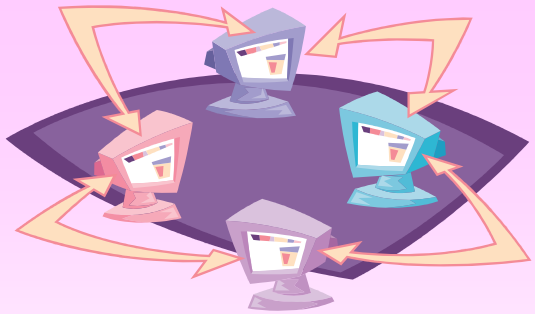
Dasar Pemrograman Java

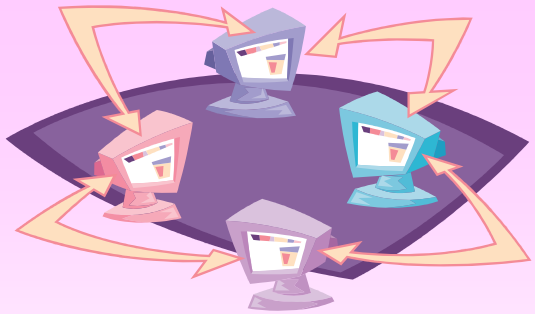
Tessy Badriyah, SKom. MT.

http://lecturer.eepis-its.edu/~tessy



Tujuan Pembelajaran

- Penggunaan Komentar dalam program
- Memahami perbedaan identifier yang valid dan yang tidak valid
- Memahami Keyword pada JAVA
- Memahami 8 tipe data dasar
- Menggunakan nilai literal untuk tipe numerik dan teks
- Memahami istilah variabel primitif dan variabel referensi
- Memahami penggunaan Operator dalam JAVA

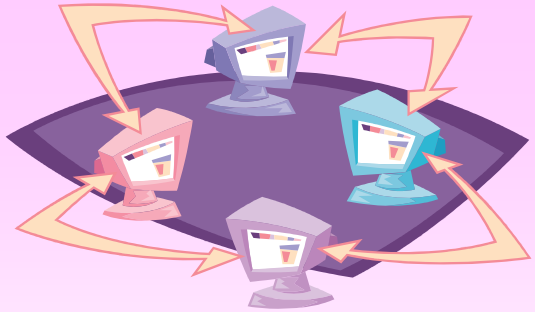


Penggunaan Komentar

- Komentar digunakan untuk tulisan berupa keterangan dan tidak ikut diproses pada saat program dijalankan
- Komentar digunakan untuk mengingat kembali perintah yang pernah ditulis
- Komentar pada Java menggunakan dua cara :
  - Komentar satu baris

    // ini adalah komentar satu baris
  - Komentar lebih dari satu bari

```
/*   ini adalah komentar baris pertama
     dan ini komentar baris kedua
*/
```

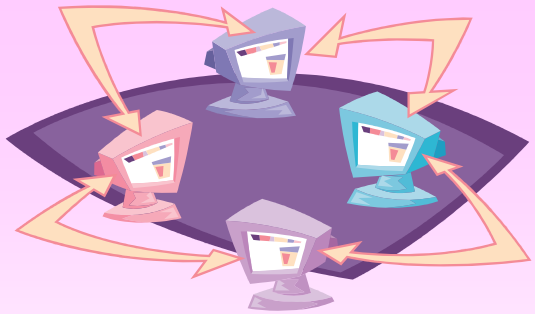


Percobaan 1

- Memberi komentar pada program
- Berikan komentar pada Class Mobil yang dibuat pada bab sebelumnya

Percobaan 1

```
class Mobil {
  // atribut
  String aktifitas;
  String warna;
  int kecepatan;
  // method untuk memeriksa kecepatan
  void cekKecepatan() {
    if (kecepatan==0)
      aktifitas="parkir";
  }
  // method untuk mencetak atribut
  void cetakAtribut() {
    System.out.println("Aktifitas    = "+aktifitas);
    System.out.println("warna        = "+warna);
    System.out.println("Kecepatan    = "+kecepatan);
  }
  public static void main(String [] args) {
    // membuat obyek baru bernama mobilku
    Mobil mobilku = new Mobil();
    // memberi nilai awal pada atribut
    mobilku.kecepatan=0;
    mobilku.warna="merah";
    // memeriksa kecepatan
    mobilku.cekKecepatan();
    // mencetak atribut
    mobilku.cetakAtribut();
   }
}
```



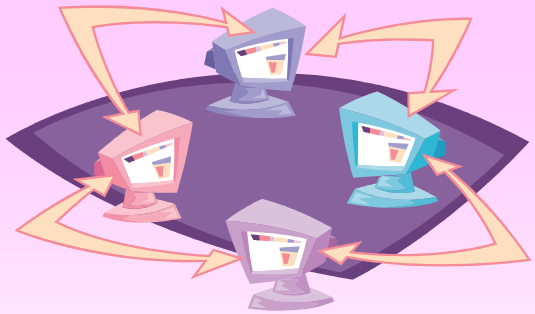
Baris perintah blok program (1)

- Baris perintah pada Java selalu diakhiri dengan tanda (;)

  total = total + x;

- Blok adalah kumpulan perintah yang diapit dengan tanda kurung buka { dan kurung tutup }

```
{
    a = b + c;
    a = a + 1;
}
```



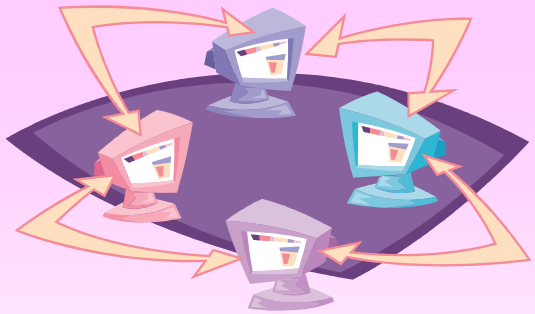
Baris perintah blok program (2)

- Definisi sebuah class, diletakkan dalam blok.

```
public class Tanggal {
        private int tgl;
        private int bulan;
        private int tahun;
}
```

- Blok program bisa bersarang (nested) => di dalam blok program terdapat blok program yang lain

```
public class CekTanggal {
        if (bulan==2)
        {   tgl=28;
        }
}
```



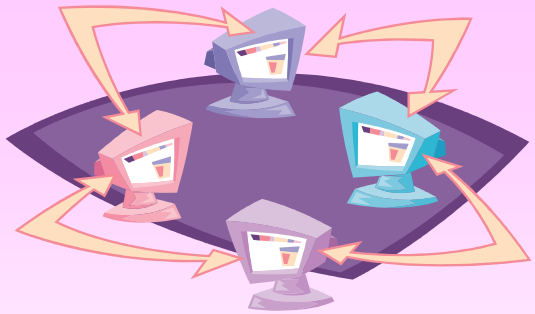
Identifier

- Identifier digunakan untuk memberi nama variabel, class dan method
- Identifier dimulai dengan sembarang huruf, underscore(_) atau dollar ($)
- Contoh penamaan :

  namaku

  _var1

  $harga

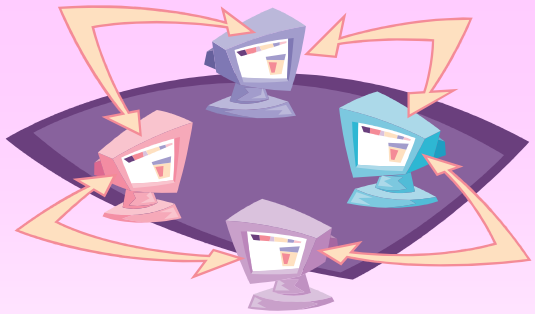


Percobaan 2

- Yang mana penggunaan identifier yang tidak valid ? Betulkan !

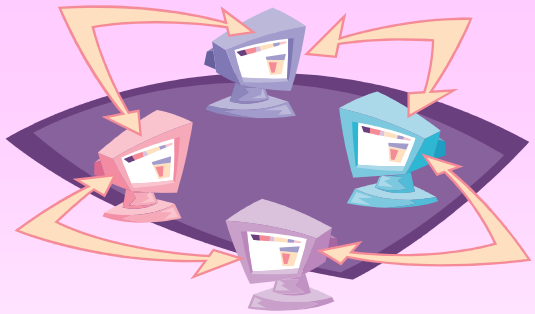
Tes.java *

```
public class Tes {
    public static void main(String [] args) {
        int 123Bilangan=0;          // identifier 1
        String $aku="namaku";       // identifier 2
        double _angka=3934.38;      // identifier 3
        short &bil=123;             // identifier 4
    }
}
```



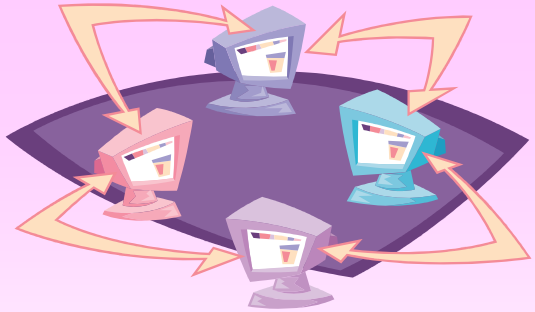
Keyword yang dimiliki JAVA

abstract	default	if	private	this
boolean	do	implements	protected	throw
break	double	import	public	throws
byte	else	instanceof	return	transient
case	extends	int	short	try
catch	final	interface	static	void
char	finally	long	strictfp	volatile
class	float	native	super	while
const	for	new	switch	
continue	goto	package	synchronized	



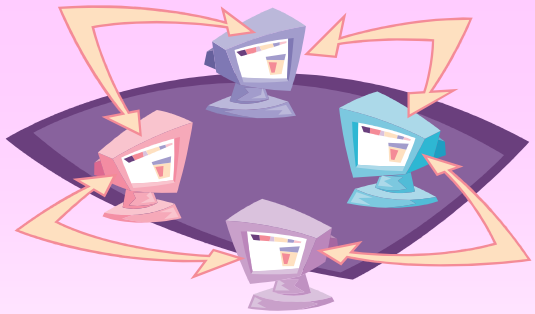
Tipe Data Dasar

- Dalam JAVA, tipe data dasar ada 8 yaitu :
  - Logika – boolean (1)
  - Teks – char (2)
  - Bilangan bulat – byte (3), short (4), int (5) dan long (6)
  - Bilangan pecahan – float (7) dan double (8)



Tipe data Boolean

- Tipe data boolean mempunyai dua kemungkinan nilai : true atau false
- Contoh :
  - boolean ada = true;

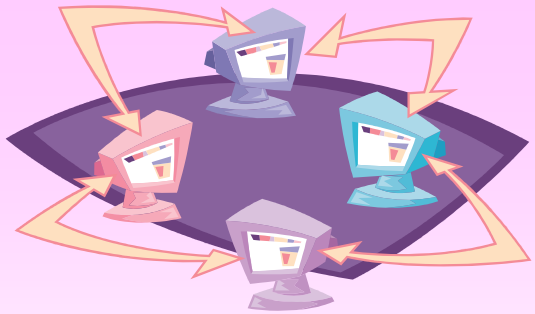


Percobaan 3

- Penggunaan tipe data boolean

Tes.java

```
public class Tes {
    public static void main(String [] args) {
        String tanya = "Apakah daisy ada di rumah ?";
        boolean ada=true;
        System.out.println(tanya);
        if (ada==true)
            System.out.println("ada");
        else
            System.out.println("tidak ada");
    }
}
```



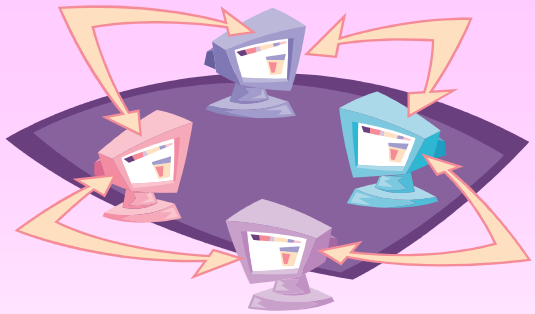
Tipe data teks => char

- Direpresentasikan dalam 16 bit unicode
- Nilai karakter diapit dengan tanda petik tunggal
- Contoh :
  - ‘a’

    huruf a
  - ‘\t’

    tab
  - ‘\u0063’

    unicode karakter untuk huruf c



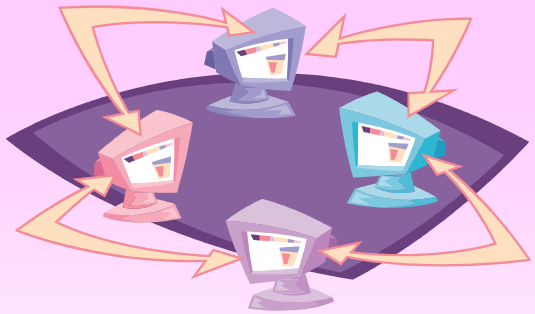
Percobaan 4

- Penggunaan Tipe Data char

TipeChar.java *

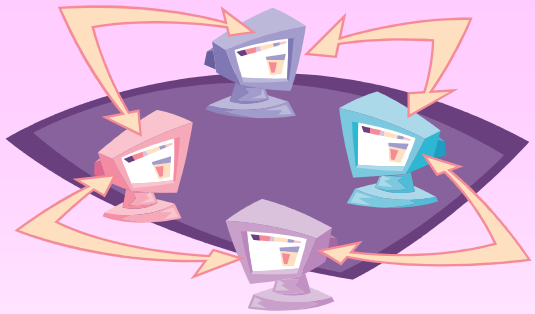
```
public class TipeChar {
    public static void main(String [] args) {
        // menampilkan variabel karakter
        char huruf='a';
        System.out.println("ini huruf "+huruf);
        // \t = tab
        System.out.println("Belajar\tJava");
        // menampilkan unicode karakter abc
        System.out.println("\u0061\u0062\u0063");
    }
}
```

```
C:\j2sdk1.4.1_01\bin\java.exe  -classpath "C:\j2
ini huruf a
Belajar       Java
abc
Finished executing
```



Tipe data teks => String

- Sebenarnya bukan tipe data dasar tapi sebuah class
- Perhatikan huruf besar pada karakter pertama yang merupakan ciri class
- Nilai string diapit dengan tanda petik ganda
- Contoh :
  - String salam="Assalamu'alaikum"
  - String pesanerror="Record tidak ditemukan"



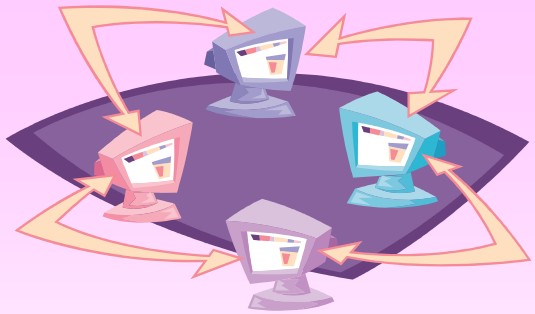
Tipe data bilangan bulat => byte, short, int, long

- Menggunakan tiga bentuk => desimal, oktal, hexadesimal
- Contoh :
  - 2

    Bentuk desimal untuk integer 2
  - 077

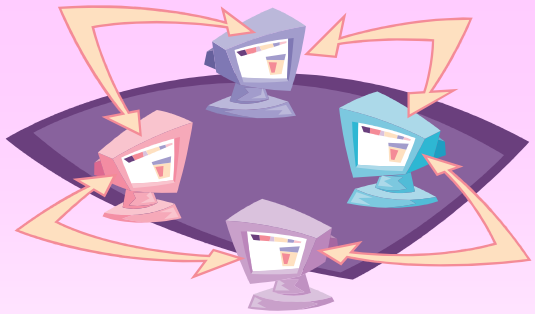
    Diawali dengan angka 0, menandakan bilangan oktal
  - 0xBAAC

    Diawali dengan 0x menandakan bilangan hexadesimal
- Defaultnya adalah *int*
- Untuk mendefinisikan tipe data long digunakan L atau l di belakang nilai



Range untuk tipe data bilangan bulat

Integer Length	Name or Type	Range
8 bits	`byte`	-2^{7} to $2^{7}-1$
16 bits	`short`	-2^{15} to $2^{15}-1$
32 bits	`int`	-2^{31} to $2^{31}-1$
64 bits	`long`	-2^{63} to $2^{63}-1$



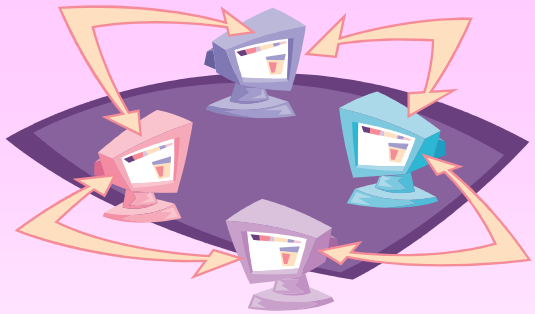
Percobaan 5

- Penggunaan tipe bilangan bulat
- Jelaskan program di bawah ini bagaimana outputnya ?

Tes.java

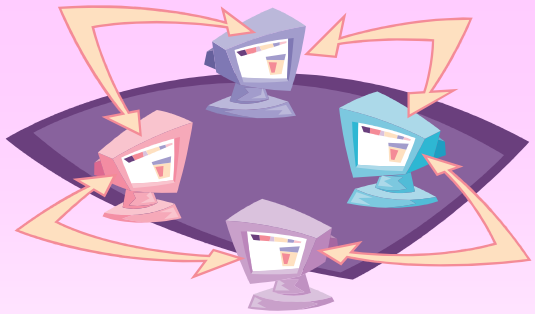
```
public class Tes {
    public static void main(String args []) {
        int a = 12;
        int b = 012;
        int c = 0x12;
        System.out.println(a);
        System.out.println(b);
        System.out.println(c);
    }
}
```

```
C:\j2sdk1.4.1_01\bin\java.exe  -classpath "C:\j2sdk1.4.1
12
10
18
Finished executing
```



Bilangan pecahan => float, double

- Defaultnya adalah double
- Dapat dituliskan dalam bentuk :
  - Bilangan desimal : 12.345
  - Floating point : 6.02E34
  - Float : 2.425F
  - Double : 123.4E+342D



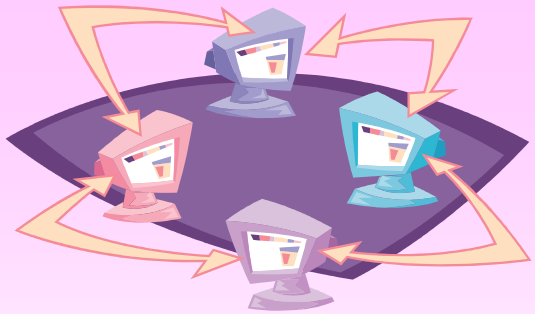
Percobaan 6

- Penggunaan tipe bilangan pecahan

Tes.java

```
public class Tes {
   public static void main(String args []) {
     float harga=150000;
     double total;
     System.out.println("Harga barang = "+harga);
     total=harga + 0.1*harga;     // harga + 10% pajak
     System.out.println("Total penjualan = "+total);
   }
}
```

```
C:\j2sdk1.4.1_01\bin\java.exe  -classpath "C:\j2sdk1.4.1_01
Harga barang = 150000.0
Total penjualan = 165000.0
Finished executing
```



Konversi

- Konversi tipe data terjadi pada saat :
  - Suatu nilai diberikan pada variabel yang berbeda tipe datanya
- Aturan pada konversi :
  - Tipe data boolean tidak bisa dikonversi ke tipe data lain
  - Selain boolean bisa dikonversi dengan prinsip : **widening** => tipe data variabel di sebelah kiri harus memiliki range (jangkauan) yang lebih luas daripada tipe data variabel di sebelah kanannya

    => jika syarat ini tidak bisa dipenuhi maka konversi tidak bisa dilakukan dan harus dilakukan **casting**.

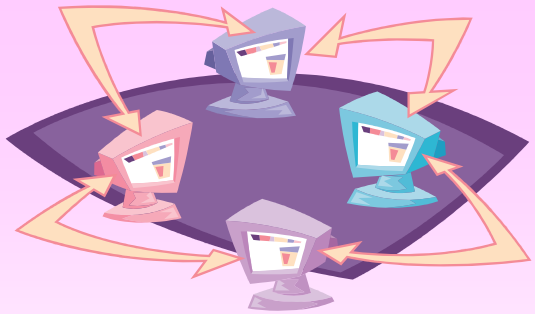
Casting

- Jadi casting adalah perubahan data yang dilakukan oleh user karena tidak bisa dilakukan konversi.
- Contoh penulisan :

```
(Data Type)  Expression
```

- Contoh casting :

```
(int) 3.75          ===>  3
(float) 3           ===>  3.0
(float) (1 / 2)     ===>  0.0
(float)1/2          ===>  0.5
```



Prinsip widening pada Casting

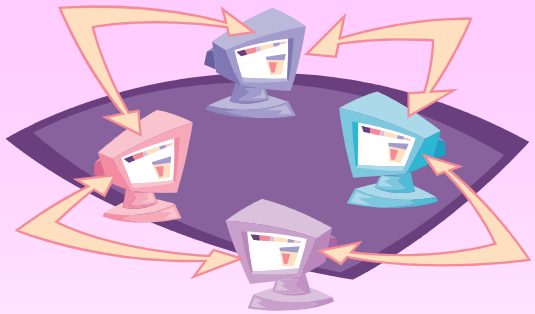
- Contoh casting :

  short s = 5;

  int i = 100;

  s = (short) i;

  => dilakukan casting karena tipe data sebelah kiri lebih kecil jangkauannya (short) dibandingkan tipe data sebelah kanan (int) atau dengan kata lain prinsip widening tidak bisa dipenuhi



Percobaan 7

- Apa yang terjadi jika program ini dijalankan ?
  Jika terjadi error betulkan !

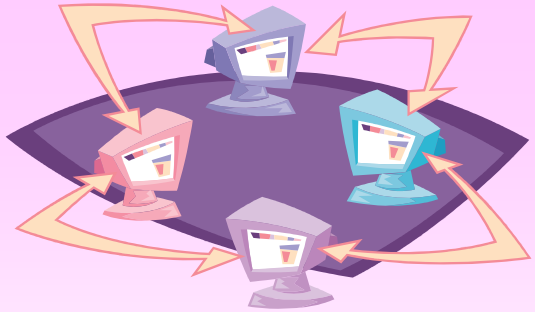
Tes.java

```
public class Tes {
    public static void main(String args []) {
        short s = 9;
        int i=10;
        float f=11.1f;
        double d=12.2;
        s = i;
        d = f;
        i = d;
    }
}
```

pembetulan

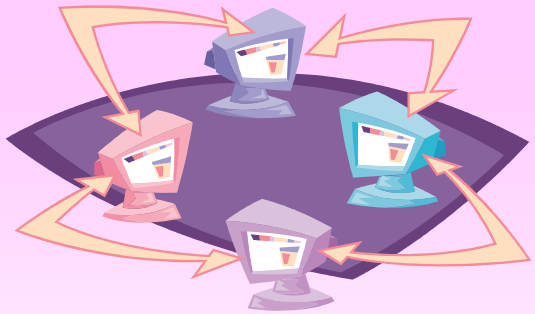
Tes.java

```
public class Tes {
    public static void main(String args []) {
        short s = 9;
        int i=10;
        float f=11.1f;
        double d=12.2;
        s = (short) i;
        d = f;
        i = (int) d;
    }
}
```



Promotion dari tipe primitif

- Promotion terjadi pada saat operasi aritmatik dimana kompiler berusaha mencari tipe data yang sesuai dengan tipe data operan yang berbeda-beda.



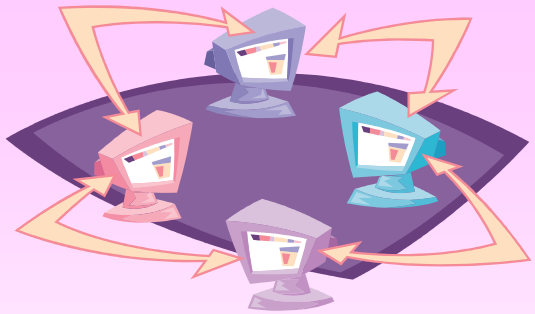
Percobaan 8

- Contoh Promotion

Tes.java

```
public class Tes {
   public static void main(String args []) {
         short s = 9;
         int i=10;
         float f=11.1f;
         double d=12.2;
         double hasil;
         hasil=((-s*i)*(f/d));
    }
}
```

- Short s dipromosikan ke int, kemudian dikalikan negatif hasilnya dikalikan dengan int i kemudian hasilnya (hasil pertama) disimpan. Float f dipromosikan jadi double selanjutnya dibagi dengan double d hasilnya disimpan (hasil kedua) menjadi double. Hasil pertama (int) dipromosikan menjadi double selanjutnya dikalikan dengan hasil kedua, hasil terakhir disimpan dalam variabel hasil dalam bentuk double.

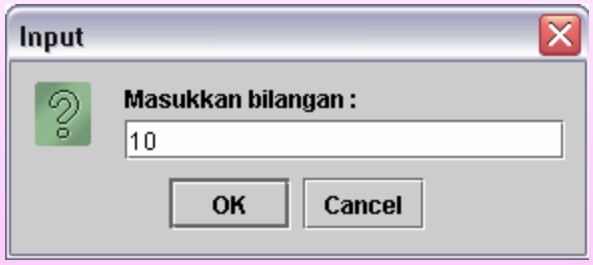


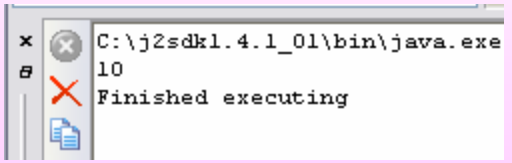
Menginputkan suatu Nilai

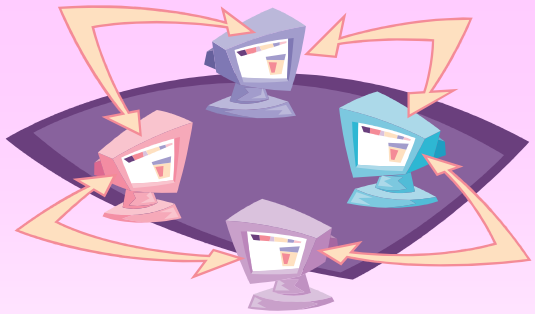
- Untuk menginputkan suatu nilai dari keyboard, dapat menggunakan JOptionPane
- Berikut ini program untuk menginputkan bilangan integer dari keyboard

InputBilangan.java

```
import javax.swing.JOptionPane;
public class InputBilangan {
   public static void main(String [] args) {
      int bil;
      String str=JOptionPane.showInputDialog("Masukkan bilangan : ");
      bil=Integer.parseInt(str);
      System.out.println(bil);
      System.exit(0);
    }
}
```

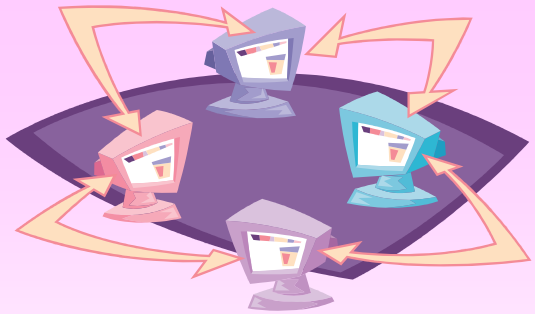






Menginisialisasi Obyek dengan Constructor

- Suatu obyek dapat diberi nilai awal atau diinisialisasi dengan menggunakan constructor.
- Jadi constructor dijalankan hanya sekali yaitu saat suatu obyek diciptakan.
- Contoh : memberi nilai awal 1-Mei-2007 pada obyek yang diciptakan dari class Tanggal

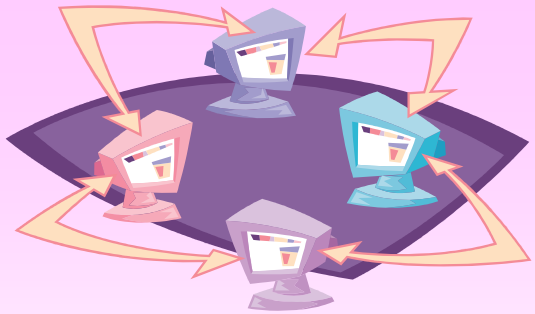


Percobaan 9

- Menginisialisasi obyek dengan Constructor

Tanggal.java

```
public class Tanggal {
        public int tgl;
        public int bulan;
        public int tahun;
        public Tanggal() {
            tgl=1;
            bulan=5;
            tahun=2007;
        }
}
```



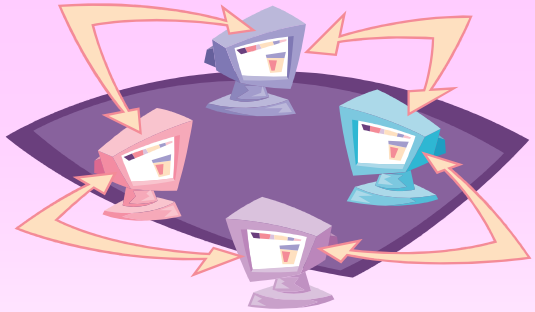
Percobaan 10

- Menguji class Tanggal

Tanggal.java | TesTanggal.java

```
public class TesTanggal {
    public static void main(String [] args) {
            Tanggal tanggalku = new Tanggal();
            System.out.println(tanggalku.tgl);
            System.out.println(tanggalku.bulan);
            System.out.println(tanggalku.tahun);
    }
}
```

```
C:\j2sdk1.4.1_01\bin\java.exe  -classpath "C:\j2s
1
5
2007
Finished executing
```

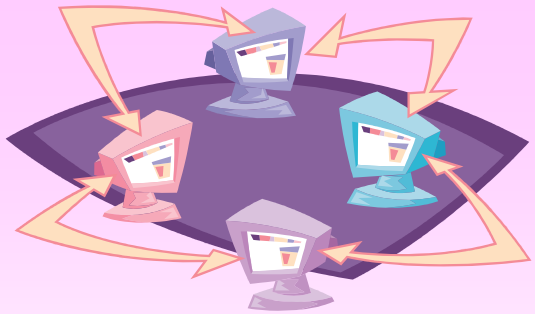


Percobaan 11

- Constructor juga dapat diberi parameter (argumen)
- Modifikasi class Tanggal sebelumnya :

Tanggal.java *

```
public class Tanggal {
        public int tgl;
        public int bulan;
        public int tahun;
        public Tanggal(int stgl, int sbulan, int stahun) {
           tgl=stgl;
           bulan=sbulan;
           tahun=stahun;
        }
}
```



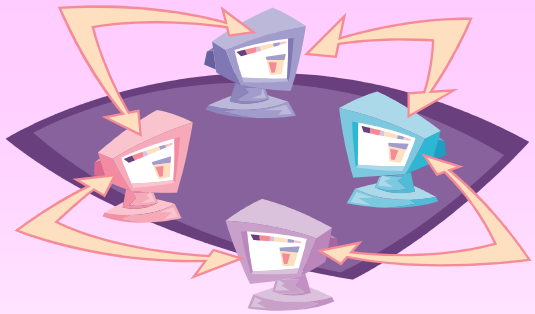
Percobaan 12

- Menguji constructor dengan parameter

Tanggal.java | TesTanggal.java

```
public class TesTanggal {
    public static void main(String [] args) {
            Tanggal tanggalku = new Tanggal(3,5,2007);
            System.out.println(tanggalku.tgl);
            System.out.println(tanggalku.bulan);
            System.out.println(tanggalku.tahun);
    }
}
```

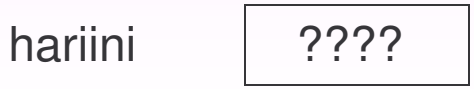
```
C:\j2sdk1.4.1_01\bin\java.exe  -classpath "C:\j2sdk1.
3
5
2007
Finished executing
```



Ilustrasi dari Alokasi Memori pada Constructor

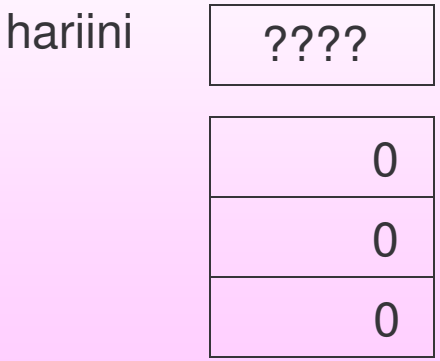
- Deklarasi variabel untuk obyek baru

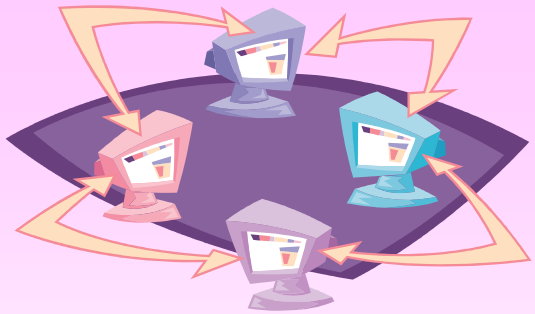
  Tanggal hariini = new Tanggal(3,5,2007);



- Menggunakan operator new untuk mengalokasikan memori => constructor dijalankan.

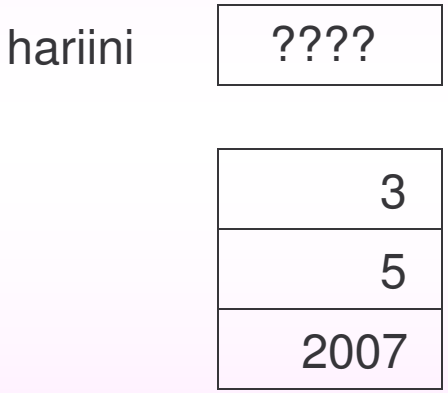
  Tanggal hariini = new Tanggal(3,5,2007);

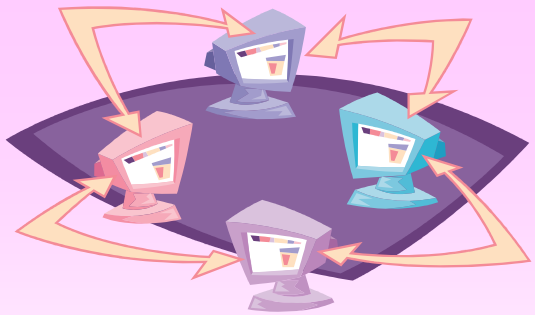




Ilustrasi dari Alokasi Memori pada Constructor

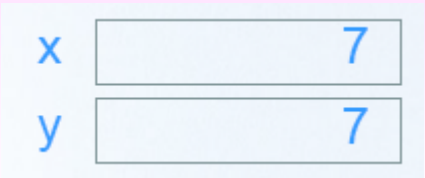
- Penandaan variable dibuat untuk merefer ke obyek



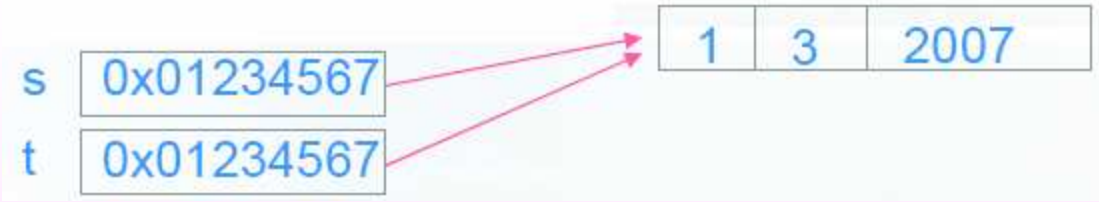


Merujuk ke alamat variabel lain (*Assigning Reference Type*)

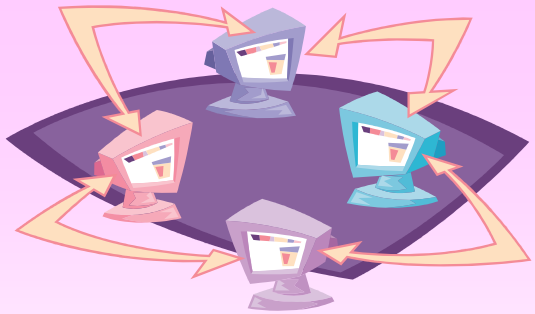
- int x = 7;
  int y = x;



- Tanggal s = new Tanggal(1,3,2007);
  Tanggal t = s;

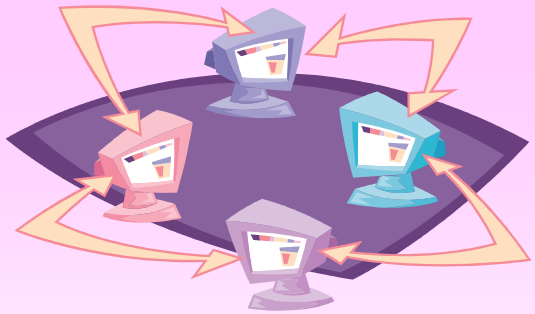


- t = new Tanggal(4,6,2007)



Pass by Value

- Pada pemrograman Java, pada argumen hanya dilewatkan suatu nilai bukan alamat (pass by value)
- Jika argumen berupa obyek, maka nilai dari argumen tersebut adalah referensi ke obyek yang dilewatkan
- Isi dari obyek bisa berubah karena pemanggilan sebuah method, tapi pemanggilan suatu method dengan argumen referensi ke obyek tidak akan merubah nilai obyek pada bagian program yang memanggil

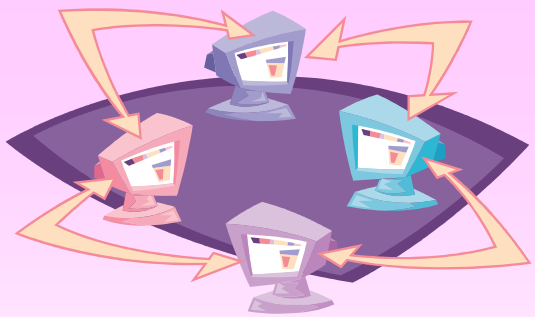


Percobaan 13

- Contoh pass by value

- **Tanggal.java**

```
public class Tanggal {
  int tgl=1;
  int bulan=1;
  int tahun=2001;
  public Tanggal(int Tgl,int Bulan,int Tahun)
  { tgl=Tgl; bulan=Bulan; tahun=Tahun; }
  public void setTgl(int Tgl) {
    tgl = Tgl;
  }
  public void setBulan(int Bulan) {
    bulan = Bulan;
  }
  public void setTahun(int Tahun) {
    tahun = Tahun;
  }
  public void cetak() {
    System.out.println("tanggal : "+tgl);
    System.out.println("bulan   : "+bulan);
    System.out.println("tahun   : "+tahun);
  }
}
```

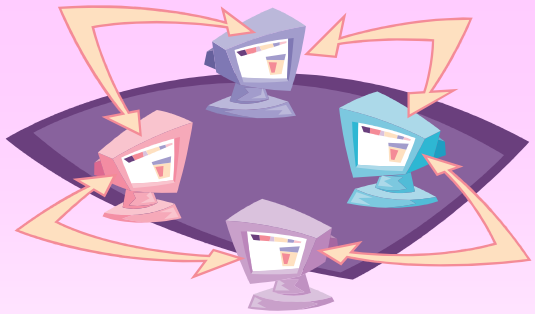


Percobaan 13

- Contoh pass by value

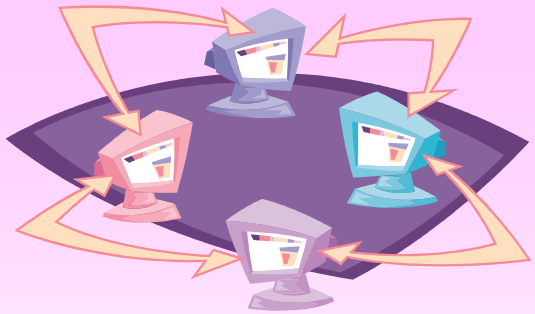
- TesPass.java

```
public class TesPass {
  public static void ubahInt(int nilai) {
    nilai = 55;
  }
  public static void ubahRefObyek(Tanggal ref) {
    ref = new Tanggal(22,2,2007);
  }
  public static void ubahAttrObyek(Tanggal ref) {
    ref.setTgl(23);
  }

  public static void main(String [] args) {
    Tanggal tanggalku;
    int val;
    // tandai nilai variabel
    val=11;
    // ubah nilainya dengan suatu method
    ubahInt(val);
    // periksa nilainya
    System.out.println("Nilai integer : "+val);
    // tandai nilai tanggal
    tanggalku = new Tanggal(1,1,2006);
    // ubah nilai referensi obyek
    ubahRefObyek(tanggalku);
    // tampilkan nilai tanggal
    tanggalku.cetak();
    // ubah nilai atribut melalui referensi
    ubahAttrObyek(tanggalku);
    // tampilkan nilai tanggal
    tanggalku.cetak();
  }
}
```



Keyword this

- Beberapa kegunaan dari keyword this :
  - Untuk merefer ke atribut dan method lokal
  - Keyword this tidak membedakan antara method lokal atau variabel constructor dengan variabel instance
  - Keyword this digunakan untuk melewatkan **current object** sebagai parameter ke method atau constructor yang lain

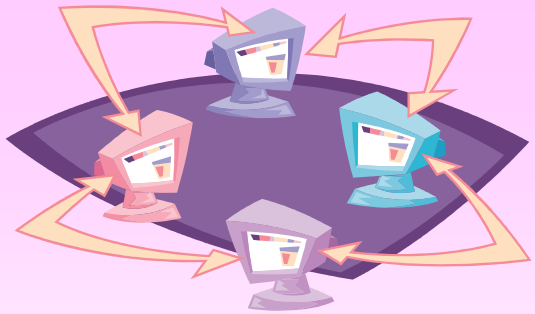


Percobaan 14

- Contoh penggunaan keyword this

Tanggal2.java

```
public class Tanggal2 {
 int tgl=1;
 int bulan=1;
 int tahun=2001;
 public Tanggal2(int Tgl, int Bulan, int Tahun) {
  this.tgl=Tgl; this.bulan=Bulan; this.tahun=Tahun; }
 public Tanggal2(Tanggal2 tanggalku) {
  this.tgl=tanggalku.tgl;
  this.bulan =tanggalku.bulan;
  this.tahun=tanggalku.tahun; }
 public Tanggal2 tambah_tgl(int banyaknya) {
  Tanggal2 tanggalbaru = new Tanggal2(this);
  tanggalbaru.tgl = tanggalbaru.tgl + banyaknya;
  return tanggalbaru;
 }
 public void cetak() {
  System.out.println("tanggal : "+tgl);
  System.out.println("bulan   : "+bulan);
  System.out.println("tahun   : "+tahun);
 }
}
```

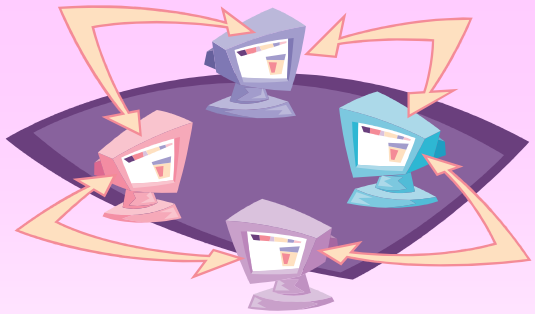


Percobaan 14 (lanjutan)

- Contoh penggunaan keyword this

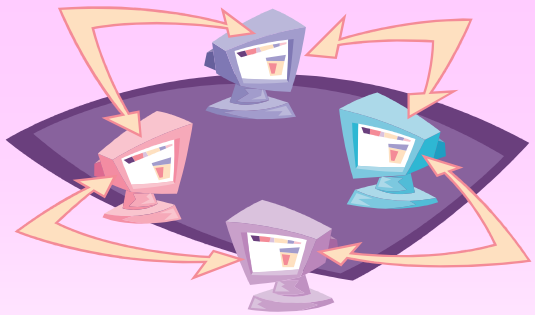
TesTanggal2.java

```
public class TesTanggal2 {
  public static void main(String [] args) {
    Tanggal2 tanggalku=new Tanggal2(14,9,70);
    Tanggal2 minggudepan=tanggalku.tambah_tgl(7);
    minggudepan.cetak();
  }
}
```



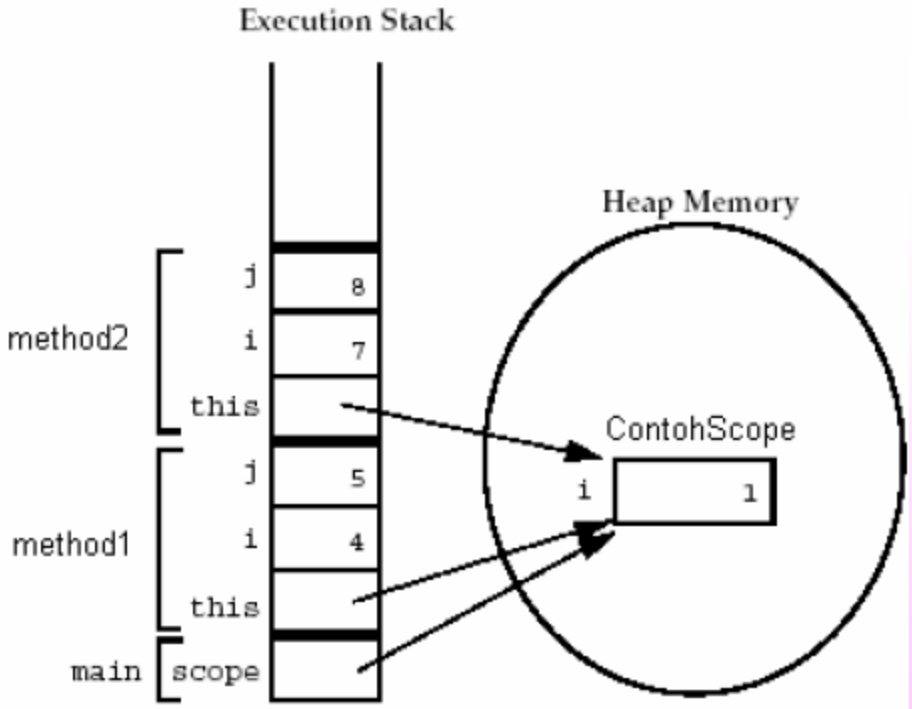
Definisi Variabel Lokal

- Variabel yang didefinisikan di dalam method
- Variabel diciptakan pada saat method dijalankan dan variabel dihapus pada saat keluar dari method
- Variabel harus diinisialisasi sebelum digunakan



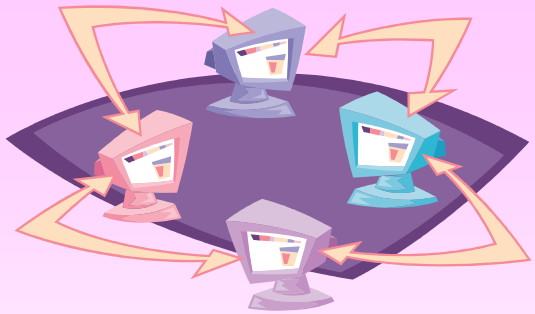
Percobaan 15

- Ruang lingkup variabel lokal



ContohScope.java

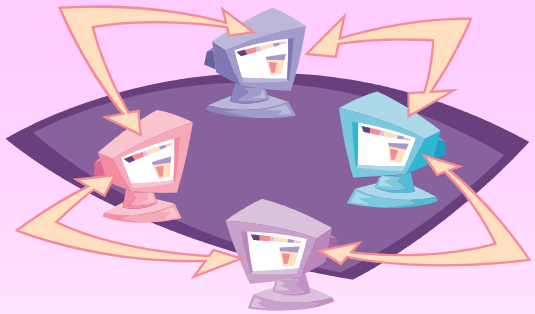
```
public class ContohScope {
   private int i=1;
   public void method1() {
      int i=4, j=5;
      this.i=i+j;
      System.out.println(i);
      System.out.println(j);
      System.out.println(this.i);
      method2(7);
      System.out.println(i);
      System.out.println(j);
      System.out.println(this.i);
    }
    public void method2(int i) {
       int j=8;
       this.i=i+j;
    }
}
class TesScope {
   public static void main(String [] args) {
      ContohScope scope = new ContohScope();
      scope.method1();
    }
}
```



Inisialisasi variabel by Java

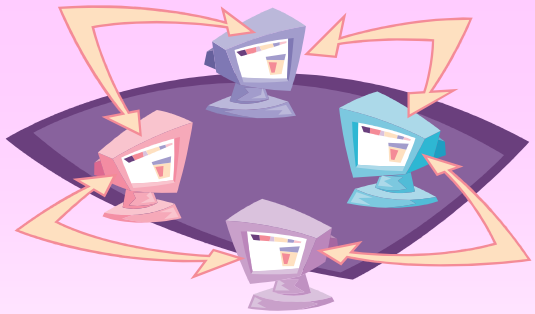
- Jika sebelumnya variable tidak terikat dengan nilai apapun (tidak diinisialisasi), kemudian variabel tersebut digunakan, maka variabel akan diinisialisasi secara otomatis oleh Java.

Variable	Value
byte	0
short	0
int	0
long	0L
float	0.0F
double	0.0D
char	'\u0000'
boolean	false
All reference types	null



Bentuk Operator

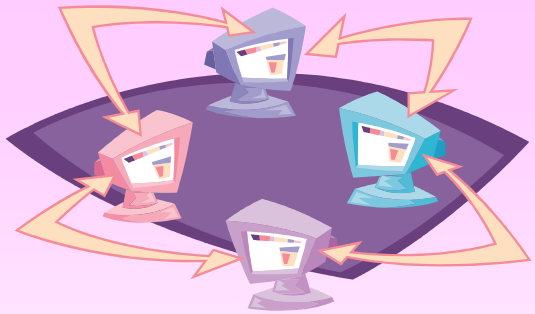
- Operator dapat digolongkan ke dalam dua bentuk yaitu : unary operator dan binary operator
- Unary operator adalah operator yang hanya melibatkan 1 operan
- Binary operator adalah operator yang melibatkan dua operan
- Sedangkan jenis operator dalam Java ada banyak macamnya, yaitu operator aritmatika, operator logika, operator bitwise, dll.



Operator dalam Java

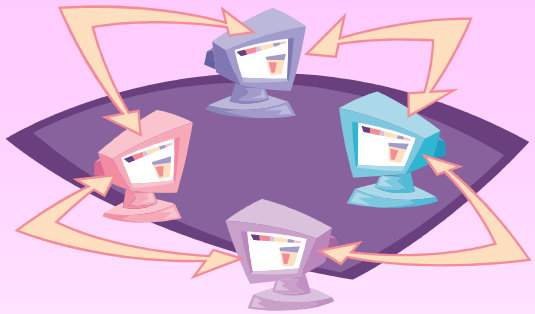
Separator	. [] () ; ,

Associative	Operators
R to L	++ -- + - ~ ! (data type)
L to R	* / %
L to R	+ -
L to R	<< >> >>>
L to R	< > <= >= instanceof
L to R	== !=
L to R	&
L to R	^
L to R	|
L to R	&&
L to R	||
R to L	?:
R to L	= *= /= %= += -= <<= >>= >>>= &= ^= |=



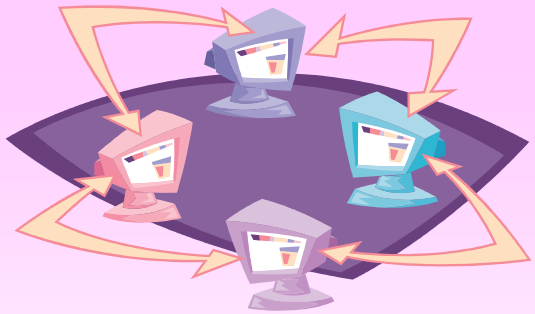
Operator Aritmatika

- Operator aritmatika adalah operator yang berfungsi untuk operasi aritmatika
- Yang termasuk dalam operator aritmatika adalah : +, - , *, /, % (modulus – sisa bagi)



Operator Increment - Decrement

- Operator Increment adalah operator yang digunakan untuk menaikkan satu nilai (--)
- Operator decrement adalah operator yang digunakan untuk menurunkan satu nilai (++)

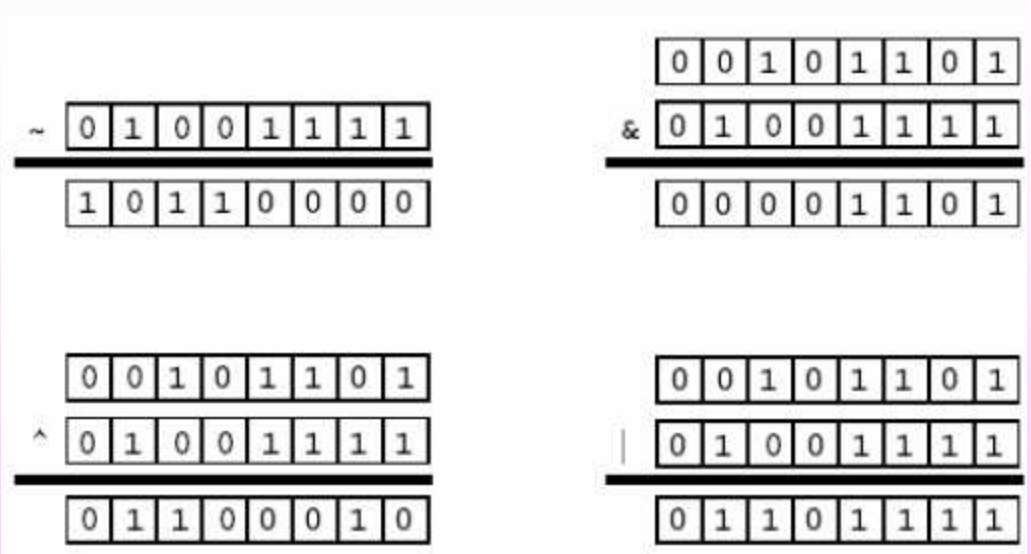


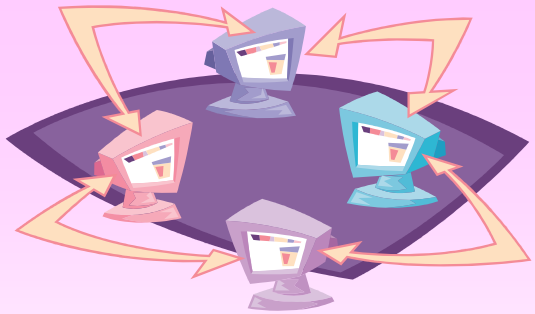
Operator bitwise

- Operator bitwise

```
~ - Complement        & - AND
^ - XOR               | - OR
```

- Contoh penggunaan :

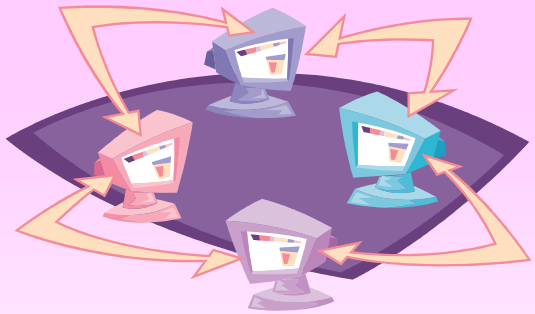




Percobaan 16

- Penggunaan Operator bitwise

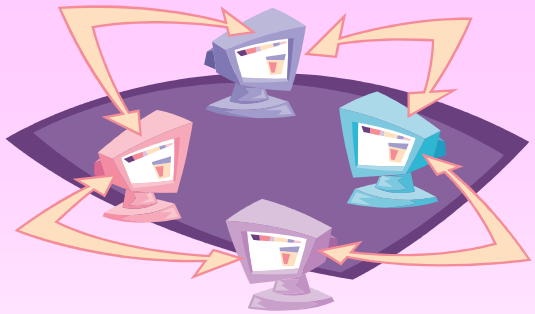
```
public class Complement {
    public static void main(String args[] ) {
        int i;
        i=~7;
        System.out.println(i);
    }
}
```



Operator Boolean

- Operator boolean adalah operator yang menghasilkan nilai true (benar) atau false (salah).

```
!  - NOT          &  - AND
|  - OR           ^  - XOR
```



Percobaan 17

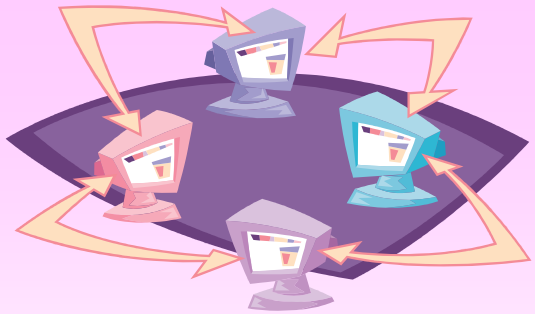
- Penggunaan Operator Boolean

ShortAnd.java

```
public class ShortAnd {
    public static void main(String args[] ) {
        int a=5, b=7;
        if ((a<2) && (b++<10)) b+=2;
           System.out.println(b);
    }
}
```

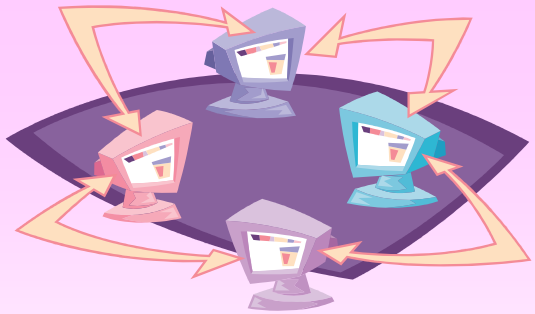
ShortOr.java *

```
public class ShortOr {
   public static void main(String args[] ) {
      int a=5, b=7;
      if ((a<2) || (b++<10)) b+=2;
         System.out.println(b);
   }
}
```



Operator Logika

- Operator logika adalah operator yang dipakai untuk operasi perbandingan dan selalu menghasilkan tipe boolean
- Yang termasuk operator logika : ==, !=, >, >=, <, <=

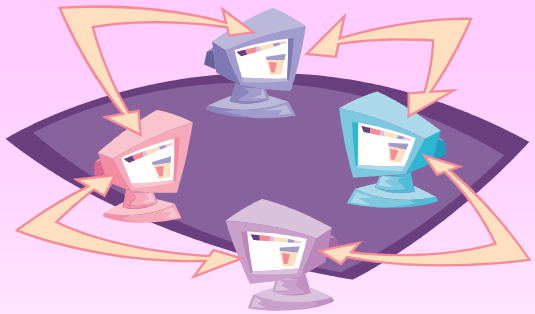


Operator shift

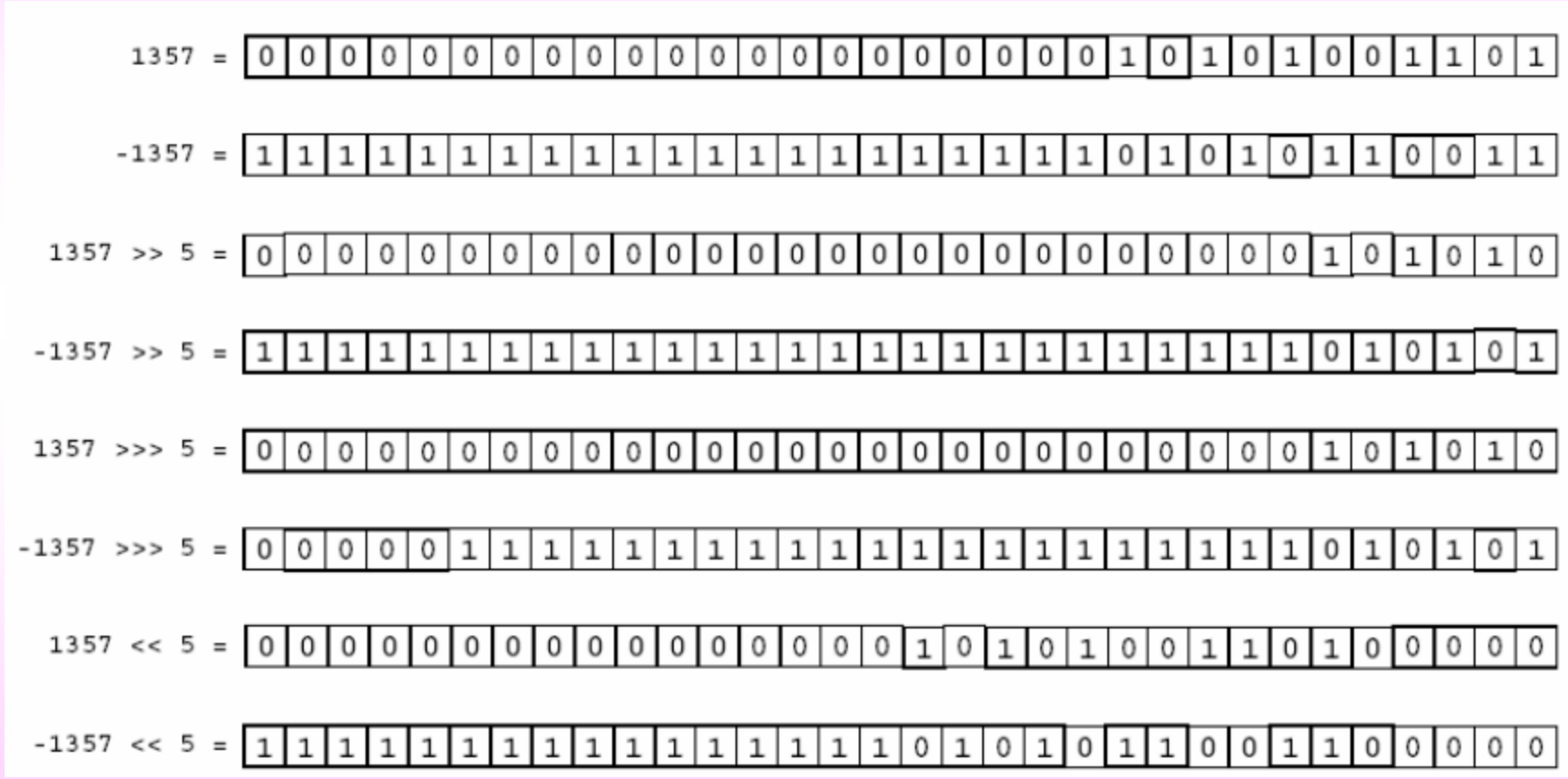
- Operator shift adalah operator yang berfungsi untuk menggeser susunan bit pada suatu nilai, yaitu >> (right shift), << (left shift)
- Contoh penggunaan :

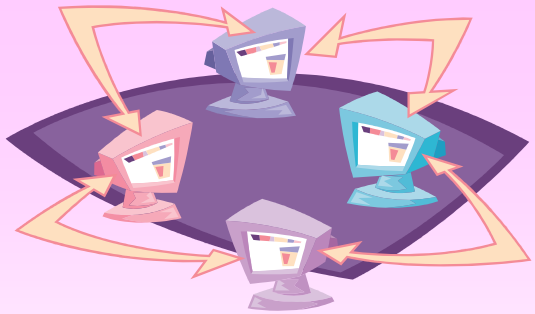
```
128 >> 1 returns 128/2^1 = 64
256 >> 4 returns 256/2^4 = 16
-256 >> 4 returns -256/2^4 = -16

128 << 1 returns 128 * 2^1 = 256
16  << 2 returns 16 * 2^2 = 64
```

- Operator >>> digunakan untuk :
  - Bit patterns
  - Sign bit tidak dikopi selama proses shift



Ilustrasi Operator shift

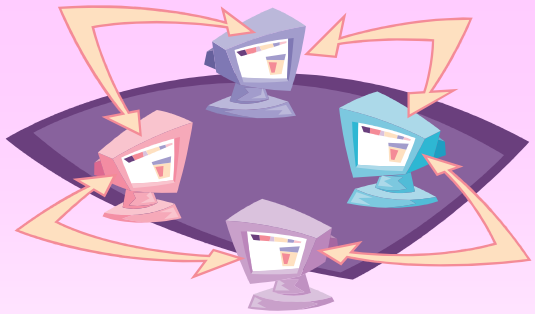




Percobaan 18

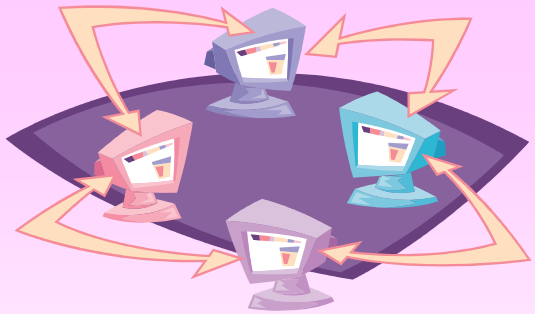
- Penggunaan Operator Shift

```
public class LeftShift {
    public static void main(String args[] ) {
        int i=3;
        i = i << 2;
        System.out.println(i);
    }
}
```



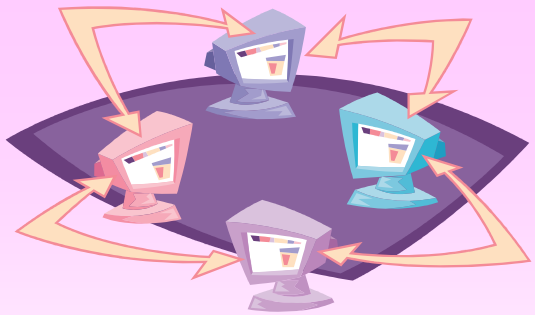
Operator Kombinasi

- Operator yang terdiri dari gabungan dua operator
- Contoh :
  - Operator += adalah gabungan dari operator = dan +



Operator Presedence

Operator	Precedence
() [] . ! ~ ++ -- + - (Data Type) * / % + - << >> >>> < <= > >= instance == != & ^ | && || ? : = += -= *= /= %= &= ^= |= <<= >>= >>>=	Tertinggi Terendah



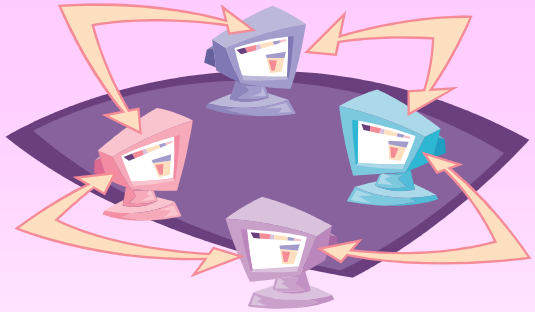
Ternary Operator

- Expr1 ? Expr2 : Expr3

```
if (x > y)  max = x;
else  max = y;
```

```
max = x > y ? x : y ;
```

```
m = a > b ? (c > a ? c : a) : (c > b ? c : b) ;
```



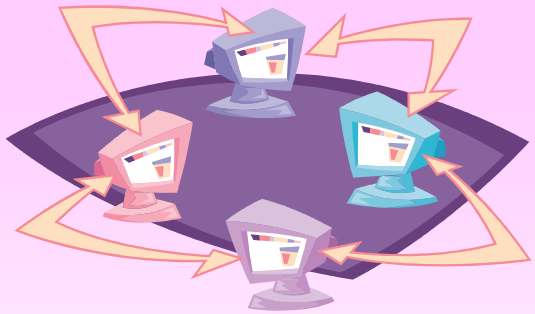
Percobaan 19

- Penggunaan Operator Ternary

ContohTernary.java *

```
public class ContohTernary {
    public static void main(String [] args) {
        int x=2,y=6,m;
        int a=1,b=2,c=3;
        m = x > y ? x : y;
        System.out.println(m);
        m = a > b ? (c > a ? c : a):(c > b ? c : b);
        System.out.println(m);
    }
}
```

Latihan

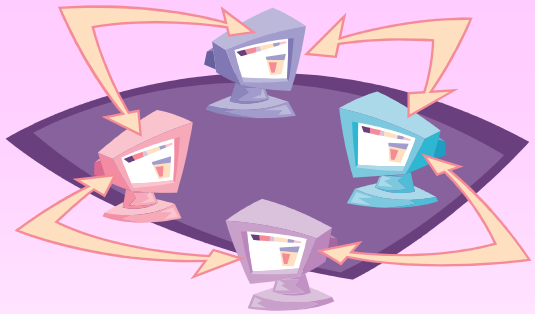


Latihan 1

- Modifikasi program berikut agar dapat menginputkan nilai tanggal, bulan dan tahun kemudian jika terjadi kesalahan pada input tanggal, program akan memunculkan pesan kesalahan.

Tanggal.java *

```
public class Tanggal {
        public int tgl;
        public int bulan;
        public int tahun;
        public Tanggal(int stgl, int sbulan, int stahun) {
           tgl=stgl; bulan=sbulan; tahun=stahun;
        }
        public static void main(String [] args) {
           Tanggal tanggalku = new Tanggal(32,3,2007);
           if ((tanggalku.tgl != '\0') && (tanggalku.tgl>31))
           {
              System.out.println("Ada yang salah dengan tanggal");
            }
        }
}
```



Latihan 2

- Jelaskan perbedaan antara kedua program berikut :

ShortAnd.java * | ShortAnd2.java

```
public class ShortAnd {
    public static void main(String args[] ) {
        int a=5, b=7;
        if ((a<2) && (b++<10)) b+=2;
           System.out.println(b);
    }
}
```

ShortAnd.java * | ShortAnd2.java *

```
public class ShortAnd2 {
    public static void main(String args[] ) {
        int a=5, b=7;
        if ((a<2) & (b++<10)) b+=2;
           System.out.println(b);
    }
}
```

selesai

24 – Maret – 2007 dan

3 – Mei – 2007