

Pertemuan 3

Operator, Sequence, Flowchart, dan Pseudocode

Tim Ajar Dasar Pemrograman 2025

Tujuan

- Mahasiswa mampu memahami dan menjelaskan tentang operator
- Mahasiswa mampu memahami dan menerapkan sequence di pemrograman Java
- Mahasiswa mampu memahami dan membuat algoritma dalam bentuk flowchart
- Mahasiswa mampu memahami dan membuat algoritma dalam bentuk pseudocode

OPERATOR

Jenis operator

1. Operator Aritmatika
2. Operator Increment dan Decrement
3. Operator Assignment
4. Operator Relasi
5. Operator Logika
6. Operator Bitwise

1. Operator Aritmetik (sudah dibahas di pertemuan 2)

2. Operator Increment dan Decrement

Operator Increment dan Decrement digunakan untuk menaikkan atau menurunkan suatu nilai integer (bilangan bulat) sebanyak satu satuan, dan hanya dapat digunakan pada variabel.

Operator	Use	Description
++	a++	Menaikan/menambah 1 nilai setelah operasi dilakukan
	++a	Menaikan/menambah 1 nilai sebelum operasi dilakukan
--	a--	Penurunan/mengurangi 1 nilai setelah operasi dilakukan
	--a	Penurunan/mengurangi 1 nilai sebelum operasi dilakukan



```
*/  
  
public class OperatorIncrementdanDecrement {  
    public static void main(String[] args) {  
        int i = 1;  
  
        //increment  
        System.out.println("i : " + i);  
        System.out.println("++i : " + ++i);  
        System.out.println("i++ : " + i++);  
  
        //decrement  
        System.out.println("--i : " + --i);  
        System.out.println("i-- : " + i--);  
        System.out.println("i : " + i);  
    }  
}
```

variabeltipedataoperator.OperatorIncrementdanDecrement > main >

out - variabeltipedataoperator (run) ✖

```
run :  
i : 1  
++i : 2  
i++ : 2  
--i : 2  
i-- : 2  
i : 1
```



Contoh

```
int x = 5;  
int y = ++x; // x naik jadi 6, lalu y = 6  
  
// Hasil:  
// y = 6  
// x = 6
```

- **++x** berarti "naikkan **x** dulu, baru gunakan hasilnya".
- **x** berubah dari **5** menjadi **6** **sebelum** nilai tersebut diberikan ke **y**

```
int x = 5;  
int y = x++; // y = 5, lalu x naik jadi 6  
  
// Hasil:  
// y = 5  
// x = 6
```

- **x++** berarti "gunakan nilai **x** dulu, baru naikkan **x**".
- **y** mendapatkan nilai **x** (yaitu 5), dan kemudian **x** berubah menjadi 6 setelah baris tersebut selesai dieksekusi.

3.Operator Assignment

Operator assignment dalam Java digunakan untuk memberikan sebuah nilai ke sebuah variabel. Operator assignment hanya berupa '=', shortcut assignment operator yang penting, yang digambarkan dalam tabel berikut :

Operator	Penggunaan	Ekuivalen Dengan
+=	Op1 += Op2	Op1 = Op1 + Op2
-=	Op1 -= Op2	Op1 = Op1 - Op2
*=	Op1 *= Op2	Op1 = Op1 * Op2
/=	Op1 /= Op2	Op1 = Op1 / Op2
%=	Op1 %= Op2	Op1 = Op1 % Op2
&=	Op1 &= Op2	Op1 = Op1 & Op2
=	Op1 = Op2	Op1 = Op1 Op2
^=	Op1 ^= Op2	Op1 = Op1 ^ Op2
<<=	Op1 <<= Op2	Op1 = Op1 << Op2
>>=	Op1 >>= Op2	Op1 = Op1 >> Op2
>>>=	Op1 >>>= Op2	Op1 = Op1 >>> Op2



Contoh kode program

```
public class operatorassignment2 {  
    public static void main(String[] args) {  
        int a = 10;  
        // Demo operator assignment  
        a += 5;  
        System.out.println("value a [10] += 5 = " + a);  
  
        int b = 10;  
        b -= 5;  
        System.out.println("value b [10] -= 5 = " + b);  
  
        int c = 10;  
        c *= 5;  
        System.out.println("value c [10] *= 5 = " + c);  
  
        int d = 10;  
        d /= 5;  
        System.out.println("value d [10] /= 5 = " + d);  
  
        int e = 10;  
        e %= 5;  
        System.out.println("value e [10] %= 5 = " + e);  
    }  
}
```

Output - variabeltipedataoperator (run) ☒

```
run:  
value a [10] += 5 = 15  
value b [10] -= 5 = 5  
value c [10] *= 5 = 50  
value d [10] /= 5 = 2  
value e [10] %= 5 = 0  
BUILD SUCCESSFUL (total time: 0 seconds)  
|
```



Contoh kode program

`a = a+5;` bisa dipersingkat menjadi `a += 5;`

`b = b-5;` bisa dipersingkat menjadi `b -= 5;`

`c = c*5;` bisa dipersingkat menjadi `c *= 5;`

`d = d/5;` bisa dipersingkat menjadi `d /= 5;`

`e = e%5;` bisa dipersingkat menjadi `e %= 5;`

4. Operator Relasi

Operator relasi dalam Java digunakan untuk menghasilkan nilai boolean yang sering digunakan untuk mengatur alur jalannya sebuah program.

Operator	Penggunaan	Deskripsi
>	Op1 > Op2	Menghasilkan true jika Op1 lebih besar dari Op2
<	Op1 < Op2	Menghasilkan true jika Op1 lebih kecil dari Op2
>=	Op1 >= Op2	Menghasilkan true jika Op1 lebih besar atau sama dengan Op2
<=	Op1 <= Op2	Menghasilkan true jika Op1 lebih kecil atau sama dengan Op2
==	Op1 == Op2	Menghasilkan true jika Op1 sama dengan Op2
!=	Op1 != Op2	Menghasilkan true jika Op1 tidak sama dengan Op2



```
public class operatorrelasi {  
    public static void main(String[] args) {  
        int x,y,z;  
        x = 100;  
        y = 99;  
        z = 99;  
        System.out.println("Nilai x = "+x);  
        System.out.println("Nilai y = "+y);  
        System.out.println("Nilai z = "+z);  
        // operator sama dengan  
        if(y == z ){  
            System.out.println("y sama dengan z");  
        }else {  
            System.out.println("y tidak sama dengan z");  
        }  
        // operator tidak sama dengan  
        if(x != y ){  
            System.out.println("x tidak sama dengan y");  
        }else {  
            System.out.println("x sama dengan y");  
        }  
        // operator lebih besar dari  
        if(x > y ){  
            System.out.println("x lebih besar dari y");  
        }  
    }  
}
```

menghasilkan

```
Nilai x = 100  
Nilai y = 99  
Nilai z = 99  
y sama dengan z  
x tidak sama dengan y  
x lebih besar dari y  
y lebih kecil dari x  
x lebih besar dari atau sama dengan y  
y lebih kecil dari atau sama dengan x
```

5. Operator Logika

Operator ini digunakan untuk ekspresi logik yang menghasilkan nilai boolean.

Operator-operator yang digunakan adalah AND (&&), OR (| |) dan NOT (!).

Operator	Deskripsi	Contoh
&&	and	x=6 y=3 (x < 10 && y > 1) hasil true
	or	x=6 y=3 (x==5 y==5) hasil false
!	not	x=6 y=3 !(x==y) hasil true

Contoh kode program

```
public class operatorlogika {  
    public static void main(String[] args) {  
        boolean _true = true;  
        boolean _false = false;
```

run:

Relation with OR (||)

_true || _true : true

_true || _false : true

_false || _true : true

_false || _false : false

Relation with AND (&&)

_true && _true : true

_true && _false : false

_false && _true : false

_false && _false : false

Relation with NOT (!)

inverse of (NOT) _true is: false

inverse of (NOT) _false is: true

BUILD SUCCESSFUL (total time: 1 second)

```
    System.out.println("Relation with OR (||)");  
    System.out.println("_true || _true : " + (_true || _true));  
    System.out.println("_true || _false : " + (_true || _false));  
    System.out.println("_false || _true : " + (_false || _true));  
    System.out.println("_false || _false : " + (_false || _false));  
  
    System.out.println("Relation with AND (&&)");  
    System.out.println("_true && _true : " + (_true && _true));  
    System.out.println("_true && _false : " + (_true && _false));  
    System.out.println("_false && _true : " + (_false && _true));  
    System.out.println("_false && _false : " + (_false && _false));  
  
    System.out.println("Relation with NOT (!)");  
    System.out.println("inverse of (NOT) _true is: " + !_true);  
    System.out.println("inverse of (NOT) _false is: " + !_false);
```

6. Operator Bitwise

Operator ini digunakan untuk melakukan manipulasi bit dari sebuah bilangan

- Bitwise OR(|)

Hasil bit bernilai 1 ketika salah satu bit-bit bernilai 1, selain itu bernilai 0.

Contoh:

```
int a = 5; //0101
int b = 7; //0111
System.out.println(a|b); //output 7
//0101
//0111
//____
//0111 -> 7
```


6. Operator Bitwise(2)

- Bitwise AND(&)

Hasil bit bernilai 1 ketika semua bit-bit bernilai 1, selain itu bernilai 0.

Contoh:

```
int a = 5; //0101
int b = 7; //0111
System.out.println(a&b); //output 5
//0101
//0111
//----
//0101 -> 5
```

6. Operator Bitwise(3)

- Bitwise XOR(^)

Nilai bit bernilai 1 ketika ada bit bernilai 1 dan 0, selain itu bernilai 0.

Contoh:

```
int a = 5; //0101
int b = 7; //0111
System.out.println(a^b); //output 2
//0101
//0111
//____
//0010 -> 2
```



6. Operator Bitwise(4)

- Bitwise Complement(~)

Nilai bit yang berkebalikan, ketika nilai bit bernilai 1 maka menghasilkan 0 sedangkan bernilai 0 menghasilkan 1.

Contoh:

```
int a = 5; //0101
System.out.println(~a); //output 10
//0101
//____
//1010 -> 10
```

SEQUENCE

Sequence...(1)

- Sequence adalah **alur eksekusi instruksi** di mana setiap instruksi dikerjakan secara **berurutan** sesuai dengan urutan penulisannya dalam kode program.
- Suatu program akan mengikuti **urutan instruksi** yang telah ditentukan oleh programmer. **Misalnya**, jika ada tiga instruksi A, B, dan C, maka A akan dieksekusi terlebih dahulu, diikuti oleh B, dan kemudian C.
- Urutan eksekusi ini **sangat penting** untuk memastikan program berfungsi **sesuai dengan logika** yang diharapkan. Salah urut dalam menjalankan instruksi dapat menghasilkan output yang salah atau perilaku program yang tidak diinginkan.



Sequence...(2)

Perhatikan kode berikut!

```
int a = 10;  
int b = a + 5;
```

1. **Baris pertama:** Menginisialisasi variabel `a` dengan nilai `10`.
2. **Baris kedua:** Menginisialisasi variabel `b` dengan hasil dari `a + 5` (yaitu `10 + 5 = 15`).

“Ini adalah contoh sederhana tentang bagaimana instruksi dikerjakan dalam urutan yang telah ditentukan untuk mencapai hasil yang benar”

Sequence...(3)

Menara Hanoi (Contoh untuk 3 Cakram)

Sebuah tiang (A) berisi tiga cakram (merah, biru, hijau) dengan merah paling bawah, lalu biru, dan hijau di atas. Tujuan Anda adalah memindahkan seluruh tumpukan cakram dari tiang A ke tiang C, dengan menggunakan tiang B sebagai tiang bantu, serta mengikuti aturan berikut:

1. **Satu Cakram per Langkah:** Hanya satu cakram yang boleh dipindahkan pada satu waktu.
2. **Cakram Lebih Besar di Bawah:** Cakram yang lebih besar tidak boleh ditempatkan di atas cakram yang lebih kecil.

Sequence...(4)

Langkah-Langkah Penyelesaian (untuk 3 Cakram)

1. Pindahkan cakram terkecil: (**hijau**) dari A ke C.
2. Pindahkan cakram tengah: (**biru**) dari A ke B.
3. Pindahkan cakram **hijau**: dari C ke B.
4. Pindahkan cakram terbesar: (**merah**) dari A ke C.
5. Pindahkan cakram **hijau**: dari B ke A.
6. Pindahkan cakram **biru**: dari B ke C.
7. Pindahkan cakram **hijau**: dari A ke C.

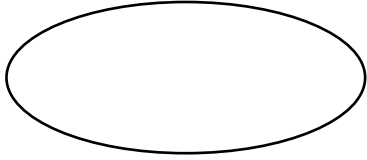


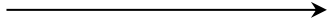
Hasil: Setelah 7 langkah, tumpukan cakram berhasil dipindahkan dari tiang A ke tiang C, dengan setiap cakram berada pada posisinya masing-masing.

FLOWCHART


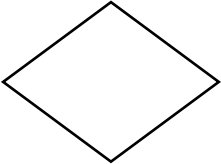
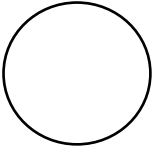
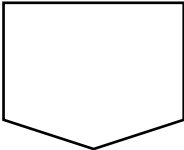

Flowchart

- Flowchart merupakan sebuah bagan dengan **simbol-simbol** tertentu yang digunakan untuk menjelaskan **urutan proses** dan hubungan antar proses lainnya pada sebuah program.
- Software untuk membuat flowchart draw.io

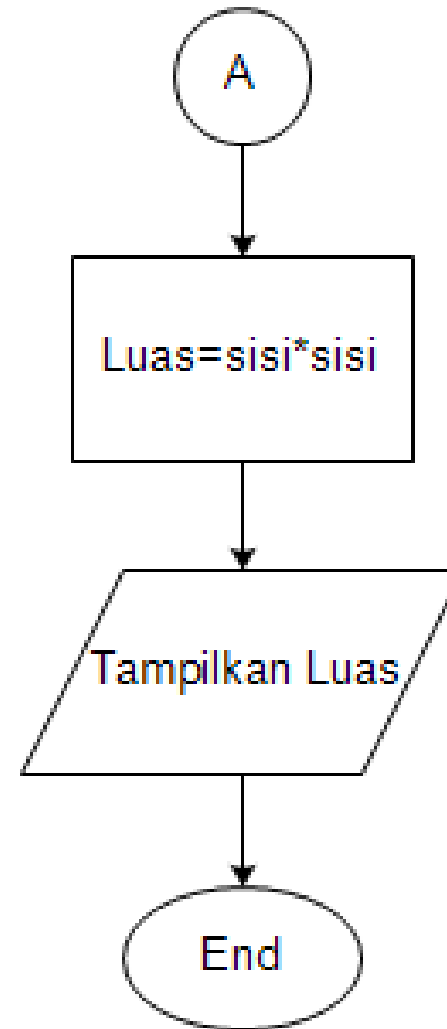
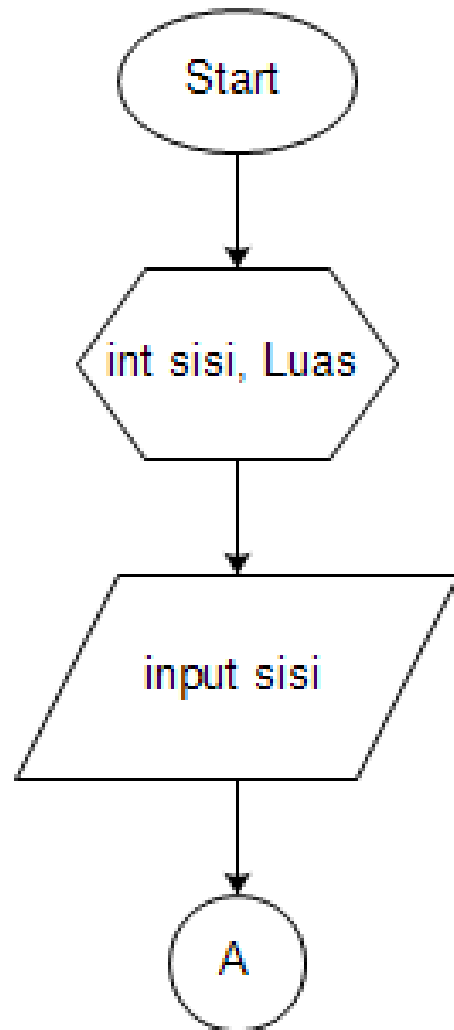
Symbol-simbol Flowchart...(1)

Symbol	Nama	Deskripsi
	Terminator	Simbol untuk permulaan (start) dan akhir (end) dalam sebuah proses
	Preparation	Simbol untuk mempersiapkan penyimpanan yang akan digunakan
	Input-output	Simbol yang menyatakan proses input dan output
	Flow Line (Garis Alir)	Arah aliran program

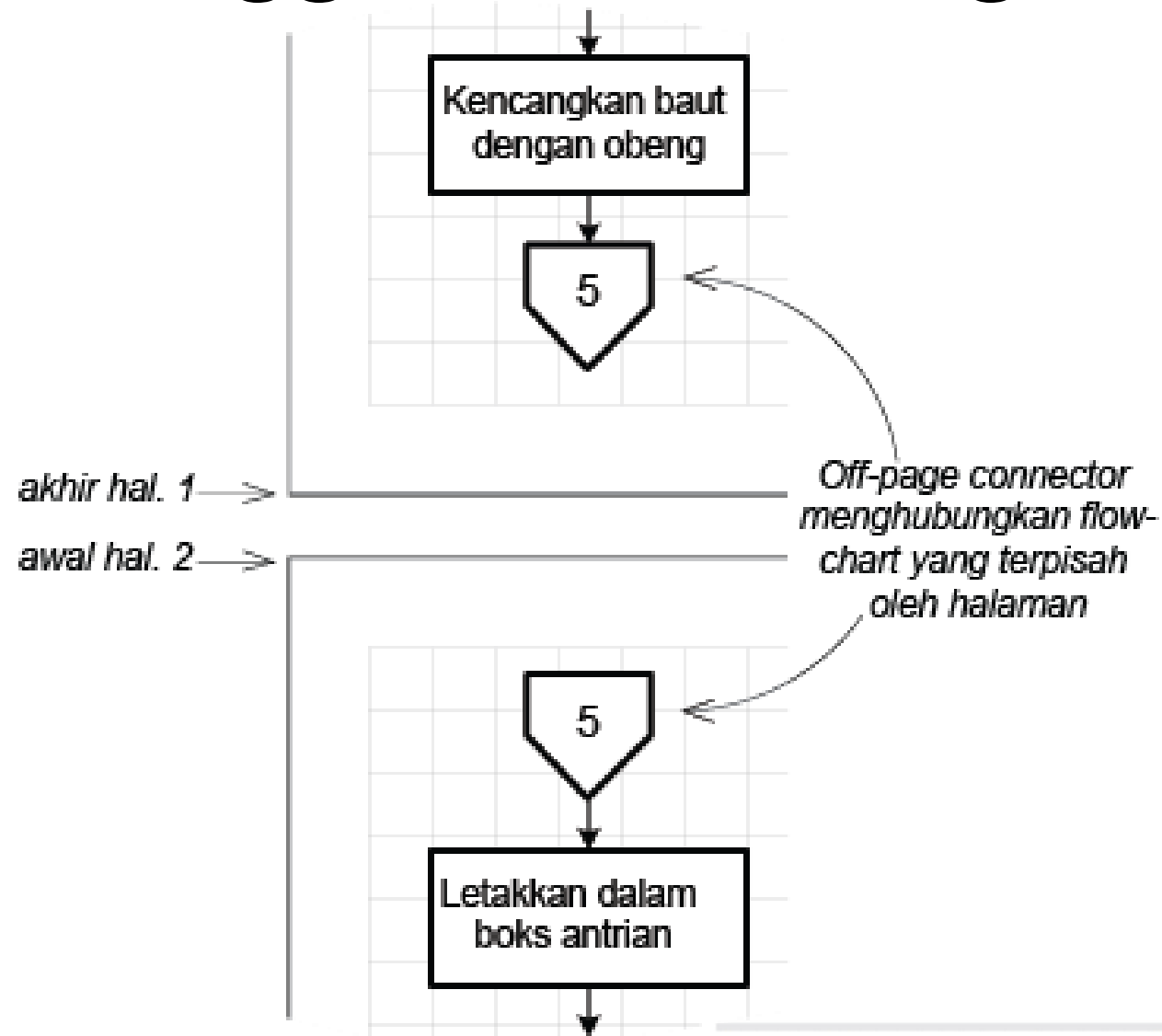
Symbol-simbol Flowchart...(2)

Symbol	Nama	Deskripsi
	Processing simbol	Simbol yang menunjukkan pemrosesan oleh komputer
	Percabangan	Simbol yang menunjukan terdapat dua pilihan output
	On Page Connector	Penghubung bagian-bagian flowchart yang ada pada satu halaman
	Off Page Connector	Penghubung bagian-bagian flowchart yang berada pada halaman yang berbeda
	Predefined Process	Simbol untuk pelaksanaan suatu bagian (sub-program)/fungsi/prosedur

Contoh Penggunaan On Page Connector



Contoh Penggunaan Off Page Connector



PSEUDOCODE

DEFINISI PSEUDOCODE

- **Pseudocode** adalah **representasi algoritma dalam bentuk deskripsi terstruktur** yang menyerupai bahasa pemrograman, tetapi **tidak terikat pada aturan sintaks tertentu**.
- Pseudocode berfungsi sebagai jembatan antara perancangan algoritma menggunakan bahasa manusia dan implementasinya ke dalam bahasa pemrograman yang sesungguhnya.
- Disebut “pseudo” (semu) karena bukan bahasa pemrograman sungguhan, melainkan bentuk perantara antara **bahasa manusia** dan **kode program**.

Pseudocode: Masukan

- Sewaktu komputer menerima informasi atau *input*, maka *statement* yang biasa digunakan adalah **“Read”, “Get”, “Baca” ,”Input”**
- Contoh:
 - Input bilangan1
 - Baca bilangan2
 - Read jariJari
 - Get panjang

Pseudocode: Keluaran

- Pada saat komputer menampilkan informasi ataupun *output*, maka *statement* yang biasa digunakan adalah **“Print”**, **“Write”**, **“Put”**, **“Output”**, **“Display”** ataupun **“Cetak”**
- Contoh:
 - Print luas
 - Output total
 - Display gaji
 - Cetak nilai

Pseudocode: Perhitungan Aritmatika

- Untuk melakukan operasi aritmetika digunakan pseudocode berikut:

+	Penjumlahan (Add)
-	Pengurangan (subtract)
*	Perkalian (multiple)
/	Pembagian (Devide)
()	Kurung

- Statement **“Compute”**, **“Calculate”** ataupun **“Hitung”** juga dapat digunakan.
- Contoh:

Add bilangan1 and bilangan2 to total
total = bilangan1 + bilangan2

Pseudocode: Memberikan Nilai

- Ada beberapa cara untuk memberikan nilai ke dalam variabel :
 - Memberikan nilai awal, menggunakan *statement* **“Initialize”** atau **“Set”**
 - Memberikan nilai sebagai hasil dari suatu proses, maka tanda **“=”** atau **“←”** bisa digunakan
 - Untuk menyimpan suatu nilai maka *statement* **“Save”** atau **“Store”** digunakan

- Contoh:

Set counter to 0

Initialize counter to 0

total ← bilangan1 + bilangan2

luas = panjang * lebar

Pseudocode : Operasi Pembandingan

- Untuk membandingkan nilai antara 2 variabel, digunakan

<	Kurang dari
<=	Kurang dari sama dengan
>	Lebih dari
>=	Lebih dari sama dengan
==	Sama dengan
!=	Tidak sama dengan

- Operasi ini menghasilkan kondisi benar atau salah
- Contoh:

a > b

c == d

Pseudocode : Operasi Relasional

- Biasanya digunakan untuk me-relasikan 2 keadaan atau kondisi
- Menghasilkan nilai true dan false
- Kata kunci: AND, OR
- Contoh:
umur < 15 AND nilai > 70
nilai == 'A' OR nilai == 'B'



Pseudocode: Pemilihan

- Salah satu operasi terpenting yang dapat dilakukan komputer adalah membandingkan dan memilih salah satu alternatif solusi.
- Keyword yang digunakan : **“IF”**, **“THEN”** dan **“ELSE”**
- Contoh

INPUT harga

IF harga >100 **THEN**

 harga = harga-(0.5*harga)

ELSE

 harga = harga-(0.1*harga)

ENDIF

OUTPUT harga

Pseudocode: Pengulangan

- Jika ada beberapa perintah atau proses yang harus diulang, maka dapat digunakan *keyword* “**DOWHILE**” dan “**ENDDO**”.
- Contoh

bil \leftarrow 0

DOWHILE bil < 10

OUTPUT bil

bil \leftarrow bil +1

ENDDO

CONTOH

Buatlah pseudocode untuk menghitung rata-rata tiga bilangan!

JAWAB

PROGRAM

Menghitung rata-rata tiga bilangan

DEKLARASI

a,b,c ,rata_rata : float

ALGORITMA

Input a,b,c

rata_rata= (a+b+c)/3

Output rata_rata

Contoh Soal

Contoh 1

Polinema mempunyai sebuah lapangan sepak bola berbentuk persegi panjang, buatlah flowchart, pseudocode untuk menghitung luas lapangan tersebut!

Contoh 1

1. Menentukan Algoritma

Input: panjang, lebar

Output: luas

Proses:

1. input panjang, lebar **2**

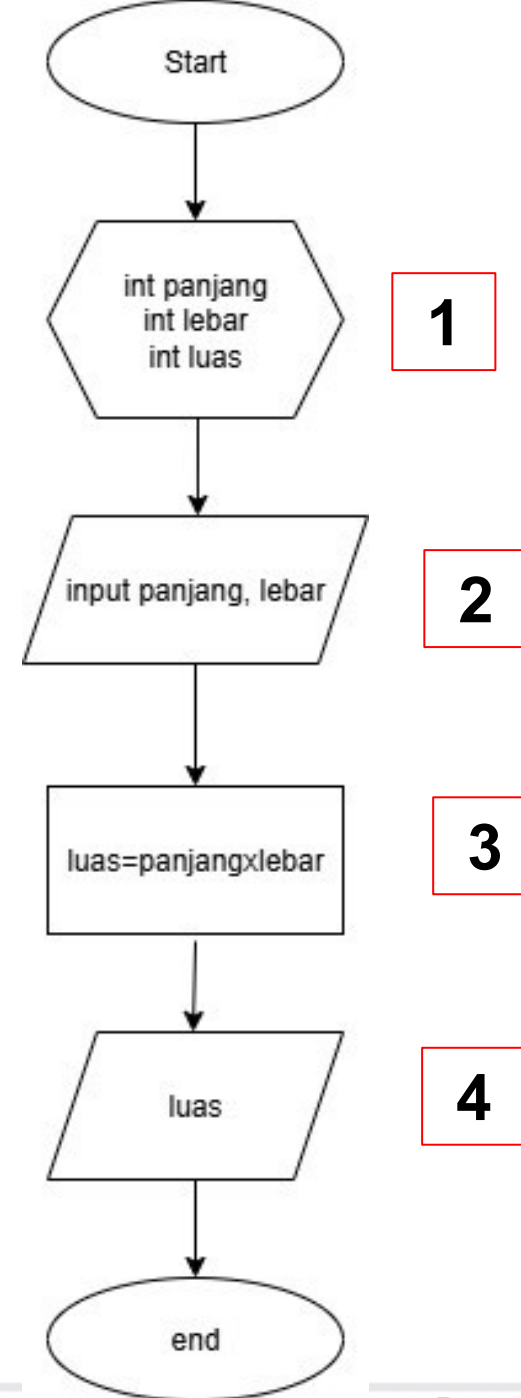
2. $luas = Panjang \times lebar$ **3**

3. Output luas **4**

2. Mengidentifikasi variable dan jenis tipe data berdasarkan algoritma

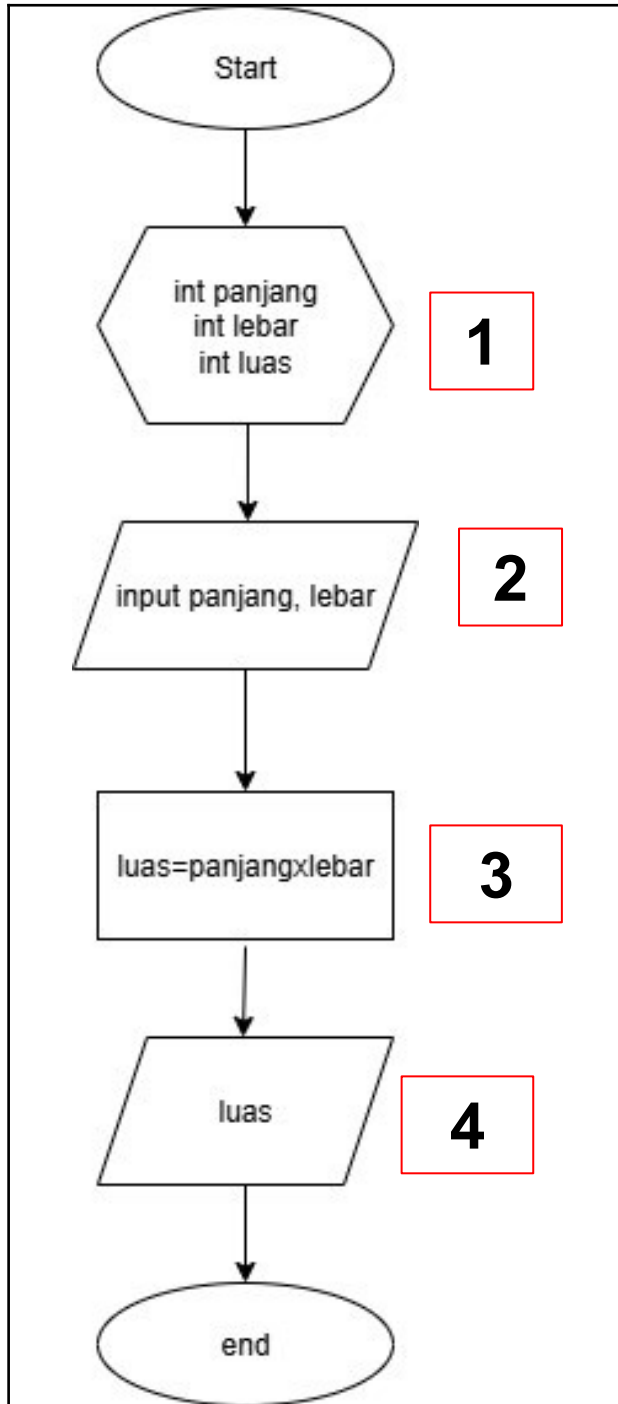
Variabel	Tipe data
panjang	int
lebar	int
luas	int

1





Contoh 1



PROGRAM

Menghitung LuasPersegiPanjang

DEKLARASI

panjang:int
lebar:int
luas:int

ALGORITMA

Input panjang

Input lebar

luas <- panjang * lebar

Output (luas)

Contoh 2

- Ibu Lani berbelanja di sebuah toko dan membeli pakaian seharga **Rp. xxx**. Toko tersebut memberikan **diskon 15%** untuk setiap pembelian.

Berapakah besar diskon dan jumlah yang harus dibayar Ibu Lani? buatlah flowchart, pseudocode untuk menghitung luas lapangan tersebut!

Contoh 2

1. Menentukan Algoritma

Input: harga

Output: potongan, jml_bayar

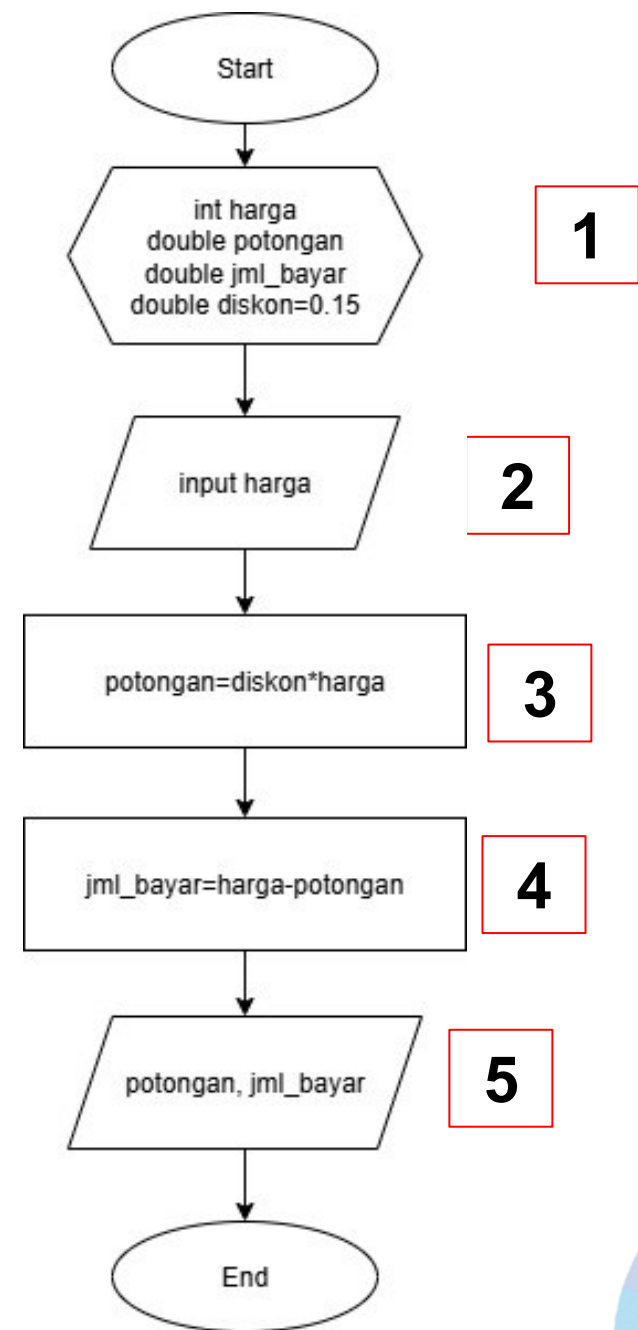
Data Lain: diskon=0.15

Proses:

1. input harga **2**
2. potongan=diskon x harga **3**
3. jml_bayar= harga – potongan **4**
4. Output potongan, jml_bayar **5**

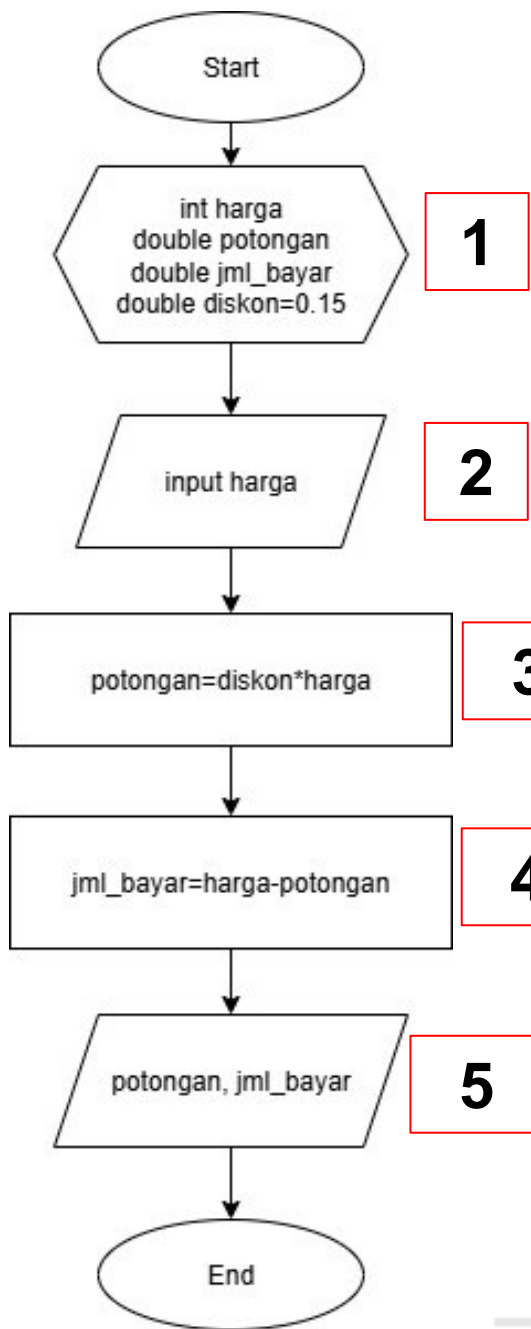
2. Mengidentifikasi variable dan jenis tipe data berdasarkan algoritma

Variabel	Tipe data
harga	int
potongan	double
jml_bayar	dpuble
Diskon=0.15	double





Contoh 2



PROGRAM

Menghitung Total Bayar

DEKLARASI

harga:int

potongan: double

jml_bayar: double

diskon=0.15: double

ALGORITMA

Input harga

potongan <- diskon * harga

Jml_bayar <- harga - potongan

Output (potongan, jml_bayar)

Tugas...(1)

Pak Ali membeli sebuah motor dengan harga **Rp. x** secara kredit. Ia membayar uang muka sebesar **Rp. y** dan sisanya dicicil selama **z bulan** dengan bunga tetap **1% per bulan** dari sisa harga yang belum dibayarkan.

Berapakah jumlah cicilan per bulan yang harus dibayar Pak Ali?

- Buatlah algoritma flowchart dari studi kasus tersebut!
- Buatlah algoritma pseudocode dari studi kasus tersebut!

Tugas...(2)

Sebuah mobil menempuh perjalanan dari Malang ke Surabaya sejauh **x km**. Mobil tersebut menghabiskan rata-rata **1 liter bensin untuk 2 km**. Jika harga bensin adalah **Rp. 10.000 per liter**, **berapa biaya bensin yang diperlukan untuk perjalanan tersebut?**

- Buatlah algoritma flowchart dari studi kasus tersebut!
- Buatlah algoritma pseudocode dari studi kasus tersebut!