

### Template Lembar Kerja Individu dan Kelompok

Nama & NPM	Topik:	Tanggal:
Diodo Arrahman G1A022027	Operator Java	7 – 9 – 2022

#### [No. 1] Identifikasi Masalah

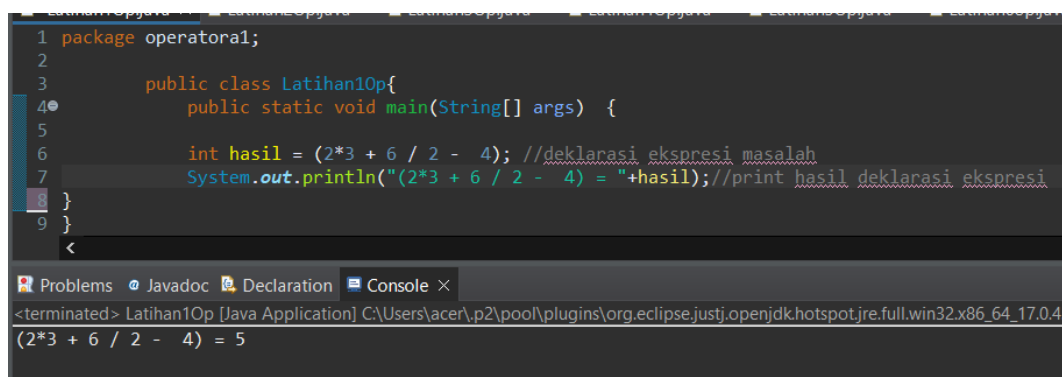
Pada soal, diberikan sebuah permasalahan matematika dengan ekspresi,  $(2*3 + 6 / 2 - 4)$   
Kami diminta untuk menyusun kode Java untuk melakukan perhitungan terhadap masalah tersebut.

#### [No.1] Analisis dan Argumentasi

Saya mengusulkan masalah ini dapat diatasi dengan cara memasukkan ekspresi permasalahan tersebut ke program eclipse setelah menyusun struktur dasar program java. Caranya dengan membuat deklarasi untuk ekspresi tersebut.

#### [No.1 ] Penyusunan Algoritma dan Kode Program

- 1) Algoritma
  - a. Susun struktur dasar kode java, seperti public class, public static void
  - b. Deklarasikan ekspresi yang akan kita hitung
  - c. Print deklarasi ekspresi yang akan kita hitung
- 2) Kode Program dan Luaran



```
1 package operator1;
2
3 public class Latihan1Op{
4     public static void main(String[] args) {
5
6         int hasil = (2*3 + 6 / 2 - 4); //deklarasi ekspresi masalah
7         System.out.println("(2*3 + 6 / 2 - 4) = "+hasil); //print hasil deklarasi ekspresi
8     }
9 }
```

Problems Javadoc Declaration Console x

<terminated> Latihan1Op [Java Application] C:\Users\acer\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.4  
(2\*3 + 6 / 2 - 4) = 5

Luaran yang dihasilkan sudah sesuai, menghasilkan perhitungan yang juga sudah benar.

#### [No.1] Kesimpulan

Pada soal tersebut, perhitungan ekspresi  $(2*3 + 6 / 2 - 4) = 5$ . Dari hasil perhitungan tersebut, dapat kita simpulkan bahwa urutan prioritas Operator yang dijalankan pada program tersebut adalah dimulai dari operator perkalian (\*), yaitu  $2*3$  dijalankan terlebih dahulu, dilanjutkan pembagian (/), yaitu  $6 / 2$ , selanjutnya hasil kali dijumlahkan dengan hasil bagi, dilanjutkan dengan pengurangan.

Urutan :

$$\begin{aligned} &2*3 + 6 / 2 - 4 \\ &= 6 + 3 - 4 \\ &= 9 - 4 \\ &= 5 \end{aligned}$$

## [No. 2] Identifikasi Masalah

Pada soal diberikan kode

```
public class OperatorPenugasan {  
    public static void main(String[] args) {  
        // deklarasi nilai  
        int a = 20, b = 3;  
        //operator penugasan  
        b += a;  
        System.out.println("Penambahan : " + b);  
    }  
}
```

Dengan menghasilkan luaran

Penambahan : 23

Terdapat perintah untuk menampilkan perhitungan dengan operator ( -=, \*=, /=, %=).

## [No. 2] Analisis dan Argumentasi

Pada soal ini telah menampilkan operator penugasan berupa penambahan (+=). Untuk menyelesaikan soal yang diberikan, kita cukup menambahkan operator penugasan lain, yaitu pengurangan, perkalian, pembagian, dan modulus. Untuk menambahkan nya, kita cukup mengganti simbol di depan simbol sama dengan.

## [No. 2 ] Penyusunan Algoritma dan Kode Program

### 1) Algoritma

- Salin dan tempel kode program pada soal ke Eclipse atau Jdoodle
- Tambahkan kode perhitungan baru, beri perintah print untuk operator baru
- Ulangi sampai seluruh operator yang diperintahkan telah ditambahkan

### 2) Kode Program dan Luaran

```
1 package operatora1;  
2  
3 public class Latihan2Op {  
4     public static void main(String[] args) {  
5         // deklarasi nilai  
6         int a = 20, b = 3;  
7         //operator penugasan  
8         b += a; //Penambahan  
9         System.out.println("Penambahan : " + b);  
10        b -= a; //Pengurangan  
11        System.out.println("Pengurangan : " + b);  
12        b *= a; //Perkalian  
13        System.out.println("Perkalian : " + b);  
14        b /= a; //Pembagian  
15        System.out.println("Pembagian : " + b);  
16        b %= a; // Modulus atau sisa  
17        System.out.println("Modulus : " + b);  
18    }  
19 }  
20 }  
21
```

Problems Javadoc Declaration Console ×

<terminated> Latihan2Op [Java Application] C:\Users\acer\.p2\pool\plugins\org.eclipse.justj

Penambahan : 23  
Pengurangan : 3  
Perkalian : 60  
Pembagian : 3  
Modulus : 3

Luaran yang dihasilkan sudah sesuai dengan program yang disusun. Pada penambahan menghasilkan 23 karena nilai b ditambahkan dengan nilai a, yaitu  $3 + 20 = 23$ . Pada pengurangan menghasilkan nilai 3, karena nilai b yang diambil adalah nilai setelah penjumlahan sebelumnya, maka b bernilai 23. Sehingga, jika b dikurang a maka  $23 - 20 = 3$ . Untuk perkalian, b dikalikan dengan a, nilai b yang diambil adalah nilai yang telah mengalami pengurangan sebelumnya, yaitu 3. Maka  $b * a = 3 * 20 = 60$ . Untuk pembagian, b dibagi dengan a, nilai b yang diambil adalah nilai yang telah mengalami perkalian sebelumnya, yaitu 60. Maka  $b / a = 60 / 20 = 3$ . Untuk modulus, modulus adalah sisa dari pembagian. Bilangan yang dibagi adalah b dan a. Nilai b yang diambil adalah nilai b yang telah mengalami pembagian sebelumnya, yaitu 3. Maka,  $b \% a$ , modulus atau sisa dari nilai b terhadap pembagian nilai a. Nilai b 3, dibagi nilai a 20, maka  $3/20$  menghasilkan sisa atau modulus sebesar 3.

### [No. 2] Kesimpulan

Pada program tersebut, perhitungan yang dilakukan sesuai dengan operator nya masing-masing. Dari hasil perhitungan dapat kita lihat, jika kita melakukan perhitungan, kemudian melakukan perhitungan lagi dengan deklarasi yang sama, maka yang dihitung oleh program bukan nilai pada deklarasi awal, melainkan hasil setelah perhitungan.

### [No. 3] Identifikasi Masalah

Pada soal diberikan kode program

```
public class OperatorRelasional {
    public static void main(String[] args) {
        int nilaiA = 12;
        int nilaiB = 4;
        boolean hasil;

        System.out.println(" A = " + nilaiA + "\n B = " + nilaiB);
        // apakah A lebih besar dari B?
        hasil = nilaiA > nilaiB;
        System.out.println("\n Hasil A > B = "+ hasil);

        // apakah A lebih kecil dari B?
        hasil = nilaiA < nilaiB;
        System.out.println("\n Hasil A < B = "+ hasil);

        // apakah A lebih besar samadengan B?
        hasil = nilaiA >= nilaiB;
        System.out.println("\n Hasil A >= B = "+ hasil);

        // apakah A lebih kecil samadengan B?
        hasil = nilaiA <= nilaiB;
        System.out.println("\n Hasil A <= B = "+ hasil);

        // apakah nilai A sama dengan B?
        hasil = nilaiA == nilaiB;
        System.out.println("\n Hasil A == B = "+ hasil);

        // apakah nilai A tidak samadengan B?
        hasil = nilaiA != nilaiB;
        System.out.println("\n Hasil A != B = "+ hasil);
    }
}
```

Menghasilkan luaran

```
A = 12
B = 4
```

```
Hasil A > B = true
Hasil A < B = false
Hasil A >= B = true
Hasil A <= B = false
Hasil A == B = false
Hasil A != B = true
```

Pada soal, kita diperintahkan untuk menyusun kode program menggunakan operator relasional (<, >, <=, >=, =, ==, !=) agar nilai a dan b menghasilkan luaran true.

### [No. 3] Analisis dan Argumentasi

Untuk menghasilkan seluruh nilai a dan b berluaran true, cara yang saya usulkan adalah menambahkan NOT (!) di kode yang menghasilkan luaran false, agar menjadi true.

### [No.3 ] Penyusunan Algoritma dan Kode Program

- 1) Algoritma
  - a. Salin dan tempel kode program pada soal ke Eclipse atau Jdoodle
  - b. Run kode program
  - c. Periksa kode mana yang menghasilkan luaran false
  - d. Tambahkan operator NOT (!) pada kode program yang berluaran false
- 2) Kode Program dan Luaran

```
package operatora1;

public class Latihan30p {
    public static void main(String[] args) {
        int nilaiA = 12;
        int nilaiB = 4;
        boolean hasil;

        System.out.println(" A = " + nilaiA + "\n B = " + nilaiB);
        // apakah A lebih besar dari B?
        hasil = nilaiA > nilaiB;
        System.out.println("\n Hasil A > B = " + hasil);

        // apakah A tidak lebih kecil dari B?
        hasil = !(nilaiA < nilaiB);
        System.out.println("\n Hasil !(A < B) = " + hasil);
        //karena sebelumnya menghasilkan luaran false, kita balikkan menggunakan not agar menjadi true
        // apakah A lebih besar samadengan B?
        hasil = nilaiA >= nilaiB;
        System.out.println("\n Hasil A >= B = " + hasil);
        // apakah A tidak lebih kecil samadengan B?
        hasil = !(nilaiA <= nilaiB);
        System.out.println("\n Hasil !(A <= B) = " + hasil);
        //karena sebelumnya menghasilkan luaran false, kita balikkan menggunakan not agar menjadi true
        // apakah nilai A tidak sama dengan B?
        hasil = !(nilaiA == nilaiB);
        System.out.println("\n Hasil !(A == B) = " + hasil);
        //karena sebelumnya menghasilkan luaran false, kita balikkan menggunakan not agar menjadi true
        // apakah nilai A tidak sama dengan B?
        hasil = nilaiA != nilaiB;
        System.out.println("\n Hasil A != B = " + hasil);
    }
}
```

```
A = 12
B = 4

Hasil A > B = true

Hasil !(A < B) = true

Hasil A >= B = true

Hasil !(A <= B) = true

Hasil !(A == B) = true

Hasil A != B = true
```

Luaran yang dihasilkan sudah sesuai dengan permintaan soal, yaitu menghasilkan luaran true pada nilai a dan b.

### [No. 3] Kesimpulan

Pada program ini, Saya merubah nilai boolean (true atau false) pada program menggunakan operator NOT (!). Nilai boolean yang awalnya false, setelah mendapatkan operator NOT (!) menjadi true. Artinya, operator NOT (!) dapat membalikkan nilai yang ada. Sehingga semua nilai pada a dan b menghasilkan luaran true.

### [No. 4] Identifikasi Masalah

Pada soal, kita diberikan kode program

```
public class operator {  
    public static void main(String[] args) {  
        // deklarasi nilai  
        int a = 5;  
  
        System.out.println("a: " + a);  
        System.out.println("b: " + (a++));  
    }  
}
```

Dengan menghasilkan luaran

a: 5

b: 5

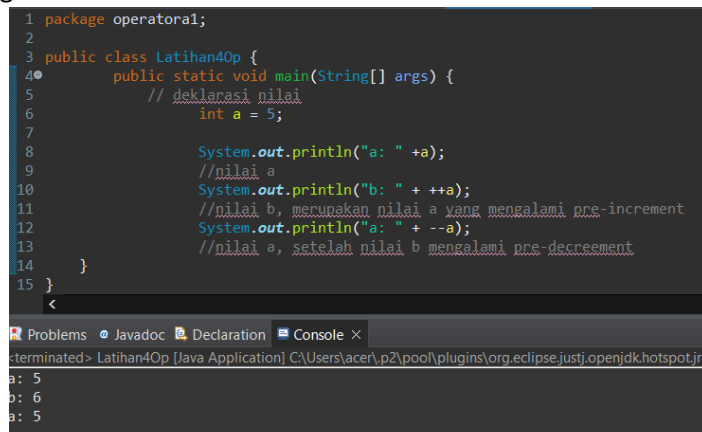
Kita diminta untuk merubah nilai menjadi 5 dan b menjadi 6 menggunakan pre/post increment dan pre/post decrement.

### [No.4] Analisis dan Argumentasi

Saya mengusulkan permasalahan ini dapat diatasi dengan cara merubah kode program dengan mengganti post increment pada soal menjadi pre increment. Alasan nya, agar perubahan nilai pada kode kita langsung muncul. Kemudian, agar melibatkan penggunaan decreement, Saya akan menggunakan pre-decreement untuk merubah kembali nilai yang telah mengalami pre-increment menjadi nilai aslinya.

### [No.4 ] Penyusunan Algoritma dan Kode Program

- 1) Algoritma
  - a. Salin dan tempel kode program pada soal ke Eclipse atau Jdoodle
  - b. Ganti kode program pada baris ke-10 menjadi pre-increment
  - c. Tambahkan perintah untuk merubah kembali kode b menjadi kode a dengan nilai 5 menggunakan pre-decreement
- 2) Kode program dan luaran



```
1 package operator1;  
2  
3 public class Latihan4Op {  
4     public static void main(String[] args) {  
5         // deklarasi nilai  
6         int a = 5;  
7  
8         System.out.println("a: " + a);  
9         //nilai a  
10        System.out.println("b: " + ++a);  
11        //nilai b, merupakan nilai a yang mengalami pre-increment  
12        System.out.println("a: " + --a);  
13        //nilai a, setelah nilai b mengalami pre-decrement  
14    }  
15 }
```

terminated> Latihan4Op [Java Application] C:\Users\acer\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre

a: 5  
b: 6  
a: 5

Pada program tersebut, luaran yang dihasilkan sesuai dengan permintaan. Dengan melibatkan pre/post increment dan pre/post decrement Saya telah menghasilkan nilai a = 5 dan nilai b = 6.

#### [No. 4] Kesimpulan

Pada kode program ini, kita dapat mengambil kesimpulan mengenai fungsi dari pre-increment dan pre-decrement. Dengan nilai awal a = 5, kita bisa mendapatkan nilai b = 6 dengan menambahkan pre-increment pada a. Kemudian, kita bisa mendapatkan kembali nilai a = 5 dengan menambahkan pre-decrement pada nilai b. Artinya, pre-increment akan menambahkan 1 nilai pada nilai kita dan pre-decrement akan mengurangi 1 nilai pada nilai kita.

#### [No. 5] Identifikasi Masalah

Pada soal, kita diberikan kode program

```
public class operator {  
    public static void main(String[] args) {  
        // deklarasi nilai  
        boolean a = true;  
        boolean b = false;  
  
        System.out.println("Hasil logika (a && b) : " + (a && b));  
    }  
}
```

Dengan menghasilkan luaran

```
Hasil logika (a && b) : false
```

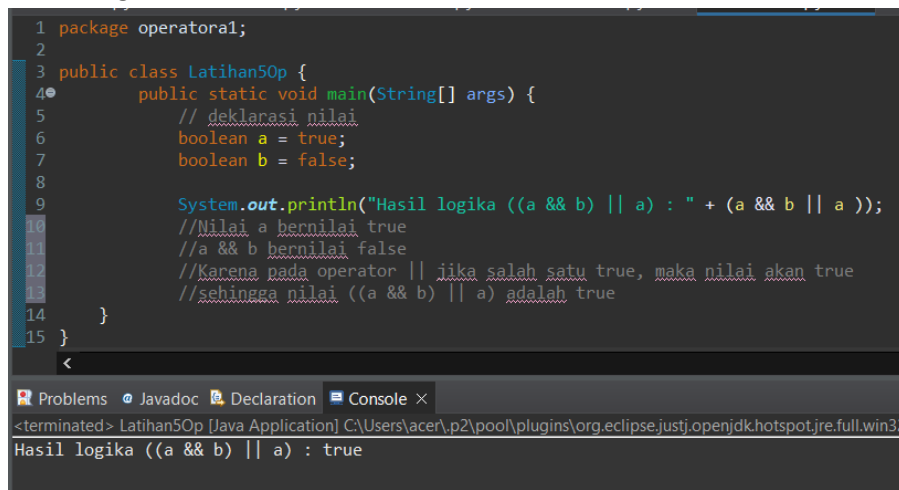
Kita diminta untuk membuat kode program yang menghasilkan luaran true dengan melibatkan operator && dan operator ||.

#### [No. 5] Analisis dan Argumentasi

Saya mengusulkan masalah ini dapat diselesaikan dengan menambahkan operator || dengan variabel a setelah melakukan operasi a && b. Alasan nya, karena nilai a && b menghasilkan luaran false, sedangkan variabel a bernilai true, dengan menggunakan operator || jika ada salah satu yang bernilai true, maka nilainya akan true.

#### [No.5 ] Penyusunan Algoritma dan Kode Program

- 1) Algoritma
  - a. Salin dan tempel kode program pada soal ke Eclipse atau Jdoodle
  - b. Tambahkan operasi || dengan variabel a setelah operasi a && b
- 2) Kode Program dan Luaran



```
1 package operator1;  
2  
3 public class Latihan5Op {  
4     public static void main(String[] args) {  
5         // deklarasi nilai  
6         boolean a = true;  
7         boolean b = false;  
8  
9         System.out.println("Hasil logika ((a && b) || a) : " + (a && b || a));  
10        //Nilai a bernilai true  
11        //a && b bernilai false  
12        //Karena pada operator || jika salah satu true, maka nilai akan true  
13        //sehingga nilai ((a && b) || a) adalah true  
14    }  
15 }
```

Problems Javadoc Declaration Console ×

<terminated> Latihan5Op [Java Application] C:\Users\acer\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32

Hasil logika ((a && b) || a) : true

Pada program ini, luaran yang dihasilkan sudah sesuai permintaan. Luaran telah menghasilkan nilai true dengan melibatkan operator `||`. Prioritas operatornya adalah `a&&b` terlebih dahulu yang menghasilkan false. Sehingga menjadi `(false) || a (true)`. Karena `a` bernilai true dan pada operator `||` jika salah satu bernilai true maka akan menghasilkan true.

#### [No. 5] Kesimpulan

Pada program tersebut, kita dapat mengambil kesimpulan bahwa untuk penggunaan operator `&&` jika nilai adalah `(true) && (false)` akan menghasilkan luaran false. Sedangkan untuk penggunaan operator `||` jika nilai adalah `(false) || true` akan menghasilkan luaran true.

#### [No. 6] Identifikasi Masalah

Pada soal, kita diberikan kode program

```
public class OperatorKondisi{  
    public static void main( String[] args ){  
        String status = "";  
        int nilai = 80;  
        status = (nilai > 60)?"Lulus":"Gagal";  
        System.out.println( status );  
    }  
}
```

Dengan luaran

Lulus

Kita diminta untuk membuat kode program apabila `jam < 12` maka tampil "Selamat Pagi", apabila `jam > 12` maka tampil "Selamat Malam", dengan diketahui nama variabel `Jam = 12`.

#### [No. 6] Analisis dan Argumentasi

Saya mengusulkan masalah ini dapat diselesaikan dengan menggunakan operator Kondisional atau Ternary. Caranya dengan menyatakan nya dalam bentuk `ekspresi1?ekspresi2:ekspresi3`.

Dengan,

`ekspresi1` = boolean dengan hasil true atau false.

`ekspresi2` = jika nilai boolean `ekspresi1` bernilai true.

`ekspresi3` = jika nilai boolean `ekspresi1` bernilai false.

Ekspresi 1 disini adalah `jam < 12`

Ekspresi 2 adalah selamat malam

Ekspresi 3 adalah selamat pagi

#### [No. 6] Penyusunan Algoritma dan Kode Program

##### 1) Algoritma

- a. Salin dan tempel kode program pada soal ke Eclipse atau Jdoodle
- b. Ubah deklarasi nilai
- c. Ubah ekspresi pada operator ternary

## 2) Kode Program dan Luaran

```
1 package operatoral1;
2
3 public class Latihan6op {
4     public static void main( String[] args ){
5         String status = "";
6         int jam = 12; //deklarasi nilai
7         status = (jam > 12)?"Selamat Malam":"Selamat Pagi";//operator ternary
8         System.out.println( status );
9     }
10 }
11
```

<

Problems Javadoc Declaration Console x

<terminated> Latihan6op [Java Application] C:\Users\acer\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x

Selamat Pagi

Pada program ini, luaran yang dihasilkan sudah sesuai permintaan. Pada program nilai yang dideklarasikan sebesar 12. Karna pada ekspresi1 dinyatakan jam > 12, nilai jam = 12 dinyatakan sebagai false. Karena false, maka program akan menampilkan ekspresi 3, yaitu selamat pagi.

### [No. 6] Kesimpulan

Pada program tersebut penggunaan operator ternary digunakan sebagai operator kondisional. Pada operator ternary terdapat 3 ekspresi, ekspresi pertama menyatakan boolean dengan hasil true atau false. Jika ekspresi pertama dinyatakan sebagai true, maka program akan menampilkan ekspresi kedua, sebaliknya jika ekspresi pertama dinyatakan sebagai false, maka program akan menampilkan ekspresi ketiga.

### [No. 7] Identifikasi Masalah

Pada soal kita diberikan kode program,

```
public class operatorBitwise {
    public static void main(String[] args) {
        int a = 10;
        int b = 7;
        int hasil;

        hasil = a & b;
        System.out.println("Hasil dari a & b : " + hasil );

        hasil = a | b;
        System.out.println("Hasil dari a | b : " + hasil );

        hasil = a ^ b;
        System.out.println("Hasil dari a ^ b : " + hasil );
    } }
```

Dengan Luaran

```
Hasil dari a & b : 6
Hasil dari a | b : 7
Hasil dari a ^ b : 1
```

Kita diperintahkan untuk menambah perhitungan dengan menggunakan operator bitwise : >>,<<



### [No. 7] Analisis dan Argumentasi

Pada soal ini, saya mengusulkan untuk menambahkan kode yang sama namun dengan operator >>, << kemudian menjalankannya dan menganalisis luaran setelah kode tersebut dijalankan

### [No. 7] Penyusunan Algoritma dan Kode Program

- 1) Algoritma
  - a. Salin dan tempel kode program pada soal ke Eclipse atau Jdoodle
  - b. Tambahkan kode program untuk operator <<, >>
  - c. Jalankan kode program
- 2) Kode Program dan Luaran

```
1 package operator1;
2
3 public class Latihan7Op {
4     public static void main(String[] args) {
5         int a = 10; // 1010
6         int b = 7; // 0111
7         int hasil;
8
9         hasil = a & b; |
10        System.out.println("Hasil dari a & b : " + hasil );
11
12        hasil = a | b;
13        System.out.println("Hasil dari a | b : " + hasil );
14
15        hasil = a ^ b;
16        System.out.println("Hasil dari a ^ b : " + hasil );
17
18        hasil = a >> b;
19        System.out.println("Hasil dari a >> b : " + hasil );
20
21        hasil = a << b;
22        System.out.println("Hasil dari a << b : " + hasil );
23
24    } }
25
```

```
<terminated> Latihan7Op [Java Applicatio
Hasil dari a & b : 2
Hasil dari a | b : 15
Hasil dari a ^ b : 13
Hasil dari a >> b : 0
Hasil dari a << b : 1280
<
```

Luaran telah sesuai dengan perintah yang dihasilkan, juga telah mendapatkan penambahan kode untuk operator << dan >>

### [No. 7] Kesimpulan

Pada soal no 7, kode program yang dijalankan menggunakan operator bitwise. Fungsi dari operator bitwise adalah menjalankan operasi secara langsung terhadap bit suatu bilangan. Pada bilangan biner, nilai 1 berarti true dan nilai 0 berarti false. Dari operasi-operasi tersebut, menghasilkan luaran yaitu :

Hasil dari  $a \& b$  : 2

Hasil dari  $a | b$  : 15

Hasil dari  $a \wedge b$  : 13

Hasil dari  $a \gg b$  : 0

Hasil dari  $a \ll b$  : 1280

Dengan  $a = 10$ , dan  $b = 7$

Untuk mengetahui darimana asal hasil-hasil pada luaran di atas, kita perlu mengkonversikan nilai  $a$  dan  $b$  dari desimal menuju biner

Konversi  $a = 10$  menuju biner

Caranya dengan membagi 10 dengan 2 sampai tidak dapat dibagi lagi

$10/2 = 5$              $\rightarrow$  sisa 0

$5/2 = 2$              $\rightarrow$  sisa 1

$2/2 = 1$              $\rightarrow$  sisa 0

$1/2 = 0$              $\rightarrow$  sisa 1

Nilai biner diambil dari sisa terakhir ke awal, yaitu 1,0,1,dan 0.

Maka konversi biner dari 10 adalah 1010

Dengan cara yang sama, kita bisa mendapatkan nilai biner dari 7.

Konversi  $b = 7$  menuju biner

$7/2 = 3$              $\rightarrow$  sisa 1

$3/2 = 1$              $\rightarrow$  sisa 1

$1/2 = 0$              $\rightarrow$  sisa 1

Maka, konversi biner dari 7 adalah 111, agar digitnya sama dengan 10, kita tambahkan 0 di kiri menjadi 0111

Tiap-tiap operator bitwise memiliki arti sebagai berikut.

Operator  $\&$  : Bernilai true jika kedua nya true

1010	0010
0111	

Operator  $|$  : Bernilai true jika salah satunya true

1010	1111
0111	

Operator  $\wedge$  : Bernilai true jika keduanya tidak true atau keduanya tidak false

1010	1101
0111	

Operator >> : menggeser ke kanan

a >> 7, berarti nilai a digeser sebanyak 7 kali ke kanan, Contoh :

Nilai a = 10 = 1010

Ketika a >> 1 maka, 0101

Ketika a >> 2 maka, 0010

Dengan cara yang sama, jika a digeser sampai 7 kali akan menyisakan 0000

Operator << : menggeser ke kiri

a << 7, berarti nilai a digeser sebanyak 7 kali ke kiri, contoh

Nilai a = 10 = 1010

Ketika a << 1 = 10100

Ketika a << 2 = 101000

Dengan cara yang sama, jika a << 7 maka akan menghasilkan 10100000000

Selanjutnya konversi tiap-tiap operator dapat dilihat di tabel berikut

Desimal a	Desimal b	Biner a	Biner b	Operator	Hasil Operasi (biner)	Hasil Operasi (Desimal)
10	7	1010	0111	a & b	0010	2
10	7	1010	0111	a   b	1111	15
10	7	1010	0111	a ^ b	1101	13
10	7	1010	0111	a >> b	0000	0
10	7	1010	0111	a << b	10100000000	1280

Cara mengkonversi bilangan biner menjadi desimal adalah dengan membuat  $2^n$  dari bagian paling kanan bilangan biner kemudian dikalikan ke atasnya. Contoh :

Konversi 0010 menjadi desimal :

0	0	1	0
$2^3$	$2^2$	$2^1$	$2^0$
$2^3 * 0$	$2^2 * 0$	$2^1 * 1$	$2^0 * 0$
0	0	2	0

Setelah dikalikan, jumlahkan hasilnya  $0 + 0 + 2 + 0 = 2$

Jadi, hasil konversi 0010 menuju bilangan desimal adalah 2

Untuk mengonversi bilangan biner lain nya menuju desimal, menggunakan cara yang sama