



DASAR PEMROGRAMAN

Tim Ajar Algoritma dan Struktur Data
Genap 2020/2021



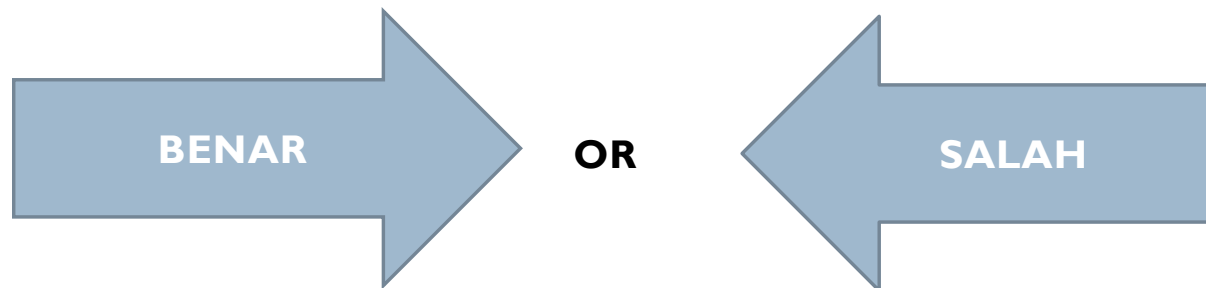
PEMILIHAN



PEMILIHAN

- Pemilihan(selection) adalah instruksi untuk yang dipakai untuk memilih satu kemungkinan dari beberapa kondisi

Kondisi : suatu pernyataan atau ekspresi (pernyataan logika)



Sintaks Pemilihan IF

► Bentuk Umum:

```
if (Kondisi)
{
    Pernyataan;
}
```

- ✓ apabila *kondisi* bernilai benar, maka *pernyataan* akan dilaksanakan.
- ✓ apabila *kondisi* bernilai salah, maka *pernyataan* tidak akan dilaksanakan.



Pemilihan if...Else

- ▶ Struktur pemilihan IF-ELSE minimal harus mempunyai 2 pernyataan. Jika **Kondisi** bernilai TRUE atau terpenuhi, maka **Pernyataan-1** akan dijalankan. Namun, jika **Kondisi** bernilai FALSE, maka **Pernyataan-2** yang akan dijalankan.
- ▶ Bentuk umum:

```
if (Kondisi)
{
    Pernyataan-1;
}
else
{
    Pernyataan-2;
}
```



Pemilihan If...else if...else

► Bentuk umum:

```
If ( kondisi 1 )
{
    pernyataan-1;
}
else if ( kondisi 2)
{
    pernyataan-2;
}
else if ( kondisi 3)
{
    pernyataan-3;
}
.....
.....
else if ( kondisi X)
{
    pernyataan-X;
}

Else
{
    pernyataan;
}
```

Pada bentuk if...else if...else

- pernyataan 1 akan dijalankan apabila “kondisi 1” bernilai BENAR.
- Jika “kondisi 1” bernilai SALAH, maka akan dicek “kondisi 2”.
- Jika “kondisi 2” BENAR maka akan dijalankan statement 2, begitu seterusnya.
- Dan apabila tidak ada satupun syarat yang terpenuhi, barulah statement else terakhir yang akan dikerjakan.

Contoh

```
public static void main(String[] args) {
    Scanner input= new Scanner (System.in);

    int bayar;

    System.out.println("Masukkan total belanja anda: ");
    bayar=input.nextInt();

    if(bayar>=2000000){
        System.out.println("Selamat anda mendapatkan hadiah kompor gas");
    }
    else if(bayar>=1000000){
        System.out.println("Selamat anda mendapatkan hadiah teflon");
    }
    else if (bayar>=500000) {
        System.out.println("Selamat anda mendapatkan hadiah piring");
    }
    else{
        System.out.println("Maaf anda belum beruntung, tingkatkan belanja anda!");
    }

}
```



Pemilihan SWITCH-CASE

► Bentuk Umum:

```
switch (Kondisi)
{
    case Konstanta-1:
        Pernyataan-1;
        break;
    case Konstanta-2:
        Pernyataan-2;
        break;
    ...
    ...
    case Konstanta-n:
        Pernyataan-n;
        break;
    default:
        Pernyataan;
}
```

Sintaks pemilihan ini akan menjalankan salah satu dari beberapa pernyataan “case” sesuai dengan nilai kondisi yang ada di dalam “switch”. Selanjutnya proses akan dilanjutkan sampai ditemukan pernyataan “break”.

Contoh

```
public static void main(String[] args){
    Scanner sc = new Scanner(System.in);
    int angka;

    System.out.print("Masukkan angka: ");
    angka = sc.nextInt();

    switch(angka){
        case 1:
            System.out.println("Hari Senin");
            break;
        case 2:
            System.out.println("Hari Selasa");
            break;
        case 3:
            System.out.println("Hari Rabu");
            break;
        case 4:
            System.out.println("Hari Kamis");
            break;
        case 5:
            System.out.println("Hari Jumat");
            break;
        case 6:
            System.out.println("Hari Sabtu");
            break;
        case 7:
            System.out.println("Hari Minggu");
            break;

        default:
            System.out.println("Maaf, angka yang Anda masukkan salah");
    }
}
```



Operator Ternary

- ▶ Digunakan dalam Sintaks pemilihan
- ▶ Bentuk Umum:

sintaks (Condition) ? (kondisi jika true) : (kondisi jika false)

```
public static void main(String[] args) {  
  
    Double angka = 5.5;  
    String hasil;  
    hasil = (angka > 0.0) ? "Bilangan positif" : "Bilangan Negatif";  
    System.out.println(angka + " adalah " + hasil);  
}
```

Pemilihan Bersarang

► Bentuk Umum

```
if (kondisi 1){  
    if (kondisi 2){  
        pernyataan 1;  
        ...  
        ...  
        if (kondisi n){  
            pernyataan 2;  
        } else {  
            pernyataan  
3;  
        }  
    } else {  
        pernyataan n;  
    }  
} else {  
    pernyataan x;  
}
```

➤ Kondisi yang akan diseleksi pertama kali adalah kondisi IF yang berada di posisi terluar (kondisi 1).

➤ Jika kondisi 1 bernilai salah, maka pernyataan ELSE terluar (pasangan dari IF yang bersangkutan) yang akan diproses.

➤ Jika ternyata kondisi 1 bernilai benar, maka kondisi berikutnya yang lebih dalam (kondisi 2) akan diseleksi. Jika kondisi 2 bernilai salah, maka pernyataan ELSE (pasangan dari IF yang bersangkutan) yang akan diproses.

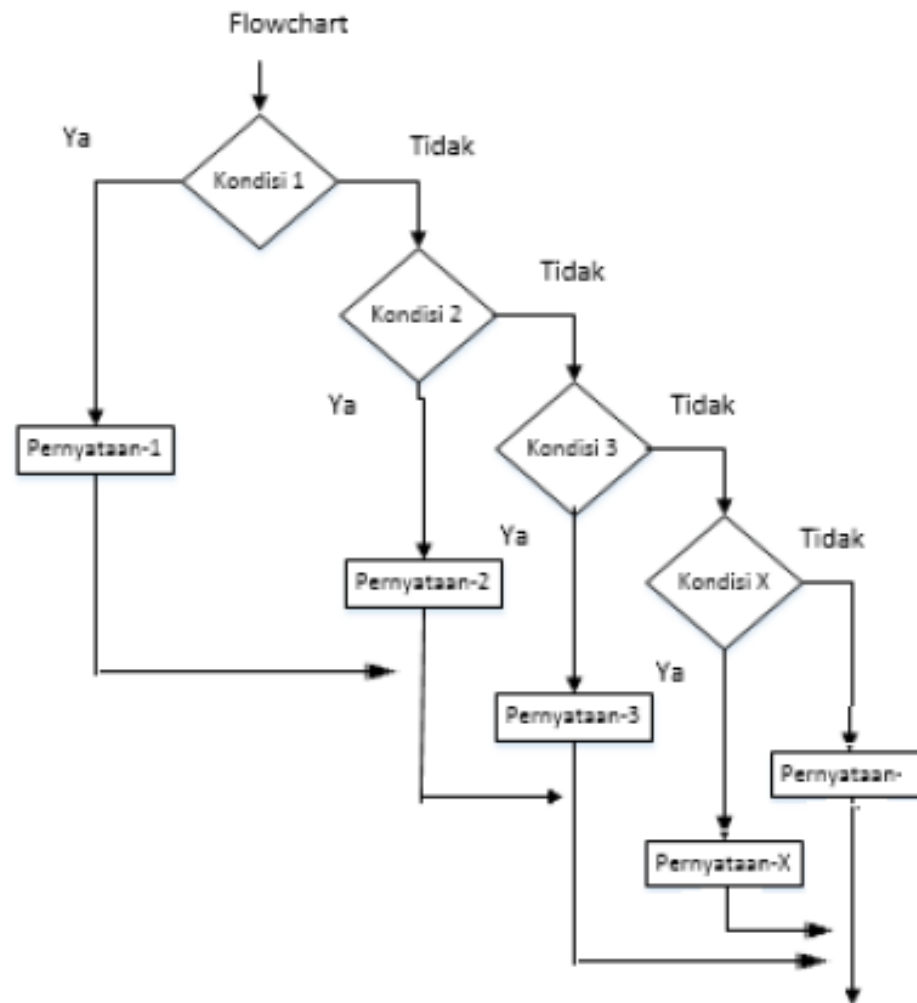


Contoh

```
import java.util.Scanner;

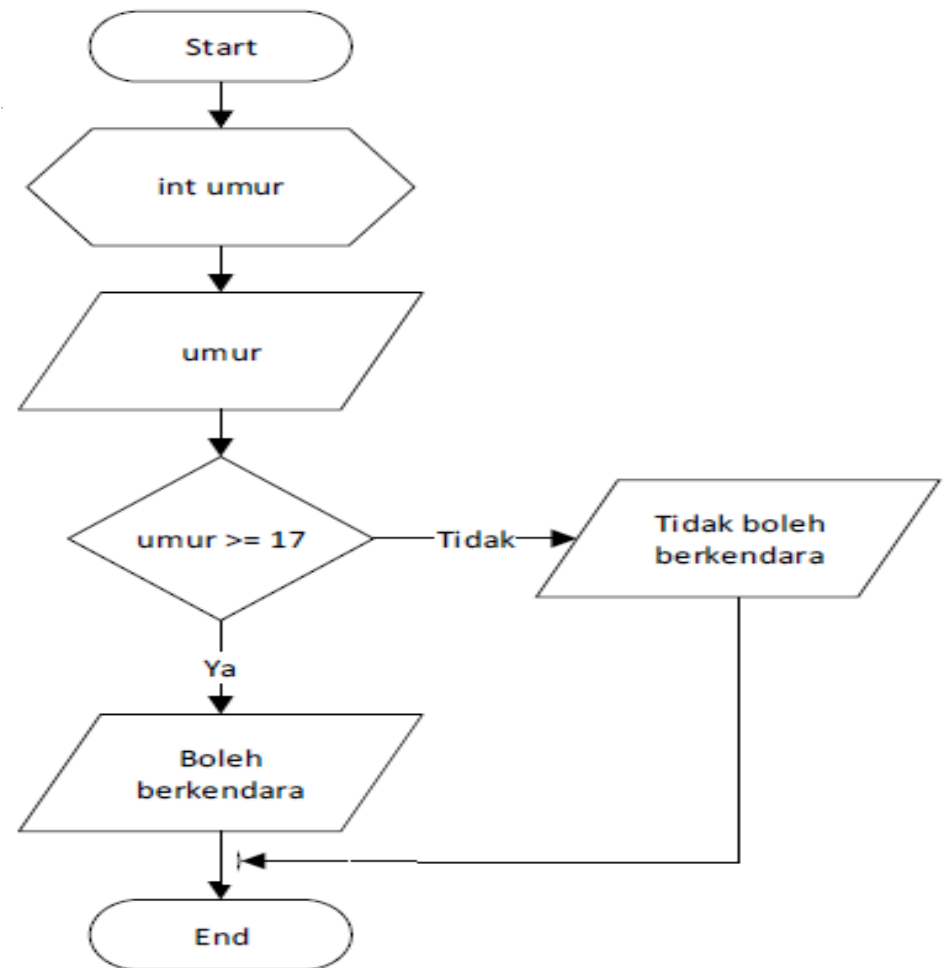
public class kasir {
    public static void main(String[] args) {
        int total, diskon, bayar;
        String kartu;
        Scanner sc = new Scanner (System.in);
        System.out.print("Apakah pelanggan mempunyai kartu anggota (y atau t)? ");
        kartu = sc.nextLine();
        System.out.print("Berapa total harga barang belanjaan? Rp ");
        total = sc.nextInt();
        if (kartu.equals("y")) {
            if (total > 500000) {
                diskon = 50000;
            } else {
                diskon = 25000;
            }
        } else {
            if (total > 200000) {
                diskon = 10000;
            } else {
                diskon = 0;
            }
        }
        bayar = total - diskon;
        System.out.println("Total yang harus dibayar: Rp " + bayar);
    }
}
```

FLOWCHART PEMILIHAN



Contoh Flowchart

- ▶ Didalam aturan tata tertib berkendara kendaraan bermotor maka terdapat aturan dimana orang yang boleh berkendara bermotor yaitu orang yang umurnya minimal 17 tahun. Buatlah flowchart untuk mengecek apakah umur seseorang diperbolehkan berkendara kendaraan bermotor!





PERULANGAN



Perulangan For

- **Inisialisasi nilai** adalah tempat dimana kita akan memberikan nilai awal pada variabel counter (variabel yang digunakan untuk menghitung jumlah perulangan).
- **Syarat perulangan** adalah syarat yang harus dipenuhi agar perulangan tetap dilakukan.
- **Perubahan nilai** adalah perubahan yang akan dilakukan pada tiap putaran.

```
for (inisialisasi; kondisi; iterasi) {  
    //statement yang akan diulang  
}
```



Contoh

- ▶ Program untuk menjumlahkan 5 bilangan positif pertama.

```
public class DemoFor3 {  
  
    public static void main(String[] args) {  
        int n = 5; // 5 bilangan positif pertama  
  
        int hasil = 0;  
        for (int i = 1; i <= n; i++) {  
            hasil = hasil + i;  
            if (i != n) {  
                System.out.print(i + " + ");  
            } else {  
                System.out.print(i + " = ");  
            }  
        }  
        System.out.println(hasil);  
    }  
}
```

```
Compile Single.  
run-single:  
1 + 2 + 3 + 4 + 5 = 15  
BUILD SUCCESSFUL (total time: 1 second)  
|
```

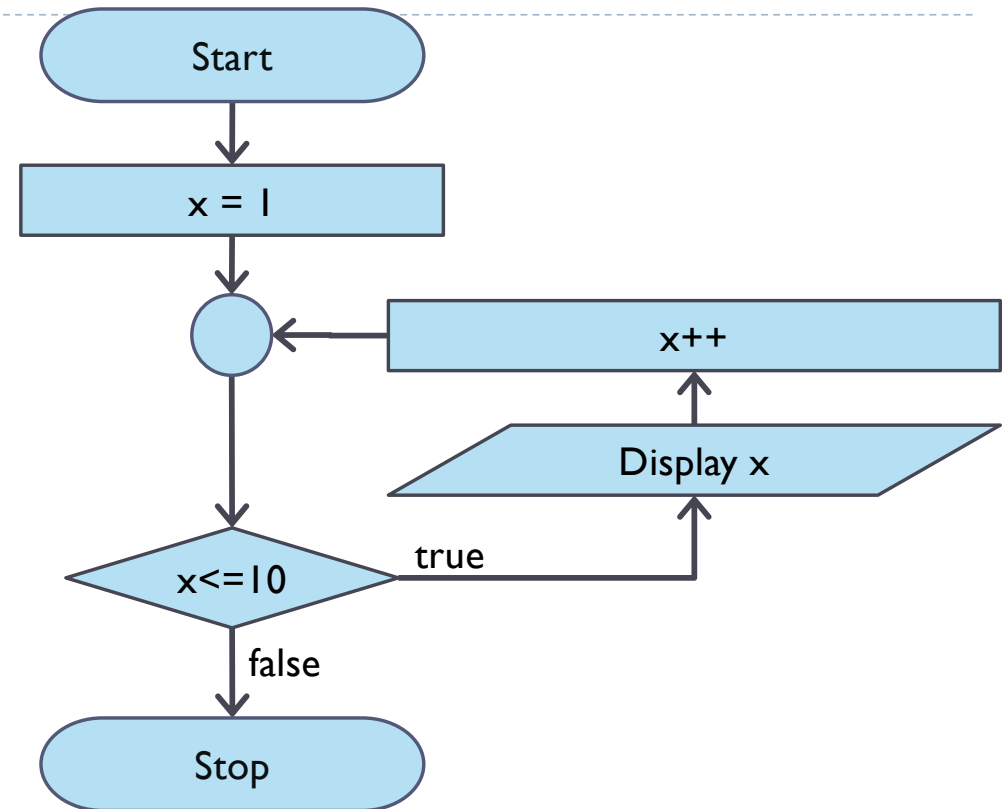
Contoh Flowchart

Buatlah program untuk mencetak ke layar angka 1 sampai 10.

```
int x;  
for (x=1; x<=10; x++)  
System.out.printf("%d\n", x);
```

Output:

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10
```



Perulangan while

- Syntax dari perintah while():

```
inisialisasi
while (kondisi) {
    //statement yang diulang
    iterasi
}
```

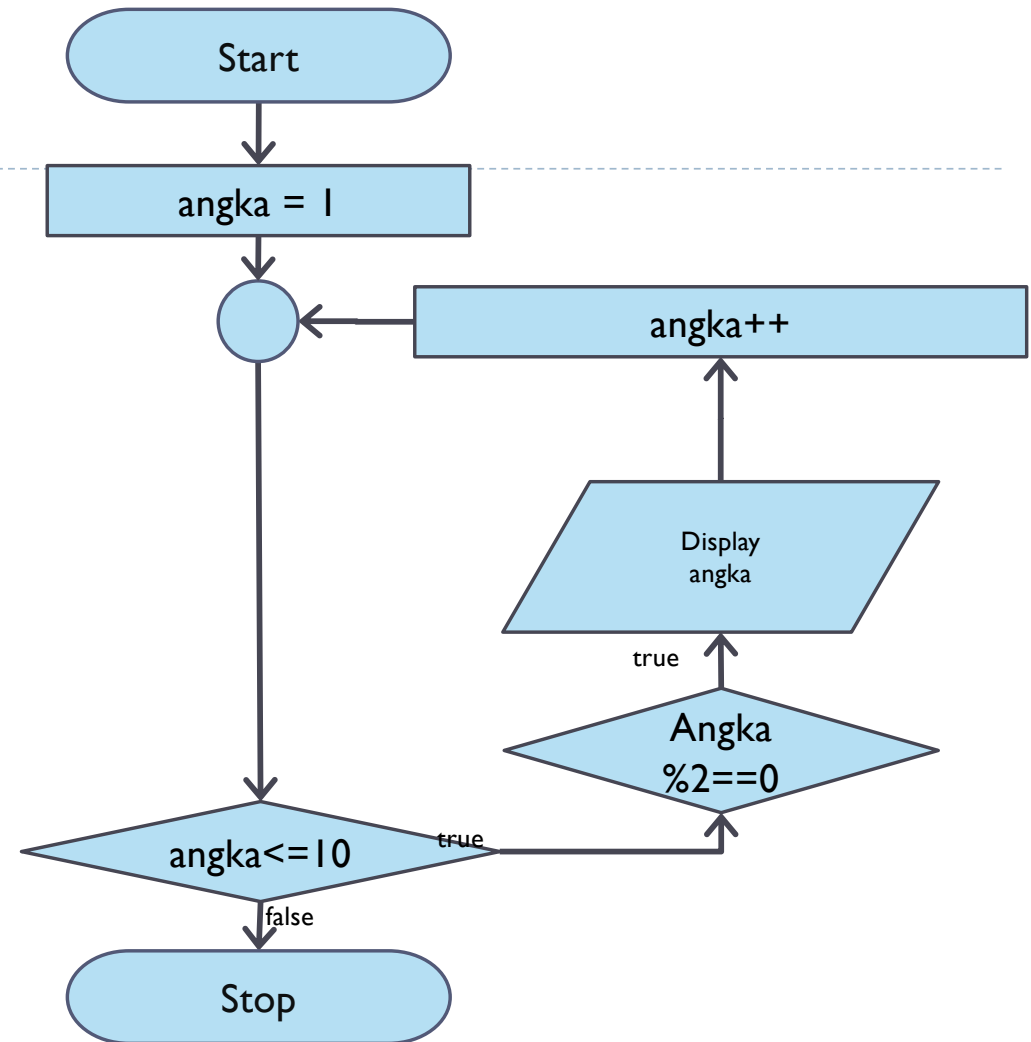
- Syarat perulangan adalah syarat yang harus dipenuhi agar perulangan tetap dilakukan
- Perulangan while akan terus dijalankan selama syarat perulangan bernilai TRUE



Contoh

- mencetak bilangan genap dari 2 sampai 10

```
public static void main(String[] args) {  
    int angka;  
    angka = 1;  
  
    while (angka <= 10)  
    {  
        if (angka%2==0)  
        {  
            System.out.println (angka);  
        }  
        angka ++;  
    }  
}
```



Perulangan Do-while

- ▶ perintah `do-while()` akan menjalankan statementnya sebanyak satu kali, meskipun syarat pengulangan tidak terpenuhi.

```
inisialisasi  
do {  
  //statement yang akan diulang  
  ...  
  iterasi  
} while (kondisi);
```



Contoh

► Mencetak Angka 0 - 3

```
public class demoDoWhile {  
    public static void main(String[] args) {  
        int i = 0;  
        do {  
            System.out.println(i);  
            i++;  
            if (i == 4) {  
                break;  
            }  
        } while (i < 10);  
    }  
}
```

```
0  
1  
2  
3  
BUILD SUCCESSFUL (total time: 1 second)|
```



Perulangan Bersarang (1)

► “for bersarang”

```
1  for (int i = 0; i < n; i++) {    // loop level 1
2      for (int j = 0; j < n; j++) { // loop level 2
3          for (int k = 0; k < n; i++) { // loop level 3
4              // statement
5          }
6      }
7      for (int l = 0; l < n; l++) { // loop level 2
8          // statement
9      }
10 }
```

Outer loop

Inner loop

Perulangan Bersarang (2)

► “while bersarang”

Outer loop

```
1  int i = 0;
2  // pengecekan loop. Selama kondisi (i < n) bernilai true, loop terus berjalan
3  while (i < n) {      // loop level 1
4      int j = 0;
5
6      // pengecekan loop. Selama kondisi (j < n) bernilai true, loop terus berjalan
7      while (j < n) {   // loop level 2
8          // statement
9          j++;
10     }
11     i++;
12 }
```

Inner loop

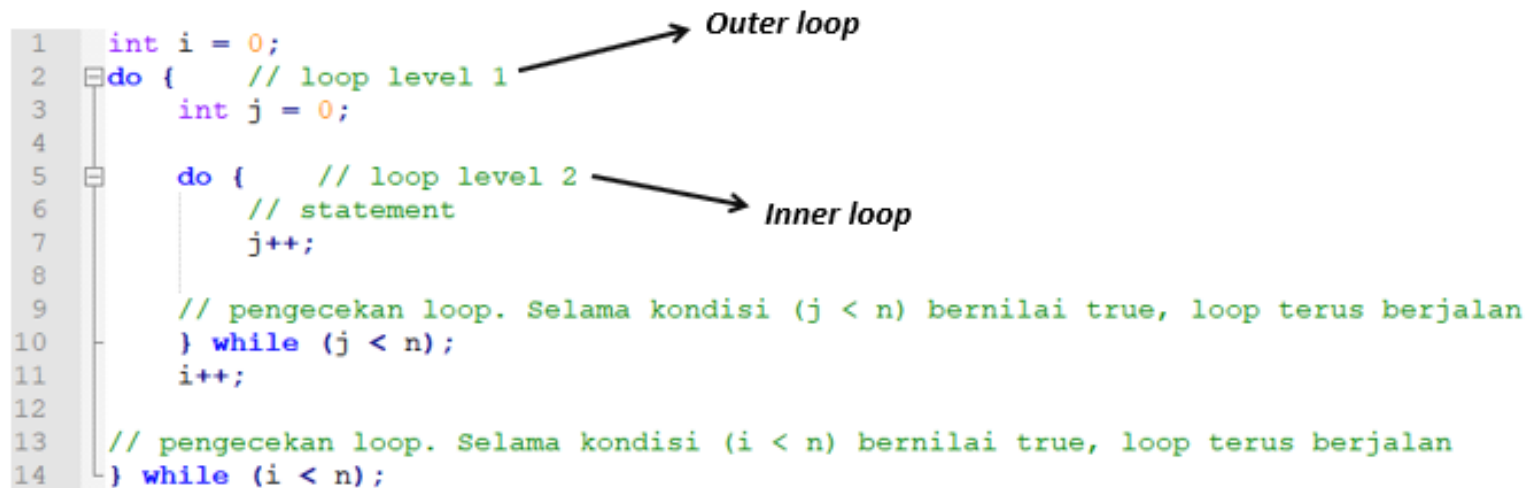
Perulangan Bersarang (3)

► “do-while bersarang”

```
1  int i = 0;
2  do {    // loop level 1
3      int j = 0;
4
5      do {    // loop level 2
6          // statement
7          j++;
8
9          // pengecekan loop. Selama kondisi (j < n) bernilai true, loop terus berjalan
10         } while (j < n);
11         i++;
12
13         // pengecekan loop. Selama kondisi (i < n) bernilai true, loop terus berjalan
14     } while (i < n);
```

Outer loop

Inner loop

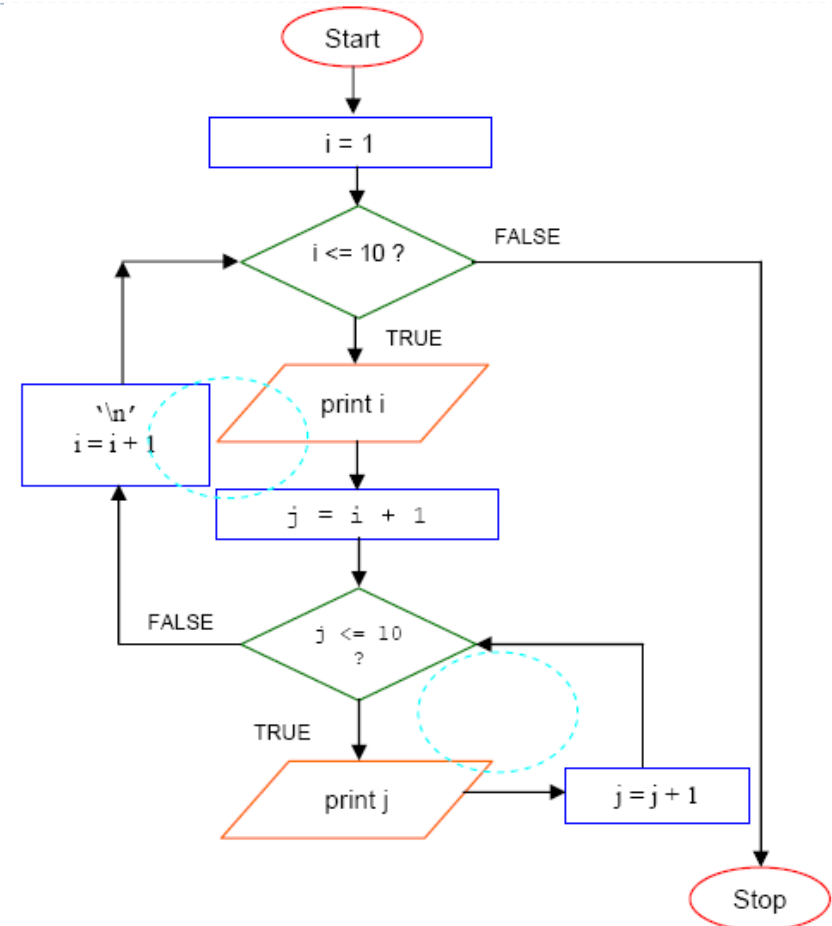


Contoh Flowchart Nested LOOP

```
public class OpTriangle {  
  
    public static void main(String[] args) {  
        for (int i = 1; i <= 10; i=i+1) {  
            System.out.print(i);  
            for (int j = i + 1; j <= 10; j=j+1) {  
                System.out.print(j);  
            }  
            System.out.println("");  
        }  
    }  
}
```

```
run-single:  
12345678910  
2345678910  
345678910  
45678910  
5678910  
678910  
78910  
8910  
910  
10
```

BUILD SUCCESSFUL (total time: 1 second)





ARRAY

The diagram consists of two horizontal rectangular boxes. The top box has a dark blue vertical bar on its left side and the word 'ARRAY' in a serif font on its right side. The bottom box has a light blue vertical bar on its left side and is otherwise empty.

Array Satu Dimensi

► Deklarasi

type namaArray[];

Atau

type[] namaArray;

Contoh : int a[]; int[] a;

- **Type** adalah tipe data dari array yang akan dibuat.
- **namaArray** adalah nama dari array yang akan dibuat.



Array Satu Dimensi

- ▶ Instansiasi objek array:

- ▶ Ketika sebuah array dideklarasikan, hanya referensi dari array yang dibuat. untuk alokasi memori dilakukan dengan menggunakan kunci kata *new*
- ▶ Cara Instansiasi variabel array:

namaArray = new tipe[jumlah elemen];

contoh: a = new int[10];



Array Satu Dimensi

- ▶ Deklarasi dan instansiasi objek array dapat digabungkan dalam sebuah instruksi sbb.:

type[] namaArray = new *type*[jumlah_elemen];

atau

type namaArray[] = new *type*[jumlah_elemen];

- ▶ Contoh :

- ▶ `int[] a = new int[10];`

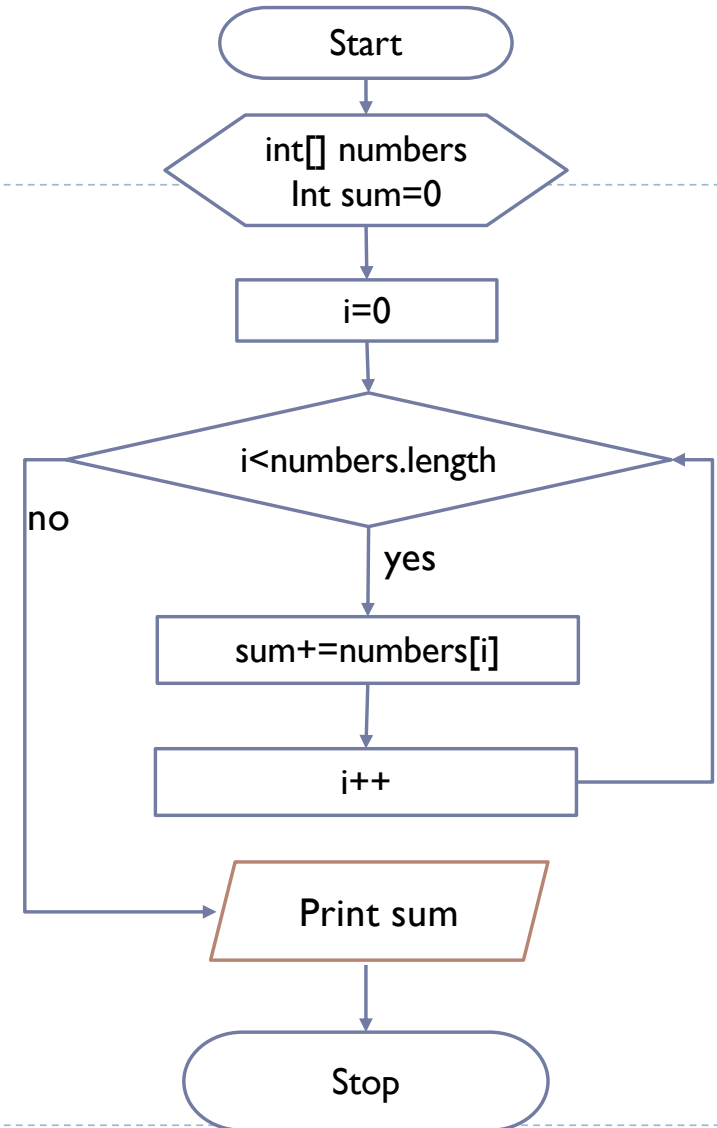
- atau*

- ▶ `int a[] = new int[10];`



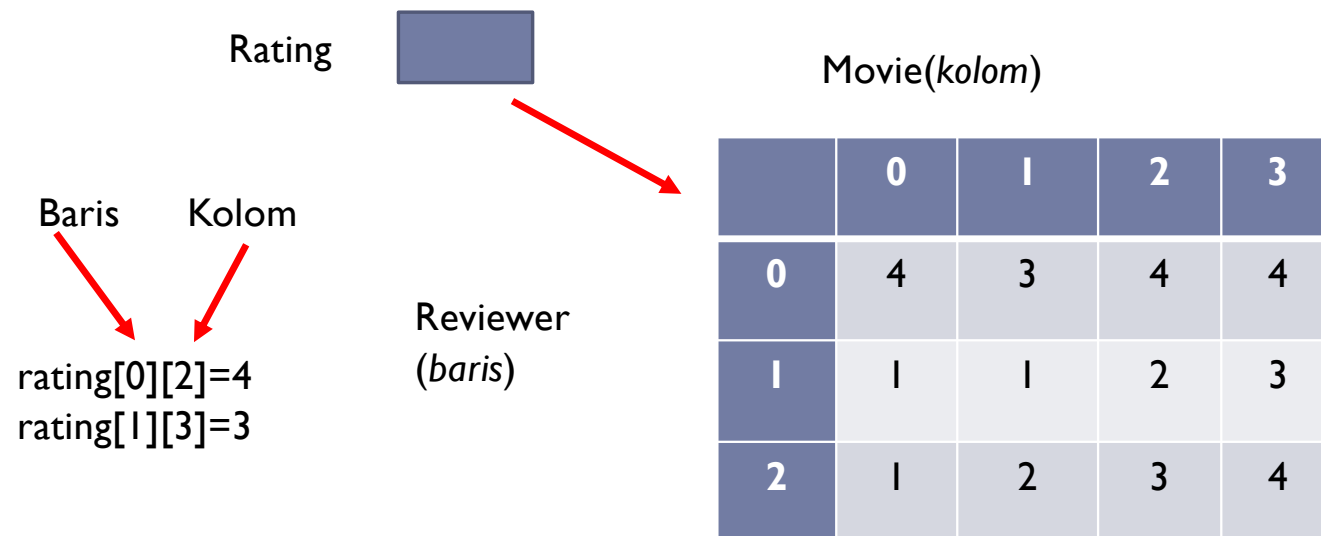
Contoh Array

```
public static void main(String[] args) {  
    int[] numbers = {1,2,3,4,5};  
    int sum = 0;  
    for (int i = 0; i < numbers.length; i++) {  
        sum += numbers[i];  
    }  
    System.out.println(sum);  
}
```



Array 2 Dimensi(2)

- ▶ Array 2 dimensi adalah sebuah array yang penomoran indeksinya menggunakan 2 angka, satu untuk baris dan satu lagi untuk kolom
- ▶ Contoh



Mendeklarasikan Array 2D

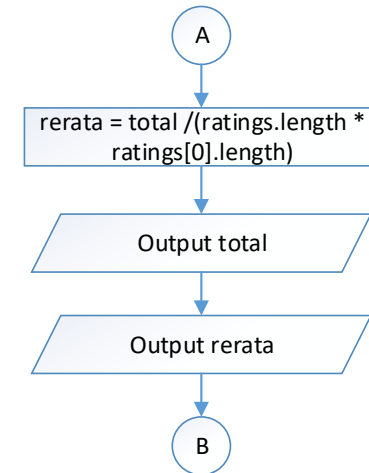
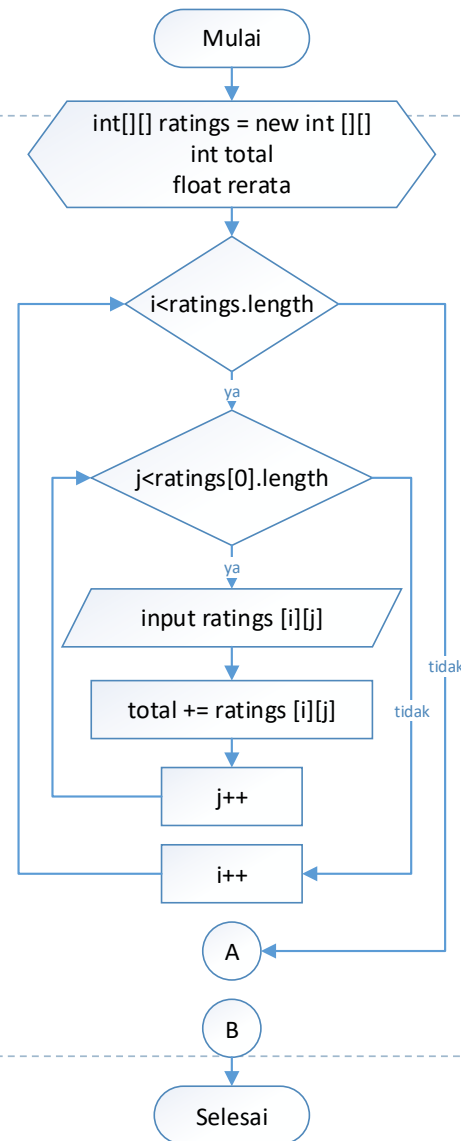
- ▶ Untuk mendeklarasikan variable array 2D, sama dengan array 1D. Hanya berbeda dengan jumlah kurung sikunya “[]”
- ▶ Bentuk umumnya

```
data_type[][] array_name = new data_type[x][y];  
x = jumlah baris  
Y = jumlah kolom  
Contoh  
int[][] arr = new int[10][20];
```



Contoh Flowchart

- Buatlah flowchart untuk menghitung rata-rata Array 2 Dimensi pada tabel rating film yang terdiri dari 3 baris (penonton pemberi rating) dan 4 kolom (judul film)!



Contoh kode program

► Menghitung rata-rata Array 2D

```
int[][] ratings = new int[3][4];
Scanner scanner = new Scanner(System.in);
int total = 0;
for (int i = 0; i < ratings.length; i++) {
    for (int j = 0; j < ratings[0].length; j++) {
        System.out.print("Masukan array[" + i + "," + j + "]: ");
        ratings[i][j] = scanner.nextInt();
        total += ratings[i][j];
    }
}

for (int rate[] : ratings) {
    for (int i : rate) {
        System.out.print(i + " ");
    }
    System.out.println();
}

float rerata = (float) total / (ratings.length * ratings[0].length);
System.out.println("Nilai total : " + total);
System.out.println("Nilai rerata: " + rerata);
```

```
Masukan array[0,0]: 1
Masukan array[0,1]: 2
Masukan array[0,2]: 3
Masukan array[0,3]: 4
Masukan array[1,0]: 2
Masukan array[1,1]: 3
Masukan array[1,2]: 4
Masukan array[1,3]: 4
Masukan array[2,0]: 3
Masukan array[2,1]: 2
Masukan array[2,2]: 3
Masukan array[2,3]: 4
1 2 3 4
2 3 4 4
3 2 3 4
Nilai total : 35
Nilai rerata: 2.9166667
```

Nilai rata-rata didapatkan dari penjumlahan seluruh nilai elemen matriks dibagi dengan hasil

► kali ukuran array baris dan kolom



FUNGSI



DEKLARASI FUNGSI

```
static TypeDataKembalian namaFungsi() {  
    // statement  
    //statement  
}
```

Keterangan:

- ▶ **Static** : Jenis fungsi yang dibuat bersifat static, agar dapat secara langsung di panggil di fungsi main yang juga bersifat static
- ▶ **TypeDataKembalian**: tipe data dari nilai yang dikembalikan (*output*) setelah fungsi dieksekusi
- ▶ **namaFungsi()**: nama fungsi yang dibuat



Fungsi

Contoh :

Pembuatan Fungsi:

```
static void berisalam() {  
    System.out.println("Halo! Selamat Pagi");  
}
```

Pemanggilan Fungsi:

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



Fungsi

- ▶ Fungsi **void tidak memerlukan return.**
- ▶ Fungsi yang memiliki tipe data fungsi **selain void memerlukan return.**
- ▶ **Nilai yang di-return-kan** dari suatu fungsi harus **sesuai dengan tipe data fungsi.**

- ▶ Variabel Lokal: variabel yang dideklarasikan dalam suatu fungsi, dan hanya bisa **diakses** atau dikenali dari **dalam fungsi itu sendiri.**
- ▶ Variabel Global: Variabel yang dideklarasikan di **luar blok fungsi**, dan bisa diakses atau **dikenali dari fungsi manapun.**



Fungsi yang mengembalikan Nilai

Pembuatan Fungsi dengan parameter dan return value:

```
static int luasPersegi(int sisi){  
    int luas = sisi * sisi;  
    return luas;  
}
```

Pemanggilan Fungsi dan memberi nilai parameter:

```
System.out.println("Luas Persegi dengan sisi 5 = " + luasPersegi(5));  
  
int luasan = luasPersegi(6);
```



Contoh Fungsi

```
public class Fungsi1 {
```

```
    static int a = 10, b = 5;  
    static double c;
```

Variabel Global

```
    static int Kali() {  
        int hasilKali = a * b;  
        return hasilKali;
```

Variabel Lokal

```
    }
```

```
    static void Tambah() {
```

Variabel Lokal

```
        int hasilTambah = a + b;  
        System.out.println("Hasil Tambah adalah " + hasilTambah);  
    }
```

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

```
        System.out.println("Hasil Kali adalah " + Kali());  
        Tambah();
```

```
    }
```

```
}
```

Variable Length Argument (Varargs).

- ▶ Varargs dipakai dengan cara menempatkan parameter-parameter dalam sebuah array dan array tsb yang akan menjadi parameter dari fungsi.
- ▶ Apabila tidak diketahui berapa jumlah dari parameter suatu fungsi dengan pasti, kita bisa menggunakan **Variable Length Argument** (Varargs).
- ▶ Deklarasi:

```
accessModifier methodName(datatype... arg) {  
    // method body  
}
```



Contoh Varargs

```
static void tampil(String str, int... a) {  
    System.out.println("parameter string: " + str);  
    System.out.println("jumlah parameter: " + a.length);  
  
    for (int i : a) {  
        System.out.print(i + " ");  
    }  
  
    System.out.println();  
}  
  
public static void main(String[] args) {  
    tampil("daspro", 100, 200);  
    tampil("daspro2", 1, 2, 3, 4, 5);  
    tampil("daspro3");  
}
```



Fungsi Rekursif



- Biasanya sebuah fungsi akan dipanggil (di-CALL) oleh fungsi lain
- Pada fungsi rekursif, di dalam sebuah fungsi terdapat perintah untuk memanggil fungsi itu sendiri (dirinya sendiri). Dengan demikian, proses pemanggilan fungsi akan terjadi secara berulang-ulang
- Bentuk umum:

```
static tipe_data_kembalian nama_fungsi  
(parameter) {  
    ...  
    nama_fungsi (...)  
    ...  
}
```

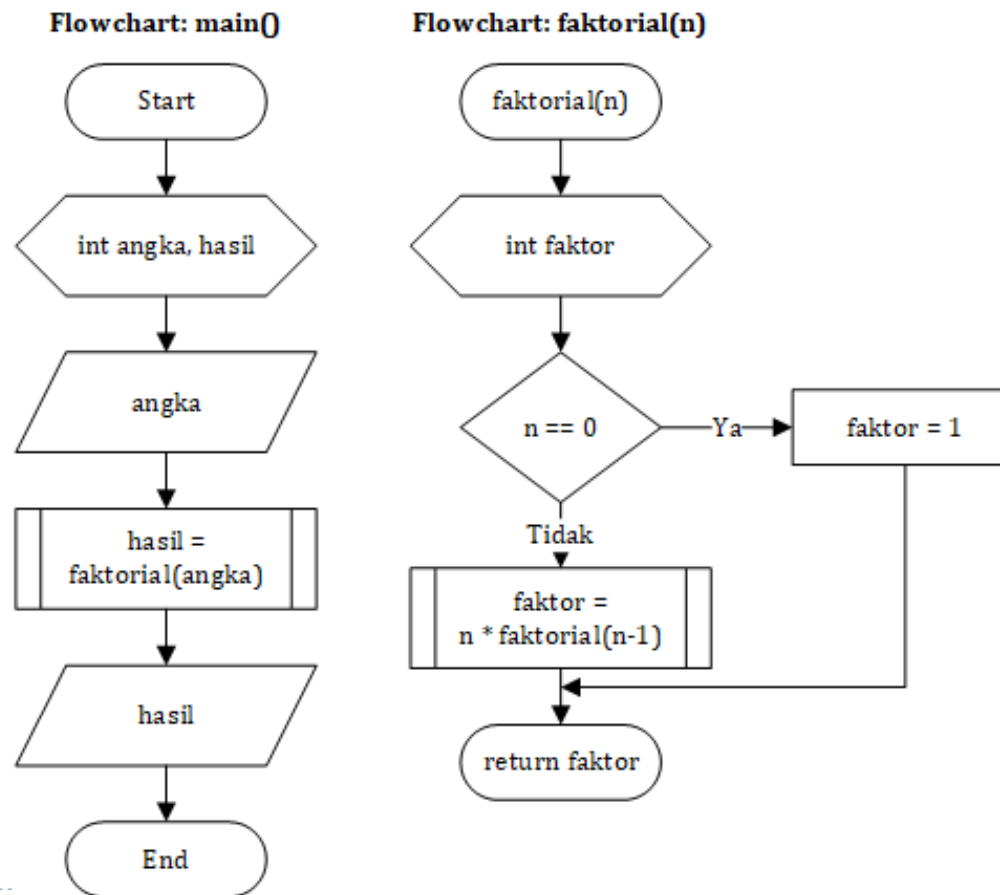


Fungsi faktorial

- Base case: $n = 0$
- Recursion call: $f(n) = n * f(n-1)$

```
public class faktorial {  
  
    public static void main(String[] args) {  
        System.out.println(faktorialRekursif(5));  
    }  
  
    static int faktorialRekursif(int n) {  
        if (n == 0) {  Base case  
            return (1);  
        } else {  
            return (n * faktorialRekursif(n - 1));  Recursion call  
        }  
    }  
}
```

Flowchart fungsi faktorial



LATIHAN

Buatlah flowchart untuk menyelesaikan permasalahan berikut ini :

1. Menampilkan deretan bilangan ganjil dari angka 1 sampai 20 kecuali angka 11 dan 17
 1. Contoh : 1 3 5 7 9 13 15 19
2. Membalik kata yang ada pada array dengan tipe data char
 1. Contoh kata = {'S','E','L','A','M','A','T'} : output :TAMALES
3. Konversi Detik ke Hari, jam dan menit
 1. Contoh : input 500000 detik
 2. Output : 5 hari, 18 jam, 53 menit, 20 detik

