleştir(HashSet<Integer>a,HashSet<Integer>b)  
 {  
 HashSet<Integer>birlesik =new HashSet<>();  
 birlesik.addAll(a);  
 birlesik.addAll(b);  
 return birlesik; **dikkat etmen gereken yer.**  
  
 *// return işlemin sonucunu veriyormuş işlemin kendisini vermiyormuş o yüzden yukarıda başka bir değişken tanımladık  
 // işlemi yukarıda yaptık sonucu return ettik*

}   
 public static HashSet<Integer> farkiBul (HashSet<Integer>a, HashSet<Integer>b)  
  
 { HashSet<Integer> fark= new HashSet<>(a); *// yeni tanımlanan HashSet i a ya eşitledik. yeni HashSet ten b de kileri   
 çıkaracağız*

fark.removeAll(b); *anın içinden b de olan elemanları çıkarır a yı verir.*  
 return fark; dikkat etmen gereken yer.   
 }  
 *// return işlemin sonucunu veriyormuş işlemin kendisini vermiyormuş o yüzden yukarıda başka bir değişken tanımladık  
 // işlemi yukarıda yaptık sonucu return ettik*

*//retainall kesişim  
 // removeall fark ı verir.* public static HashSet<Integer> kesisimBul (HashSet<Integer>a, HashSet<Integer> b)  
 {  
 HashSet<Integer> kesisim =new HashSet<>(a);*// aya eşit oluyor.* kesisim.retainAll(b); *// retain muhafaza etmek.* return kesisim; **dikkat etmen gereken yer.**  
 }  
}

**evJavaSet5 {**

*2 add ile mainde tanıtılan elemanlar metod kısmında eklendi varargs kullanılarak  
3 set array e dönüştürüldü for each ile* public static void main(String[] args) {  
 *// new HashSet de yazabilirdik.****1*** TreeSet<Integer> rakamlar= *generateSet*();  *// generateSet(); bunu main den call ediyormuşsun gibi****2***  *addElements*(rakamlar,,…values:10,100,100);  
 *addElements*(rakamlar, ,…values 50); *// rakamlar TreeSet e eleman ekleme*

System.*out*.println("rakamlar = " + rakamlar); *// rakamlar = [4, 6, 10, 14, 50, 58, 60, 74, 80, 82, 85, 96, 100]* System.*out*.println("rakamlar.size() 2 = " + rakamlar.size()); *// 2 = 13* ***3***int []ints =*convertToArray*(rakamlar);  
 System.*out*.println(Arrays.*toString*(ints));  
}

convertToArray metod ismi mainden çağırıldığında Array diziye atandığı için. Ve ya set diziye dönüştürüldüğü için mainde ki return türü Array olarak tanıtıldı.  
 public static int [] convertToArray(TreeSet<Integer>set) *// set i diziye çeviren metod* {int [] yeniDizi= new int[set.size()]; ***set i array e dönüştürme 1. Yol***   
  
 int i=0; iteratör de kullanılabilirmiş  
 for (Integer sayi: set)  set in elemanlarında get olmadığı için fori yerine foreach kullanıldı  
 {  
 yeniDizi[i++]=sayi; *// önce işlemi yap sonra sayacı arttır. i artmalı ki her seferinde yeni bir alana sayının element leri atansın  
   
// yenidizi[i]=sayi  
// i ++ şeklinde de yapılabilirdi.* } return yeniDizi;

// set ler normal olarak so denilip string şeklinde yazdırılabiliyor. bu yüzden **Set i Array e dönüştürme işlem sırası**

set diye isimlendirilen Treeset i yazdırmak için set i sayı değişkenine atayıp aşağıda so sayı diye yazdırarak set in çıktısını integer şeklinde alabiliriz.

Ancak burada bir set i Array a dönüştürmemiz gerektiği için yukarıda açıklaması yapılan işlemlerden sonra ekstra

1 Array bir dizi tanımlandı. uzunluğu set size kadar tutuldu

2 Integer değişkenin elemanlarını Array dizinin içine doldurduk

3 doldururken de her bir element i yeni bir yere yazmak için yeri belirtirken ++ kullandık.

mainde parametre olduğu için parametreli işlem yapmak istediğimiz için parantez içine parametre tanıtıldı  
 public static void addElements(TreeSet<Integer> set, int...values) *// kaç tane parametre geleceğini bilmediğimiz için vararrgs* {  
for (int i = 0; i < values.length ; i++) { *values in lenght i kadar set.e eleman ekle kaç tane parametre değeri geleceğini bilmediğimiz için* set.add(values[i]); i ye eleman ekle  
  
 }  
 *// burada gerçekleşen işlemleri mainde metod etiketi ile çağırınca etiketin parametre kısmına hangi değişkeni yazarsan  
 // o değişkene buradaki işlemleri uygulayacak addElements(rakamlar)* }

}**ÖZELLİKLE DİKKAT EDİP YÖNTEMİNİ ÖĞRENMEN GEREKEN YER.**  
 **public static HashSet<Integer> farkiBul (HashSet<Integer>a, HashSet<Integer>b)  
  
 { HashSet<Integer> fark= new HashSet<>(a); *// yeni tanımlanan HashSet i a ya eşitledik. yeni HashSet ten b de kileri çıkaracağız* fark.removeAll(b);  
 return fark; *anın içinden b de olan elemanları çıkarır a yı verir.*  
 }  
 *// return işlemin sonucunu veriyormuş işlemin kendisini vermiyormuş o yüzden yukarıda başka bir değişken tanımladık  
 // işlemi yukarıda yaptık sonucu return ettik***

*birleştir*(setA,setB);

HashSet<Integer>birlesik =new HashSet<>();  
birlesik.addAll(a);

*farkiBul*(setA,setB);

HashSet<Integer> fark= new HashSet<>(a); *// yeni tanımlanan HashSet i a ya eşitledik. yeni HashSet ten b de kileri çıkaracağız*fark.removeAll(b);

YALIN A YI BULACAĞIZ

*kesisimBul*(setA,setB);

HashSet<Integer> kesisim =new HashSet<>(a);*// aya eşit oluyor.*kesisim.retainAll(b);

public static int [] convertToArray(TreeSet<Integer>set) *// set i diziye çeviren metod* {int [] yeniDizi= new int[set.size()]; ***set i array e dönüştürme 1. Yol***   
 set to array  
 int i=0; iteratör de kullanılabilirmiş  
 for (Integer sayi: set) set in elemanlarında get olmadığı için fori yerine foreach kullanıldı  
 { set convert to array  
 yeniDizi[i++]=sayi;

#### Gün 28 RECAP : Anlatılmadı HashSet, LinkedHashSet, TreeSet | Part 1 - 07/01/2020

***dizilerde sabit olduğu için lenght.******büyüyüp küçülebilen değişebilen yerlerde ise size().*** *size yi metod şeklinde yapmışlar çünkü sabit değil büyüyüp kısalabiliyor. O Anda metod gidip işleme girip sayıp geliyor. her zaman değişebildiği için.  
  
// setlerde elemana direk ulaşılmıyor elemanı direkt getir yok . bu işlem için set in bir üstünü kullanacağız Map*

Set s=new HashSet(); *// şeklin de de set tanımlanabilir. türünü belirtmezsen integer double diye içine herşeyi atarsın"*

*\*\*\* Tekrarlı Kayıtlara izin vermez \*\*\**

*add (Object o) : Set içerisine bir eleman ekler.*

*clear: Set içerisindeki bütün elemanları siler.*

*contains (Object o): Bir eleman dizi içinde mi kontrol eder.*

*remove (Object o): Bir elemanı siler.*

*size: O ana kadar kaç eleman eklendiğini döndürür*

*Kaç tane elemanın ve hangilerinin eklenmediğini bulabiliriz.*

İf( ! variableName.add(dizi[i]){

So(duplicate Detected)

}

*// dikkat edilmesi gereken yerlerden bir tanesi ıteratör de elemanlara ulaşılarak silme yapıldığı gibi clear komutu ile tek başına döngü olmadan da silme işlemi yapılabiliyor.  
  
 // list ler ile setler arasındaki soru çözüm yöntemlerine karşılaştır . for döngüsü içinde yapılıyordu.  
Set s=new HashSet(); // şeklin de de set tanımlanabilir. türünü belirtmezsen integer double diye içine her şeyi atarsın"*

HashSet<Integer> hs1=new HashSet<Integer>(); *// sadece ınteger alabilen bir set .*

Set<Integer>hs2= new HashSet<>(); *ÖNERİLEN sadece ınteger alabilen bir set . en pratik tanımla ma şekli budur setler hashSet linkedSet ve TreeSet in atası olduğu için başta set diye tanımlayıp eşitlikten sonra hangi set türü olacaksa tanımlanabilir* Set hsObject=new HashSet(); *// Bu tanımlamada her türlü değişken atanabilir. Bütün tipler  
 bütün tipleri tanımlayan tipin adı : OBJECT-> NESNE -> HERŞEYİN ATASI  
 //bu tanımlama diğerlerine göre daha yavaş çalışır ve içine başka şeylerde atıldığından güvenli değildir. veriler tutarsız ve kontrol dışı olabilir. ÇÜNKÜ*

hsObject.add("12");  
 hsObject.add(12); *// kullanıcı string de Integerde girebilir.*

Set<Integer>hs1= new HashSet<>(Arrays.*asList*(1,2,3,4,5)); *böylede eleman atanabiliyor. hs1Add((1,2,3) ve hsAdd(4,5); ile aynı int [] dizi= {1,2,3}; gibi başlangıçta değer atadığımız tarzda değer atadık*

#### Mentoring HashSet, LinkedHashSet, TreeSet lerin metodlarda kullanımı | 03/07/2020 zeynep

**evDeneme {**  
 public static void main(String[] args) {  
 */\* 1  
 main method altinda bir double hashSet olusturunuz.  
  
 ve bu seti, adi setOlustur ve return tipi hashSet double olan  
  
 ayri bir metod altın da 3.23 , 3.10 , 5.12 , 10.12 , 23.12 değerlerini kullanarak doldurun.  
 2  
 adini toplaminiAl koyacagimiz ve parametre olarak bir Double hashSet kabul edecek ayrı bir method oluşturarak  
  
 hashSetin değerlerinin toplamını alacaksınız.  
 \*/*

1. ***Yöntem ile tanıtılan 1. Soru ve çıktı alma şekli***Set<Double> hs = new HashSet<>(); // set tanımlanırken parametre kısmına aşağıdaki örnekteki gibi metod un ismi

Yazılmadığın dan yani metod set e asign edilmediğin den  **sadece** set in elemanı alma işlemi yapılırken (başka işlem de değil) çıktı alırken hem metod ismi hem de set ismi gerekiyor. Aşağıdakinden farklı olarak

*setOlustur*(hs);  
 System.*out*.println("hs = " +*setOlustur*(hs)); *// so=hs = [5.12, 3.23, 3.1, 23.12, 10.12]* ***1. Yöntem ile tanıtılan 2. Soru ve çıktı alma şekli*** *toplamınıAll*(hs);  
 System.*out*.println("toplamınıAll(hs) = " + *toplamınıAll*(hs));*// so=toplamınıAll(hs) = 44.69*

1. ***Yöntem ile tanıtılan 1 soru ve çıktı alma şekli***

Set<Double> parameyresizDEneme=*parametresizMetod*();*//* // set tanımlanırken parametre kısmına metod un ismi yazıldığından yani metod set e asign edildiğinden çıktı alırken sadece set in ismini yazmak yetiyor yukarıdakinden farklı olarak.System.*out*.println("parameyresizDEneme = " + parameyresizDEneme); *//so =parameyresizDEneme = [2.5]*

*2.* ***Yöntem ile tanıtılan 2 soru ve çıktı alma şekli***

*toplamınıAll*(parameyresizDEneme); burada sadece set in ismini yazarak çıktı almak yetmedi çünkü set in elemanlarını almak istemiyoruz işlemi istiyoruz bu yüzden set ve hangi işlem yapılacaksa o metod un ismini almak gerekiyor.

System.*out*.println("toplamınıAll(hs) (parameyresizDEneme) = " + *toplamınıAll*(parameyresizDEneme));*//toplamınıAll(hs) (parameyresizDEneme) = 8.2*

*.* }  
 public static Set<Double> setOlustur(Set<Double>mHs) //parametreyi burada yazdığımız için içeride yazmaya gerek kalmadı  
 {  
 //Set<Double> mHs=new HashSet<>();  
 mHs.add(3.23);  
 mHs.add(3.10);  
 mHs.add(5.12);  
 mHs.add(10.12);  
 mHs.add(23.12);  
 return mHs;  
 }  
 public static Set<Double> parametresizMetod()   
 {  
 Set<Double> c=new HashSet<>();  
 c.add(2.5);  
 c.add(5.7);  
 return c;  
 } *// metodlar reuseAbility tekar kullanılabilecek imkanı sunar.* public static Double toplamınıAll(Set<Double>hs)*//toplamını alabileceğim bir şey olabilmesi için parametre kısmına bir şeyler yazılmalı* { double sum=0; *//Double yi büyük yazdığımda 0.0 a eşitliyordu so for a eşitlemiyordu*  *// wrapper miydi box in un boxing miydi bak* for(Double str: hs)  
 sum+=str;  
 return sum;  
 }

**evDeneme2 {**

*adi elementEkle , parametresi* ***Integer set*** *ve* ***integer array*** *olan bir method yaziniz (return tipi olmayacak)*

public static void main(String[] args) {

HashSet<Integer> set =new HashSet<>();

*elementEkle*(set, new int[]{10,27,3,40,57,6,7,8,9,10});

System.*out*.println("set 1 = " + set); set 1 = [3, 6, 7, 40, 8, 57, 9, 10, 27]

}

public static void elementEkle**(HashSet <Integer> set , int [] elements)** *// multiple parameters* { for (int ekelenecekler: elements) *// set in içerisine int Arraylar eklendi*  
 {  
 set.add(ekelenecekler);  
 } System.*out*.println("set 2= " + set); set 2 = [3, 6, 7, 40, 8, 57, 9, 10, 27]

#### Gün 29 | HashMap, LinkedHashMap, TreeMap | Part 1 - 07/02/20

**Maplerin farklarına ekle**

HashMap<String,String> map=new HashMap<>();map.put("Amazon", "296 Euro");

*// kopya oluşturma*HashMap<String ,String> map2= new HashMap<>();  
  
*// 1. yol*map2=new HashMap<>(map); *// kavramlara ekle ve map kullanım farklarına  
  
//2.yol*map2.putAll(map);

*entry set key ve value yi kısadan ikisini bir arada veya ayrı ayrı yazdırabilmede kolaylık sağlar foreach ın aksine*

foreach ta bazen key i görmek istediğimizde sadece value yi yazdırdığımız değişkeni ona göre ayarladığımız için key i alırken value yi göremeyebiliyoruz

for (Map.Entry<String,String> entryYazdırma : map.entrySet()) *// entry komutu ile yazdırılıyor alt alta* {  
 String value= entryYazdırma.getValue();  
 System.*out*.println( value);

String key= entryYazdırma.getKey(); bu şekilde de yazılabilir  
 System.*out*.println( key);  
}*1. for u bitirir çıkar 2. for a girer onu bitirir çıkar*

for (Map.Entry<String,String> entryYazdırma : map.entrySet())  
 {  
 String key= entryYazdırma.getKey();  
 System.*out*.println( key);  
 }

*// yukarıdakinden farklı komut ile yazdırma*

for (String key : map.keySet())  
 {  
 System.*out*.print( key);  
 }

System.*out*.println();  
  
 for (String value : map.values())  
 {  
 System.*out*.println( value);  
 }

System.*out*.println("Değer value = " + map.get(key));*// so= 296 Euro*

*valueleri key e yüklediğimiz için Bir value ye ulaşmak istediğimizde onu key ile aratmalıyız*

Check etmek istediğimiz şeyi seçerken contains ile yaparız true false döner.

map.containsValue("296 Euro")); true

*keyleride value leri de toplamını alabiliriz.*

Gün 31

Kavramlara eklenecekler

// class oluşturma

// classların komutları attribute fields

// classların yazdırılması.

Arabaa araba1= new Arabaa(); *// araba sınıfından türetildi artık nesne oldu*

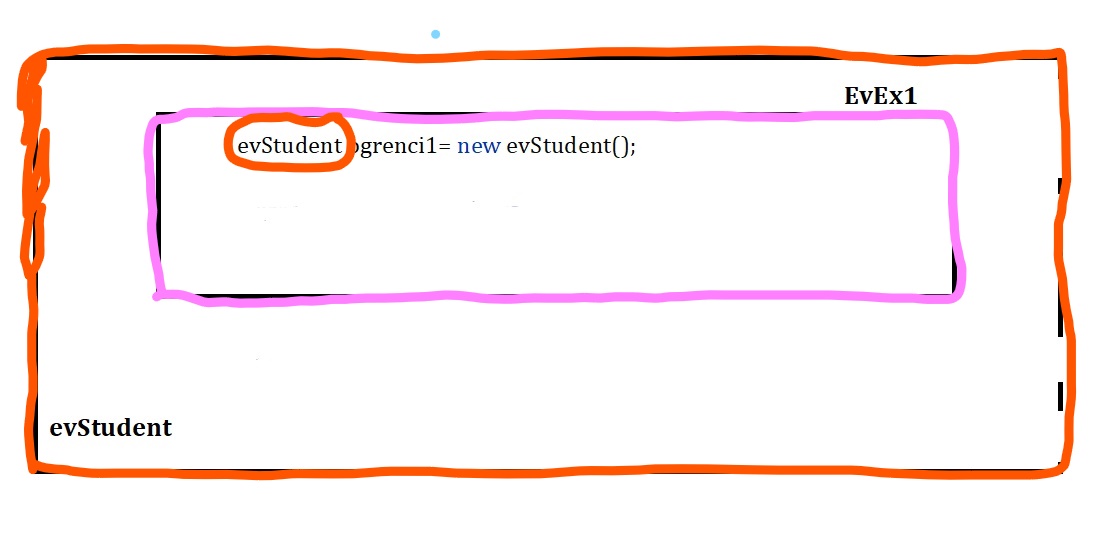
*1. Yöntem aynı dosyada var olan en üstteki class ın altında main ve main altında class açılıp new denilip içine attribute fields eklenip mainden call ettik*

*2. yöntem de en üstte var olan class ı yani daha dosyayı ilk açarken oluşturmuş olduğumuz class ın içine fields attribute oluşturup mainden call ettik ve kullandık**3. Yöntemde 3. Class ı 2 . class a tanıttık ve ardın dan 2. Class taki ismi ile 3. Class tada erişime açtık başka dosya tanıtılmış class ı call ettik*

*// aynı dosyanın içinde 1 tane public olabilir*

evStudent ogrenci1= new evStudent(); *1 öğrenci başlatıldı class artık object*  
 *başka bir dosya içinde oluşturulan fields attribute ler uygulanabilmek için ana klasördeki mainden çağırıldı.* ogrenci1.name ="Ayşe";  
 ogrenci1.grade=3;  
  *student dosyasında class ında oluşturulan değişkenler. Yukarı da oluşturmuş olduğumuz öğrencimiz ile işlendi*

*//1. yöntem* evSchool okulu =new evSchool(); *// school oluşturulup dolduruldu*  *burdan yazdırılabilir.* okulu.adress="göztepe";  
 okulu.name= "Marmara Üniversitesi";  
 okulu.principalName="Rektör";  
 okulu.tuitionFees=0;



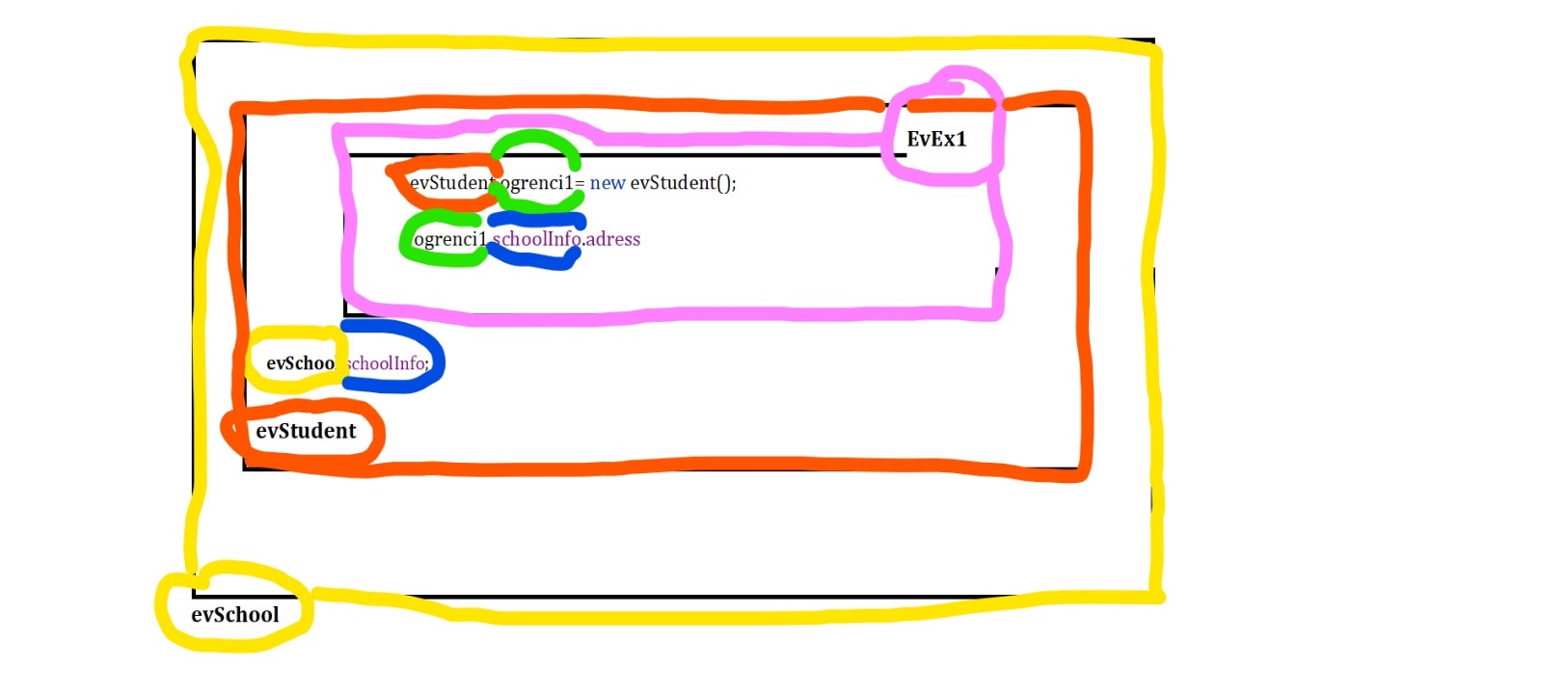
ogrenci1.schoolInfo=okulu; bu ne ne gerek vardı analayamadım *// oluşturulup dolduruldu öğrenciye atandı  
 kendi dosyasında olan school . önce öğrenciye eklendi gerektiğinde kullanılsın diye ancak tasarım aşamasındaydı kullanılmıyordu gerektiğinde kullanılabilsin diye eklendi. Mainde new deyip doğum gerçekleşince kullanıma başlayabilir.*

*//2. yöntem  
 variable name ve class* ogrenci1.schoolInfo= new evSchool(); *// öğrencinin school infosu yeni oluşturulup direkt atama işlemi yapıldı* ogrenci1.schoolInfo.adress="göztepe";  
 ogrenci1.schoolInfo.name="Marmara Üniversitesi";  
 ogrenci1.schoolInfo.principalName="Rektör";  
 ogrenci1.schoolInfo.tuitionFees=0;

*// bu yazdırma işlemini bir class aluşturcaz yazdırcaz hepsini burada yazcak* System.*out*.println("ogrenci1.name = " + ogrenci1.name); *//Ayşe* System.*out*.println("ogrenci1.grade = " + ogrenci1.grade); *//3* System.*out*.println("ogrenci1 = " + ogrenci1.schoolInfo.name);

student sınıfının içinde öğrenci 2 olmaz birisi nesne birisi class .. class kendi başına tanımlanmalı öğrenci 2 mainde tanıtılmalı

ogrenci1.schoolInfo= new evSchool(); 1. Ve 2. Class ların tanımlanmış isimleri ile birlikte ç new . Class yazmazsan yukarıda erişime açılsa bile burada kullanamazsın çünkü tek tek doğumlara ilaveten özeliklerinin kisinin birlikte de doğumuna ihtiyaç var.

**Return yapmayınca main de so deyip yazdıramazsın

*Gün 32*

***Main {***

***class larda yazdırma 4 şekildedir***

***/1. yöntem String şeklinde her bir değişkeni ve attributess i tek tek uzun uzun yazma***

System.*out*.println("kisi2 = " + kisi2.name); her farklı kişi için bunları yapmak çok uğraştırıcı  
 System.*out*.println("kisi2 = " + kisi2.surname);  
 System.*out*.println("kisi2 = " + kisi2.age);

*//2.Yöntemin kullanılması : class ın dışında bir metod oluşturup yazdırma, her nesne için böyle yazarsak oldukça karışır  
 BilgiYazdır*(kisi1);  
 *BilgiYazdır*(kisi2); *//bu yöntemde parametre göndermek zorunda kalıyorum, uğraştırıcı  
  
 //3.yöntem* kendi class ı içinde bir metod oluşturarak yazdırma en kolayı kisi2.BilgiYazdır(); *// hem davranışları kendi içinde hem metodu kendi içinde* kisi1.BilgiYazdır(); *// içine parametre yazmaya gerek yok. Tanımla mış çıkıyor*

*olduğumuz değişkeni( kisi1, kişi2) yazınca hangi işlemi yapmak istiyorsam yanında*

*}*

public static void BilgiYazdır(Personn kisi) kendi başına metod  
 {  
 System.*out*.println("kisi metod = " + kisi.name); içine parametre aldığı için main de de parametre isteyecek  
 System.*out*.println("kisi metod= " + kisi.surname);  
 System.*out*.println("kisi metod = " + kisi.age);  
 }  
}   
class Personn {

void BilgiYazdır() *class içindeyken tipini verip yazdırmak yeterli bir şey döndürürse return yazıcaz aynı mantık*{  
 System.*out*.println("kisi class = " + name);  
 System.*out*.println("kisi class = " + surname);  *class ın kendi değerleri gelecek zaten class ın içindeki kendi metod u* System.*out*.println("kisi class = " + age);  
}

}

String DenemBilgiYazdır(){  
 System.*out*.println("kisi class = " + surname);System.*out*.println("kisi class = " + age);

return name + surname;

return a so yapmadan **sadece** name + surname yazarak ta mainden ulaşılıp yazdırılabilir istenilen işlem yapılabilir  
so kısmında name yazmamasına rağmen . mainde kişi.DenemeBilgiYazdır yapınca return edilenler geliyor.  
}

**Gun32.evTask2;{**  
 *// 1.Adım :ayrı dosyada olmak üzere Customer isimli (fields: name) isimli bir class tanımlayınız.  
 //2.Adım :ayrı bir dosyada ElectricityAccount isimli (fields : totalKw, rate(double), bill)  
 //3.Adım :Customer a bir ElectricityAccount ekleyiniz.  
 //4.Adım :1 tane aboneye oluşturunuz, Account u dahil.* public static void main(String[] args) {  
 *// 4. adımda hem müşteriyi hem de elektrik aboneliğini oluşturduk* Customerr abone1=new Customerr();   
  
 abone1.name="Kaya";  
 abone1.ElectricAbonesi.rate  
 yukarıda variable name . fonksiyonlar çıkabilir ama yapmadan çalışmıyor. Çünkü olanlar henüz doğmadı. Tasarım aşamasında

abone1.ElectricAbonesi= new ElectrictyAccounttttt(); bunu yapmadan çalışmıyor anlamaya çalış

*- 3. Class da olan* ElectrictyAccounttttt *2. Klass a sadece tanıtılıyor. İnstanceler ekleniyor tasarıma*

*Gerekirse 2. Class a ilave olarak new yazdıktan sonra kullanılabilsin diye*

*Gerekmezse kullanılmayacağı için class ın özellikleri bir kenarda bekliyor*

*Gerektiği esnada 2.class ı oluşturmuş olduğumuz variable name + 2. Class ta çağırmış olduğumuz ve oluşturmuş olduğumuz class ismine ilave ten mainde new denilerek kullanılacak*

*New deyince doğum burada gerçekleşiyor. Customer class ının altına yeni bir class ekleniyor ancak hala tasarım aşamasındalar*

*2. Yöntem ide öğren  
 // new demeden bir class ı kullanamaz sın ancak new olayını class ın içinde de yapabilirmişiz.*

System.*out*.println(" denemeee = " +abone1.ElectricAbonesi.rate);

bir class birden fazla class tan çağırılabiliyor. Main class tan önce bir class ta tanıtılsa bile daha sonra tekrar main class tan çağırılabiliyor. Ve başka bir isimle tanıtılabiliyor.

*yukarıya bir class ı çağırıp variable name ini yazıp new class yazdıktan sonra . Oluştur muş olduğun diğer class lardaki metod ve fonksiyonlarında çıkmasını istiyorsan main den erişime açılan class ın içine gidip . başka dosyalardaki class larıda yukarıya tanıtmış olduğumuz classs ın içinden çağırıp daha sonra o class ın ismi ve 3 class ın ismini main e yazııp erişime açmalıyız noktaya basınca başka class larda yaptığın işlemlerin çıkmasını istiyorsan*

}  
}

1 public class Customerr {  
 String name;  
  
3 ElectrictyAccounttttt ElectricAbonesi; *// oluşturulan electrricity class customer klasına tanıtıldı ilişkilendirildi.*

*doğum mainde olacak*}

2 public class ElectrictyAccounttttt {  
 int totalKw=0;  
 double rate=0.7;  
 double bill;

}

classlarda metodlar ın yazılışı

void + metodname()

variabletyp +metodname()

4 tür yazdırma şekli vardır.

1- String olarak uzun uzun

2- Ayrı bir metod içinde (Foreach, For) main i karışık gösterir ve her değişkeni metod un parametre kısmına yazacağın için aklında tutman gerekir

3- kendi class ı içinde bir metod oluşturarak yazdırma en kolayı

4-Arraysto string metod

*sınıf class ının içine bu metod u açtığımızda sınıfı çağırdığımız her yerden sınıf class ının içindeki oluşturulmuş olan metodlar ve sınıf class ının dışında oluşturulmuş olan ancak sınıf class ı içerisinden çağırılmış olan tüm class ların özelliklerine ulaşılabilecek*

*sadece* System.*out*.println("sınıff = " + sınıff);

*dediğimizde bütün classlar ve metodlar ve instanceler objectler fieldsler yazdırılacak*

*kullanılabilecek mainde yapmadığımız için main daha kullanışlı ve temiz görünecek* @Override  
 public String toString() { sağ click+ generate+ to string yapınca açılıyor String toString(); yazılışı gerekli  
 return "Dersin Konusu :" + konu +  
 "\n ogrenci ismi : " +ogrencii.isim +  
 "\n ogrenci bolumu :" +ogrencii.bolum +  
 "\n ogrenci yas: " + ogrencii.yas +  
 "\n oğrenci birinciSınav: " +ogrencii.birinciSınav +  
 "\n ogrenci ikinci Sınav : "+ ogrencii.ikinciSınav +  
 "\n ogrenci ikinci sınav: "+ ogrencii.ucuncuSınav +  
 "\n ogrenci notOrt: " +ogrencii.notOrtt() + her bir dosyada generate string clik ederek bir yerde toplayabilirsin  
 "\n ogretmen isim :" + ogretmenn.isim +  
 "\n ogretmen deneyim " +ogretmenn.deneyim +  
 "\n ogretmenn.ogretmenimNasil : " + ogretmenn.ogretmenimNasil();*// bu niye olmadı bak* }  
}

*// parametrenin ismini ve for yazıp enter e basınca çıkıyor direkt.*

*//komutlara çalış ne işlem yapıyor .Türkçesine bak ing söylemeye çalış*

Gün 33 kavramlar

**Metodların basında static varsa main metod da (ve belki heryerde emin ol) direkt kullanılabiliyor yoksa bulunduğu class main metod da new denilip oluşturulup öyle kullanılabiliyor**

package Gun33.Evgun33.evExAmple;  
  
public class evJavaStaticAndNonStaticMethods {

*// bütün programlarımda kullancağım tip dönüşümlerim için metodlara ihtiyacım var mesela int to string string to int vs. vs.*

*bunları yapan bir sınıf yazınız  
bunları mainde kullanarak gösteriniz*  
 *// 2 public metod a typcasting gösterilecek bunu metod un tasarım halinden çıkarıp new diyerek call edip doğumun gerçekleşmesi ile yapıyoruz  
// 3 public staic metod da typ casting yapıldı mainde new demeden kullanabildik . direkt olarak görebildik çünkü*

public static void main(String[] args) {

*// 2. yöntem type casting için kendi sınıfımı yazdım onu kullancağım* evUtility util =new evUtility();  
 static metod olmadığı için mainden class new denildikten sonra kullanılabiliyor  
 strRakam=util.*getString*(sayı);intRakam=util.*getınt*(strRakam);  *//3. Yöntem yeniden oluşturmadığımız için sınıf ı direkt adı ile kullanıyoruz*

strRakam= evUtility.*getString*(sayı); static olduğu için mainde new demeye gerek yok

intRakam = evUtility.*getınt*(strRakam);

*burda bir sorun çıkarsa static metod dan komudu değiştirip bu komutun kullanıldığı her yerden bunu düzeltmiş olacağız örenek trim*

*// aynı Math sınıfı gibi* Math.*random*();  
 }  
  
}

package Gun33.Evgun33.evExAmple; main dosyada gerçekleşecek işlemler in yazılacağı dosya  
  
public class evUtility {

// 2. yöntem için kullanıldı bu yöntemi bilmeye gerek yok farkı görmek için anlatıldı  
static olmadığı için kullanılmaı gereken pakette veya class ta buradaki tasatım halinden çıkarıp  
 new deyip doğum haline getirmeliyiz doğmalı

public String getString(int valueDeğer) static metod olmadığı için mainden class new denildikten sonra kullanılabiliyor  
 {  
  
 return String.*valueOf*(valueDeğer); *// kendine gelen int değerleri hep stringe çevirip geriye vercek* }  
   
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------  
 *// static olacağı için istediğimiz class da new demeden kullanabiliriz.* public static String getString(int valueDeğer) static olduğu için mainde new demeye gerek yok  
 {  
  
 return String.*valueOf*(valueDeğer).trim(); *// kendine gelen int değerleri hep stringe çevirip geriye vercek* }

---------------------------------------------**2. örnek** ------------------------------------------------------------------------------------------------------------------

package Gun33.Evgun33.**evTask;**  
*// Math sınıfndaki metodlarıdan 5 tanesini kendi isimlendirmenizle ayrı bir sınıfta yazarak,  
// (kendi metodunuzun içinde Math.fonksiyonlarını kullanabilirsiniz)  
// mainden nesne oluşturmadan direk kullanılmasını sağlatacak örnek kullanımlarını yazınız.*public class evTask {  
 public static void main(String[] args) {  
  
 System.*out*.println("evMyMath.getMin(3,45) = " + evMyMath.*getMin*(3,45));  
 System.*out*.printf(" yüzde işareti 5 hane olsun noktadan sonra 4 hane olsun f ve virgül = %5.4f ", evMyMath.*getKarekok*(7));  
 *//formülde f den sonrasını yazmaya başlıyor. string notlarını ona göre al* }  
}

package Gun33.Evgun33.evTask;  
  
public class ev**MyMath {**  
  
 public static int getMin(int value1, int value2 ) static oldukları için mainde new denmeden kullanılabildiler  
 {  
 return Math.*min*(value1,value2);  
 }  
  
 public static double getKarekok(int value1) *// return double döndürürken parametre int. alabilir.*  
 {  
 return Math.*sqrt*(value1);  
 }  
}

---------------------------------------------örnek3-----------------------------------------------------------------------------------------------------------

package Gun33.Evgun33.evExAmple.evTask2;  
*// Bir Üniversitede öğrencilere ders kaydı yapılacaktır.  
// 1- Ders sınınıfı ayrı dosyada : adı: Lesson , fields : name, credit (1-3 arasında değer alıyor)*

*// 2- Öğrenci Sınıfı ayrı dosyada: adı: Student, fields : name, maxCredit, ilaveten birde dersleri listesini tutacak bir liste ekleyiniz   
// 3- 3 adet ders oluşturunuz. Önemli lesson class ının derslerini tutacağı için listenin türü lesson olmalı  
// 4- 1 adet öğrenci tanımlayınız alabileceği maxCredi si 10 olsun.  
// 5- Bu öğrencinin ders listesine açılmış dersleri ekleyiniz.   
// ders eklerken max alabileceği Crediyi geçmemeli, geçerse uyarı versin, alabileceğiniz max credi doldu şeklinde. max crediyi hesaplayacak bir metod yazılmalı*

*// Ödev  
// 6- Aşağıdaki ders oluşturma kısmı için ilgili sınıfta lessonCreate adında bir metod  
// oluşturunuz  
// 7- Öğrenciye ders ekleme bölümünü yine ilgili sınıfta bir metod olarak yazınız.  
  
// fields attributies properties instance*import java.util.ArrayList;  
  
public class **evTask2 {**  
 public static void main(String[] args) {  
sr 3 evLesson mat101= new evLesson(); *// lesson class ı doğdu ve içindeki instanceleri attribute ve fieldsleri görebilceiğiz* mat101.name="Matematik"; class name + varianle nname+ =new class name;  
 mat101.credit=3; variable name + class fields= deişken değer.

sr 3 evLesson java105= new evLesson();  
 java105.name="Java Programing"; *// her eklediğimiz lesson da lessson class ının altındaki instanceler çıkacak* java105.credit=6;  
  
  
 sr 3 evLesson fizik109=new evLesson(); *// her eklediğimiz lesson yeni bir object yeni bir doğum* fizik109.name="Fiziik dersi";  
 fizik109.credit=5;  
  
SR4 evStudent ogr1=new evStudent();  
   
 ogr1.name="Alpay";  
 ogr1.maxCredit=10;

sr 5 ogr1.dersListesi= new ArrayList<>(); *// doğum gerçekleşti lesson class ında da sadece liste oluşturulmuştu . liste ismi ile türü   
 içine lesson verileri geleceği için lessondu*

*// buraya kontrol koyulacak toplam aldığı krediye bakılarak eklenecek ders ile öğrencinin o ana kadar ki kredi toplamı  
 // öğrencinin alabileceği max krediyi aşmıyorsa topla* if (ogr1.totalcredit()+ mat101.credit< ogr1.maxCredit) { // sr5 student class metod kısmına bak  
sr5 ogr1.dersListesi.add(mat101);  
 }  
 else  
 {  
 System.*out*.println(" mat101 alabileceği kredi miktarı doldu");  
 }  
  
 if(ogr1.totalcredit()+ fizik109.credit< ogr1.maxCredit){  
 ogr1.dersListesi.add(java105);  
 }  
  
 else  
 {  
 System.*out*.println("java105 alabileceği kredi miktarı doldu");  
 }  
  
 if(ogr1.totalcredit()+ java105.credit< ogr1.maxCredit) {  
 ogr1.dersListesi.add(fizik109);  
 }  
 else  
 {  
 System.*out*.println("fizik109 alabileceği kredi miktarı doldu");  
 }  
 }  
}

package Gun33.Evgun33.evExAmple.evTask2;  
  
public **class evLesson {**  
 String name; *// dersler bur da oluşturulmaz burası class dersler object olacak yani doğum* int credit;

burası ders te çözülmedi  
*/\*  
 // türü evlesson  
 public static evLesson creatLesson(String name, int credit) // dersi ve kredisini hesaplayacak bir metod  
 {  
 evLesson ders=new evLesson(); // tasarım halindeki class bject oluyor doğdu içine atttribute fields instanceleri alabilmemiz için  
 ders.name=name;  
 ders.credit=credit;  
  
 return ders;  
 }  
  
 // metod u neden yazdık ? class ı direkt new class diyerek ana klasorde mainde çağırabilirdik  
 // amaç burada static ve non static metod u öğrenmek o yüzden metod oluşturduk ve bu metod u new demeden mainde kullanabilmek için  
  
 // evlesson class ının içindeki instanceleri alabilelim diye metoddan class ı call ettik class ı burada  
 // dünyaya getirdik  
  
 // ana klasörde neden yapmadık ? her değişkeninn yanında çıksın diye new demek zorunda kalmayalım diye.  
 // amaç burada static ve non static metod u öğrenmek o yüzden metod oluşturduk ve bu metod u new demeden mainde kullanbilmek için  
  
 \*/*}

package Gun33.Evgun33.evExAmple.evTask2;  
  
import java.util.ArrayList;  
  
public class **evStudent {**  
 String name;  
 int maxCredit;  
  
  *derslerin listesi olacağı için türü evLesson fieldsleri String name; int credit otomatik gelecek* ArrayList<evLesson> dersListesi; *//=new ArrayList<>(); doğum burada olmayacağı için gerek yok*public int totalcredit()  
{  
 int total=0;  
 for (evLesson ders: dersListesi) // sr5 in class kısmı  
 {  
 total+=ders.credit;  
 }  
 return total;  
}  
}