



23 EKİM 2021
DERS 1

Genel Hatırlatmalar
Java Giriş

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Genel Hatırlatmalar



1. Derslere Hazirlanin ve Zamaninda Katilin
2. Dersi Dikkatli Dinleyin
3. Derste Aktif Olun
4. Anlamadiklarinizi Sorun
5. Ödevlerinizi Yapin (Kod yazma araba kullanma gibidir)
6. Her Dersten Sonra Tekrar Yapin

Genel Hatırlatmalar

7. Basari = Egitim + Calismak
8. Grup calismalari yapin, En iyi ogrenme yontemi ogretmektir
9. Mentoring toplantilarini kacirmayin
10. Maillerinizi gunluk kontrol edin
11. Yoklama yapiliyor zooma isminizle girin
<https://class.techproed.com/>
12. Teknik destek slack @technical support
13. Ders esnasinda canli destek
Haluk Bilgin, Fatih Kilic, Ebubekir Sahin
14. Customer service +1 917 768 74 66

**"TEACHERS
CAN OPEN
THE DOOR,
BUT YOU
MUST ENTER
IT YOURSELF."**

~ CHINESE PROVERB

Mentoring

Mentoring toplantıları her hafta team tarafından ortak belirlenen gün ve saatte düzenli şekilde yapılmaktadır.

- ✓ Mentoring faaliyetleri STUDENT COACHING (öğrenci danışmanlığı) olarak yapılmaktadır.
- ✓ Mentoring faaliyetlerinde...
 - Haftanın görülen derslerin değerlendirmesi...
 - Derslerle ilgili döküman desteğinin sağlanması....
 - Ödev proje vs çalışmaların takip edilmesi...
 - Team work'lerin takip edilmesi...
 - FlipGrid çalışmalarının takip edilmesi...
 - Java verbal çalışmalarının takip edilmesi...
 - Java coding çalışmalarının takip edilmesi...
 - Interview çalışmalarının takip edilmesi...

DÜZENLİ OLARAK YAPILMAKTADIR....

Gorulecek Dersler

Automation Engineer:

Java	Selenium Grid
Selenium	Git, GitHub
SDLC	HTML & CSS
API	Bootstrap
SQL	Java Script
Jenkins	Lambda
JDBC	Project

Java Developer

Core Java	UML Diagram
Advance Java	Multi Thread
Oracle SQL	Hibernate
JDBC	MongoDB
HTML5 & CSS	SpringMVC
Bootstrap	Restful API
JavaScript	Micro Services with Spring Boot
React.js	Git-GitHub
SDLC	
Market Session	

Mobile Developer

Core Java	Git-GitHub
Oracle SQL	Bootstrap
SDLC	React.js
HTML5 & CSS	JavaScript
Advance Java	React Native
	Project

Ders Isleyisi Bilmeniz Gerekenler

1. Maillerinizi gunluk kontrol edin
2. Dersleri zoom'dan izliyoruz ama mesajlasma icin slack kullaniyoruz



- Iki slack kanalimiz var
- Direk mesaj
- Kod paylasma (**snippet**)
- Mesaj silme ve edit
- Pin yapma



Google Classroom

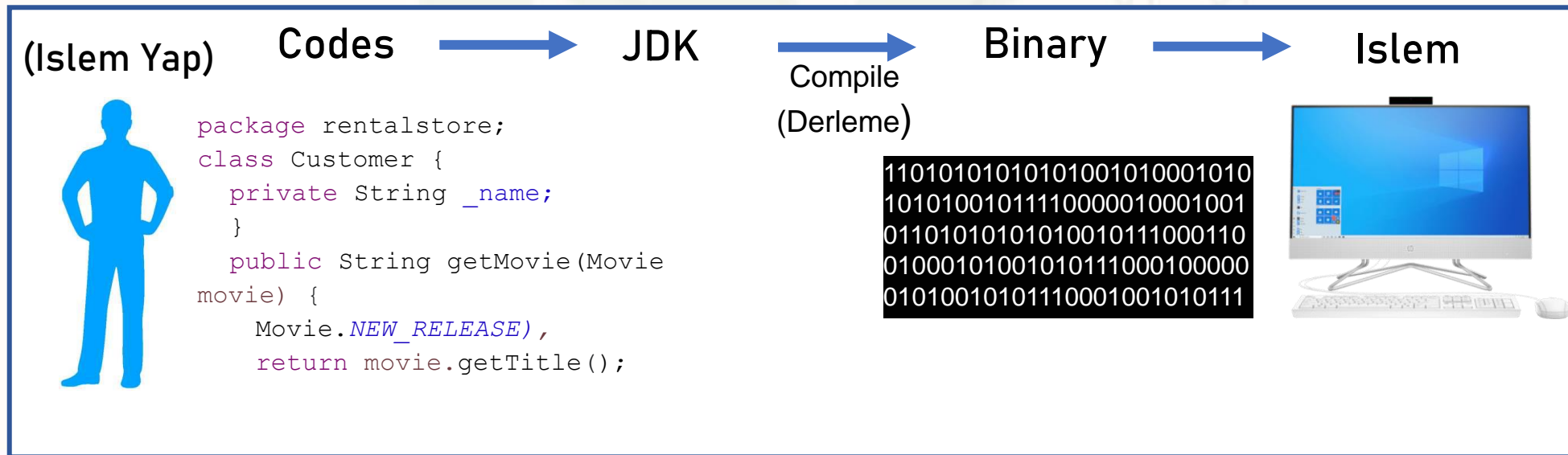
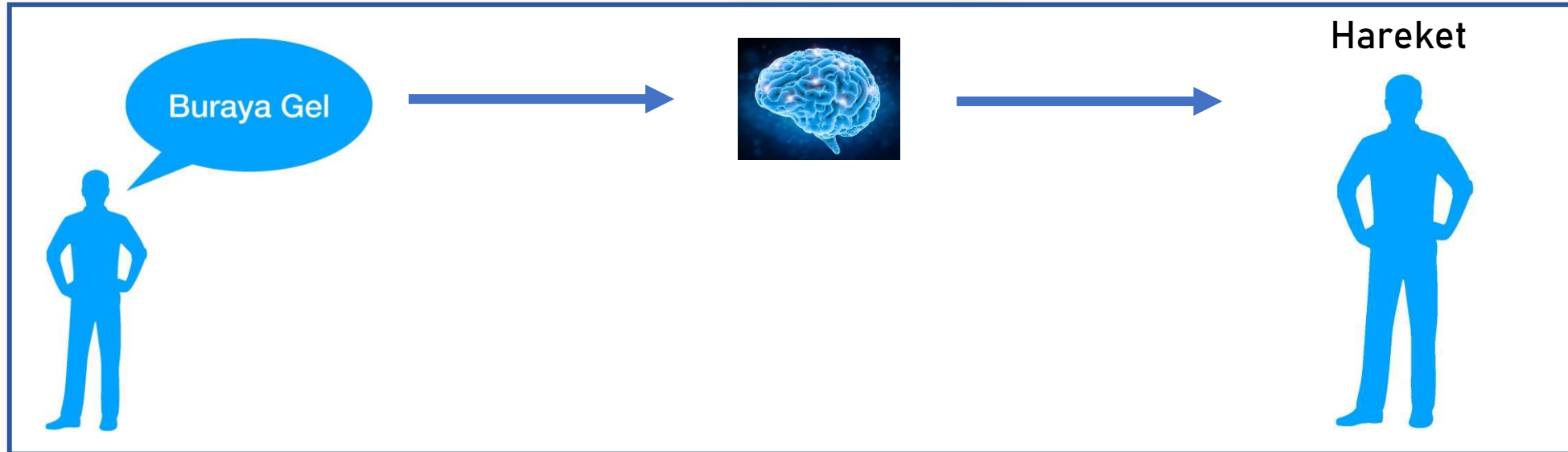
3. Google Classroom
 - Tum ders notlari, zoom linki ve videolar Google Classroom'dan paylasilacak
 - Maillerinize davetiye gonderildi

Ders Isleyisi Bilmeniz Gerekenler

1. Ders tam zamaninda baslar.
2. Dersin basinda 10 dakika bir onceki gunun kisa tekrari yapilir
3. Her konu bittiginde ertesi gun kisa tekrardan sonra Socratic test i yapilir (10 -15 dk) sonra o sorular cozulerek konu tekrari yapilir



Programlama Dili Nedir ?



Nicin Java ?

1- Öğrenmesi kolay

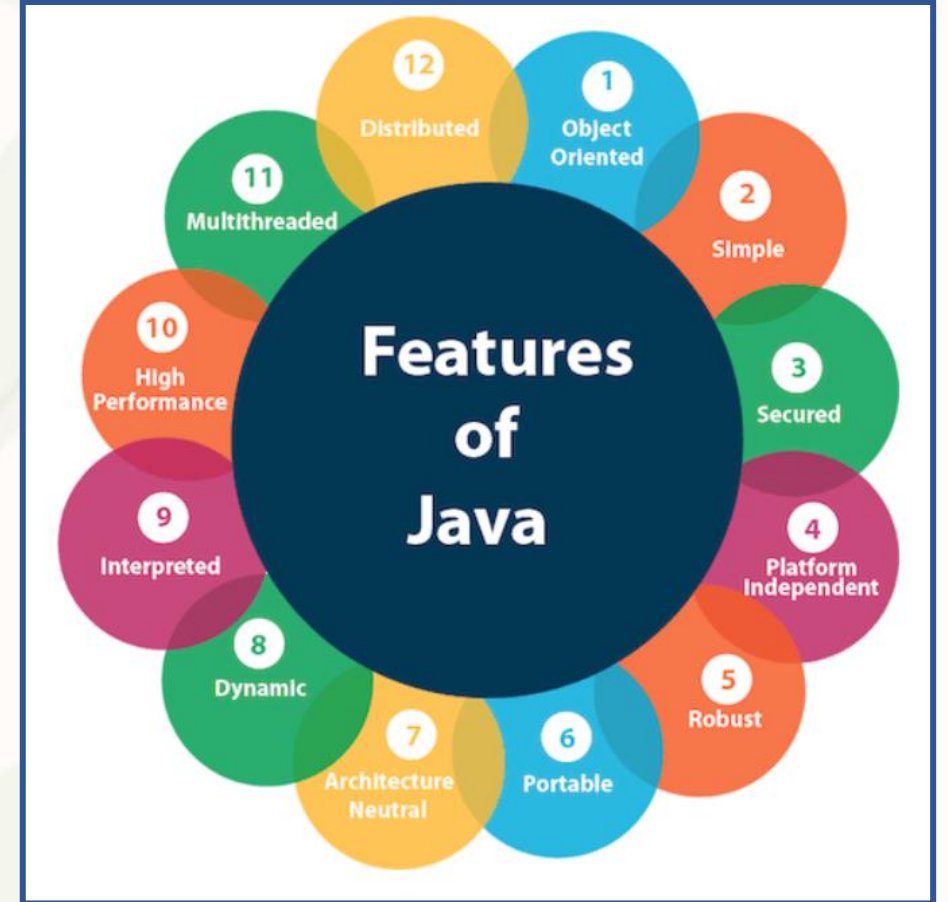
2- Dünyada en çok kullanılan programlama dili

Sun'a göre 3 milyar cihaz Java kullanıyor. Şu anda Java'nın kullanıldığı birçok cihaz var.

Bunlardan bazıları şu şekildedir:

- Acrobat reader, medya oynatıcı, antivirüs vb.
- Masaüstü Uygulamaları
- Bankacılık uygulamaları gibi Kurumsal Uygulamalar
- Cep Telefonu
- Akıllı kart uygulamaları
- Robotik uygulamaları
- Oyunlar

3- Java “**Object Oriented Programming (OOP)**” Language’ dir.

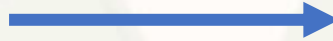




Object Oriented Programming Nedir?



Objects (Nesne)



Application (Urun)

- 1- Feature (Fields veya Variables)
Pasif ozellik (renk,sekil,isim)
- 2- Functionality (Method)
Aktif ozellik (tasima,degistirme)

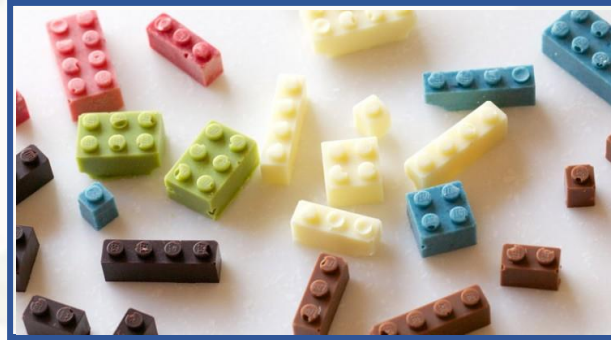


Bir Object Nasıl Olusturulur?



Class(Object Kalibi)

Field
(Variables) Method
(Functions)



Object

Birden fazla Obje birlestirilir



Application



Object Nasıl Kullanılır ?



Ogretmen



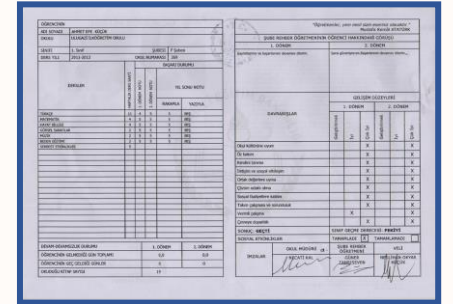
Ogrenci

Dersler

09:00	TÜRKÇE-1
09:30	MATEMATİK-1
10:00	TÜRKÇE-2
10:30	MATEMATİK-2
11:00	TÜRKÇE-3
11:30	MATEMATİK-3
12:00	TÜRKÇE-4
12:30	MATEMATİK-4
13:00	İYEP TÜRKÇE



Personel



Notlar



Bir Class Hangi Bolumlerden Olusur?

```
public class C2_MethodCreation2 {
```

1

2

3

2

```
} // Class sonu
```

- 1 – Class Declaration
- 2 – Curly braces : Suslu parantez
- 3 – Class Body : Suslu parantezler arasında kalan ve kodlarımızı yazdığımız bölüm



Bir Class'ın İçinde Neler Bulunur?

```
public class C2_MethodCreation2 {
```

```
    private double ortalama;  
    public int sonuc;
```

1

1 – Field / Variables

```
    public static void main(String[] args) {  
        ortalama(85.2 ,90.3); // method call  
    }
```

2

2 – Main Method

```
    public static void ortalama(double sayi1, double sayi2) {  
        System.out.println("girdiginiz iki sayinin ortalamasi : " + (sayi1+sayi2)/2);  
    }
```

3

3 – Method

```
} // Class sonu
```



Class Olustururken (Declaration) Kullanilan Keyword'ler Nelerdir?

```
public class MyFirstClass {}  
1      2      3      4
```

- 1 public** : Access Modifier (Erisim duzenleyici) : class'a kimlerin erisebilecegini belirler. Public olursa her yerden erisilebilir
default : Sadece bulunduгу Package'den kullanilabilir
- 2 class** : Yazdigimiz kodun class oldugunu belirtir
- 3 MyFirstClass** : Olusturdugumuz class'in ismidir. Class'a istedigimiz ismi verebiliriz ancak isim verilirken genelde class'da yapilan isleme uygun bir isim secilmesine dikkat edilir.
Isim mutlaka buyuk harfle baslar, birden fazla kelimedenden olursa sonraki kelimelerin ilk harfleri de buyuk harf yazilir (Camel Case)
- 4 Body (Class Body)** : { } arasinda kalan kodlarimizi yazdigimiz bolumdur



Method Olusutururken Kullanilan Keyword'ler Nelerdir?

```
public int myFirstMethod () {}  
1      2      3      4  5
```

- 1 **public** : Access Modifier (Erisim duzenleyici):methoda'a kimlerin erisebilecegini belirler
private: Sadece bulunduğu class'da kullanılabilir
protected : Sadece icinde bulunduğu class ve child class'lardan kullanilir
- 2 **int** : Return Type, methodun ne urettigini ve bize dondurdugunu belirtir
- 3 **myFirstMethod** :Olusturdugumuz method'un ismidir. Isim mutlaka kucuk harfle baslar, birden fazla kelimeden olusursa sonraki kelimelerin ilk harfleri buyuk harf yazilir (Camel Case)
- 4 **()** **parantez**: Methodlarda isimden sonra parantez kullanilir ve gerektiginde parantez icinde parametre yazilir.
- 5 **Body (Method Body)** : { } arasinda kalan kodlarimizi yazdigimiz bolumdur



Main Method

```
public static void main(String[] args) {}
```



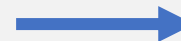
- main method, java'nin calismaya basladigi giristir. (Entry Point)
- main method olusturulurken yazilmasi gereken syntax (kod dizimi) degistirilemez
- Parantez icinde yazilan (String[] args) java'nin calismasi icin gerekli olan parametreleri barindirir ve olmasi sarttir.

Araba



Motor

Java Project



Main Method



Yorum Cümlesi (Comment) Nasıl Eklenir ?

```
public class Example {  
    // Bir satiri comment haline getirmek icin // kullanilir  
    String isim ="Mehmet";  
    /*  
    Eger birden fazla  
    satiri yorum haline  
    getirmek istiyorsak  
    kullanilir  
    int sayi=10;  
    double not=75.70;  
    */  
    boolean ogrenciMi =false;  
}
```

➤ **Comments** : Java tarafından çalıştırılmayan, amacı kodların açıklanması veya bir konuda bilgi vermek olan cümlelerdir

➤ Genelde iki kullanım vardır

1) Tek satirlik comment

2) Çok satirlik comment

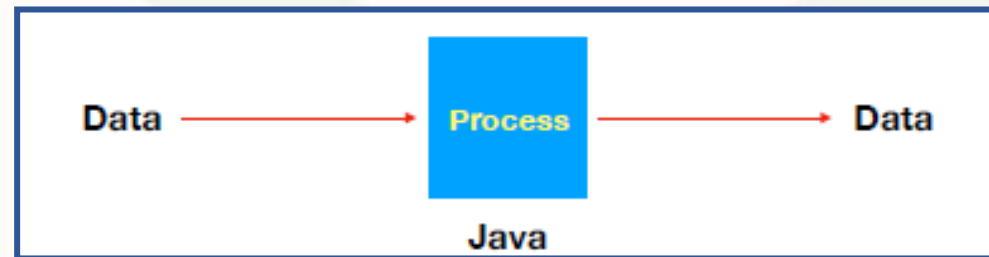


Data

Data bilgisayar tarafından işlenen (**processed**) veya depolanan (**stored**) bilgidir.

Joe, Smith, 1234 Daire, SLC, UT, 8404,8015553211
0143 0157 0155 0160 0165 0164 0145 0162 0040 0150 0157 0160 0145
011000110110111101101101110000011101010111010001100101011100100010000001101000000 101

Java'nin kullandığı (**use**) veya ürettiği (**produce**) her şey data'dır.

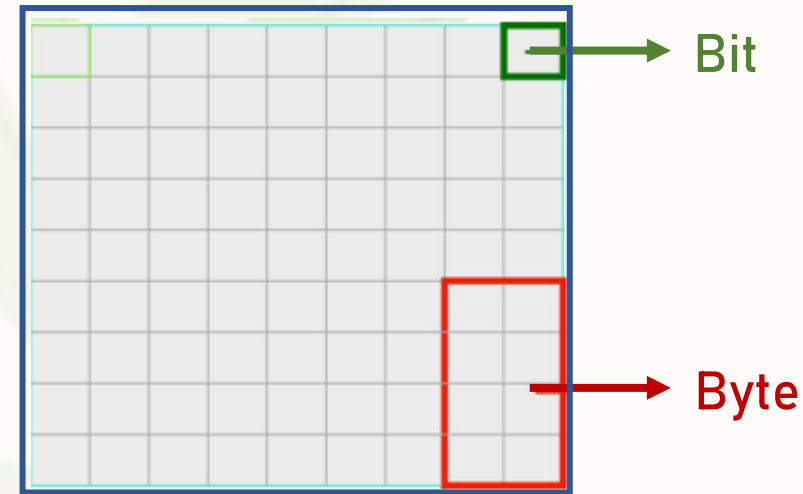
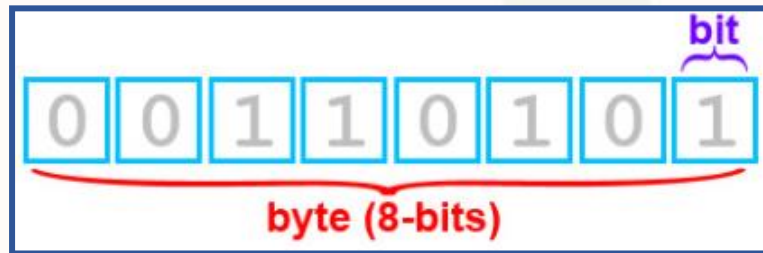




Bit

bit hafızadaki en küçük data parçasıdır. Her “bit” bir binary value içerir, 0 veya 1.

Note: 8 bit =1 byte



Memory
(Hafıza)



23 EKİM 2021
DERS 2

Java Giriş
Variables

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.



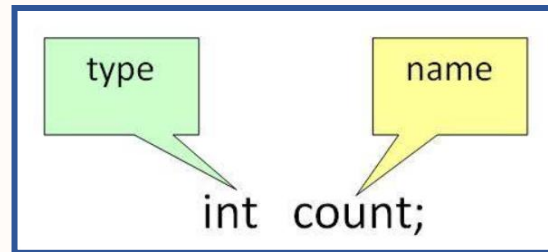
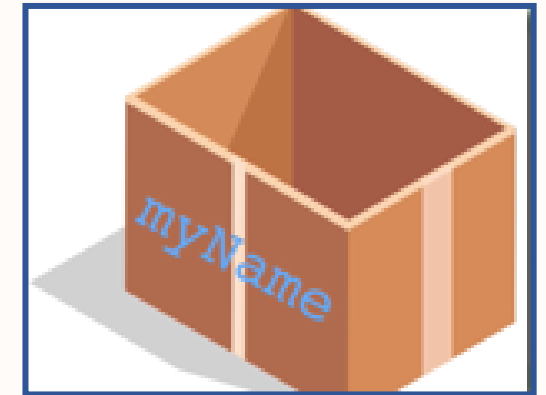
Variables (Degisken) Olusturma Declaration

Variable bellekte (memory) ayrılmis olan alanin (reserved area) adidir.

Variable icinde deger saklayan bir konteynirdir (container).

Bir deęişkende saklanan deęer, program yürütölürken deęiştirilebilir.

Java'da, tüm deęişkenler kullanılmadan önce deklare edilmelidir (variable declaration)



Variable declaration icin iki seyi belirtmemiz gerekiyor

- 1- Data type (data turu)
- 2- Variable Name (degisken ismi)



Variables Deger Atama Assignment

Varolan bir variable'a deger atamaya assignment (atama) denir.

1- Deger atamasi yapilirken data turune uygun deger atanmalidir. Diger turlu Java hata verir.

```
5 public class Example {  
6  
7     String isim ="Mehmet";  
8     boolean ogrenciMi =false;  
9     int not=85;  
10    double ortalama= 78.3;  
11  
12    String ad =75;  
13    boolean emekliMi ="true";  
14    int maas=true;  
15    double yas= "kuru";  
16 }
```



Variables Deger Atama Assignment

2- İlk once declaration, daha sonra atama yapılabilir.

```
String isim ;  
boolean ogrenciMi;  
int not;  
double ortalama;  
  
isim ="Mehmet";  
ogrenciMi =false;  
not=85;  
ortalama= 78.3;
```

3- Bir defa declaration yapıldıktan sonra, birden fazla atama yapılabilir. Java son degeri tutar, oncekini siler.

```
5 public class Example {  
6     public static void main(String[] args) {  
7  
8  
9         int level=1;  
10  
11  
12  
13         level=2;  
14  
15  
16  
17         level=3;  
18  
19  
20     }  
21 }
```




Variables Deger Atama Assignment

4- Ayni data turunde birden fazla variable tek komutla deklare edilebilir.

```
9      int level,yas,maas;  
10  
11     level=5;  
12     yas=20;  
13     maas=10000;
```

5- Ayni data turunde birden fazla variable tek komutla deklare edilip deger atanabilir.

```
9      int level=5, yas=20, maas=10000;
```

Eclips Kullanim

1- Proje olusturma

File -- New -- Project -- (Java Project) Next -- java2021FallTr -- finish

2- Package (paket) olusturma

src dosyasina sag click -- New -- Package -- day01variables -- finish

3- Class olusturma

day01variables dosyasina sag click -- New -- Class -- C01_Variables01 -- finish

4- Main method olusturma

public static void main(String[] args) yazarak main methodu olusturalim



Data Types

Java'da iki data tipi kullanılmaktadır

- **Primitive Data Types** : boolean, char, byte, short, int, long, float ve double
- **Non- Primitive Data Types** : String,

ilerleyen derslerde göreceğimiz primitive olmayan Array, List, Object gibi her data non-primitive'dir.



Primitive Data Types

1) **boolean** Data Type: true veya false barindirir. Hafizada **1 bit** kullanir

Sadece dogru veya yanlis seklinde cevap verilebilecek variable'larda kullanilir

```
boolean isExpensive = true;
```

```
boolean isCold = false;
```

2) **char** Data Type : Tek karakter barindirir. Hafizada **16 bit** kullanir

Harf, sayi veya sembol bakilmaksizin sadece 1 karakter kullanacak variable'larda kullanilir

```
char letter = 'a';
```

```
char digit = '3';
```

```
char cymbol = '#';
```

Note: char degerlerini single quote arasina yazilir.



Primitive Data Types

3) **byte** Data Type: -128 den 127'e (dahil) tamsayilar icin kullanilabilir. Hafizada **8 bit** kullanir

```
byte age = 73;
```

4) **short** Data Type: -32.768 den 32.767'e (dahil) tamsayilar icin kullanilabilir. Hafizada **16 bit** kullanir

```
short ilceNufusu = 27,324;
```

5) **int** Data Type: -2.147.483.648 den 2.147.483.647'e (dahil) tamsayilar icin kullanilabilir. Hafizada **32 bit** kullanir

```
int turkiyeNufusu = 67,324.564;
```

6) **long** Data Type: -9,223,372,036,854,755,808 den ,223,372,036,854,755,807'e (dahil) tamsayilar icin kullanilabilir. Hafizada **64 bit** kullanir



Primitive Data Types

7) **float** Data Type: Kucuk ondalik sayilar icin kullanilabilir. Hafizada **64 bit** kullanir

```
float floatVar2 = -2.123456f;
```

Not: float sayilarin sonunda " f " yazilmalidir, yazilmazsa java sayiyi double kabul eder

8) **double** Data Type: Buyuk ondalik sayilar icin kullanilabilir. Hafizada **64 bit** kullanir

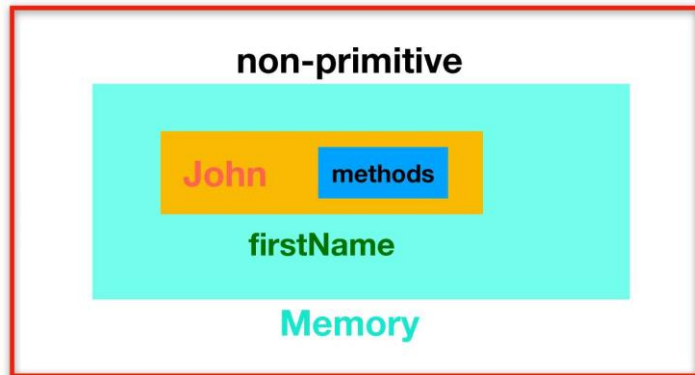
```
double doubleVar2 = -2.12345679078000000000123
```



Non-Primitive Data Type

String Data Type:

String pes pese dizilmiş char'lardan oluşur. Kelimeler, cümleler, matematiksel işlem yapılmayacak sayısal değerler de String olarak tanımlanabilir



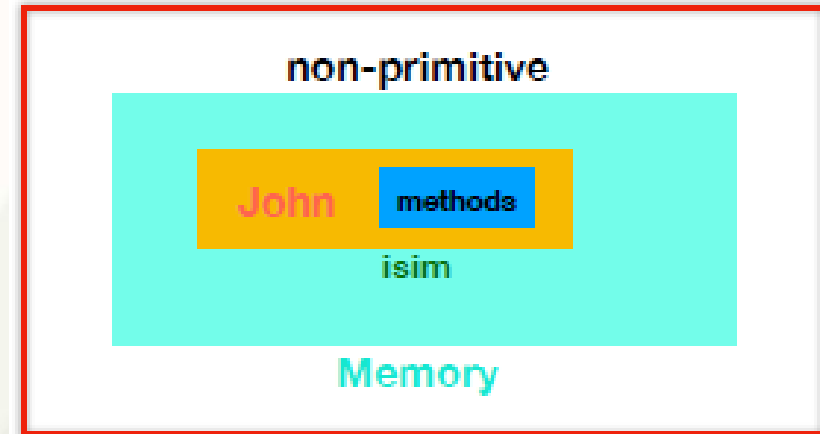
```
String okulAdi = "Yildiz Koleji, Cankaya Ankara #";  
String telNo   = "5321234567";  
String ilkHarf = "A";
```

Note: String'ler çift tırnak (double quotes) arasına yazılır.

Note: Baska non-primitive data type'lar da var, daha sonra öğreneceğiz.



Primitive VS Non-Primitive Data Types



- 1) Primitive'ler sadece value içerir, non-primitive'ler value ve methodlar içerir.
- 2) Primitive'ler küçük harf ile, non-primitive'ler büyük harf ile baslar.
- 3) Primitive'leri Java olusturur biz primitive olusturamayiz.
Non-primitive'leri biz de olusturabiliriz, Java da olusturabilir. Or: String'i Java olusturmudur.
- 4) Primitive'lerin buyuklukleri data type'ing gore degisir ve sabittir. non-primitive'ler icin sabit buyukluk soz konusu degildir.



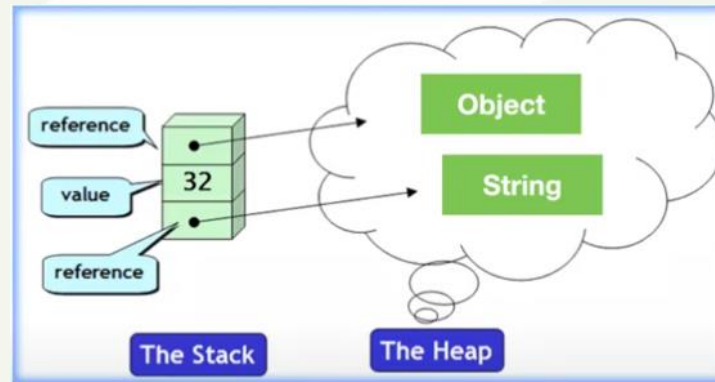
Variable ve Method'lar Nasıl Adlandırılır

1. Java variable isimleri **case sensitive (Buyuk kucuk harfe duyarlidir)**dir.
"money", "Money" veya "MONEY" birbirinden farklıdır
2. Java variable isimleri "harf", "\$" veya "_" ile başlamalıdır.
Fakat "\$" ve "_" ile başlamak tavsiye edilmez.
3. Java variable isimlerinde, ilk harften sonra sayı, "\$" ve "_" kullanılabilir.
4. Variable isimleri için Java'ya özel terimler (key word) kullanılamaz. (int, for, if, import vb).
5. Variable isimleri küçük harflerle yazılır.
6. Variable isimleri 1'den fazla kelime içeriyorsa, ilk kelimeden sonraki her kelimenin ilk harfi büyük harf ile başlamalıdır. firstName, bigApple, ageJohnWalker gibi. Buna camelCase denir.



Memory (Hafıza) Kullanımı

Javada kullanılan iki hafıza vardır

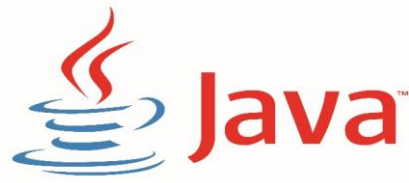


Stack => small

Heap => huge

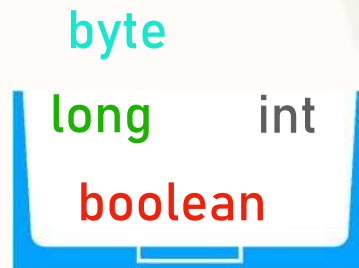
1- **Stack Memory** : primitive data tiplerine ait degerleri ve Non-primitive datalara (Object) ait referanslari(adres) barindirir

2- **Heap Memory** : Non-primitive data'lari depolamak(store) icin kullanilir

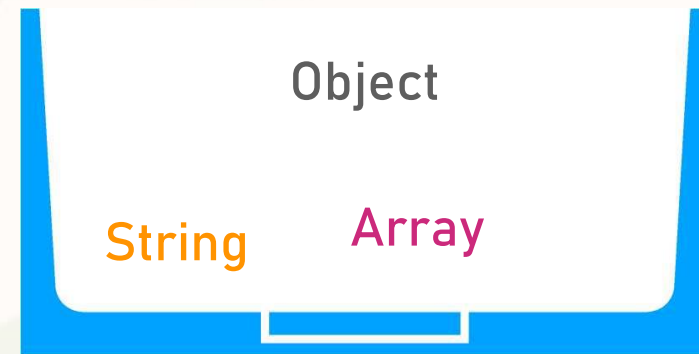


Memory (Hafiza) Kullanimi

reference of an Object



Stack



Heap

Variables Class Work

- 1- Farkli 3 data turunde variable olusturun ve bunlari yazdirin
- 2- isim ve soyisim icin iki variable olusturun ve bunlari
isminiz : Mehmet
soyisminiz : Bulutluoz
sekinde yazdirin
- 3- Iki farkli tamsayi data turunde 2 variable olusturun bunlari toplamini yazdirin
- 4- Bir tamsayi ve bir ondalikli variable olusturun ve bunlari toplamini yazdirin
- 5 - char data turunde bir variable olusturun ve yazdirin
- 6- Bir tamsayi, bir de char degisken olusturun ve bunlari toplamini yazdirin.

Variables Class Work

Interview Question

1- Verilen sayi1 ve sayi2 variable'larinin degerlerini degistiren (SWAP) bir program yaziniz

Orn : sayi1=10 ve sayi2=20;

kod calistiktan sonra

sayi1=20 ve sayi2=10

2- Verilen sayi1 ve sayi2 variable'larinin degerlerini 3.bir variable olmadan degistiren (SWAP) bir program yapiniz

ASCII Table

ASCII control characters			ASCII printable characters			Extended ASCII characters		
00	NULL	(Null character)	32	space	64	@	96	`
01	SOH	(Start of Header)	33	!	65	A	97	a
02	STX	(Start of Text)	34	"	66	B	98	b
03	ETX	(End of Text)	35	#	67	C	99	c
04	EOT	(End of Trans.)	36	\$	68	D	100	d
05	ENQ	(Enquiry)	37	%	69	E	101	e
06	ACK	(Acknowledgement)	38	&	70	F	102	f
07	BEL	(Bell)	39	'	71	G	103	g
08	BS	(Backspace)	40	(72	H	104	h
09	HT	(Horizontal Tab)	41)	73	I	105	i
10	LF	(Line feed)	42	*	74	J	106	j
11	VT	(Vertical Tab)	43	+	75	K	107	k
12	FF	(Form feed)	44	,	76	L	108	l
13	CR	(Carriage return)	45	-	77	M	109	m
14	SO	(Shift Out)	46	.	78	N	110	n
15	SI	(Shift In)	47	/	79	O	111	o
16	DLE	(Data link escape)	48	0	80	P	112	p
17	DC1	(Device control 1)	49	1	81	Q	113	q
18	DC2	(Device control 2)	50	2	82	R	114	r
19	DC3	(Device control 3)	51	3	83	S	115	s
20	DC4	(Device control 4)	52	4	84	T	116	t
21	NAK	(Negative acknowl.)	53	5	85	U	117	u
22	SYN	(Synchronous idle)	54	6	86	V	118	v
23	ETB	(End of trans. block)	55	7	87	W	119	w
24	CAN	(Cancel)	56	8	88	X	120	x
25	EM	(End of medium)	57	9	89	Y	121	y
26	SUB	(Substitute)	58	:	90	Z	122	z
27	ESC	(Escape)	59	;	91	[123	{
28	FS	(File separator)	60	<	92	\	124	
29	GS	(Group separator)	61	=	93]	125	}
30	RS	(Record separator)	62	>	94	^	126	~
31	US	(Unit separator)	63	?	95	_		
127	DEL	(Delete)						
128	Ç		160	á	192	Ł	224	Ó
129	ü		161	í	193	ł	225	ô
130	é		162	ó	194	Ł	226	õ
131	â		163	ú	195	ł	227	ö
132	ä		164	ñ	196	—	228	ø
133	à		165	Ñ	197	†	229	ō
134	á		166	ª	198	ä	230	µ
135	ç		167	º	199	Å	231	þ
136	ê		168	¿	200	Ł	232	þ
137	ë		169	®	201	Œ	233	ú
138	è		170	™	202	ℒ	234	û
139	ï		171	½	203	Œ	235	ü
140	î		172	¼	204	ℒ	236	ý
141	ì		173	¡	205	=	237	ÿ
142	Ā		174	«	206	†	238	—
143	Ă		175	»	207	▣	239	˙
144	É		176	⌘	208	ø	240	≡
145	æ		177	⌘	209	Ð	241	±
146	Æ		178	⌘	210	Ê	242	≡
147	ô		179	⌘	211	Ë	243	¾
148	ö		180	⌘	212	È	244	¶
149	ò		181	Å	213	Ì	245	§
150	û		182	Ā	214	Í	246	÷
151	ù		183	Ă	215	Î	247	˚
152	ÿ		184	©	216	Ï	248	°
153	Û		185	⌘	217	Ĵ	249	ˆ
154	Ü		186	⌘	218	Œ	250	˘
155	ø		187	⌘	219	⌘	251	˙
156	£		188	⌘	220	⌘	252	˚
157	Ø		189	¢	221	⌘	253	²
158	×		190	¥	222	⌘	254	■
159	f		191	Œ	223	⌘	255	nbsp

Kullanıcıdan Deger Alma

1) `Scanner scan = new Scanner(System.in);`

`scan` : olusturdugumuz scanner'in ismidir ve istedigimiz ismi vermemiz mumkundur. Ancak genelde scan ismi kullanilir.

Bu tur isimlendirmelerde genel kurallara uymamiz kodumuzun anlasilabilir olmasi acisinden faydali olacaktir.

2) `System.out.println("Lutfen 100'den kucuk pozitif iki tamsayi giriniz");`

Kullaniciya girmesini istedigimiz degerler icin aciklayici bilgi vermeliyiz.

Burada aciklama olarak ne yazdirsak kodumuz calisir, hatta birsey yazdirmasak da calisir ancak kullanici kendisinden ne istedigimizi bilmezse deger girmesi gerektigini veya ne tur bilgi girmesi gerektigini bilemez

Kullanıcıdan Deger Alma

3) **scan.nextInt()** ile girilen degerleri alabiliriz. Istedigimiz data tipine gore next'ten sonra yazilacak kisim degisir.

```
int num1 = scan.nextInt()  
int num2 = scan.nextInt()
```

nextBoolean()	→ Reads a boolean value from the user
nextByte()	→ Reads a byte value from the user
nextDouble()	→ Reads a double value from the user
nextFloat()	→ Reads a float value from the user
nextInt()	→ Reads a int value from the user
nextLine()	→ Reads a String value from the user
nextLong()	→ Reads a long value from the user
nextShort()	→ Reads a short value from the user



29 HAZIRAN 2021
DERS 3

Scanner
Data Casting
Increment-Decrement

Mehmet BULUTLUOZ
Elk.Elektronik Muh.

Önceki Dersten Aklimizde Kalanlar

- 1) Data : bilgisayarda işlenen(processed) veya depolanan (stored) herseye data denir
- 2) Compile : derleme, bilgiyi işlemek demektir. Java'nin kullanıcının yazdığı kodları bilgisayarın anlayacağı binary kodlara çevirmesidir.
- 3) OOP Concept : Object oriented Programming , java'da Class'ları kullanarak objeler üretiriz, sonra bu objeleri kullanarak application'lar üretiriz. (Lego gibi)
- 4) Class'larda neler bulunur? : fields (pasif,hareketsiz) , methods (dinamik,hareketli)
- 5) Variable nedir ? : Data değerlerini saklamak (store) için kullanılan container'dır.
- 6) Variable için kullanılır ? : Datayı program içerisinde kullanabilmek için variable'lara atarız. Programımız içinde ne zaman variable ismini yazsak, Java bize o variable'a atanan son değeri getirir.

Önceki Dersten Aklımızda Kalanlar

7) Variable nasıl oluşturulur? : Variable oluşturmak için declaration (tanımlama) yapılır. Veri türü (data type) ve variable ismini yazılmalıdır

8) Variable değer atama : Assignment denir. Variable isminin karşına = isareti konularak istenen değer assign edilir.

9) Variable declaration ve assignment için nelere dikkat etmeliyim ?

Declaration ve assignment sırasıyla yapılmalıdır.

Önce declaration, sonra assignment olmalı.

İstersek tek satırda ikisini birden yapabiliriz

```
int sayi = 20 ;
```

istersek de önce declare edip sonra değer atayabiliriz

```
int sayi;
```

```
sayi = 10;
```

Onceki Dersten Aklimizda Kalanlar

10) Class olusturmak icin hangi keyword'lar kullanilir ? :

```
public class ClassIsmi { Class Body }
```

11) Method olusturmak icin hangi keyword'lar kullanilir ? :

```
public String methodIsmi (parametre) { method Body }
```

12) Main method Nedir ?

Java'nin kodlari calistirmaya basladigi giris nokasidir (Entry Point).

13) Main method olusturmak icin kullanılan syntax nedir ? :

```
public static void main (String[] args){ main method Body }
```

Önceki Dersten Aklımızda Kalanlar

14) Java'da kaç çeşit data type vardır ? Java'da temel iki data tipi vardır.

- primitive (ilkel) data types : boolean (true /false) ,
char (tek karakter),
byte, short, int, long (tam sayılar),
float, double (ondalıkli sayılar)
- non-primitive (object): String (simdilik)

15) Java'da kaç çeşit hafıza vardır ? :

stack- (small) primitive datalar ve non-primitive data tiplerinin reference'lerinin store edildiği hafızadır.

Heap memory (huge) : non-primitive dataların store edildiği hafızadır.

Önceki Dersten Aklımızda Kalanlar

1) Print yaparken

- variable'in değerini yazdırmak için `syso` içine variable'in ismini yazarız
- açıklama yazdırmak istersek, `""` içine istediğimiz her şeyi yazabiliriz
- hem açıklama hem variable varsa açıklama `""` içinde, variable'in sadece ismi ve aralarında `+`
- eğer açıklama ile birlikte matematiksel bir işlem yapıyorsak, matematiksel ifadeyi `()` içine almalıyız
- Java'da variable isimleri CASE SENSITIVE'dir. `money`, `Money`, `MONEY` farklıdır
- İki farklı sayı türünde değer toplamını yazdırmak istersek, Java daha geniş olanı tercih eder
- Eğer `char` bir variable matematiksel işlemde kullanılırsa, Java o karakterin ASCII tablosundaki değerini kullanır.

2) Scanner ile kullanıcıdan değer almak için 3 adım takip edilir

- Scanner objesi oluşturmak (Scanner Class'ı import etmeliyiz)
- Kullanıcıdan istediğimiz bilgiyi açık bir şekilde konsolda yazdırmak
- `next method`'unu kullanarak kullanıcıdan aldığımız değeri, oluşturduğumuz variable'a atmalıyız. Oluşturduğumuz variable ve kullandığımız `next method` kullanıcıdan istediğimiz dataya uygun olmalıdır.

Kullanıcıdan Deger Alma Sorular

- Soru 1)** Kullanıcıdan iki tamsayı alıp bu sayıların toplam,fark ve carpimlerini yazdırın
- Soru 2)** Kullanıcıdan karenin bir kenar uzunlugunu alın ve karenin cevresini ve alanini hesaplayip yazdırın
- Soru 3)** Kullanıcıdan yarıcap isteyip cemberin cevresini ve dairenin alanini hesaplayip yazdırın
- Soru 4)** Kullanıcıdan dikdortgenler prizmasının uzun, kısa kenarlarını ve yuksekligini isteyip prizmanın hacmini hesaplayip yazdırın
- Soru 5)** Kullanıcıdan ismini ve soyismini isteyip asagidaki sekilde yazdırın
- İsminiz : Mehmet
Soyisminiz : Bulut
Kursumuza katiliminiz alınmistir,tesekkur ederiz
- Soru 6)** Kullanıcıdan ismini ve soyismini alıp aralarında bir bosluk olusturarak asagidaki sekilde yazdırın
- İsim – soyisim : Mehmet Bulutluoz
- Soru 7)** Kullanıcıdan ismini alıp isminin bas harfini yazdırın.

Data Casting

Veri Sinifi Degistirme

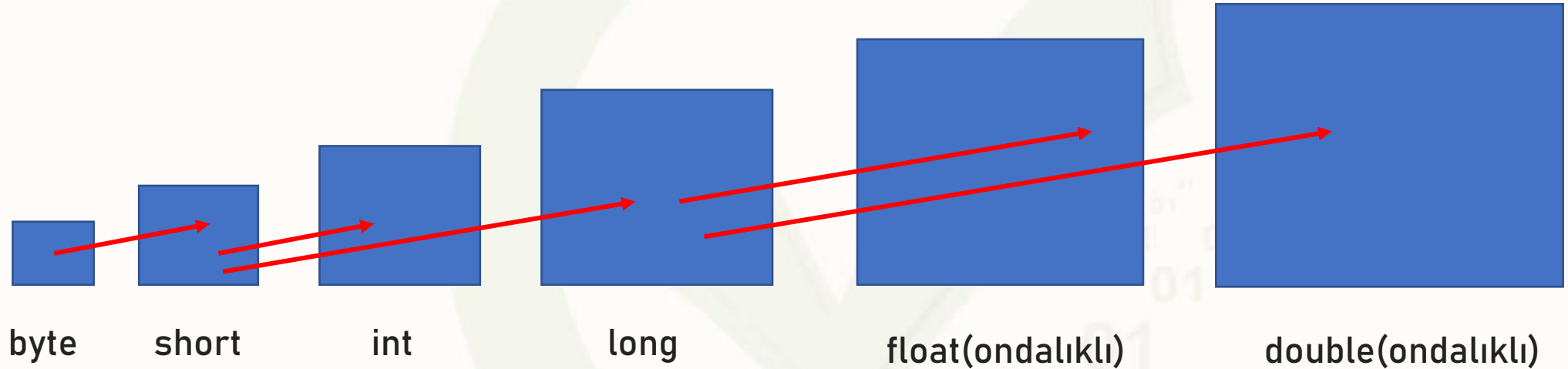
- Java'da kod yazarken bir veri tipinden diğer bir veri tipine aktarım yapmamız gerekebilir.
- Veri tiplerinde bir variable'a , olusturulduđu data tipinden farklı bir data turunden deđer atanmasına Data Casting denir.
- Data casting yaparken aklimizdan cikarmamamız gereken konu data tiplerinin sinirlaridir. Data tipinin sinirlarini asan data casting islemlerinde hata almamamız için dikkat etmemiz gereken bazı durumlar olacaktır.
- Hatırlayacağımız şekilde Java'da sayılarla ilgili data tiplerinin sıralaması şu şekildeydi

byte < short < int < long < float(ondalıklı) < double(ondalıklı)

Data Casting

1) Auto Widening (Otomatik Genisletme)

Dar veri tipinden daha geniş bir veri tipine geçmek istediğimizde Java bunu otomatik olarak yapacaktır.



Orn :

```
byte num1 = 12;  
short num2 = num1;    // yazdirirsak 12 olarak yazdirir  
int num3 = num2;      // yazdirirsak 12 olarak yazdirir  
float num4=num3;      // yazdirirsak 12.0 olarak yazdirir  
double num5=num4;     // yazdirirsak 12.0 olarak yazdirir
```

Data Casting

2) Explicit Narrowing (Manuel Daraltma)

```
public class Main {  
    public static void main(String[] args) {  
        double myDouble = 9.78;  
        int myInt = (int) myDouble; // Manual casting: double to int  
  
        System.out.println(myDouble); // Outputs 9.78  
        System.out.println(myInt);    // Outputs 9  
    }  
}
```

- Genis veri tipinden daha dar bir veri tipine gecmek istedigimizde Java donusumu otomatik olarak YAPMAYACAKTIR.
- Bu durumda Java Casting'in bir problem olusturabilecegini varsayarak sizden MANUEL ONAY isteyecektir.
- Narrowing Casting bazi datalari kaybetmemize yol acabilir, bazen de sayiyi kendi sinirlari icinde kalan baska bir sayiya donusturebilir

Data Casting

- Soru 1)** byte veri tipinde bir degisken olusturun, short,int,float ve double data tiplerinde birer degisken olusturup adim adim widening yapin ve yazdirin
- Soru 2)** int veri turunde bir degisken olusturun ve adim adim narrowing yapin ve yazdirin
- Soru 3)** Float data turunde bir variable olusturun ve yazdirin
- Soru 4)** double 255.36 sayisini int'a ve sonra da olusturdugunuz int sayiyi byte'a ceviris yazdirin
- Soru 5)** int 2 sayiyi birbirine boldurun ve sonucu yazdirin
- Soru 6)** int bir sayiyi double bir sayiya bolun ve sonucu yazdirin
- Soru 7)** Farkli data tipleri ile islem yapip, sonuclarini yazdiralim



30 HAZIRAN 2021
DERS 4

Increment-**D**ecrement
Matematiksel **O**peratorler

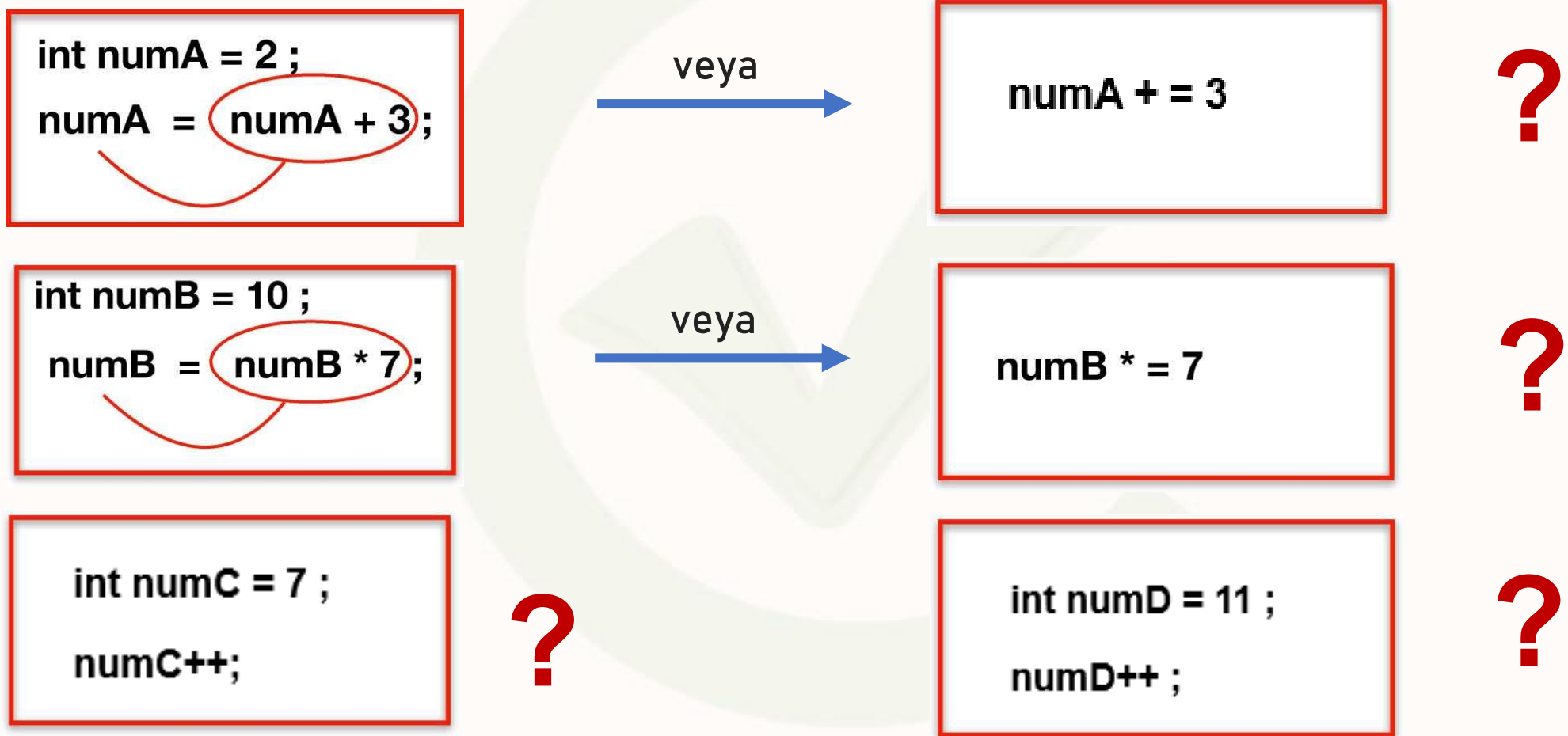
Mehmet BULUTLUOZ
Elk.Elektronik Muh.

Önceki Dersten Aklimizde Kalanlar

- 1) Scanner ile kullanıcidan String bir deger aliyorsak iki secenegimiz var
 - next() kullanirsak ilk space'e kadar olan kısmi alir
 - nextLine() tum satiri alir
 - Eger kullanıcidan tek karakter almak istiyorsak nextChar() method'u olmadigi icin next() ile ilk kelimeyi alip, next()'ten sonra . Koyup charAt(0) kullaniriz
 - scan objesi ile isimiz bittiginde scan.close() method'u ile scanner kapatilir
- 2) Syso ile bir metin yazdirirken, konsolda alt satira gecmek istiyorsak, ikinci satirin baslamasini istedigimiz harften once bosluk birakmadan, \n yazmaliyiz.
- 3) Java'da dizilerdeki elementlere ulasmak icin index kullanilir. Index 0'dan baslar. Örneğin Mehmet kelimesindeki h harfi 3.harftir ama index'i 2 dir.
- 4) Data Casting : bir variable'a kendi data turunden baska turde bir deger atamak istersek Data Casting yapmaliyiz. 2 sekilde data casting olabilir
 - i) Auto Widening (otomatik genisletme) : Eger variable'in data turu atamak istedigimiz degerin data turunden genisse Java otomatik olarak bu atamayi yapar.
 - ii) Explicit Narrowing (Manuel onay ile daraltma) Eger variable'in data turu atamak istedigimiz degerin data turunden dar ise, Java bu islemi otomatik olarak yapmaz. Bizden bu islem icin onay ister,
Çunku explicit narrowing data kayiplarina yol acabilir, hatta variable'in sinirlari disinda ise datanın degistirlmesine sebep olabilir

Increment

Bir Variable'in Degerini Artirma Yontemleri



Decrement

Bir Variable'in Degerini Azaltma Yontemleri

```
int numA = 2 ;  
numA = numA - 3 ;
```

veya

```
numA -= 3
```

?

```
int numB = 20 ;  
numB = numB / 5 ;
```

veya

```
numB /= 5
```

?

```
int numD = 7 ;  
numD -- ;
```

?

```
int numE = 11 ;  
numE -- ;
```

?

Pre Increment & Post Increment

- Pre Increment ve Post Increment operatorlerinin her ikisi de artirma islemi icin kullanilir
- Pre Increment isleminde variable statement'da kullanilmadan once artirilir veya azaltilir

```
public static void main(String[] args) {  
    int a=15;  
    int b=++a;  
    System.out.println(b);  
}
```

Output : 16

- Post Increment isleminde variable statement'da kullanilir, sonra artirilir veya azaltilir

```
public static void main(String[] args) {  
    int a=15;  
    int b=a++;  
    System.out.println(b);  
}
```

Output : 15

Javada Matematiksel Operatorler

- 1- Ustel islemler
- 2- Parantez ici
- 3- Carpma-Bolme
- 4- Toplama-cikarma

Ornek 1 :

$$38 / 2 * (4 + 3) * 2 =$$

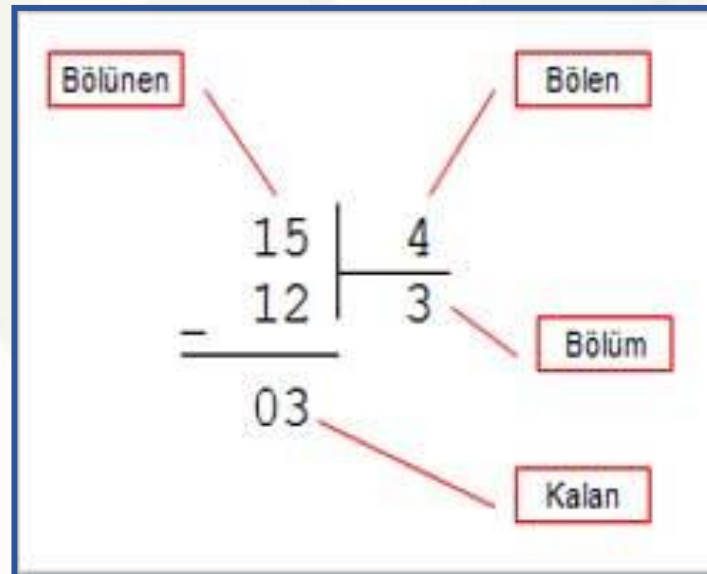
Ornek 2 :

$$8 + 2 * (14 - 6 / 2) - 12 =$$

Modulus %

Modulus islemi bir bolme isleminde kalan sayiyi bize verir

```
public static void main(String[] args) {  
    int a=15 % 4;  
    System.out.println(a);  
}
```



Modulus %

Soru 1) Kullanıcıdan 4 basamaklı bir sayı alın ve rakamlar toplamını bulup yazdırın

İpucu 1:

Sayı % 10 => Bize son basamağı verir

$$538 \% 10 = 8$$

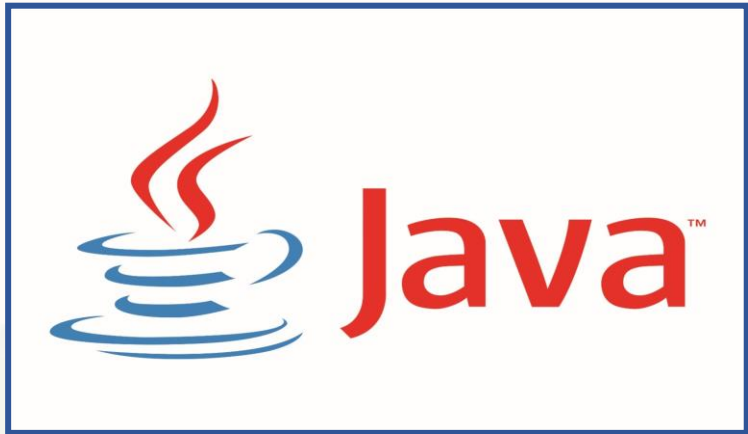
İpucu 2:

Int Sayı /10 => Bize son basamak hariç sayıyı verir

int sayı=538;

sayı = sayı / 10 =>

sayı'ya 53 değerini atar



1 TEMMUZ 2021
DERS 5

Wrapper Class
Concatenation
Karsilastirma Operatorleri

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Önceki Dersten Aklımızda Kalanlar

1) Increment / Decrement : Javada variable'ların değerini artırıp azaltmak demektir.

`int sayi=10;`

`sayi = sayi +1;` bu en basit yazım ama tercih etmiyoruz

`sayi +=1 ;` Bu iki yazımda `=` olmak zorunda yani assignment olmalıdır

`sayi++;` assignment olmadan direk sayının değerini bir artırır.

2) Pre / post : `++` veya `-` variable isminden önce yazılırsa buna pre-increment denir. Bu durumda önce variable'in değeri 1 artırılır, sonra o satırdaki diğer işlem yapılır.

`++` veya `-` variable isminden sonra yazılırsa buna post-increment denir. Bu durumda önce satırdaki diğer işlem yapılır, sonra sayının değeri artırılır.

3) Matematiksel işlemler

- işlem sırası önemlidir ve Java matematikte geçerli işlem önceliğine göre çalışır

i) parantez - us

ii) carpma - bolme

iii) Toplama - cikarma

Aynı öncelikli iki işlem olursa önce soldaki yapılır

4) Modulus operatörü

`sayi % 10` bize sayının 10 ile bölümünden kalanı verir. (yani son basamağı verir)

Wrapper Class

Java **primitive** data turleri ile **methodlari** kullanabilmemiz icin **Wrapper class**'lari olusturmudur.

Character,Byte,Integer,Short,Float,Double primitive data turleri icin olusturulan wrapper class'lardir.

```
public class Example {  
    public static void main(String[] args) {  
  
        int num1 = Integer.MIN_VALUE;  
        System.out.println(num1);  
  
        int num2 = Integer.MAX_VALUE;  
        System.out.println(num2);  
  
        int num3 = Byte.MIN_VALUE;  
        System.out.println(num3);  
  
        int num4 = Byte.MAX_VALUE;  
        System.out.println(num4);  
  
    }  
}
```

-2147483648

2147483647

-128

127

Concatenaion (String Datalari Birlestirme)

Birden cok String'i + isareti ile topladiginizda Java bu String degiskenleri birlestirerek yeni bir String olusturur

```
String a = "Hello";  
String b = "World";  
System.out.println(a+b);  
System.out.println(a+" "+b);
```

→ HelloWorld
→ Hello World

Not : Eger matematiksel bir islemin icinde String kullanilrsa, matematikteki oncelikler dikkate alinarak islem yapilir. Sira String ile toplamaya geldiginde toplama yerine Concatenation uygulanir

```
String a = "Hello";  
int b = 2;  
int c = 3;  
  
System.out.println(a+b+c);  
System.out.println(c+b+a);  
System.out.println(a+(b+c));  
System.out.println(a+b*c);
```

→ Hello23
→ 5Hello
→ Hello5
→ Hello6

Concatenation (String Datalari Birllestirme)

Soru 1) Asagida verilen variable'lari kullanarak istenen sonuclari yazdiran programlari yaziniz.

Variables

```
String str1= "Java";  
String str2= "Guzel";  
int sayi1=5;  
int sayi2=4
```

Istene Yazilar

- 1) Java Guzel 54
- 2) Java 5 Guzel
- 3) Java 94
- 4) Java 19
- 5) 54 Guzel



2 TEMMUZ 2021
DERS 6

Karsilastirma Operatorleri
If Statements

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Önceki Dersten Aklımızda Kalanlar

- 1) Wrapper Classes : primitive data türleri değer alırlar ama method'ları yoktur. Primitive data türleri ile hazır method'lar kullanabilmek için Java Wrapper Class'ları oluşturmıştır. Character, Boolean, Byte, Short, Integer, Long, Float ve Double
- 2) Eğer bir String sadece sayılardan oluşuyorsa Integer.parseInt() method'u ile int'a çevirilebilir.
- 3) Concatenation : Java, String bir variable ile + işaretinin kullanıldığını gördüğünde , toplama işlemi yapmaz + işaretinin iki tarafındaki değerleri BİRLEŞTİRİR.
*** Java'nın concatenation yapması için + işaretinin bir tarafında String olması yeterlidir.
- 4) Matematiksel bir değeri String'e çevirmek istersek "" ile concat etmemiz yeterlidir.
- 5) İki sayıyı toplamak değil de birleştirmek (yan yana yasmak) istersek basa veya araya "+" yaparız.
""+sayi1+sayi2 sayi1+ "" + sayi2
- 6) Eğer char bir değişken matematiksel bir işlemde kullanılırsa, Java o karakterin ASCII değerini matematiksel işlemde kullanır. Eğer ascii değerinin devreye girmesini istemiyorsak öncesinde "" ile concat yapabiliriz.
- 7) Concat yapılırken Java matematikteki işlem öncelik sırasına uygun hareket eder. Örneğin carpma ve toplama varsa önce carpma yapar.

Relational Operators (Karsilastirma Operatorleri)

= Assignment (Atama yapar) operatoru

`int num1=3;` num1 degiskenine 3 degerini atar

`String str1 = "Ali" + " " + "Can";` str1'e Ali Can degeri atar

`c = c+5;` c'nin degerini 5 artirir ve son degeri c'ye atar

== Cift esittir isareti / karsilastirma (Comperison) operatoru

`boolean sonuc1 = 5+2 == 7;` sonuc1 degeri **true** olur

`boolean sonuc2 = 5*2 == 15;` sonuc2 degeri **false** olur

Relational Operators (Karsilastirma Operatorleri)

!= Esit degildir isareti

boolean sonuc1= 5+2 != 7; sonuc1 degeri **false** olur

System.out.println(5*2 != 15); **true** yazdirir

> Buyuktur , >= Buyuk veya esittir

boolean sonuc1= 5+2 >= 7; sonuc1 degeri **true** olur

System.out.println(5*2 > 15); **false** yazdirir

< Kucuktur , <= Kucuk veya esittir

boolean sonuc1= 5+2 < 7; sonuc1 degeri **false** olur

System.out.println(5*2 < 15); **true** yazdirir

Conditional Operators (Sart Operatorleri)

&& AND (ve) isareti

&& isareti ile birlestirilen tum ifadeler dogru ise sonuc true olur.
Diger tum durumlarda false doner. (&& operatoru mukemmeliyetcidir)

```
boolean sonuc1= (5+2 == 7) && (4+3 !=5) ;  
System.out.println((5*2 != 15) && (5>7));
```

sonuc1 degeri **true** olur
false yazdirir

|| OR (veya) isareti

|| isareti ile birlestirilen tum ifadeler yanlis ise sonuc false olur.
Diger tum durumlarda truee doner. (|| operatoru iyimserdir)

```
boolean sonuc1= (5+2 == 7) || (4+3 !=5) ;  
System.out.println((5*2 == 15) || (5>7));
```

sonuc1 degeri **true** olur
false yazdirir

If Statements (If cumleleri)

Eger hava guzel olursa piknige gidecegiz. (guzel olmazsa karar yok)

Eger (hava guzel olursa) {piknige gideriz} her durumda alt satira gecer

If (boolean sart) {sart saglanirsa istenen kod} her durumda alt satira gecer

```
public static void main(String[] args) {  
  
    int a = 2;  
    int b = 3;  
  
    if (a>b) {  
        System.out.println(a+b);  
    }  
    if (a==b) {  
        System.out.println(a*b);  
    }  
}
```

If Statements (If cumleleri)

Not : If statement birden fazla olursa hepsi birbirinden bagimsiz olur. If cumlelerini birbirine baglamayi da ogrenecegiz.

Eger hava guzel olursa piknige gidecegiz. (guzel olmazsa karar yok)

Eger Ali ararsa ona kizacagim. (aramazsa karar yok)

Eger aksam mac varsa onu izleriz. (mac yoksa karar yok)

```
8      int a=10;
9      int b=8;
10
11     if (a==b) {
12         System.out.println("iki sayi esit");
13     }
14
15     if (a+b<100) {
16         System.out.println("sayilarin toplami yuzden kucuk");
17     }
18
19     if (a*b>1000) {
20         System.out.println("sayilarin carpimi bin'den buyuk");
21     }
```

If Statements (Sorular)

Soru 1) Kullanıcıdan bir tamsayı isteyin ve sayının tek veya çift olduğunu yazdırın

Soru 2) Kullanıcıdan gün isimlerinden birinin ilk harfini isteyin ve o harfle başlayan gün isimlerini yazdırın

Örnek: ilkHarf=P output = "Pazar, Pazartesi veya Perşembe"
ilkHarf=S output = "Salı"

***** Büyük küçük harf problem olmaması için toUpperCase methodunu kullanın**

Soru 3) Kullanıcıdan gün ismini alın ve hafta içi veya hafta sonu olduğunu yazdırın

Örnek: gün=Pazar output = "Hafta sonu"
gün=Salı output = "Hafta içi"

***** String için equals method'unu kullanın**

Soru 4) Kullanıcıdan dikdörtgenin kenar uzunluklarını isteyin ve dikdörtgenin kare olup olmadığını yazdırın

Soru 5) Kullanıcıdan bir gün alın eğer gün "Cuma" ise ekrana "Müslümanlar için kutsal gün" yazdırın. "Cumartesi" ise ekrana "Yahudiler için kutsal gün" yazdırın. "Pazar" ise ekrana "Hristiyanlar için kutsal gün" yazdırın



3 TEMMUZ 2021
DERS 7

If Else Statements

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Önceki Dersten Aklımızda Kalanlar

1) Javada kullanılan karşılaştırma operatorleri

= Assignment operatoru, eşitliğin sol tarafına variable ismi, sağ tarafına ise değer yazılır

==, != karşılaştırma operatorleri denir.

<, <= karşılaştırma operatorleri matematiksel işlem değil, mantıksal karşılaştırma yapar

>, >= sonuçları true veya false olur. (true veya false dönerler)

2) Conditional Operatorler && (And), || (OR)

&& mukemmeliyetçidir, and ile birbirine bağlı şartların tamamı True olursa sonuç True olur, bunun dışındaki tüm durumlar False'dur.

|| iyimserdir, OR ile birbirine bağlanan şartların tümü F olursa sonuç F olur, bunun dışındaki tüm durumlar True'dur.

3) If Statements : If(boolean sart) { sart doğru ise çalışacak kod}

sart'ın sonucu true ise {body} çalışır, false ise {body} çalışmaz

{body} çalışsa da çalışmasa da her durumda {body}den sonraki satır çalışır

if cümleleri birbirinden bağımsızdır, hepsi de çalışabilir, hiçbirisi de çalışmayabilir

(boolean sart) bölümüne boolean sonuç döndürmeyen bir kod yazıldığında Java hata verir

Socrative Quiz

- 1) <https://www.socrative.com/> adresine gidin
- 2) **Login** butonuna basin
- 3) **Student Login** butonuna basin
- 4) Room Name **BULUTLUOZ** yazin
- 5) Isminizi yazin
- 6) **Done** butonuna basin

Sure : 15 Dakika

If Else Statements

Eger hava guzel olursa piknige gideriz, yoksa evde otururuz.

Eger (hava guzel olursa) {piknige gideriz} **yoksa** {evde otururuz}

If (boolean sart) {sart saglanirsa istenen kod} **else** {sart saglanmazsa istenen kod}

```
public static void main(String[] args) {  
  
    int a = 2;  
    int b = 3;  
  
    if (a>=b) {  
        System.out.println(a+b);  
    } else {  
        System.out.println(a*b);  
    }  
}
```

If Else Statements (Sorular)

- Soru 1) Kullanıcıdan dikdörtgenin kenar uzunluklarını isteyin ve dikdörtgenin kare olup olmadığını yazdırın
- Soru 2) Kullanıcıdan bir karakter girmesini isteyin ve girilen karakterin harf olup olmadığını yazdırın
- Soru 3) Kullanıcıya yaşını sorun, eğer yaş 65'den küçük ise "emekli olamazsın, çalışmalısın", 65'den büyükse "Emekli olabilirsiniz" yazdırın
- Soru 4) Kullanıcıdan bir üçgenin üç kenar uzunluğunu alın eğer üç kenar uzunluğu birbirine eşit ise ekrana "Eşkenar üçgen" yazdırın. Diğer durumlarda ekrana "Eşkenar değil" yazdırın.

& ile && Arasindaki Fark

& isareti kullanildiginda Java isaretin iki yanindaki mantiksal ifadelerin ikisini de kontrol eder. Bu islem kodumuzu yavaslatir

40<30 & 50==50 & 60>50

ilk karsilastirma yanlis olmasina ragmen Java tum karsilastirmalari kontrol etmeye devam eder.

&& isareti kullanildiginda ise Java en bastan kontrol etmeye baslar, mantiksal ifadelerin birinde yanlisi bulursa sonrakileri kontrol etme ihtiyaci duymaz. Bu islem kodumuzu hizlandirir

40<30 && 50==50 && 60>50

ilk karsilastirma yanlis oldugunu gorunce Java diger karsilastirmalari kontrol etmeden alt satira gecer.

If Else If ... Statements

Eger soruyu biliyorsa Ali soruyu cozsun , o bilmiyorsa Veli biliyorsa Veli cozsun, o da bilmiyorsa Ayse biliyorsa, Ayse cozsun, o da bilmiyorsa Fatma biliyorsa, Fatma cozsun, o da bilmiyorsa kim isterse o cozsun.

Eger soruyu biliyorsa Ali soruyu cozsun , **o bilmiyorsa Veli biliyorsa** Veli cozsun, **o da bilmiyorsa Ayse biliyorsa**, Ayse cozsun, **o da bilmiyorsa Fatma biliyorsa**, Fatma cozsun, **o da bilmiyorsa** kim isterse o cozsun.

```
If (sart) {sart saglanirsa istenen kod} else if {sart saglanmazsa istenen kod}  
else if {sart saglanmazsa istenen kod} else if ( kac tane durum varsa else if ..... )  
else {sart saglanmazsa istenen kod}
```

If Else If Statements (Sorular)

- Soru 5) Kullanıcıdan gun ismini yazmasını isteyin. Girilen isim geçerli bir gün ise gün isminin 1.,2. ve 3.harflerini ilk harf büyük diğer ikisi küçük olarak yazdırın, gün ismi geçerli değilse “Geçerli gün ismi giriniz” yazdırın
- Soru 6) Kullanıcıdan iki sayı isteyin, sayıların ikisi de pozitif ise sayıların toplamını yazdırın, sayıların ikisi de negatif ise sayıların carpımını yazdırın, sayıların ikisi farklı işaretlere sahipse “farklı işaretlerde sayılarla işlem yapamazsın” yazdırın, sayılardan sıfıra eşit olan varsa “sıfır carpıma göre yutan elemandır” yazdırın.
- Soru 7) Kullanıcıdan 100 üzerinden notunu isteyin. Not’u harf sistemine çevirip yazdırın. 50’den küçükse “D”, 50-60 arası “C”, 60-80 arası “B”, 80’nin üzerinde ise “A”
- Soru 8) Kullanıcıdan maaş için bir teklif isteyin ve aşağıdaki değerlere göre cevap azdırın.
Teklif 80.000’in üzerinde ise “Kabul ediyorum” ,
60 – 80.000 arasında ise “Konusabiliriz”,
60.000’nin altında ise “Maalesef Kabul edemem” yazdırın

Nested If Else Statements

Eger calisan kadinsa 60 yasından büyük olduğunda emekli olabilir, calisan erkekse 65 yasından büyükse emekli olabilir

Eger (calisan kadinsa) {Kadin yasini kontrol et} ,
yoksa {erkek yasini kontrol et}

```
If (calisan kadinsa)
    {if (yas>60) {emekli olabilirsiniz} else {emekli olamazsin}}
else
    {if (yas>65) {emekli olabilirsiniz} else {emekli olamazsin}}
```

If Else Statements

Sorular

Soru 11) Nested If kullanarak asagidaki soruyu cozen kodu yaziniz.

Kullanıcıdan bir sifre girmesini isteyin

Eger ilk harf buyuk harf ise "A" olup olmadigini kontrol edin. Ilk harf A ise "Gecerli Sifre" degilse "Gecersiz Sifre" yazdirin.

Eger ilk harf kucuk harf ise "z" olup olmadigini kontrol edin. Ilk harf z ise "Gecerli Sifre" degilse "Gecersiz Sifre" yazdirin.

Soru12)Kullanıcıdan 4 basamakli bir sayi girmesini isteyin. Girdiği sayi 5'e bölünüyorsa son rakamını kontrol edin. Son rakamı 0 ise ekrana "5'e bölünen çift sayı" yazdırın. Son rakamı 0 değil ise "5'e bölünen tek sayı" yazdırın. Girdiği password 5'e bölünmüyorsa ekrana "Tekrar deneyin" yazdırın.



5 TEMMUZ 2021
DERS 8

Ternary Operator
Switch Case

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Önceki Dersten Aklimizde Kalanlar

- 1) If Else Statements : If cümleleri eğer else ile birbirine bağlanmazsa bağımsız olarak çalışırlar. Bu durumda her bir if body'si çalışabilir, çalışmayabilir.
- 2) If cümlelerini else ile birbirine bağladığımızda, bağlı if statement'lerden sadece biri çalışır, diğerleri çalışmaz
- 3) Bazen tek bir if else ile soruları çözemeyiz, bu durumda if else statement'leri iç içe (nested) kullanmamız gerekir

If Else If Statements (Sorular)

Soru 9) Interview Question

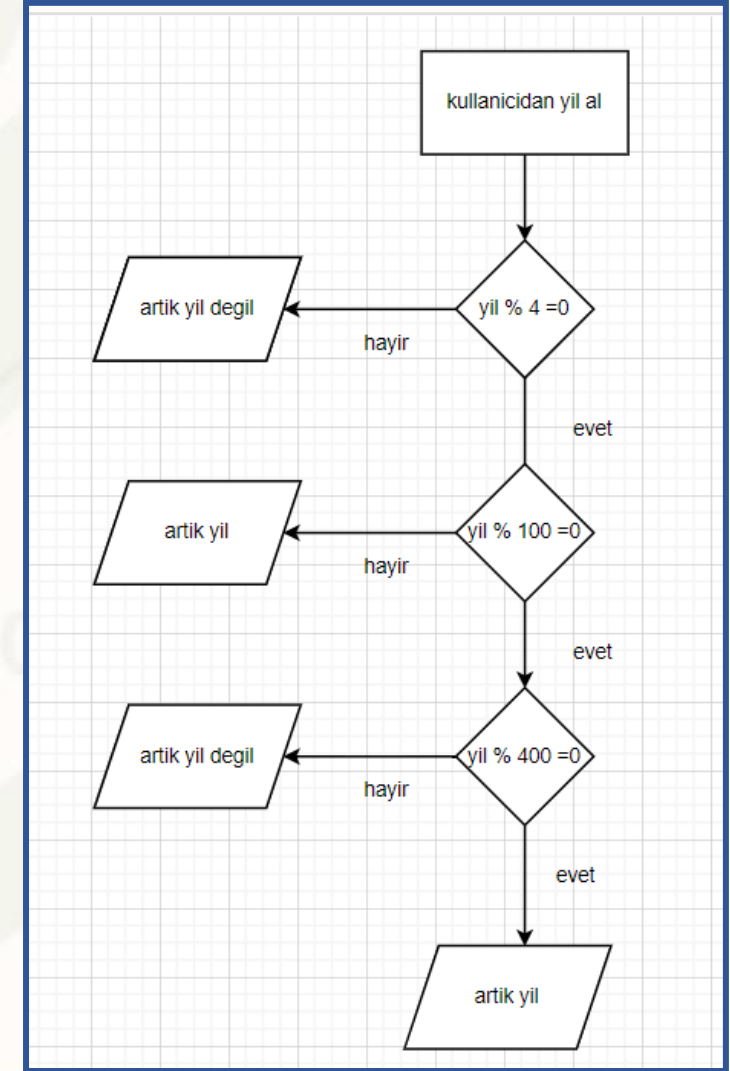
Kullanıcıdan artık yıl olup olmadığını kontrol etmek için yıl girmesini isteyin.

Kural 1: 4 ile bölünemeyen yıllar artık yıl değildir

Kural 2: 4 ile bölünüp 100 ile bölünemeyen yıllar artık yıldır

Kural 3: 4'ün kati olmasına rağmen 100 ile bölünebilen yıllardan sadece 400'ün kati olan yıllar artık yıldır

<https://app.diagrams.net/>



Nested If Else Statements

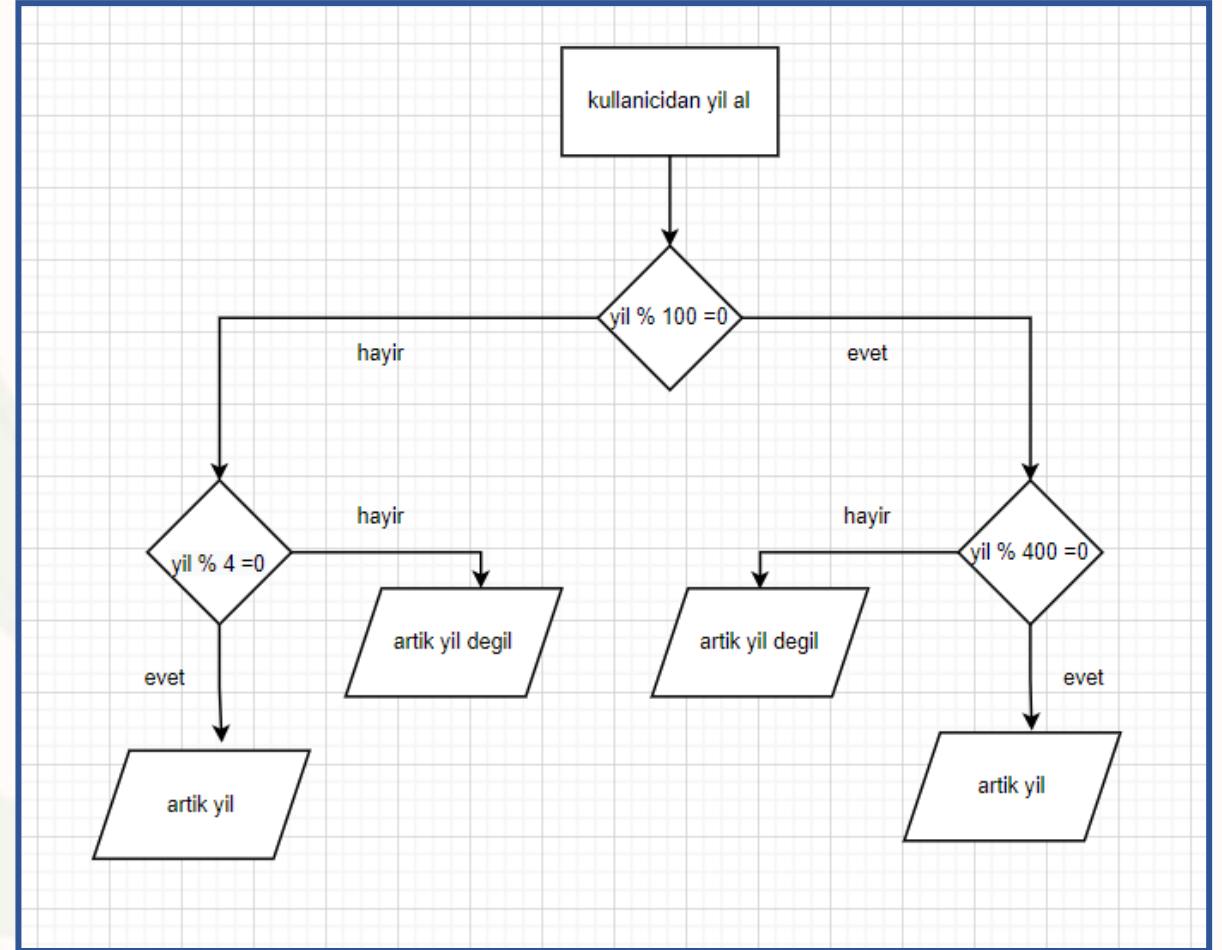
Sorular

Soru 10) Interview Question

Kullanıcıdan artık yıl olup olmadığını kontrol etmek için yıl girmesini isteyin.

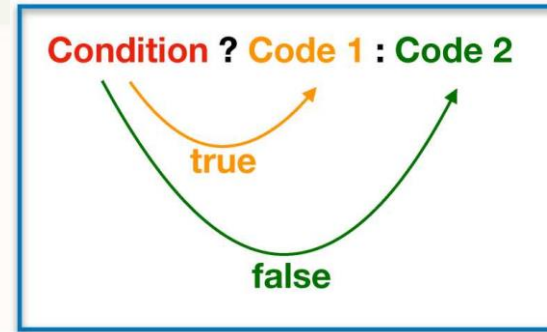
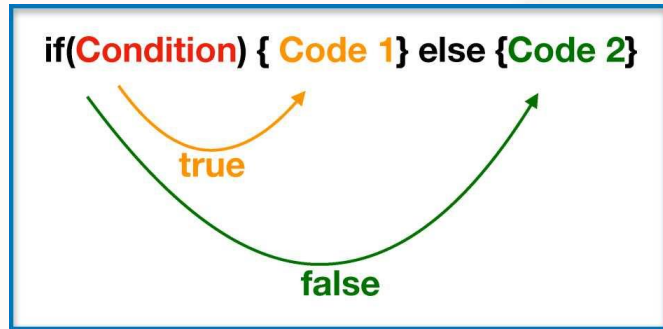
Kural 1: 4 ile bölünemeyen yıllar artık yıl değildir

Kural 2: 4'ün kati olmasına rağmen 100 ile bölünebilen yıllardan sadece 400'ün kati olan yıllar artık yıldır



<https://app.diagrams.net/>

Ternary Operator



Not1 : Ternary islemi If Statement ile yapacagimiz islemleri basit olarak yapmamizi saglar

Not2 : Ternary islemi bize bir sonuc donecegi icin, bu islemi bir variable'a atamaliyiz.

```
public static void main(String[] args) {  
    int x=10;  
  
    (x/2==0) ? "cift sayi" : "tek sayi";  
}
```

```
public static void main(String[] args) {  
    int x=10;  
  
    String sonuc = (x/2==0) ? "cift sayi" : "tek sayi";  
    System.out.println(sonuc);  
}
```

Ternary Operator

Ekranda Ne Goruruz ?

Soru1 : int y = 112;

```
System.out.println( (y > 5) ? ("Inek") : ("Koyun") );
```

Soru2 : int y = 112;

```
System.out.println((y < 91) ? 9 : 11);
```

Soru3 : int y = 1;

```
int z = 1;
```

```
int a = y<10 ? y++ : z++;
```

```
System.out.println(y + "," + z + "," + a);
```


Ternary Operator Sorular

Soru1) Kullanıcıdan iki sayı alın ve büyük olmayan sayıyı yazdırın

Soru2) Kullanıcıdan bir tamsayı alın ve sayının tek veya çift olduğunu yazdırın

Soru3) Kullanıcıdan bir sayı alın ve sayının mutlak değerini yazdırın

Soru4) Kullanıcıdan bir sayı alın. Sayı pozitifse "Sayı pozitif" yazdırın, negatifse sayının karesini yazdırın

Nested Ternary

Condition ? (Kod 1) : (Kod 2) ;

Condition1 ? Durum1 : Durum2

Condition2 ? Durum1 : Durum2

Soru1 : Kullanıcıdan bir tamsayı alın ve sayı 10'dan küçükse "Rakam" , 100'den küçükse "iki basamaklı sayı" değilse "uc basamaklı veya daha büyük sayı" yazdırın

Soru2 : Kullanıcıdan bir harf isteyin küçük harf ise consola "Küçük Harf" , büyük harfse consola "Büyük Harf" yoksa "girdiğiniz karakter harf değil" yazdırın.

Nested Ternary

Ekranda Ne Goruruz ?

Soru1 : int y = 8;

$(y > 5) ? (y < 10 ? 2 * y : 3 * y) : (y > 10 ? 2 + y : 3 + y);$

Soru2 : int y = 12;

$(y > 5) ? (y < 10 ? 2 * y : 3 * y) : (y > 10 ? 2 + y : 3 + y);$

Soru3 : int y = 5;

$(y > 5) ? (y < 10 ? 2 * y : 3 * y) : (y > 10 ? 2 + y : 3 + y);$

Soru4) Kullanıcıdan dikdörtgenin uzunlugunu ve genisligini alin, girilen degerlere gore dikdorgenin kare olup olmadigini yazdirin.

Soru5) Kullanıcıdan bir sayi alin ve sayi 3 basamakli ise “uc basamakli sayi”, yoksa “Uc basamakli degil” yazdirin



6 TEMMUZ 2021
DERS 9

Switch Case

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Önceki Dersten Aklımızda Kalanlar

- 1) Ternary : if ile aynı görevi yapar ama if'e göre daha basit yapıdadır,
- 2) Ternary'nin içine kodlar yazamayız ama if gibi düşünüp sonuç değerleri atayabiliriz
- 3) Ternary her zaman bize bir sonuç döndürecek için sonuca uygun bir variable'a atama yapabilir veya direkt sonucu yazdırmak için `syso` içine yazabiliriz
- 4) Eğer bir variable'a atama yapacaksak şartın `true` ve `false` olması durumlarında alacağı değerlerin data türleri aynı olmak zorundadır, direkt yazdıracaksak sonuçlar aynı data türünden olmayabilir

```
String sonuc = sayi%2==0 ? "sayi çift" : "sayi tek" ;
```

```
syso( sayi%2==0 ? "sayi çift" : 20) ;
```

- 5) Şart kısmi boolean olmalı
- 6) Eğer tek bir şart ile soruyu çözmiyorsak iç içe ternary (nested) kullanabiliriz
- 7) boolean şart ? `True` olursa sonuç : `False` olursa sonuç ;

Switch Statement

If else ile cozdugumuz sorularda **kontrol etmemiz gereken** sart sayisi cok oldugunda **switch Statement** kullanilir.

```
public static void main(String[] args) {  
    int sayi = 3;  
  
    switch(sayi) {  
        case 1 :  
            System.out.println("sayı = 1");  
            break;  
        case 2 :  
            System.out.println("sayı = 2");  
            break;  
        case 3 :  
            System.out.println("sayı = 3");  
            break;  
        case 4 :  
            System.out.println("sayı = 4");  
            break;  
        default :  
            System.out.println("sayı bunlardan biri değil");  
    }  
}
```

Switch Statement

break komutu yapacagimiz islem bittiginde switch statement'in sonuna gitmemizi saglar.

Java istenen case'e gittikten sonra **break** komutunu gorene kadar tum case'leri calistirir.

default komutu basta tanimlanan degisken icin hic bir case calismazsa calistirmek isedigimiz kodlari yazdigimiz bolumdur.

(If else statements da en sonda yazdigimiz else gibi calisir)

Switch Statement'da long,double,float ve boolean **kullanilamaz**

Switch Statement

Sorular

Soru1 : Kullanıcıdan haftanın kacinci gunu oldugunu sorun ve gun ismini yazdirin

Soru2 : Kullanıcıdan kacinci ay oldugunu sorun ve ay ismini yazdirin

Soru3 : Kullanıcıdan bir sayi girmesini isteyin

Girilen sayi

10 ise “İki basamakli en kucuk sayi

100 ise “uc basamakli en kucuk sayi”

1000 ise “dort basamakli en kucuk sayi”

diger durumlarda “Girdigin sayiyi degistir” yazdirin

Soru4 : Kullanıcıdan SDET kisaltmasindaki harflerden birini yazmasini isteyin.

Kullanici S girerse “Software”

D girerse “Developer”

E girerse “Engineer”

T girerse “In Testing” yazdirin

Soru5 : Kullanıcıdan gun ismini alip haftaici veya hafta sonu yazdiralim

String Manipulation / Methods

1- concatenation

Birden fazla String'i birleştirerek tek bir String haline getirmek için kullanılır.

İki şekilde kullanılır.

i) + (toplama) isareti ile

```
public static void main(String[] args) {  
  
    String isim= "Ali";  
    String soyisim="Can";  
  
    System.out.println(isim + " " + soyisim);  
}
```

Output :

Ali Can

ii) concat() methodu kullanarak

```
public static void main(String[] args) {  
  
    String isim= "Ali";  
    String soyisim="Can";  
  
    System.out.println(isim.concat(soyisim));  
}
```

Output :

AliCan

String Manipulation / Methods

1- charAt()

İstenen indexdeki karakteri (char) döndürür. Index 0'dan başlar, maximum index (String'in uzunluğu - 1) dir.

```
public static void main(String[] args) {  
    String isim= "Techproeducation";  
    System.out.println(isim.charAt(3));  
}
```

Output :

h

Eğer method'da index olarak maximum indexden büyük bir sayı kullanılırsa Java hata verir (**StringIndexOutOfBoundsException**).

```
public static void main(String[] args) {  
    String isim= "Techproeducation";  
    System.out.println(isim.charAt(20));  
}
```

```
Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String index out of range: 20  
    at java.lang.String.charAt(Unknown Source)  
    at _00_anlik.asd.main(asd.java:11)
```

String Manipulation / Methods

3-toUpperCase()

4-toLowerCase()

Girilen String degiskendeki tum harfleri istenen bicime ceviri.

```
public static void main(String[] args) {  
    String isim= "TechProEducation";  
  
    System.out.println(isim.toLowerCase());  
    System.out.println(isim.toUpperCase());  
}
```

Output :

```
techproeducation  
TECHPROEDUCATION
```

NOT : toLowerCase(Locale locale)

Girilen String degiskendeki tum harfleri istenen local dilde istenen bicime ceviri.

```
public static void main(String[] args) {  
    String isim= "TECHPROEDUCATION";  
  
    System.out.println(isim.toLowerCase(Locale.forLanguageTag("tr")));  
}
```

Output :

```
techproeducation
```



7 TEMMUZ 2021
DERS 10

String Manipulations

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Önceki Dersten Aklımızda Kalanlar

- 1) switch case : Eğer if else ile kontrol edeceğimiz durumlar fazla ise, if else yerine switch-case kullanırız
- 2) Switch'de parantez içine boolean, long, float ve double değerleri olmaz, bunun dışındaki String, int, char vb.. Değerler yazılabilir
- 3) Switch-case karşılaştırmaları case'lerde yapar, switch'de girilen değer, case'de girilen değere eşitse o case çalışır
- 4) Herhangi bir case çalışmaya başladığında break görene kadar çalışmaya devam eder, break görünce switch-case'in sonuna gider .
- 5) Case'lerde tanımlı olmayan tüm değerler için çalışmasını istediğimiz kod'lari default : bölümüne yazarız (ifelse statements daki en son else bölümü gibidir)
- 6) Birden fazla case için aynı kodlar çalışacaksa, bu case'ler alt alta yazılır, sonrasında çalışması istenen ortak kod yazılıp, ardından break; konulur

String Manipulation / Methods

5-equals

Verilen iki String'in iceriginin birbirine esit olup olmadigini kontrol eder.

Eger verilen Stringlerdeki tum karakterler (bosluk, buyuk harf, kucuk harf, ozel karakter ..) tamamen ayni ise **TRUE** doner, aksi durumda (bir karakter bile farkli olsa) **FALSE** doner.

```
public static void main(String[] args) {  
  
    String isim1= "Ali Can";  
    String isim2= "Ali Can";  
  
    System.out.println(isim1.equals(isim2));  
}
```

Output :

true

String Manipulation / Methods

equals Vs ==

(Interview Sorusu)

equals() methodu verilen iki String'in iceriginin birbirine esit olup olmadigini kontrol eder.

== karsilastirma operatoru ise verilen iki String objesinin degerinin yaninda reference(adres)'lerine da bakar,

Ayni degere sahip olsa da farkli iki objeyi == ile karsilastirdigimizda sonuc **FALSE** olur.

```
public static void main(String[] args) {  
  
    String isim1= "Ali Can";  
    String isim2= isim1+"";  
  
    System.out.println(isim1==isim2);  
  
    System.out.println(isim1.equals(isim2));  
}
```

Output :
false
true

String Manipulation / Methods

6-equalsIgnoreCase

Verilen iki String degiskeni BUYUK HARF / kucuk harf farki gozetmeksizin karsilastirir.

Buyuk / kucuk harf farklilikligi disinda herhangi bir karakter farklilikligi oldugunda equals methodunda oldugu gibi FALSE dondurur.

```
public static void main(String[] args) {  
  
    String isim1= "Ali Can";  
    String isim2= "ali can";  
  
    System.out.println(isim1.equalsIgnoreCase(isim2));  
}
```

Output :

true

String Manipulation / Methods

7-length()

Verilen String'deki karakter sayisini dondurur.

```
public static void main(String[] args) {  
    String isim= "Ali Can";  
    System.out.println(isim.length());  
}
```

Output :

7

```
public static void main(String[] args) {  
    String isim= "";  
    System.out.println(isim.length());  
}
```

Output :

0

```
public static void main(String[] args) {  
    String isim= null;  
    System.out.println(isim.length());  
}
```

Exception in thread "main" [java.lang.NullPointerException](#)
at _00_anlik.asd.main([asd.java:11](#))

String Manipulation / Methods

8-indexOf()

Verilen String'de istenen karakterin kullanildiği ilk index'i döndürür.

- 1) char'in index'i sorgulanabilir
- 2) Parametre String olabilir
- 3) Olmayan karakter sorgulanırsa
- 4) Parametre kelime olabilir
- 5) Belli bir index'ten sonrası sorgulanabilir

```
String str= "Calisirsaniz, Java ogrenmek cok kolay";  
System.out.println(str.indexOf('a'));  
System.out.println(str.indexOf("a"));  
System.out.println(str.indexOf("t"));  
System.out.println(str.indexOf("Java"));  
System.out.println(str.indexOf('a',11));
```

Output : 1
 : 1
 : -1
 : 14
 : 15

String Manipulation / Methods

indexOf() Sorular

Soru 1) Kullanıcıdan bir cumle ve bir harf isteyin, harfin cumlede var olup olmadigini yazdirin

Soru 2) Kullanıcıdan bir cumle ve bir kelime isteyin, kelimenin cumledeki kullanimina bakarak asagidaki 3 cumleden uygun olani yazdirin

- Girilen kelime cumlede kullanilmamis.
- Girilen kelime cumlede 1 kere kullanilmis.
- Girilen kelime cumlede 1'den fazla kullanilmis.

String Manipulation / Methods

9-lastIndexOf()

Verilen String'de istenen karakterin kullanıldığı son index'i döndürür.

- 1) char'in son index'i sorgulanabilir
- 2) Parametre String olabilir
- 3) Olmayan karakter sorgulanırsa
- 4) Parametre kelime olabilir
- 5) Belli bir index'ten öncesi sorgulanabilir

```
String str= "Calisirsaniz, Java ogrenmek cok kolay";  
System.out.println(str.lastIndexOf('a'));           : 35  
System.out.println(str.lastIndexOf("a"));           : 35  
System.out.println(str.lastIndexOf("t"));           : -1  
System.out.println(str.lastIndexOf("Java"));        : 14  
System.out.println(str.lastIndexOf('a',11));        : 8
```

String Manipulation / Methods

lastIndexOf() Sorular

Soru 1) Kullanıcıdan bir cumle ve bir harf isteyin, harfin cumlede var olup olmadigini yazdirin

Soru 2) Kullanıcıdan bir cumle ve bir kelime isteyin, kelimenin cumledeki kullanimina bakarak asagidaki 3 cumleden uygun olani yazdirin

- Girilen kelime cumlede kullanilmamis.
- Girilen kelime cumlede 1 kere kullanilmis.
- Girilen kelime cumlede 1'den fazla kullanilmis.



8 TEMMUZ 2021
DERS 11

String Manipulations

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Önceki Dersten Aklımızda Kalanlar

- 1) equals() verilen iki String'in içeriklerini case sensitive olarak karşılaştırır, içerikler aynı ise TRUE, aynı değilse FALSE döndürür
- 2) Equals() ile == 'in farkı : equals() sadece içeriklere bakar, == ise hem içeriklere hem de referanslara(adrese) bakar. Diğer bir deyişle == verilen iki String'in aynı String olduğunu kontrol eder.
- 3) equalsIgnoreCase() : verilen iki String'in içeriğini case sensitive olmayarak karşılaştırır. (Money, MONEY, money, MoNeY bunların hepsi için true döndürür)
- 4) length() : verilen String'deki karakter sayısını verir. Son karakterin index'ini bulmak için length()-1 kullanılır.
- 5) "" String'inin length'i 0 dir... Ancak **String str = null;** dendiğinde , null str'in değeri değildir. null bir keyword'dur, null pointer olarak bilinir ve str'in değerinin olmadığını ifade eder. Yani bize str'in bir değeri olmadığını işaret eder.

Dolayısıyla str.length(); dedikimizde Java RTE verir.

- 6) indexOf(String veya char); parametre olarak girilen char veya String'in kullanıldığı ilk index'i verir.
- 7) lastIndexOf(String veya char); parametre olarak girilen String veya char'ı sondan başa doğru arar ve ilk bulduğu kullanımin index'ini bize döndürür.

String Manipulation / Methods

10-contains()

Verilen String'in istenen karakter(ler)i icerip icermedigini kontrol eder. Iceriyorsa TRUE, icermiyorsa FALSE dondurur.

- 1) Parametre String olmalidir
- 2) Olmayan karakter sorgulanirsa
- 3) Parametre kelime olabilir

```
public static void main(String[] args) {  
    String str= "Calisirsaniz, Java ogrenmek cok kolay";  
    System.out.println(str.contains("a"));  
    System.out.println(str.contains("t"));  
    System.out.println(str.contains("Java"));  
}
```

System.out.println(str.contains("a"));	true
System.out.println(str.contains("t"));	false
System.out.println(str.contains("Java"));	true

NOT contains() methodu char icin kullanilamaz, String kullanmak zorunludur.

String Manipulation / Methods

contains() sorular

Soru 1) Kullanıcıdan email adresini girmesini isteyin, mail @gmail.com iceriyorsa “Email adresiniz kaydedildi”, icermiyorsa “Lutfen gmail mail adresinizi giriniz.. “ yazdirin

Soru 2) Kullanıcıdan bir cumle isteyin. Cumle “buyuk” kelimesi iceriyorsa tum cumleyi buyuk harf olarak, “kucuk” kelimesi iceriyorsa tum cumleyi kucuk harf olarak yazdirin, iki kelimeyi de icermiyorsa “Cumle kucuk yada buyuk kelimesi icermiyor” yazdirin.

String Manipulation / Methods

11-endsWith()

Verilen String'in istenen karakter(ler) ile bitip bitmediğini kontrol eder. İstenen karakter(ler) ile bitiyorsa TRUE, yoksa FALSE döndürür.

- 1) Parametre String olmalıdır
- 2) Yanlış karakter sorgulanırsa
- 3) Parametre kelime olabilir

```
String str= "Calisirsaniz, Java ogrenmek cok kolay";  
System.out.println(str.endsWith("y"));  
System.out.println(str.endsWith("t"));  
System.out.println(str.endsWith("olay"));
```

true

false

true

String Manipulation / Methods

12-startsWith()

Verilen String'in istenen karakter(ler) ile baslayip baslamadigini kontrol eder. Istenen karakter(ler) ile basliyorsa TRUE, yoksa FALSE dondurur.

- 1) Parametre String olmalidir
- 2) Parametre kelime olabilir
- 3) Belirli karakterden sonrasi olabilir

```
String str= "Calisirsaniz, Java ogrenmek cok kolay";  
System.out.println(str.startsWith("C"));           true  
System.out.println(str.startsWith("Calis"));       true  
System.out.println(str.startsWith("s",4));         true  
System.out.println(str.startsWith("Java",14));     true
```

String Manipulation / Methods

13-isEmpty()

Verilen String'in uzunluğu 0(sifir) ise (Hicbir karakter icermiyorsa) TRUE, yoksa FALSE dondurur.

```
String str= "Calisirsaniz, Java ogrenmek cok kolay";  
System.out.println(str.isEmpty());  
String str2="";  
System.out.println(str2.isEmpty());  
String str3=null;  
System.out.println(str3.isEmpty());
```

false

true

Hata verir

```
Exception in thread "main" java.lang.NullPointerException  
at _00_anlik.asd.main(asd.java:19)
```

String Manipulation / Methods

13- replace()

Verilen String'deki istenen karakter(ler)i istenen yeni karakter(ler) ile degistirir.

```
String str= "Java ogrenmek cok kolay";  
System.out.println(str.replace("a", "x"));  
System.out.println(str.replace("Java", "x"));  
System.out.println(str.replace("a", "xxx"));  
System.out.println(str.replace("a", ""));  
System.out.println(str.replace('a', 'x'));
```

```
Jxvx ogrenmek cok kolxy  
x ogrenmek cok kolay  
Jxxxvxxx ogrenmek cok kolxxxxy  
Jv ogrenmek cok koly  
Jxvx ogrenmek cok kolxy
```

NOT : replace() methodu char icin de kullanilabilir

String Manipulation / Methods

14- replaceAll()

replace() methodu ile benzer olarak verilen String'deki istenene karakter(ler)i istenen yeni karakter(ler) ile degistirir. Aralarindaki farklar

- replace() methodunda char kullanılabilir, replaceAll()'da char kullanilamaz
- replaceAll() methodunda Regular Expressions kullanılabilir

\\s : bosluk (space)

\\S : bosluk disindaki tum karakterler

\\w : harfler ve rakamlar (a-z , A-Z, 0-9)

\\W : harfler ve rakamlar disindaki tum karakterler

\\d : rakamlar (0-9)

\\D : rakamlar disindaki tum karakterler

String Manipulation / Methods

replaceAll()

```
public static void main(String[] args) {  
  
    String str= "Java'da rakamlar 1234567890";  
  
    System.out.println(str.replaceAll("a", "*"));  
  
    System.out.println(str.replaceAll("\\s", "*"));  
  
    System.out.println(str.replaceAll("\\S", "*"));  
  
    System.out.println(str.replaceAll("\\w", "*"));  
  
    System.out.println(str.replaceAll("\\W", "*"));  
  
    System.out.println(str.replaceAll("\\d", "*"));  
  
    System.out.println(str.replaceAll("\\D", "*"));  
}
```

J*v*'d* r*k*ml*r 123456789

Java'da*rakamlar*1234567890

Java*da*rakamlar*1234567890

Java'da rakamlar *****

*****1234567890

String Manipulation / Methods

15- replaceFirst()

Verilen String'deki istenen karakter(ler)in ilkini, istenen yeni karakter(ler) ile degistirir

```
public static void main(String[] args) {  
    String str= "Java'da rakamlar 1234567890";  
    System.out.println(str.replaceFirst("a", "*"));  
    System.out.println(str.replaceFirst("lar", "*"));  
    System.out.println(str.replaceFirst("\\s", "*"));  
    System.out.println(str.replaceFirst("\\D", "*"));  
}
```

J*va'da rakamlar 1234567890

Java'da rakam* 1234567890

Java'da*rakamlar 1234567890

*ava'da rakamlar 1234567890



9 TEMMUZ 2021
DERS 12

String Manipulations

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Onceki Dersten Aklimizda Kalanlar

- 1) Contains() parametre olarak yazdigimiz String bir baska String icinde var olup olmadiga bakar. Aranan String'in kac tane oldugu ile ilgilenmez, sadece var mi yok mu bakar. False veya true dondurur
- 2) EndsWith() verilen bir String'in, parametre olarak girilen String ile bitip bitmedigini kontrol eder. True veya False doner
- 3) startsWith() verilen bir String'in, parametre olarak girilen String ile baslayip baslamadigini kontrol eder. True veya False doner
- 4) isEmpty() parametre Kabul etmez str.isEmpty() seklinde kullanilir, ve verilen String'in bos olup olmadigini kontrol eder. True veya False doner . "" icin true doner, str=null; bunun icin hata verir.
- 5) **null bir deger degildir, ama atandigi objenin hic bir degerinin olmadigini isaret eder.**
- 6) replace(parametre1, parametre2) verilen String'deki parametre1'lerin yerine parametre2 yazar. Char icin de kullanilabilir. Parametre1 ve parametre2'nin uzunluklari esit olmak zorunda degildir. Her hangi bir karakter veya String'i ortadan kaldirmak istersek parametre1 olarak yok etmek istedigimiz String'i , parametre2 yerine ise "" yazariz.
- 7) replaceAll(parametre1, parametre2) replace ile ayni isi yapar ama char Kabul etmez, gelismis ozellik olarak regular Expressions regexp kullanilabilir
- 8) replaceFirst(parametre1, parametre2) sarta uyan ilk parametre1 yerine parametre2 yazar

String Manipulation / Methods

16- substring()

Index kullanarak verilen String'in istenen parçasını almamızı sağlar.

- Parametre olarak 1 sayı girilirse, girilen index'den String'in sonuna kadar bölümü
- Parametre olarak 2 sayı girilirse, girilen 1.sayıdaki indexden (inclusive) başlayıp, 2.sayıya kadar (exclusive) karakteri bize döndürür

```
public static void main(String[] args) {  
    String str= "Java OOP konsepti kullanır";  
    System.out.println(str.substring(0));  
    System.out.println(str.substring(10));  
    System.out.println(str.substring(26));  
    System.out.println(str.substring(29));  
}
```

Java OOP konsepti kullanır
onsepti kullanır
Hata verir

```
Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String index out of range: -3  
    at java.lang.String.substring(Unknown Source)  
    at _00_anlik.asd.main(asd.java:17)
```

String Manipulation / Methods

substring()

```
public static void main(String[] args) {  
    String str= "Java OOP konsepti kullanir";  
    System.out.println(str.substring(5,11));  
    System.out.println(str.substring(3,4));  
    System.out.println(str.substring(8,8));  
    System.out.println(str.substring(8,2));  
}
```

OOP ko

a

Hata verir

```
Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String index out of range: -6  
    at java.lang.String.substring(Unknown Source)  
    at _00_anlik.asd.main(asd.java:17)
```

Not : Java'da iki tur hata mesajı aliriz

1- Compile Time Error (CTE) : Kodumuzu yazarken kod altinin kirmizi cizgi olmasi

2- Run Time Error (RTE) : Kod calistirildiginda (Execute) karsilastigimiz hatalar

String Manipulation / Methods

17- trim()

Istedigimiz String'in basinda veya sonunda var olan bosluk / "space" leri temizler

```
String str = "  Java ogrenmek cok guzel.  ";  
  
System.out.println(str);  
  
System.out.println(str.length());  
  
System.out.println(str.trim());  
  
System.out.println(str.trim().length());
```

| Java ogrenmek cok guzel. |

28

|Java ogrenmek cok guzel.|

24

String Manipulation / Methods

Soru 1) String methodlarini kullanarak " Java ogrenmek123 Cok guzel@ " String'ini "Java ogrenmek cok guzel." sekline getirin.

Soru 2) String seklinde verilen asagidaki fiyatların toplamını bulunuz

String str1 = "\$13.99"

String str2 = "\$10.55"

ipucu : Double.parseDouble() methodunu kullanabilirsiniz.

Soru 3) Kullanıcıdan isim isteyin. Eğer

- isim "a" harfi içeriyorsa "Girdiğiniz isim a harfi içeriyor"
- isim "Z" harfi içeriyorsa "Girdiğiniz isim Z harfi içeriyor"
- ikisi de yoksa "Girdiğiniz isim a veya Z harfi icermiyor" yazdırın

Soru 4) Kullanıcıdan isim ve soyismini isteyin ve hangisinin daha uzun olduğunu yazdırın.

Soru 5) Kullanıcıdan 4 harfli bir kelime isteyin ve girilen kelimeyi tersten yazdırın.

String Manipulation / Methods

Soru 6) Kullanıcıdan bir sifre girmesini isteyin. Asagidaki sartlari sagliyorsa “Sifre basari ile tanimlandi”, sartlari saglamazsa “Islem basarisiz,Lutfen yeni bir sifre girin” yazdirin

- Ilk harf buyuk harf olmalı
- Son harf kucuk harf olmalı
- Sifre bosluk icermemeli
- Sifre uzunlugu en az 8 karakter olmalı

Soru 7) Kullanıcıdan ismini, soyismini ve kredi karti bilgisini isteyin ve asagidaki gibi yazdirin

isim-soyisim : M***** B*****

kart no : **** * 1234



10 TEMMUZ 2021
DERS 13

Method Creation
Method Call

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Önceki Dersten Aklımızda Kalanlar

- 1) `substring()` verilen bir `String`'in istedigimiz bolumlerini almamizi saglar.
- 2) İki farkli kullanimi vardır. `Substring(tekParametre)` : parametre olarak girilen indexdeki karakterden sona kodar tum karakterleri dondurur.
- 3) `substring(parametre1, parametre2)` : parametre1 index'İ dahil (inclusive), parametre2 index'i haric (exclusive) aradaki karakterleri verir.
- 4) `Str.substring(str.length()) ==>` sondaki hiclik'i verir . `Str= "Mehmet" str.substring(6)==>""`
- 5) Son harfi gormek istersek `str.substring(str.length()-1)`
- 6) Eger parametre olarak `String`'in uzunlugundan buyuk bir sayi girilirse hata verir
- 7) `Str.trim()` `String`'in aralarindaki bosluklara dokunmada, oncesinde ve sonrasinda olan bosluklari keser.

Method Creation Method Olusturma

Method Olusutururken Kullanilan Keyword'ler Nelerdir?

```
public int myFirstMethod () {}  
1      2      3      4  5
```

- 1 **public** : Access Modifier (Erisim duzenleyici):methoda'a kimlerin erisebilecegini belirler
private: Sadece bulunduгу class'da kullanilabilir
protected : Sedece icinde bulunduгу class ve child class'lardan kullanilir
- 2 **int** : Return Type, methodun ne urettigini ve bize dondurdugunu belirtir
- 3 **myFirstMethod** :Olusturdugumuz method'un ismidir. Isim mutlaka kucuk harfle baslar, birden fazla kelimeden olusursa sonraki kelimelerin ilk harfleri buyuk harf yazilir (Camel Case)
- 4 **()** **parantez**: Methodlarda isimden sonra parantez kullanilir ve gerektiginde parantez icinde parametre yazilir.
- 5 **Body (Method Body)** : { } arasinda kalan kodlarimizi yazdigimiz bolumdur



12 TEMMUZ 2021
DERS 14

Method Creation
For Loop

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Method Creation

Method Olusturma

```
public int myFirstMethod () {}
```

1 2 3 4 5

1 Access Modifier (Erisim duzenleyici):

public : methoda'a kimlerin erisebilecegini belirler

protected : Sadece icinde bulunduгу package ve child class'lardan kullanilir

default : Sadece icinde bulunduгу paket(package)'den kullanilir

private: Sadece bulunduгу class'da kullanilabilir

Access Levels				
Modifier	Class	Package	Subclass	World
public	Y	Y	Y	Y
protected	Y	Y	Y	N
no modifier	Y	Y	N	N
private	Y	N	N	N

Method Creation

Method Olusturma

2 static (Ileride detayli anlatilacak)

Bir method olusturulurken **static** kelimesinin kullanilmasi mecburi degildir.

Main method'umuz static oldugu icin main method'dan cagiracagimiz tum method'lari static yapmamiz gereklidir

```
public static void main(String[] args) {  
  
}
```

Method Creation Method Olusturma

- 3 **int (Return Type)** : methodun ne urettigini ve bize ne dondurdugunu belirtir.
- Return Type, primitive veya non-primitive tum data turlerinden olabilir
 - Eger method bir sey dondurmeyecekse (ornegin, sadece bir sey hesaplayip yazdiracaksa) return type olarak **void** secilir
 - Return Type olarak void disinda bir sey yazdiysak, methodun sonunda mutlaka **return** keyword kullanilmalidir
 - Return keyword'den sonra return type'a uygun bir **deger veya variable** yazilmalidir.
 - Return type'a sahip methodlar cagrildiklari satira, return keyword'den sonra yazilan deger veya variable'i dondururler.

```
public static void main(String[] args) {  
    int sonuc = topla(15,24);  
}  
  
public static int topla(int num1, int num2) {  
    return num1 + num2;  
}
```

Method Creation Method Olusturma

4 myFirstMethod :Olusturdugumuz method'un ismidir. Isim mutlaka kucuk harfle baslar, birden fazla kelimededen olusursa sonraki kelimelerin ilk harfleri buyuk harf yazilir (Camel Case)

5 () parantez : Methodlarda isimden sonra parantez kullanilir ve gerektiğinde parantez icinde parametre yazilir.

******* Eger bir Class'da ayni isme sahip birden fazla method olusturmamiz gerekirse parametreleri farkli yapmamiz gereklidir (Overloading)

Method Creation

Method Olusturma

6 Body (Method Body) : { } arasında kalan kodlarımızı yazdığımız bölümdür

*** Method nerede oluşturulmalıdır ?

Method Class body'si içinde Main method dışında oluşturulmalıdır

```
public class asd {  
    public static void main(String[] args) {  
        toplama(5,4);  
    }  
    private static void toplama(int i, int j) {  
        System.out.println(i+j);  
    }  
}
```


Method Call

Method Cagirma

Method olusturmak method'u calistirmek icin yeterli degildir.

Ihtiyac duyuldugunda daha onceden olusturulmus methodu calistirmek icin Method ismi (parametreler ile birlikte) yazilmalidir.

Bu isleme method cagirma denir

```
public class asd {  
    public static void main(String[] args) {  
        toplama(5,4);  
    }  
    private static void toplama(int i, int j) {  
        System.out.println(i+j);  
    }  
}
```

*** Method cagirirken parantez icine yazilan degerlere **Arguments (arguman)** denir.

*** Method cagirirken kullandigimiz argumanlar ile method parametrelerinin uyumlu olmasi gereklidir.

*** Sayi parametreleri icin char degerler de arguman olarak kullanilabilir

Method Creation

- Soru 1)** Kullanıcıdan bir sayı alın. Bu sayının tek mi çift mi olduğunu, sıfırdan büyük mü küçük mü olduğunu, ayrıca ve 100'den büyükse birler, onlar ve yüzler basamağındaki rakamların toplamını, 100'den küçükse sadece 1'ler basamağını yazdıran 3 method oluşturun.
- Soru 2)** Kullanıcıya kaç sayı toplamak istediğini sorun. Kullanıcı 2,3 veya 4 değerini girerse, kullanıcıdan bu sayıları girmesini isteyin ve sayıların toplamını yazdırın. Kullanıcı toplamak istediği sayı adedini 4'den büyük girerse "Çok sayı girdiniz, ben toplayamam" yazdırın.
- Soru 3)** Email kontrolü yapan bir program yazın. Kullanıcının girdiği şifre
- @ isareti içermiyorsa geçersiz email yazdırın
 - @gmail.com içermiyorsa "lutfen gmail adresinizi girin" yazdırın
 - @gmail.com ile bitmiyorsa "Yazımda bir sorun var, maili kontrol ediniz"
- Soru 4)** Kullanıcıdan ismini, soyisini ve boşluk bırakmadan 16 hane olarak kredi kartı numarasını alın. İsim ve soyisi ilk harfleri büyük diğer harfler küçük olacak şekilde, KK numarasını ise 4 rakamlık 4 blok ve aralarında boşluk olacak şekilde düzelden 2 method yazın, ve programda kullanabilmek için düzenlenmiş hallerini geri döndürün.

Method Overloading

Interview Sorusu

1) **Overloading nedir ?** Eger bir Class'da ismi ayni fakat parametreleri farkli olan methodlar olusturursak buna **Overloading** denir.

2) **Overloading nasil yapilir ?** Java ayni isim ve ayni parametrelerle birden fazla method olusturulmasina izin vermez. Ayni isimle birden fazla method olusturmak isterseniz **method signature (metot imzasi)**'nin degistirilmesi gerekir

3) **method signature (metot imzasi) nasil degistirilir?**

Method signature'i degistirmek icin 3 yontem kullanilabilir

- parametrelerin data tipleri degistirilebilir
- parametrelerin sayisi degistirilebilir
- parametre sayisi ayni olmak zorunda ise farkli data tipindeki parametrelerin sirasi degistirilir

*** method'un return type'ini degistirmek, access modifier'ini degistirmek veya static kelimesi eklemek method signature'i degistirmez



13 TEMMUZ 2021
DERS 15

For Loop

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Onceki Dersten Aklimizda Kalanlar

Method olustururken karar vermemiz gereken 3 adim

- 1- Method'a ne gonderecegiz (arguments)
- 2- Method'da ne yapacagiz , return olacak mi ?
- 3- Method bize ne dondurecek ve biz donen degeri ne yapacagiz ?

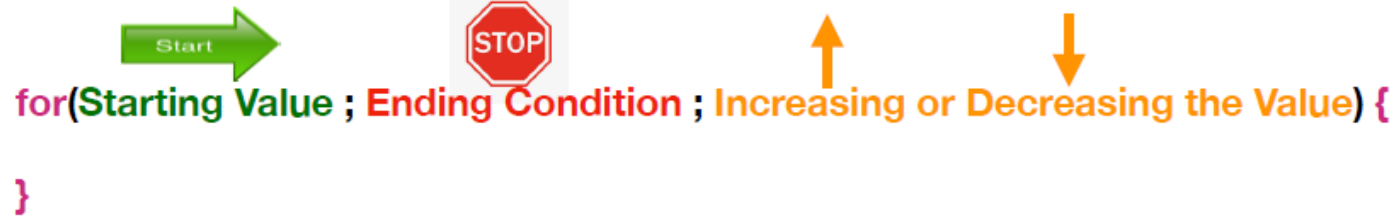
Soru : Bir oyun programinda oyuncuya level ve puanina gore bonus hesaplayan bir method yaziniz. Bonus eklendikten sonra oyuncu yeni puaniyla oyuna devam edecek. (Oyun icinde birden fazla defa bonus ekleme fonksiyonu kullanilabilir)

Bonus kurali :

- ilk 10 level icin mevcut puan 1000'den az ise 50, 1000 veya cok ise 100 bonus
- 11-50 level arasi mevcut puanin yuzde 10'u kadar bonus
- 51.levelden itibaren puan 10.000'den az ise 1000 bonus, 10.000 den coksa mevcut puanin %15 kadar bonus

For Loop

Belirli bir koşul sağlandığı sürece tekrarlanması gereken işler için kullanılan kod bloklarına LOOP(Dongu) denir. Tekrar sayısı belirli olan durumlarda for loop kullanılması tercih edilir.



The diagram shows the general syntax of a for loop: `for(Starting Value ; Ending Condition ; Increasing or Decreasing the Value) {`
Annotations include:

- A green arrow pointing right labeled "Start" above "Starting Value".
- A red octagonal "STOP" sign above "Ending Condition".
- An orange arrow pointing up above "Increasing".
- An orange arrow pointing down above "Decreasing".

The closing brace `}` is on a new line.

```
for ( int i=4; i>1; i- - ) {  
    System.out.println( i );  
}
```

For Loop



- Eger **Ending Condition** hep **true** verirse loop sonsuz donguye girer
- Eger Loop'ta **Ending Condition** hic true olmazsa loop body hic devreye girmez
- loop'da artis degeri pozitif oldugu gibi negatif de olabilir (i-- vb)
- Artis degeri 1 olmak zorunda degil, farkli da olabilir (i+=2 vb..)

For Loop

Soru 1) Ekrana 10 kez “Java guzeldir” yazdirin

Soru 2) 10 ile 30 arasindaki sayilari aralarinda virgule olarak ayni satirda yazdirin

Soru 3) 100'den baslayarak 50'ye kadar olan sayilari aralarinda virgule olarak ayni satirda yazdirin

Soru 4) Kullanicidan 100'den kucuk bir tamsayi isteyin. 1'den baslayarak girilen sayiya kadar 3'un kati olan sayilari yazdirin.

Soru 5) Kullanicidan 100'den kucuk bir tamsayi isteyin. 1'den baslayarak girilen sayiya kadar 3'un veya 5'in kati olan sayilari yazdirin.

Soru 6) Interview Question Kullanicidan 100'den kucuk bir tamsayi isteyin. 1'den baslayarak girilen sayiya kadar tum sayilari yazdirin. Ancak;

- Sayi 3'un kati ise sayi yerine “Java” yazdirin.
- Sayi 5'in kati ise sayi yerine “Guzeldir” yazdirin.
- Sayi hem 3'un hem 5'in kati ise sayi yerine “Java Guzeldir” yazdirin.

For Loop

Soru 7) Interview Question Kullanıcıdan bir String isteyin ve Stringi tersten yazdırın.

Soru 8) Interview Question Kullanıcıdan bir String isteyin ve Stringi tersine çeviren bir program yazın.

Soru 9) Interview Question Kullanıcıdan bir String isteyin. Kullanıcının girdiği String'in palindrome olup olmadığını kontrol eden bir program yazın.

Soru 10) Kullanıcıdan iki sayı isteyin. Girilen sayılar ve aralarındaki tüm tamsayıları toplayip, sonucu yazdıran bir program yazınız

Soru 11) Interview Question Kullanıcıdan 10'dan küçük bir tamsayı isteyin ve girilen sayının faktöriyelini bulun. ($5! = 5 * 4 * 3 * 2 * 1$)



14 TEMMUZ 2021
DERS 15

Nested For Loop
While Loop

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Önceki Dersten Aklımızda Kalanlar

- 1- For loop : başlangıç ve bitisi belli olan tekrarlı kod çalışmalarını yapmak için loop kullanırız.
- 2- Loop yazarken karar vermemiz gereken 3 şey
 - i) Başlangıç değeri
 - ii) bitirmek için koşulumuz nedir
 - iii) her adımda değişkeni nasıl değiştireceğiz
- 3- Eğer başlangıç değerini artırırsak veya azaltarak değiştirdiğimizde bitiş koşuluna yaklaşmıyorsa sonsuz loop oluşur,
- 4- Eğer başlangıç değeri bitiş koşulunu sağlamazsa, for loop çalışır ama loop body'si hiç devreye girmez, dolayısıyla loop anlamsız olur...
- 5- loopdaki değişken artılabildiği gibi, soruya göre azalabilir

Nested For Loop

Bazen tek bir loop ile istedigimiz sonuclara ulasamayiz.

Ozellikle iki boyutlu sekiller cizdirmek veya carpim tablosu gibi sayi ikilileri olusturmak icin nested loop kullanmamiz gerekir.

```
*           1  2  3  4
* *        2  4  6  8
* * *      3  6  9 12
* * * *
```

```
for (int i = 1; i <= 4; i++) {
    for (int j = 1; j <= 4; j++) {
        System.out.print("(" + i + "," + j + ") ");
    }
    System.out.println("");
}
```

(1,1)	(1,2)	(1,3)	(1,4)
(2,1)	(2,2)	(2,3)	(2,4)
(3,1)	(3,2)	(3,3)	(3,4)
(4,1)	(4,2)	(4,3)	(4,4)



15 TEMMUZ 2021
DERS 16

Nested For Loop
While Loop

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Nested For Loop

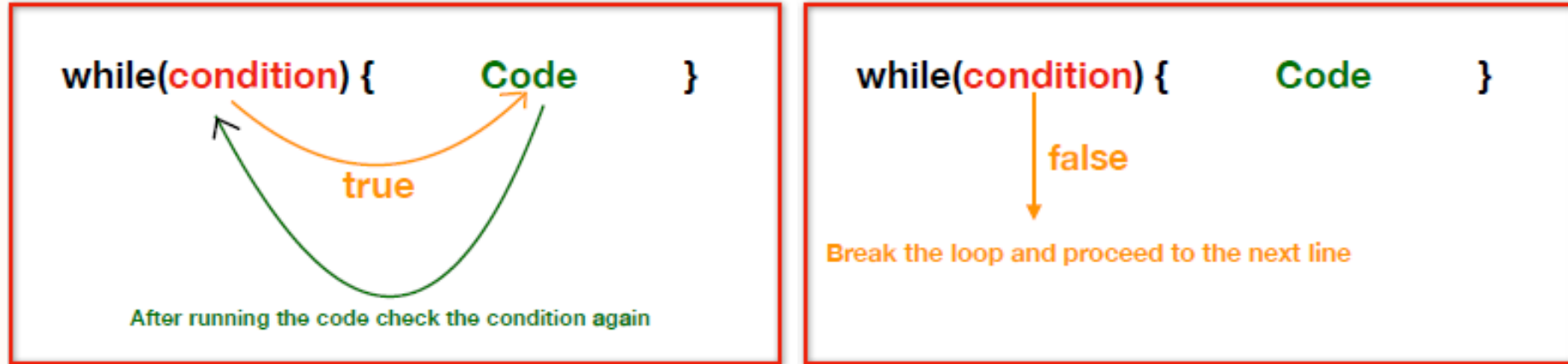
Soru 12) Kullanıcıdan pozitif bir rakam girmesini isteyin ve girilen rakama göre aşağıdaki şekli çizdirin

```
*  
* *  
* * *  
* * * *
```

Soru 13) Kullanıcıdan pozitif bir rakam girmesini isteyin ve girilen rakama göre carpim tablosu olusturun. Ornek,kullanici 3 girerse,

```
1 2 3  
2 4 6  
3 6 9
```

While Loop



```
int i = 0;  
while (i < 5) {  
    System.out.println(i);  
    i++;  
}
```

While Loop

- Soru 1)** While loop kullanarak 3 den 13 e kadar tum tek tamsayilari ekrana yazdiriniz.
- Soru 2)** For loop ve while Loop kullanarak 3 basamakli sayilardan 15, 20 ve 90'na tam bolunebilen sayilari yazdirin.
- Soru 3)** Kullanicidan baslangic ve bitis degerlerini alin. Baslangic degeri ve bitis degeri dahil aradalarindaki tum cift tamsayilari while loop kullanarak ekrana yazdiriniz.
- Soru 4)** Kullanicidan baslangic ve bitis haflerini alin ve baslangic harfinden baslayip bitis harfinde biten tum harfleri buyuk harf olarak ekrana yazdirin. Kullanicinin hata yapmadigini farz edin.

While Loop

Soru 5) Kullanıcıdan bir rakam alın ve bu rakam için carpim tablosunu ekrana yazdırın. Kullanıcının hata yapmadığını farz edin.

Örneğin kullanıcı 3 girerse;

$3 \times 1 = 3$ $3 \times 2 = 6$ $3 \times 3 = 9$ $3 \times 4 = 12$ $3 \times 5 = 15$ $3 \times 6 = 18$ $3 \times 7 = 21$ $3 \times 8 = 24$ $3 \times 9 = 27$ $3 \times 10 = 30$

Soru 6) Kullanıcıdan bir sayı alın ve bu sayıyı tam bölen sayıları ve toplam kaç tane olduklarını ekranda yazdırın

Soru 7) Kullanıcıdan bir sayı alın ve bu sayının rakamları toplamını yazdırın



16 TEMMUZ 2021
DERS 18

DoWhile Loop
Scope

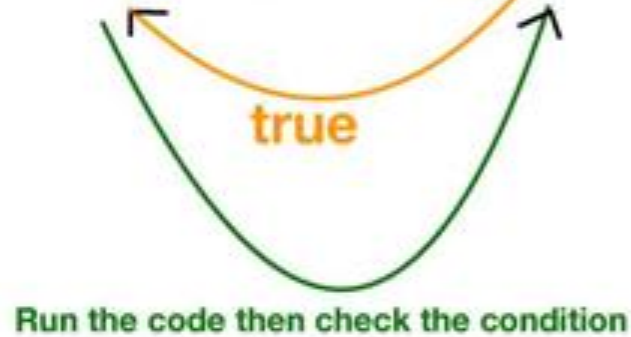
Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Onceki Dersten Aklimizda Kalanlar

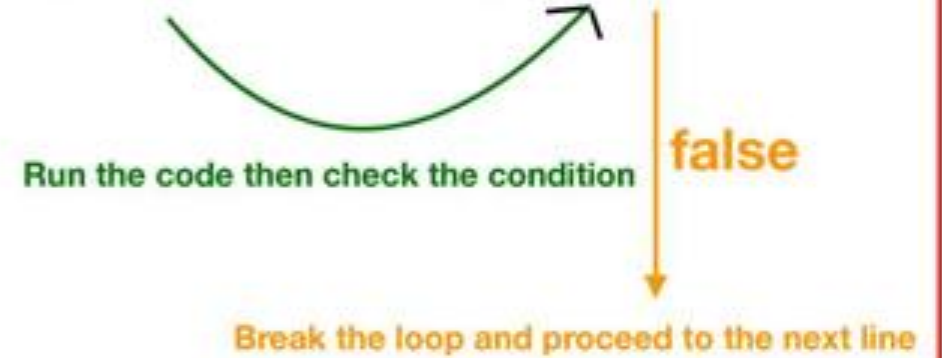
- 1- Nested For Loop : iki boyutlu sekiller veya iki degiskenli denklemler kullanmamiz gerektiğinde nested for loop'lara ihtiyacimiz olur
- 2- Outer loop'un her bir dongusu icin inner loop bastan sona calisir. Inner bittikten sonra bir alt satira gecip orada islem varsa devam eder..
- 3- While loop , sart olarak yazdigimiz boolean ifade true oldugu muddetce while loop body'si tekrar tekrar calisir. Sart false oldugunda loop sona erer ve sonraki satirdan kod calismaya devam eder
- 4- Daha once for loop da kullandigimiz baslangic degeri ve artis sekli while loop'un syntax'inde yer almaz ama loop'un calismasi icin gereklidir. Bunun icin biz her while loop icin loop baslamadan once baslangic degeri olusturacak variable'ı olusturmali ve loop'un icinde de degisim seklini ve miktarini yazmaliyiz.
- 5- For loop Vs While Loop :
adim sayisi veya baslangic ve bitis degerleri belli ise for loop daha Pratik olabilir.
Ancak adim sayisi ongorulemiyorsa, loop'un bitisi bir adima degil de sarta bagliysa bu durumda while loop daha avantajli olmaktadır.
Ornegin kullanicidan degerler aliyorsak ve kullaniciya istedigi kadar deger grime hakki veriyorsak for loop kullanmamiz mumkun degildir.

Do While Loop

do { **Code** } while(**condition**)



do { **Code** } while(**condition**)



```
public static void main(String[] args) {  
  
    int i = 0;  
  
    do {  
        System.out.println(i);  
        i++;  
    }  
    while (i<5);  
}
```

Do While Loop Vs While Loop

```
public static void main(String[] args) {  
  
    int i = 10;  
  
    do {  
        System.out.println(i);  
        i++;  
    }  
    while (i<5);  
}
```

```
public static void main(String[] args) {  
  
    int i = 10;  
  
    while (i<5){  
        System.out.println(i);  
        i++;  
    }  
}
```

Fark : While Loop, dongunun başlangıcında kosulu kontrol eder ve kosul saglanirsa body icindeki kodlari calistirir.
Do-while loop'ta ise , kosul body icerisindeki kodlar 1 kere calistiktan sonra kontrol edilir.

Sonuc : Bir while loop'daki kosul yanlissa, loop hic calismaz 'do-wile' loop'ta ise , kosul yanlissa kodlar 1 kere calisir

Do While Loop

Soru 1) 9 den 190 e kadar 7 nin kati olan tum tamsayilari ekrana yazdiriniz.

Soru 2) 'm' harfinden baslayarak 'c' harfine kadar tum harfleri yazdirin.

Soru 3) Kullanicidan toplamak uzere pozitif sayilar isteyin, islemi bitirmek icin 0'a basmasini soyleyin.

Kullanici 0'a bastiginda toplam kac pozitif sayi girdigini ve girdigi pozitif sayilarin toplamının kac oldugunu yazdirin.

Soru 4) Kullanicidan toplamak uzere pozitif sayilar isteyin, islemi bitirmek icin 0'a basmasini soyleyin.

Kullanici yanlislikla negative sayi girerse o sayiyi dikkate almayin ve "Negatif sayi giremezsiniz" yazdirip basa donun

Kullanici 0'a bastiginda toplam kac pozitif sayi girdigini, yanlislikla kac negative sayi girdigini ve girdigi pozitif sayilarin toplamının kac oldugunu yazdirin.

Do While Loop

Soru 5) Kullanıcıdan bir sifre girmesini isteyin. Girilen sifreyi asagidaki sartlara gore kontrol edin ve sifredeki hatalari yazdirin.

Kullanici gecerli bir sifre girinceye kadar bu islemi tekrar edin ve gecerli sifre girdiginde “Sifreniz Kabul edilmistir” yazdirin.

- Sifre kucuk harf icermelidir
- Sifre buyuk harf icermelidir
- Sifre ozel karakter icermelidir
- Sifre en az 8 karakter olmalidir.

Soru 6) Kullanıcıdan toplamak icin sayi isteyin ve toplam 500'e ulasincaya kadar devam istemeyi ettirin. Toplam 500'e ulastiginda veya gectiginde toplami ve kac sayi girildigini yazdirin

Scope

Instance, Class ve Local Variables





Object Nasıl Kullanilir ?



Ogretmen



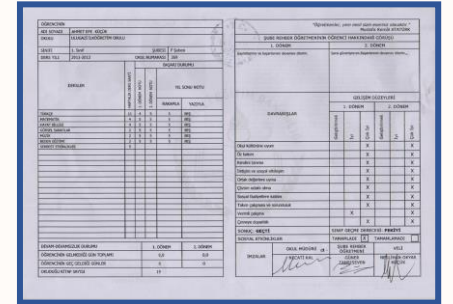
Ogrenci

Dersler

09:00	TÜRKÇE-1
09:30	MATEMATİK-1
10:00	TÜRKÇE-2
10:30	MATEMATİK-2
11:00	TÜRKÇE-3
11:30	MATEMATİK-3
12:00	TÜRKÇE-4
12:30	MATEMATİK-4
13:00	İYEP TÜRKÇE



Personel



Notlar

Scope

Instance, Class ve Local Variables

Scope (Kapsam)

- Bir Class içerisinde olusturulan variable'lar icin Scope, o variable'a nereden, nasil ulasilabilecegini ve nerede gecerli oldugunu ifade eder.
- Scope'a uymayan bir kullanimda Java Compile Time Error verir.
- Java'da olusturulan variable'lar icin 4 Scope mevcuttur
 - 1) Instance (Object) Variables // ogretmenin adi gibi, ogrencinin notu gibi
 - 2) Static (Class) Variables // okul adi, adresi gibi
 - 3) Local (Method) Variables
 - 4) Loop Variables

Scope

Instance, Class ve Local Variables

Instance (Object) Variable

Class'ın icinde ancak main method'un disinda olmalidir

Static olmamalidir

Olusturulmasi yeterlidir, deger atanmasi sart degildir.

```
public class Example {  
    int sayi;  
    public static void main(String[] args) {  
    }  
}
```

Default Value

Eger instance bir variable olusturur ama deger atamazsaniz, Java otomatik olarak default degerleri assign eder. (String icin null, sayisal data turleri 0, boolean false)

Scope

Instance, Class ve Local Variables

Instance (Object) Variable

class içerisinde veya başka class'larda direk kullanılamaz, kullanmak istediğimizde MUTLAKA object oluşturmali ve object üzerinden ulasilmalıdır.

```
public class Example {  
  
    int sayi;  
    char ilkHarf;  
    String isim;  
    boolean ogrenciMi;  
  
    public static void main(String[] args) {  
  
        Example ex1=new Example();  
        System.out.println(ex1.sayi);  
        System.out.println(ex1.ilkHarf);  
        System.out.println(ex1.isim);  
        System.out.println(ex1.ogrenciMi);  
    }  
}
```

Outputs

0
null
false

Ornek :

Bir okul uygulaması yaptığımızı düşündüğümüzde, öğretmenİsmi, öğrenciİsmi, matematiNotu gibi değişkenler bir kişi ile ilişkilendirilmedikçe anlamlı olmaz

Scope

Instance, Class ve Local Variables

Class (static) Variable

Class'ın icinde ancak main method'un disinda olmalidir.

Static olmalidir

Olusturulmasi yeterlidir, deger atanmasi sart degildir.

```
public class Example {  
    static int sayi;  
    public static void main(String[] args) {  
    }  
}
```

Scope

Instance, Class ve Local Variables

Class (static) Variable, class içerisinde direk kullanılabilir, başka class'larda kullanmak istedigimizde object oluşturmaya ihtiyaç duymadan `classIsmi.variableIsmi` ile variable'a ulaşabilir ve kalıcı olarak degistirebiliriz.

```
public class Example {  
  
    static int okulId;  
    static String okulIsmi;  
    static boolean acikMi;  
  
    public static void main(String[] args) {  
  
        System.out.println(okulId);  
        System.out.println(okulIsmi);  
        System.out.println(acikMi);  
  
    }  
}
```

Outputs

0
null
false

Ornek : Bir okul uygulaması yaptığımızı düşünün `okulIsmi`, `okulId`, `acikMi` gibi değişkenler bir kişiyi değil okulla ilgili herkesi ilgilendirir ve bir kişi okul ismini veya okul telefon numarasını değiştirirse okulla ilgili herkes için okul ismi değişir.

Scope Instance Vs Class Variables

Instance (Object) Variable, class içerisinde veya başka class'larda direk kullanılamaz, kullanmak istedigimizde MUTLAKA object oluşturmali ve object üzerinden ulasilabilir.

Class (static) Variable, class içerisinde direk kullanılabilir, başka class'larda kullanmak istedigimizde object oluşturmaya ihtiyac duymadan classIsmi.variableIsmi ile variable'a ulasabilir ve kalici olarak degistirebiliriz.

Static variable'lar herkes için ortaktır (okul ismi gibi) , instance variable'lar ise objeye bağlıdır (matematikNotu, ogrencisi gibi)

Static variable yetkisi olan herkes tarafından degistirilebilir ve bu degisim her obje için geçerlidir. Instance variable da yetkisi olan herkes tarafından degistirilebilir ancak yapılan degisiklik sadece o obje ile ilgilidir, geneli kapsamaz.

Scope

Instance, Class ve Local Variables

Local Variable

- Herhangi bir method icerisinde olusturulan variable'lardir (main method dahil).
- Sadece o method icerisinde gecerlidir
- Baska methodlarda da kullanılacak variable'lari, local olusturmak yerine **class level**'da olusturmak gereklidir.
- Class level'da olusturulacak variable, main method'da kullanilacaksa static olarak olusturulmalidir. Bu durumda bu variable kullanacak, diger method'lar da static olmalidir.

```
public class Example {  
  
    public static void main(String[] args) {  
        int sayi;  
    }  
  
    public void add() {  
        String isim;  
    }  
}
```


Scope

Instance, Class ve Local Variables

Local Variable

- Java local variable'lara default deger atamaz.
- Sadece olusturdugunuzda Java sikayet etmez. (variable olusturuldu method icerisinde deger atanacak diye bekler.)
- Olusturulan local variable'lara deger atamadan kullanmaya calisirsaniz Java sikayet eder(CTE)

```
5 public class Example {  
6  
7  
8     public static void main(String[] args) {  
9  
10        int sayi;  
11        sayi++;  
12    }  
13  
14  
15  
16    public void add() {  
17  
18        String isim;  
19        System.out.println(isim);  
20    }  
21 }
```

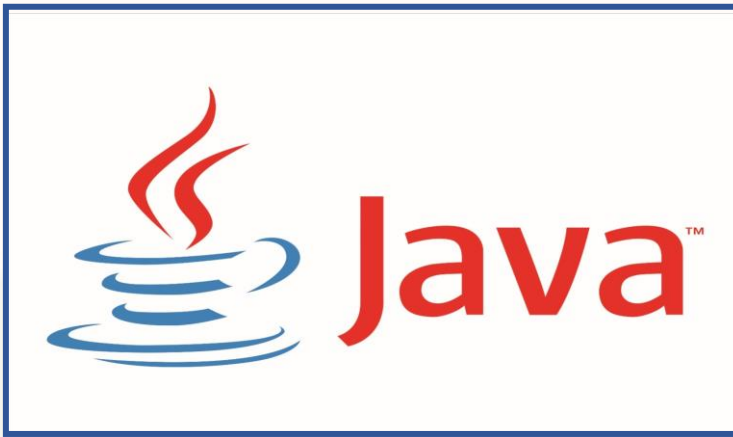
Scope

Instance, Class ve Local Variables

Loop Variables

- Bir loop icinde olusturulan variable'lar sadece o loop icerisinde gecerlidir.
- Loop icerisinde olusturulan variable'lara loop disindan ulasilamaz ve loop disinda kullanilamaz.
- Loop icerisinde olusturulan local variable'lari disarida kullanmaya calisirsaniz Java sikayet eder(CTE)

```
public static void main(String[] args) {  
    for (int i = 0; i < 10; i++) {  
        int sayi=10;  
        System.out.println(sayi);  
    }  
    System.out.println(sayi);  
}
```



17 TEMMUZ 2021
DERS 19

Arrays

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Önceki Dersten Aklimizde Kalanlar

- 1- Scope : Oluşturduğumuz variable'ların geçerli olduğu bölgedir. Bu konuda öğrenmemiz gereken şey hangi variable'a nereden ve nasıl ulaşabileceğimizdir.
- 2- Java'da bilmemiz gereken 4 scope vardır. Bunlardan ilk ikisi class level'da oluşturulan variable'lar için geçerlidir.
 - i) instance variables (Object variables): Objelere bağlı olarak değişir (öğretmen ismi veya not gibi)
 - class level'da oluşturulmalıdır.
 - static olmaması gerekir
 - static olmadıysa static alanlardan direkt kullanılamaz (object oluşturularak kullanılabilir)
 - oluşturmak yeterlidir, değer atamak mecburi değildir. (değer atanmazsa default değeri alır)
 - ii) static variables (Class variable) : Tüm objeler için ortaktır. (okul ismi gibi) Eğer static variable değiştirilirse tüm objeler etkilenir.
 - class level'da oluşturulmalıdır. (local'de static variable oluşturulamaz)
 - static kelimesi kullanılarak tanımlanmalıdır
 - Class içerisinde static olan veya olmayan her yer'den erişilebilir.
 - oluşturmak yeterlidir, değer atamak mecburi değildir. (değer atanmazsa default değeri alır)

Önceki Dersten Aklimizde Kalanlar

- 1- Scope : Oluşturduğumuz variable'ların geçerli olduğu bölgedir. Bu konuda öğrenmemiz gereken şey hangi variable'a nereden ve nasıl ulaşabileceğimizdir.
- 2- Java'da bilmemiz gereken 4 scope vardır. Bunlardan ilk ikisi class level'da oluşturulan variable'lar için geçerlidir.
 - i) instance variables (Object variables): Objelere bağlı olarak değişir (öğretmen ismi veya not gibi)
 - class level'da oluşturulmalıdır.
 - static olmaması gerekir
 - static olmadıysa static alanlardan direkt kullanılamaz (object oluşturularak kullanılabilir)
 - oluşturmak yeterlidir, değer atamak mecburi değildir. (değer atanmazsa default değeri alır)
 - ii) static variables (Class variable) : Tüm objeler için ortaktır. (okul ismi gibi) Eğer static variable değiştirilirse tüm objeler etkilenir.
 - class level'da oluşturulmalıdır. (local'de static variable oluşturulamaz)
 - static kelimesi kullanılarak tanımlanmalıdır
 - Class içerisinde static olan veya olmayan her yer'den erişilebilir.
 - oluşturmak yeterlidir, değer atamak mecburi değildir. (değer atanmazsa default değeri alır)

Önceki Dersten Aklımızda Kalanlar

3- diğer 2 scope class level'da değildir

iii) local variable : herhangi bir method içerisinde oluşturulan variable'lardır ve sadece oluşturuldukları method içerisinde direkt kullanılabilirler.

- oluşturmak için declaration yeterlidir, ancak kullanmak için declaration yetmez, mutlaka öncelikle değer ataması (assignment) yapmak gerekir.

- static olarak tanımlanamaz

- assignment yapmadan kullanmaya kalkarsak java CTE verir

iv) loop variable (bazı kaynaklarda buna da local variable denir)

- sadece oluşturuldukları loop içerisinde geçerlidirler.

- loop'un dışında erişilemez ve kullanılamazlar.

Scope

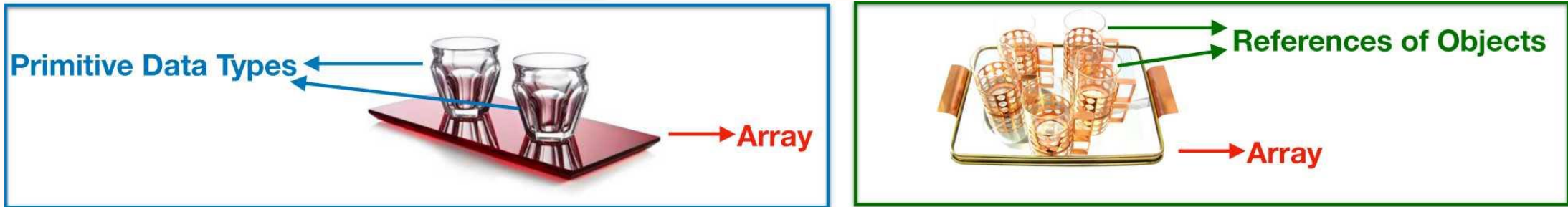
Instance, Class ve Local Variables

```
public class MyClass{
    int num1;
    String name = "Ali";
    public static void main(String args){
        add();
        product (5);
    }
    public static add(){
        num1 ++;
        int num2 = 6;
        char letter;
        System.out.println("Do addition ");
    }
    public product(int num3){
        name = "Veli";
        num2++;
        System.out.println(num3 * num3);
    }
}
```

- 1) Hangileri instance variable'dir ?
- 2) Hangileri local variable'dir?
- 3) num1 için default value nedir ?
- 4) Java hangi satirlarin altini kirmizi cizer?
- 5) Kac satir compile time error verir?

Arrays

Arrays birden fazla variable depolamak için kullanılabilen object (non-primitive data)'lerdir.



- 1) Arrays'de sadece primitive datalar veya non-primitive datalara ait referans'lar depolanabilir
- 2) Arrays içindeki tüm variable'lar aynı data type'inde olmalıdır.

Arrays

- 3) Bir Array olusturmadan once o Array'in icine kac variable koyacagimiza karar vermeliyiz.
- 4) Bir Array icine koyabilecegimiz variable sayisina o Array'in "length" i denir. O Array icine length'den fazla variable koyamayiz.



Maximum capacity (length) = 2



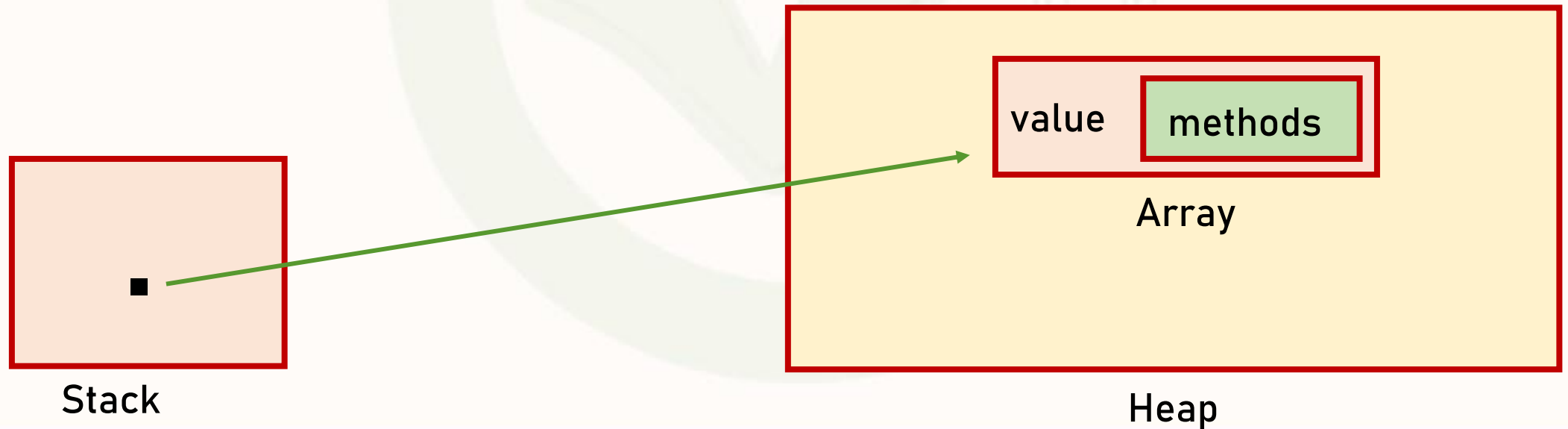
Maximum capacity (length) = 5

Arrays

5) Array'ler object (non-primitive) 'tir. Bu yuzden

- Heap Memory'de depolanirlar.
- Value ile birlikte method'lara da sahiptirler
- runtime'da olusturulurlar.

Bir Array declare edildiginde stack memory'de referans olusturulur ama Array henuz olusturulmamistir.



Arrays

6) Bir Array nasıl declare edilir?

Array declare etmek için iki yol vardır :

- `int myArray[] ;` // Bu daha çok kullanılır
- `int [] myArray;`

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

7) Bir Array nasıl oluşturulur

```
int myArray[] = new int[6];
```

- Yukarıdaki kod **length'i** 6 olan bir array oluşturur.
- Biz array'e eleman eklemesek Java elemanlar için data type'ına uygun default değerler atar.
- Eğer yukarıdaki array'i yazdırırsanız ekranda {0, 0, 0, 0, 0, 0} görürsünüz

NOT : Array oluştururken length'i yazmazsanız compile time error alırsınız.

Arrays

8) Array'e degerler nasil atanir

```
int myArray[ ] = new int[3];
```

```
myArray[0] = 9;  
myArray[1] = 10;  
myArray[2] = 11;
```

Once olusturup, sonra istedigimiz indexler icin deger atayabiliriz

Veya

```
int myArray[ ] = {9, 10, 11};
```

Olusturma ve tum indexler icin deger atamayi tek satirda yapariz.

Soru 1: Elemanlari "Ali" , "Veli", "Ayse" ve "Fatma" olan bir array olusturun ve bu array'i yazdirin.

Arrays

9) Array'in elemanlarına nasıl ulaşılır ve nasıl update edilir ?

```
int myArray[ ] = {9, 10, 11};
```

Array elemanlarına index'ler kullanılarak ulaşılır.

myArray[0] ==> 9,

myArray[1] ==> 10,

myArray[2] ==> 11,

NOT 1 : “n” array'in length'i olmak üzere myArray[n-1] son elemanı gösterir

NOT 2 : Bir Array'de olmayan index'i kullanmak isterseniz
“**ArraysIndexOutOfBoundsException**” alırsınız.

Soru 2: Soru 1'deki elemanlardan “Ali” yerine “Can”, “Ayse” yerine “Gul” atayın.

Arrays

10) Bir Array'in uzunlugu nasıl bulunur?

```
int myArray[] = {9, 10, 11};
```

```
int size = myArray.length;
```

NOT : String ve Array için length method'larında dikkatli olmak gerekir.

Strings ==> **length()**

Arrays ==> **length**



19 TEMMUZ 2021
DERS 20

Arrays

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Önceki Dersten Aklımızda Kalanlar

- 1- Array (dizi) : birden çok datayı store etmek için kullandığımız objelerdendir. (non-primitive)
- 2- NP olduğu için heap memory'de run time'da oluşturulur
- 3- Array oluşturmadan önce maximum element sayısına (length) karar vermemiz ve koyacağımız elementlerin data türünü declare etmemiz gerekir. `String arr [] = new String[5];`
- 4- Array'ı oluştururken içine koyacağımız elementlerin data türünü declare ettiğimiz için, o data türünden başka türde data koyamayız
- 5- Array'in kapasitesi sonradan değiştirilemez.. Declare edilirken length yazılmazsa CTE verir
- 6- Array'ler sadece primitive data türündeki dataları veya non-primitive data türündeki dataların referanslarını store edebilirler
- 7- Array'ı direk yazdırmak istersek Java array'ı değil referansını yazdırır. Array'deki elementleri yazdırmak için for loop kullanabiliriz veya tüm array'ı yazdırmak için Arrays class'ından toString method'unu kullanabiliriz
- 8- Array'ı oluşturduğumuzda değer atamazsak Java default değerler assign eder
- 9- Array'deki tüm elementlere index ile ulaşabilir veya update edebiliriz
- 10-

Arrays

11) Bir Array'in tüm elemanları nasıl yazdırılır?

```
int myArray[ ] = {9, 10, 11};
```

```
for(int i=0; i<size; i++) {  
    System.out.println(myArray[i]);  
}
```

```
System.out.println(Arrays.toString(myArray));
```

Soru 1: Verilen 3 elemanlı bir array'in tüm elemanlarını bir soldaki konuma taşıyacak bir program yazın. Örnek; array [1,2, 3] ise output [2, 3, 1] olacak.

Soru 2: Verilen bir array'in tüm elemanlarını toplayan bir program yazalım.

Arrays

12) Bir Array'in tüm elemanları nasıl sıralanır?

```
int myArray[ ] = {9, 15, 11};  
Arrays.sort (myArray);
```

Sıralama büyükten küçüğe nasıl yapılır ?

- Önce sort methodu kullanılır
- Sonra sıralamayı ters çevirmek için loop kullanılır

Arrays

13) Bir Array'de istenen bir elemanın varlığı nasıl kontrol edilir?

`binarySearch()` method'u belli bir elemanın bir array'de olup olmadığını kontrol etmek için kullanılır.

Ancak, `binarySearch()` methodunu kullanmadan önce mutlaka `sort()` methodu kullanılmalıdır.

```
int[ ] numbers = { 2, 8, 6, 4 };  
Arrays.sort(numbers);  
System.out.println (Arrays.binarySearch(numbers, 2)); //=====> 0  
System.out.println (Arrays.binarySearch(numbers, 4)); //=====> 1
```

Eğer bir eleman array'de yoksa output negatif olur.

- 1) 0 eleman var olsaydı sıra numarası kaç olurdu, buluruz.
- 2) Bulduğumuz sıra numarasının negatif hali, `binarySearch()`'un outputu olur.

```
System.out.println(Arrays.binarySearch(numbers, 1)); //=====> -1  
System.out.println(Arrays.binarySearch(numbers, 3)); //=====> -2  
System.out.println(Arrays.binarySearch(numbers, 9)); //=====> -5
```

Arrays

Output nedir ?

```
int[ ] numbers = { 2, 1, 7, 6 };  
Arrays.sort(numbers);  
System.out.println(Arrays.binarySearch(numbers, 2));  
System.out.println(Arrays.binarySearch(numbers, 7));  
System.out.println(Arrays.binarySearch(numbers, 3));  
System.out.println(Arrays.binarySearch(numbers, 9));
```

→ 1
→ 3
→ -3
→ -5

```
String[ ] letters = { "A", "N", "F", "C" };  
Arrays.sort(letters);  
System.out.println(Arrays.binarySearch(letters, "A"));  
System.out.println(Arrays.binarySearch(letters, "C"));  
System.out.println(Arrays.binarySearch(letters, "E"));  
System.out.println(Arrays.binarySearch(letters, "G"));
```

→ 0
→ 1
→ -3
→ -4

Arrays

14) İki array'in esit olup olmadigi nasıl kontrol edilir?

`equals()` method'u degerleri ve indexleri birlirkte kontrol edip, boolean bir deger return eder.

```
int arr1[] = {2, 1, 7, 6};  
int arr2[] = {7, 1, 6, 2};  
System.out.println(Arrays.equals(arr1, arr2)); → false  
  
int arr3[] = {3, 2, 7, 8, 11};  
int arr4[] = {7, 3, 8, 2, 12};  
Arrays.sort(arr3);  
Arrays.sort(arr4);  
System.out.println(Arrays.equals(arr3, arr4)); → false  
  
int arr5[] = {4, 2, 6, 8, 11};  
int arr6[] = {11, 4, 8, 2, 6};  
Arrays.sort(arr5);  
Arrays.sort(arr6);  
System.out.println(Arrays.equals(arr5, arr6)); → true
```

Arrays

16) Bir String nasıl array'e çevrilir ?

`split()` method'u String'e ait bir method'dur ve belirlediğimiz ayırıcıya göre String'i parçalara ayırıp bir Array'e çevirir.

```
String str = "Java öğrenmek, IT alanında yer edinmek demektir.";

String arr1[]=str.split(",");
System.out.println(Arrays.toString(arr1));
                → [Java öğrenmek, IT alanında yer edinmek demektir.]

String arr2[]=str.split(" ");
System.out.println(Arrays.toString(arr2));
                → [Java, öğrenmek,, IT, alanında, yer, edinmek, demektir.]

String arr3[]=str.split("");
System.out.println(Arrays.toString(arr3));
```

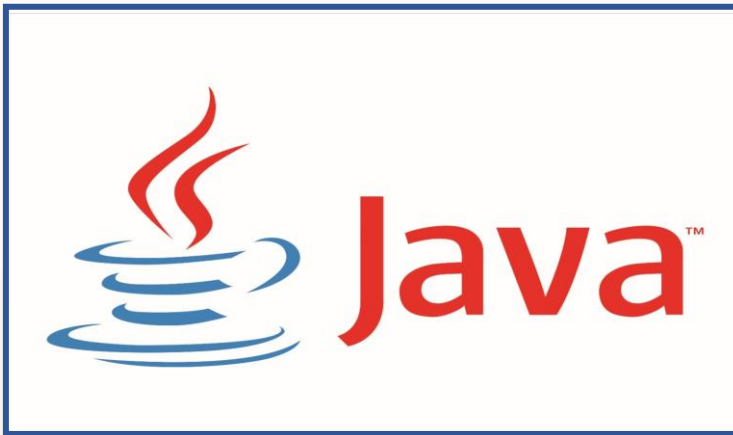
[J, a, v, a, , o, g, r, e, n, m, e, k, ,, , I, T, , a, l, a, n, d, a, , y, e, r, , e, d, i, n, m, e, k, , d, e, m, e, k, t, i, r, .]

Arrays

What is the result of the following?

```
int[] random = { 6, -4, 12, 0, -10 };  
int x = 12;  
int y = Arrays.binarySearch(random, x);  
System.out.println(y);
```

- A.** 2
- B.** 4
- C.** 6
- D.** The result is undefined.
- E.** An exception is thrown.
- F.** The code does not compile.



23 TEMMUZ 2021
DERS 21

Multi Dimensional Arrays

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Multi Dimensional Arrays (Cok Katli Array'ler)

Eger bir Array ic ice Array'lerden olusuyorsa buna Multi Dimensional Array denir

The diagram illustrates a 4D array structure. The code `int[][][] arrr = { { {1,2},{3,4},{5,6}} , { {7,8},{9,1},{2,3}} }` is shown. Annotations include:
- *child arrays*: Brackets above the innermost curly braces, pointing to {1,2}, {3,4}, {5,6} and {7,8}, {9,1}, {2,3}.
- *parent array*: Brackets below the middle curly braces, pointing to { {1,2},{3,4},{5,6}} and { {7,8},{9,1},{2,3}}.
- *grand parent array*: A large bracket below the entire array structure, pointing to the outermost curly braces.

```
int[][][] arrr = { { {1,2},{3,4},{5,6}} , { {7,8},{9,1},{2,3}} }
```

Multi Dimensional Arrays (Cok Katli Array'ler)

Array'i tanımlarken (declaration), her bir kat için bir [] kullanılır.

```
Int arr[ ][ ] = { {1,2} , {3,4}, {5,6}};
```

```
int arr[ ][ ] = new int [3][2];
```

Multi Dimensional Array oluşturma

```
arr[0][0]=1;  
arr[0][1]=2;
```

```
arr[1][0]=3;  
arr[1][1]=4;
```

Array içindeki elemanlara değer atama

```
arr[2][0]=5;  
arr[2][1]=6;
```

```
System.out.println(Arrays.toString(arr[0]));  
System.out.println(Arrays.toString(arr[1]));  
System.out.println(Arrays.toString(arr[2]));
```

Inner Array'leri yazdırma

```
System.out.println(arr[0][1]);  
System.out.println(arr[2][0]);
```

Belirli bir elemanı yazdırma

```
System.out.println(Arrays.toString(arr));
```

[[I@15db9742, [I@6d06d69c, [I@7852e922]

Multi Dimensional Arrays (Cok Katli Array'ler)

Multi Dimensional Array'in tum elemanlari nasil yazdirilir ?

```
public static void main(String[] args) {  
    int arr[][] = { {1,2} , {3,4}, {5,6}};  
    for (int i = 0; i < arr.length; i++) {  
        for (int j = 0; j < arr[i].length; j++) {  
            System.out.print(arr[i][j]+" ");  
        }  
    }  
    System.out.println(Arrays.deepToString(arr));  
}
```

→ Nested For Loop kullanılabilir

→ Arrays Class'indan method kullanılabilir

Multi Dimensional Arrays (Cok Katli Array'ler)

- Soru 1)** Asagidaki multi dimensional array'in tum elemanlarinin carpimini ekrana yazdiran bir method yaziniz. { {1,2,3}, {4,5,6} }
- Soru 2)** Asagidaki multi dimensional array'in ic array'lerindeki son elemanlarin carpimini ekrana yazdiran bir program yaziniz { {1,2,3}, {4,5}, {6} }
- Soru 3)** Asagidaki multi dimensional array'lerin ic array'lerinde ayni index'e sahip elemanlarin toplamini ekrana yazdiran bir program yaziniz. (Zor soru) arr1 = { {1,2}, {3,4,5}, {6} } ve arr2 = { {7,8,9}, {10,11}, {12} }
- Soru 4)** Asagidaki multi dimensional array'in ic array'lerindeki tum elemanlarin toplamini birer birer bulan ve herbir sonucu yeni bir array'in elemani yapan ve yeni array'i ekrana yazdiran bir program yaziniz { {1,2,3}, {4,5}, {6,7} }
- Ornek; { {1,2,3}, {4,5}, {6,7} } ==> 1+2+3=6 4+5=9 6+7=13 ==> output: {6, 9, 13}
- Soru 5)** Kullanicidan bir cumle isteyin ve cumledeki kelime sayisini yazdirin
- Soru 6)** Verilen bir Array'den isten degere esit olan elamanlari kaldirip, kalanlari yeni bir Array olarak yazdiran bir method yaziniz



24 TEMMUZ 2021
DERS 23

ArrayLists

Mehmet BULUTLUOZ
Elk.Elektronik Yuk.Muh.

Önceki Dersten Aklımızda Kalanlar

- 1- Multi Dimensional Arrays : çok katlı array, eğer bir array'in içinde eleman olarak yine array'ler bulunuyorsa buna MDA denir.
- 2- dıştaki array outer, içteki array'ler ise inner array olarak adlandırılır.
- 3- MDA kaç katlı ise declaration sırasında o kadar [] yazmamız gerekir
- 4- `int arr [] [] = new int [3][2];` bu declaration'da ilk [3] outer array'in içinde kaç tane inner array olduğunu gösterir, ikinci [2] inner array'lerin uzunluğunu verir. Bu şekilde bir declaration ile MDA oluşturulursa inner array'ler farklı uzunlukta olamaz, hepsinin uzunluğunun 2 olması şarttır... `[[0,0],[0,0],[0,0]]`
- 5- Alternatif olarak tüm array elemanlarını yazarak MDA oluşturulabilir.
`int arr [] [] = {{4,1}, {5,1,3,6},{2,7,8,9,12}}` bu durumda inner array'lerin uzunlukları aynı olmak zorunda değildir. `arr[1]` sadece 1 köşeli parantez kullanırsak outer array'in 1 index'indeki inner array'ini ifade eder. `arr[2][1]` → outer array'in 2 index'indeki inner array'in 1 index'indeki elementini verir = 7
- 6- MDA'de her bir elemente ulaşmak veya kullanmak istiyorsak nested for loop kullanmalısınız
- 7- MDA direkt yazdırmak istersek `Arrays.deepToString(arr)` kullanılabilir. Eğer `deepToString` kullanılmaz , yerine `toString` kullanılırsa inner array'lerin referanslarını yazdırır

Socrative Quiz

- 1) <https://www.socrative.com/> adresine gidin
- 2) **Login** butonuna basin
- 3) **Student Login** butonuna basin (veya <https://b.socrative.com/login/student/>)
- 4) Room Name **BULUTLUOZ** yazin
- 5) Isminizi yazin
- 6) **Done** butonuna basin

Sure : 15 Dakika

ArrayList

ArrayList nedir?

ArrayList length'i esnek olan bir Array'dir

ArrayList'e nicin ihtiyac duyariz?

- Biz array olustururken length'in en basta belirlemek zorundayiz ve daha sonra length'ini degistiremeyiz. Bu durum bizim esnek calismamiza engel olur.
- Bir array'in uzunlugunu degistirmek istedigimizde yeni bir array olusturmamiz gerekir, ArrayList de gerekmez.
- Bir array'den bir eleman silmek istedigimizde yeni bir array olusturmamiz gerekir, ArrayList de gerekmez.

ArrayList

ArrayList olusturma

```
ArrayList<String> list1 = new ArrayList<String>();
```

```
ArrayList<String> list2 = new ArrayList<>();
```

```
List<String> list3 = new ArrayList<>(); En cok bu kullanilir
```

```
ArrayList<String> list4 = new List<>();
```

Compile Time Error verir, esitligin sag tarafinda ArrayList kullanmak zorundayiz

ArrayList'i nasil yazdiririz?

ArrayList'i ekrana yazdirmek cok kolaydir.

```
System.out.println(list3);
```

ArrayList Method'lari

1) add()

add() method ArrayList'e eleman eklemek icin kullanilir

Ornek :

```
List<String> hayvan = new ArrayList<>();
```

A) add() method'u index olmadan calisabilir

```
hayvan.add("kedi"); // [kedi]
```

```
hayvan.add("yilan"); // [kedi, yilan]
```

B) add() method'u index ile de calisabilir

```
hayvan.add(1, "kartal"); // [kedi, kartal, yilan]
```

```
hayvan.add(0, "sinek"); // [sinek, kedi, kartal, yilan]
```

```
hayvan.add(1, "aslan"); // [sinek, aslan, kedi, kartal, yilan]
```

```
System.out.println(hayvan); // [sinek, aslan, kedi, kartal, yilan]
```

ArrayList Method'lari

2) size()

size() method ArrayList'de kac eleman oldugunu gosterir.

Ornek :

```
List<String> hayvan = new ArrayList<>();  
System.out.println(hayvan.size()); // 0  
  
hayvan.add("kedi"); // [kedi]  
hayvan.add("yilan"); // [kedi, yilan]  
  
System.out.println(hayvan.size()); // 2
```

3) isEmpty()

isEmpty() method'u ArrayList bos ise true, bos degilse false dondurur

ArrayList Method'lari

4) remove()

remove() method'u ArrayList'den belli bir elemani silmek icin kullanilir.

A) remove(index) kullanarak. Size'dan buyuk index yazilrsa exception verir. Index'li remove() methodu ArrayList'de verilen index'deki elemani siler.

```
List<String> hayvan = new ArrayList<>();  
hayvan.add("kedi"); // [kedi]  
hayvan.add("yilan"); // [kedi, yilan]  
hayvan.remove(1); // index'i 1 olan elemani siler  
System.out.println(hayvan); //[kedi]
```

NOT: remove(index) method'u silinen elemani dondurur. Yani method'u **System.out.println()** icinde kullanirsak silinen elemani ekrana yazdirir.

```
System.out.println(hayvan.remove(1)); //yilan
```

ArrayList Method'ları

B) remove("eleman") index'i değil elemanı kullanırsak kullandığımız elemanın ilk kullanıldığı yeri bulur ve siler.

```
List<String> hayvan = new ArrayList<>();  
    hayvan.add("kedi"); // [kedi]  
    hayvan.add("yılan"); // [kedi, yılan]  
    hayvan.add("kedi"); // [kedi, yılan, kedi]  
    hayvan.remove("kedi");  
    System.out.println(hayvan); // [yılan, kedi]
```

Not: Index'siz remove() method'u true veya false döndürür.

```
System.out.println(hayvan.remove("kedi")); //true yani kedi eleman olarak vardı ve sildim
```

```
System.out.println(hayvan.remove("tavsan")); // false yani tavsan eleman olarak yoktu  
dolayısıyla silemedim
```

ArrayList Method'lari

5) set()

set() methodu ArrayList'de var olan bir elemani degistirmeye yarar

```
List<String> hayvan = new ArrayList<>();  
hayvan.add("kedi"); // [kedi]  
hayvan.add("yilan"); // [kedi, yilan]  
hayvan.set(1, "tavsan");  
  
System.out.println(hayvan); // [kedi, tavsan]
```

NOT: set() method'u add() method'u yerine kullanilamaz .
Olmayan bir index ile set() kullanilrsa exception verir.

```
hayvan.set(2, "aslan"); // IndexOutOfBoundsException
```

ArrayList Method'lari

6) get(index)

get() methodu ArrayList'deki istenen indexdeki elemani dondurur.

```
List<String> hayvan = new ArrayList<>();
```

```
hayvan.add("kedi"); // [kedi]
```

```
hayvan.add("yilan"); // [kedi, yilan]
```

```
System.out.println(hayvan.get(0)); // kedi
```

```
System.out.println(hayvan.get(1)); // yilan
```

ArrayList Method'lari

7) contains()

contains() methodu ArrayList'de bir elemanın var olup olmadığını kontrol eder. Eleman varsa true, yoksa false return eder.

```
List<String> hayvan = new ArrayList<>();
```

```
hayvan.add("kedi"); // [kedi]
```

```
hayvan.add("yilan"); // [kedi, yilan]
```

```
System.out.println(hayvan.contains("kedi")); // true
```

```
System.out.println(hayvan.contains("tavsan")); // false
```


ArrayList Method'lari

8) **Collections.sort()** : sort() methodu ArrayList'deki elemanlari kucukten buyuge veya alfabetik siraya gore dizer.

```
List<String> hayvan = new ArrayList<>();  
hayvan.add("yilan"); // [yilan]  
hayvan.add("kedi"); // [yilan, kedi]  
hayvan.add("tavsan"); // [yilan, kedi, tavsan]
```

```
System.out.println(hayvan); //[yilan, kedi, tavsan]
```

```
Collections.sort(hayvan);
```

```
System.out.println(hayvan); // [kedi, tavsan, yilan]
```

ArrayList Method'lari

9) equals()

equals() methodu iki listteki ayni indexteki elemanlari ayni olup olmadigini kontrol eder. Ayni indexteki tum elemanlar ayni ise true return eder, farkli ise false return eder

```
List<String> first = new ArrayList<>( );  
List<String> second = new ArrayList<>( );  
System.out.println(first.equals(second)); // true  
  
first.add("a"); // [a]  
System.out.println(first.equals(second)); // false  
  
second.add("a"); // [a]  
System.out.println(first.equals(second)); // true  
  
first.add("b"); // [a,b]  
second.add(0,"b"); // [b,a]  
System.out.println(first.equals(second)); // false
```

ArrayList Method'lari

10) clear()

clear() methodu ArrayList'teki tum elemanlari siler.
Return type'i void'dir, hic bir sey donmez

```
List<String> hayvan = new ArrayList<>();  
hayvan.add("yilan"); // [yilan]  
hayvan.add("kedi"); // [yilan, kedi]  
  
System.out.println(hayvan.isEmpty()); // false  
System.out.println(hayvan.size()); // 2  
  
hayvan.clear();  
System.out.println(hayvan.isEmpty()); // true  
System.out.println(hayvan.size()); // 0
```

Sorular

ArrayList

- 1) Elemanlari A, C, E, ve F olan bir String ArrayList olusturup ekrana yazdiriniz.
- 2) indexsiz **add()** methodunu kullanarak, B'yi ekleyiniz.
index'li **add()** methodunu kullanarak, L'yi 1 numarali index'e ekleyiniz.
ArrayList'i ekrana yazdiriniz, list goyle olmalı; A, L, C, E, F, B.
- 3) **set()** methodu kullanarak, E'yi D yapiniz.
ArrayList'i ekrana yazdiriniz, list goyle olmalı; A, L, C, D, F, B.
- 4) **remove()** methodu kullanarak, F'yi siliniz.
ArrayList'i ekrana yazdiriniz, list goyle olmalı; A, L, C, D, B.
- 5) **sort()** methodu kullanarak, elemanlari alfabetik siraya diziniz.
ArrayList'i ekrana yazdiriniz, list goyle olmalı; A, B, C, D, L.
- 6) **contains()** methodu kullanarak, L'nin list'de var oldugunu ve M'nin list'de var olmadigini dogrulayiniz.
- 7) **size()** methodu kullanarak, list'in kac eleman oldugunu ekrana yazdiriniz.
- 8) **clear()** methodu kullanarak, list'deki tum elemanlari siliniz.
- 9) **isEmpty()** methodu kullanarak, list'deki tum elemanlari silindigini dogrulayiniz

Array'i ArrayList'e Cevirmek

```
String [ ] arr = {"tavsan", "serce"};
```

```
List<String> list = Arrays.asList(arr);
```

Uzunlugu degistirilemeyen bir list'e ceviriir. Yani; yeni olusturulan listte add(), remove() ve clear() methodlarini kullanamazsiniz. Exception

```
System.out.println(list.size()); // 2
```

```
System.out.println(list); // [tavsan, serce]
```

NOT: Eger array'deki bir elemani degistirirseniz list'teki eleman da otomatik olarak degisir. Listteki bir elemani degistirirseniz array de otomatik olarak degisir.

```
list.set(1, "test"); // [tavsan, test]
```

```
arr[0] = "new"; // [new, test]
```

```
System.out.println(Arrays.toString(arr)); // [new, test]
```

```
System.out.println(list); // [new, test]
```

ArrayList'i Array'e Cevirmek

```
List<String> list = new ArrayList<>();  
list.add("tavsan");  
list.add("horoz");  
System.out.println(list); //[tavsan,horoz]
```

1.yontem

```
String arr[ ] = list.toArray(new String[0]);
```

```
System.out.println(arr.length); // 2  
System.out.println(Arrays.toString(arr)); // [tavsan,horoz]
```

2.yontem

```
Object arr[ ] = list.toArray();
```

```
System.out.println(arr.length); // 2  
System.out.println(Arrays.toString(arr)); // [tavsan,horoz]
```

For Each Loop Enhanced (Gelistirilmis) For Loop

Faydaları:

Kodun daha okunabilir olmasını sağlar. Hata yapma ihtimalini azaltır.

```
public static void main(String args[]){  
  
    int arr[]={12,13,14,44};  
  
    for( int i: arr) {  
        System.out.print(i + " ");  
    }  
  
}
```

```
public static void main(String args[]){  
    ArrayList<String> list=new  
    ArrayList<String>();  
    list.add("Ali");  
    list.add("Veli");  
    list.add("Can");  
  
    for( String s : list) {  
        System.out.print(s + " ");  
    }  
  
}
```

For Each Loop

Soru 1:

Bir integer array olusturunuz ve bu array'deki tum sayilarin carpimini For-each loop kullanarak bulunuz. Sonucu ekrana yazdiriniz.

Soru 2:

Bir integer list olusturunuz ve bu list'deki tum sayilarin karesinin toplamini For-each loop kullanarak bulunuz. Sonucu ekrana yazdiriniz.

Soru 3:

iki String array olusturunuz ve bu array'lerdeki ortak elemanlari For-each loop kullanarak bulunuz. Sonucu ekrana yazdiriniz.

Ortak eleman yoksa ekrana "Ortak eleman yok" yazdiriniz

Soru 4:

Bir String olusturunuz, bu String'deki character'leri for-each loop kullanarak yazdiriniz. *ipucu: split()*

Constructor

(Java object'leri nasıl oluşturur ?)

