

# Pertemuan 3

## Variabel, Tipe Data, Operator dan Input-Output

Tim Ajar Dasar Pemrograman 2021

# Tujuan

- Mahasiswa memahami dan mampu menjelaskan tentang Tipe Data
- Mahasiswa menjabarkan dan mampu menjelaskan tentang Variabel
- Mahasiswa memahami dan mampu menguraikan tentang Operator (Penugasan Aritmatika, Penugasan Gabungan, Increment, Decrement, Relasional, Logika, Kondisional, Bitwise, Casting)



# Variabel

- Variable digunakan dalam bahasa pemrograman untuk menyimpan nilai sementara dimana untuk digunakan kembali nantinya.
- Variabel memiliki tipe data dan nama.
- tipe data mengindikasikan tipe dari nilai pada Variabel tersebut.

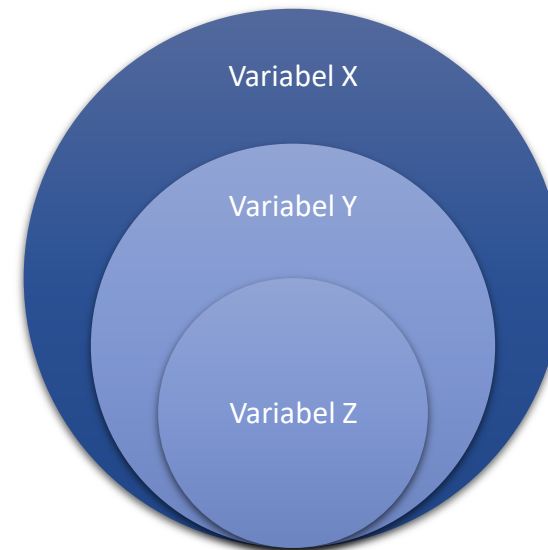


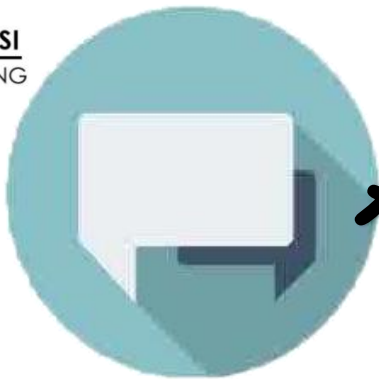
Apa yang anda bayang  
dengan gambar di samping?



# Jenis Variabel

- Variable lokal adalah variable yang hanya bisa dikenali pada sub program
- Variabel global adalah variable yang dapat dikenali pada keseluruhan program





# Aturan Penulisan Variabel

- Nama Variable tidak boleh menggunakan kata kunci Java
- Nama Variable boleh menggunakan huruf, angka(0-9), garis bawah(\_), dan symbol dolar(\$), namun sebaiknya penggunaan
- Nama Variable sebaiknya menggunakan diawali huruf kecil
- Apabila nama Variable lebih dari satu kata maka kata yang setelahnya diawali huruf besar.

## o Bentuk:

`<type data> <nama> [=nilai awal]`

nilai dalam tanda [ ] bersifat optional.

## Contoh:

```
int contVariabel;
```

```
int contohVar = 34;
```



# Tipe Data

- Tipe data adalah jenis data yang ingin kita simpan di Variabel.
- Tipe data dapat dikategorikan menjadi dua kelompok, yaitu
  1. *tipe data Primitif*
  2. *tipe data Referensi.*



Apa yang anda bayangkan dengan gambar di samping?



Bagaimana dengan gambar yang ini?

# Tipe data primitif

Jenis Data	Deskripsi	Ukuran	Minimum	Maksimum
boolean	true / false	1-bit		
char	Karakter Unicode	16-bit		
byte	Bilangan bulat	8-bit	-127	128
short	Bilangan bulat	16-bit	-32768	32767
int	Bilangan bulat	32-bit	-2147483648	2147483647
long	Bilangan bulat	64-bit	-9223372036854775808	9223372036854775807
float	Bilangan riil	32-bit	1.40129846432481707e-45	3.40282346638528860e+38
double	Bilangan riil	64-bit	4.94065645841246544e-324	1.79769313486231570e+308



# DEKLARASI

-----Deklarasi-----

```
int nilai;
```

```
double angka;
```

```
float a, b, c;
```

-----Pemberian nilai-----

```
int nilai=75;
```

```
double angka=2.5;
```



# Mencetak Variabel

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

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

-----atau-----

```
System.out.println("Nilai anda adalah" +nilai);
```

```
System.out.println("angka adalah" +a);
```

# Casting tipe data

Casting adalah ketika kita ingin memberikan nilai dari tipe data primitive ke tipe data primitive yang lain

- Widening Casting (otomatis) – mengubah tipe data dari yang ukurannya lebih kecil ke tipe data yang lebih besar

byte -> short -> char -> int -> long -> float -> double



Ilustrasi  
widening Casting

# Casting tipe data(2)

- Narrowing Casting (manual) – mengubah tipe data dari yang ukurannya lebih besar ke tipe data yang lebih kecil

double -> float -> long -> int -> char -> short -> byte



Ilustrasi narrowing  
Casting



# Contoh Casting tipe data

- Widening Casting(otomatis)

```
byte umur = 9;  
double myDouble = umur;  
System.out.println(umur);    // Outputs 9  
System.out.println(myDouble); // Outputs 9.0
```

- Narrowing Casting(manual)

```
double ipk = 3.78;  
int myInt = (int) ipk;  
System.out.println(ipk);    // Outputs 3.78  
System.out.println(myInt);  // Outputs 3
```



# Kegunaan ASCII ??

- ASCII adalah singkatan dari American Standard Code for Information InterChange.
- Sesuai dengan namanya, ASCII digunakan untuk pertukaran informasi dan komunikasi data.
- ASCII merupakan kode angka yang mewakili sebuah karakter.



## USASCII code chart

<div><div>b7b6b5</div><div>b4b3b2b1</div><div>Bits</div><div>Column</div><div>Row</div></div>					000	001	010	011	100	101	110	111
					0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	\	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	\$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	
1	1	0	1	13	CR	GS	-	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

# Tipe data referensi

- Tipe data non-primitive dibuat berdasarkan kebutuhan programmer.
- Nilai bawaan non-primitive adalah null
- Pendeklarasian tipe data ini hampir sama dengan deklarasi pada tipe data primitif.
- Tipe data non-primitive diawali dengan huruf besar







Ciri khas tipe data referensi adalah kemampuannya menampung banyak nilai.

Pada tipe data primitif, nilai yang bisa ditampung Cuma 1 saja. Perhatikan contoh berikut ini:

### *Tipe Primitif :*

`int x = 9;` (ada 1 nilai saja, yaitu angka 9)

`char hurufku = "h";` (ada 1 nilai saja, yaitu huruf h)

### *Tipe Referensi :*

`String tulisan = "Aku Belajar Java";` (ada 16 nilai, termasuk spasi)

`int[] daftar = { 1, 4, 9, 16, 25, 36, 49 };` (ada 7 nilai bertipe integer)



# operator

- Operator merupakan simbol yang biasa digunakan dalam menulis suatu pernyataan (*statement*) dalam bahasa pemrograman apapun. Operator akan melakukan suatu operasi terhadap operand sesuai dengan fungsinya.
- Contoh operasi antara lain penjumlahan, pengurangan, pembagian dan sebagainya.

3 + 8 \* 4  
3 8 4 adalah operand  
+ \* adalah 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 Aritmatika

Arithmetic operator (operator aritmatika) adalah operator yang berfungsi untuk operasi aritmatika.

Arithmetic Operator	Description
+	plus
-	minus
*	point
/	divide
%	modulus



```
* @author mayang
*/
public class operatoraritmatika {
    public static void main(String[] args) {
        int a = 20;
        int b = 10;
        System.out.println("Arithmetic Operator");
        System.out.println("bilangan pertama : "+a);
        System.out.println("bilangan kedua: "+b);
        System.out.println(" a + b = " + (a + b));
        System.out.println(" a -b = " + (a -b));
        System.out.println(" a / b = " + (a / b));
        System.out.println(" a * b = " + (a * b));
        System.out.println(" a % b = " + (a % b));
    }
}
```

ariabeltipedataoperator.operatoraritmatika > main >

it - variabeltipedataoperator (run) ✖

```
run:
Arithmetic Operator
bilangan pertama : 20
bilangan kedua: 10
a + b = 30
a -b = 10
a / b = 2
a * b = 200
a % b = 0
BUILD SUCCESSFUL (total time: 0 seconds)
```

## 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
```

## 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 -= 1;`

`c = c*5;` bisa dipersingkat menjadi `c *= 3;`

`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)
```

```
("Relation with OR (||)");  
("_true || _true : " +(_true||_true));  
("_true || _false : " +(_true||_false));  
("_false || _true : " +(_false||_true));  
("_false || _false : " +(_false||_false));  
  
("Relation with AND (&&)");  
("_true && _true : " +(_true&&_true));  
("_true && _false : " +(_true&&_false));  
("_false && _true : " +(_false&&_true));  
("_false && _false : " +(_false&&_false));  
  
("Relation with NOT (!)");  
("inverse of (NOT) _true is: " +!_true);  
("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 -6
//0101
//____
//1010 -> 10
```

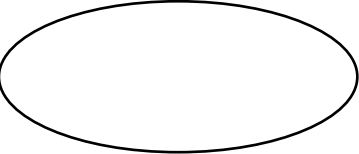



$$\sim n = -(n+1)$$


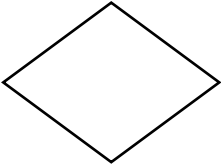
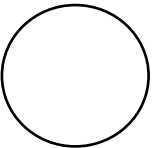
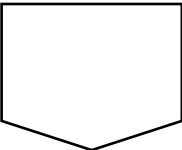

$$\sim(-n) = (n-1)$$

# FlowChart

FlowChart merupakan sebuah bagan dengan simbol-simbol tertentu yang digunakan untuk menjelaskan urutan proses dan hubungan antar proses lainnya pada sebuah program.

# Simbol-symbol Flowchart

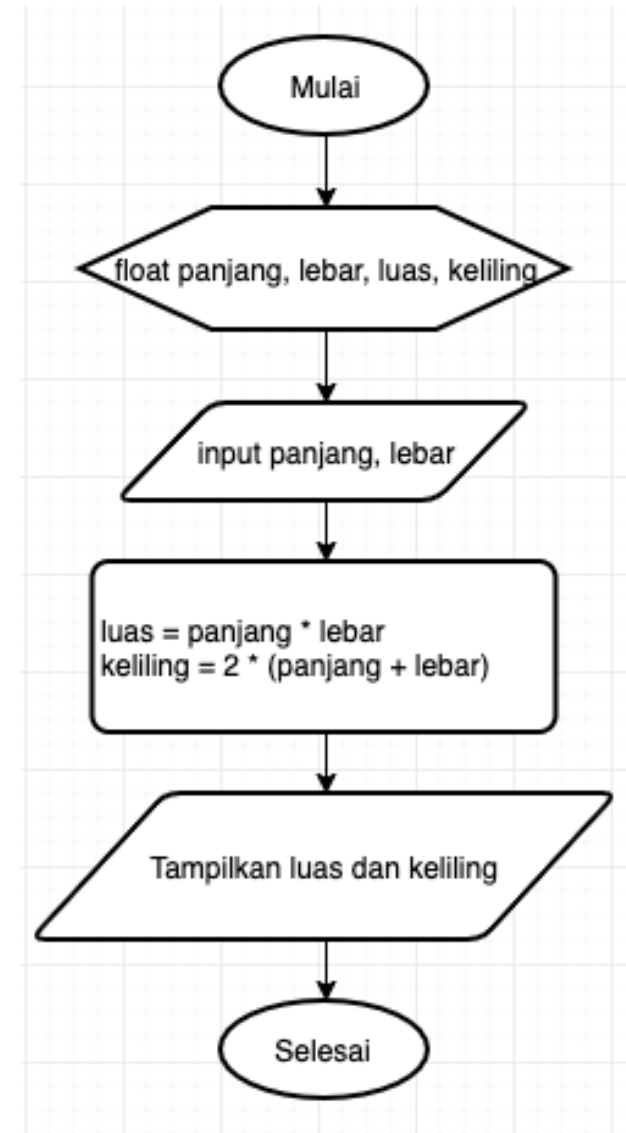
Symbol	Nama	Deskripsi
	Terminator	Simbol untuk permulaan (start) dan akhir (end) dalam sebuah proses
	Preparation	Simbol untuk mempersiapkan penyimpanan yang akan digunakan
	Input-output	Simbol yang menyatakan proses input dan output
	Flow Line (Garis Alir)	Arah aliran program

Symbol	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

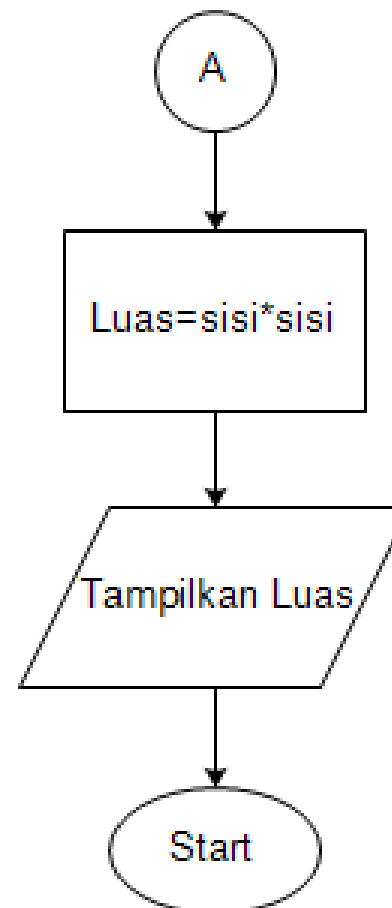
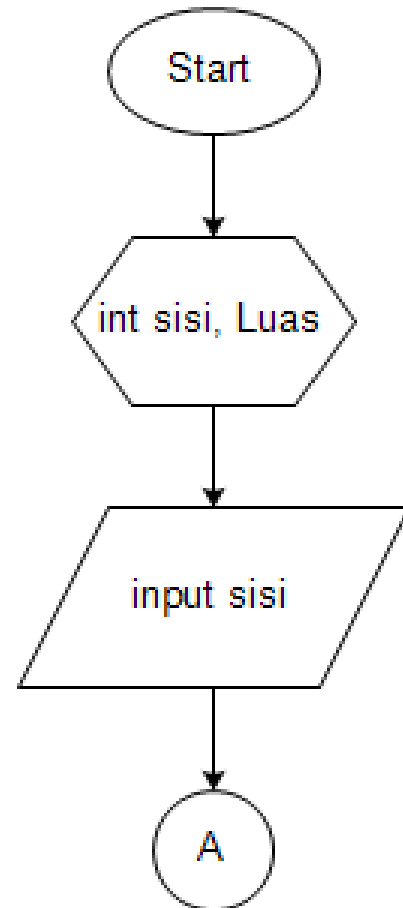
# Contoh FlowChart

## Contoh Studi Kasus:

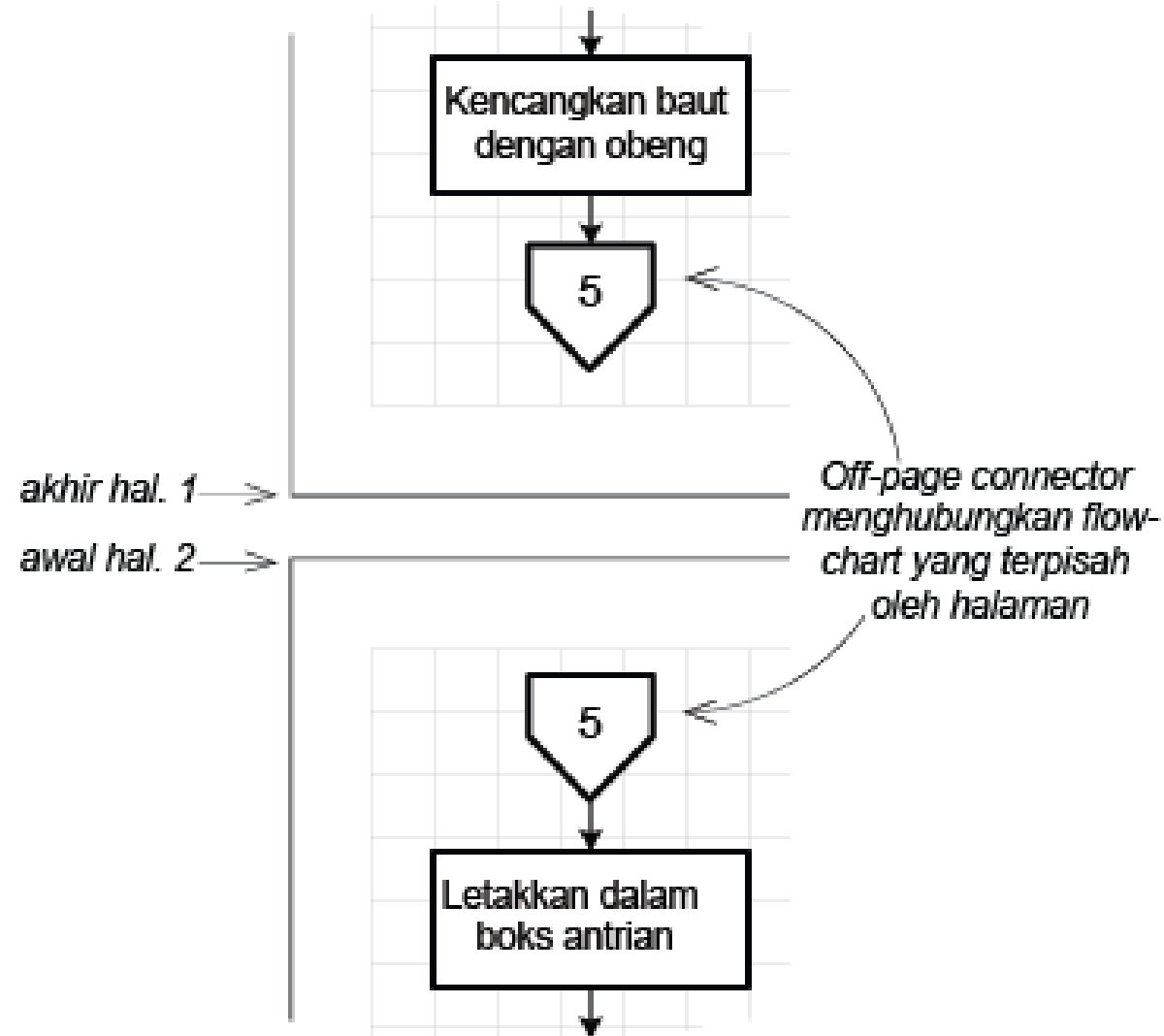
Pak Adi mempunyai sebuah kebun berbentuk persegi panjang, buatlah flowChart untuk menghitung luas dan keliling kebun pak Adi!



# Contoh Penggunaan On Page Connector



# Contoh Penggunaan Off Page Connector





# Peggunaan Inputan Pada Java

- Untuk membaca input dari keyboard yaitu dengan mengguakan library Scanner yag di import kedalam program java.
- Caranya adalah dengan menuliskan perintah **import java.util.Scanner** di baris paling atas dari kode program yang akan dibuat.
- Selanjutnya tuliskan perintah deklarasi scanner berikut ini didalam fungsi main() :

```
Scanner sc = new Scanner(System.in);
```

- Selanjutnya, tergantung dari jenis input yang akan dimasukkan, berupa bilangan bulat (int), bilangan koma (float/double), atau karakter (String).
  1. Jika input berupa bilangan bulat, maka perintahnya adalah: **nextInt();**
  2. Jika input berupa bilangan koma, maka perintahnya adalah: **nextFloat();**
  3. Jika input berupa teks, maka perintahnya adalah: **nextLine();**

# Menampilkan Output Pada Java

Untuk menampilkan output ke layar, ada beberapa cara:

1. `System.out.print("Hello world");` Perintah ini akan menampilkan kata Hello world ke layar, atau apapun yang kita tuliskan didalam tanda petik.
2. `System.out.println("Hello world");` Perintah ini akan menampilkan kata Hello world ke layar, atau apapun yang kita tuliskan didalam tanda petik, sekaligus memberi perintah ganti baris di akhir kata/kalimat.
3. `System.out.println(panjang);` Perintah ini akan menampilkan isi variabel panjang ke layar. Perhatikan bahwa untuk menampilkan isi variabel, tidak perlu menggunakan tanda petik ( " ).
4. `System.out.println("Panjang segi empat: " + panjang);` Perintah ini akan menampilkan kalimat "Panjang segi empat: " kemudian disambung dengan isi variabel panjang ke layar. Perhatikan untuk menyambung kalimat dengan isi variabel, digunakan tanda plus ( + ).

# Tugas

1. Pak Budi memiliki lahan berbentuk persegi dengan sisi 100m, di dalam lahan pak budi terdapat kolam renang berbentuk lingkaran. Buatlah flowchart untuk menghitung berapakah luas kolam renang dan lahan pak budi yang tidak dibangun kolam renang?
2. Gaji bersih karyawan PT. XYZ didapatkan dari jumlah gaji pokok dan tunjangan dikurangi pajak. Gaji pokok karyawan PT. XYZ adalah Rp. 5.000.000,-. Tunjangan karyawan dihitung 20% dari gaji pokok, sedangkan pajak adalah 5% dari gaji pokok. Buatlah flowchart untuk menghitung gaji bersih karyawan di PT. XYZ!