

Tugas 1

Makassar, 21 Februari 2025

**PRAKTIKUM  
PEMROGRAMAN BERBASIS OBJEK**



Nama : Wa Ode Namida Heiwa  
Stambuk : 13020230132  
Frekuensi : TI\_PBO-12 (B2)  
Dosen : Mardiyah Hasnawi, S.Kom., M.T., MTA

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS ILMU KOMPUTER  
UNIVERSITAS MUSLIM INDONESIA  
MAKASSAR  
2025**

## Output dari isi file tugas 1 di direktori source code

```
Command Prompt
Directory of C:\xampp\htdocs\tugas PBO\tugas 1

27/02/2025  22:12  <DIR>      .
27/02/2025  09:35  <DIR>      ..
27/02/2025  17:37                876 Asgdll.class
26/02/2025  21:39                272 Asgdll.java
27/02/2025  17:39                893 Assign.class
26/02/2025  21:45                266 Assign.java
27/02/2025  17:40            1.371 ASIGNi.class
26/02/2025  21:50            894 ASIGNi.java
27/02/2025  17:40            1.074 BacaData.class
26/02/2025  22:01            531 BacaData.java
27/02/2025  17:41            1.412 Bacakar.class
26/02/2025  22:24            1.045 Bacakar.java
27/02/2025  17:47            572 Casting1.class
26/02/2025  22:26            693 Casting1.java
27/02/2025  17:50            1.748 Casting2.class
26/02/2025  22:30            879 Casting2.java
27/02/2025  17:51            1.044 Ekspresi.class
26/02/2025  22:32            443 Ekspresi.java
27/02/2025  17:56            1.258 Ekspresi1.class
26/02/2025  22:36            815 Ekspresi1.java
27/02/2025  17:59            492 Hello.class
26/02/2025  22:59            381 Hello.java
27/02/2025  18:01            872 Incr.class
26/02/2025  22:40            270 Incr.java
27/02/2025  18:03            1.113 Oper1.class
26/02/2025  22:43            616 Oper1.java
27/02/2025  18:04            1.167 Oper2.class
26/02/2025  22:46            766 Oper2.java
27/02/2025  18:05            424 Oper3.class
26/02/2025  22:48            519 Oper3.java
27/02/2025  18:07            1.072 Oper4.class
26/02/2025  22:51            513 Oper4.java
27/02/2025  22:13            2.734 Oprator.class
27/02/2025  22:12            3.026 Oprator.java
                          32 File(s)
                          30.051 bytes
```

**Ket:** Pertama-tama masuk dulu ke folder tempat kita menyimpan file atau source yang sudah kita ketik.

### 1. Output Program Asgdll

```
Command Prompt
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

C:\Users\asus>cd C:\xampp\htdocs\tugas PBO\tugas 1

C:\xampp\htdocs\tugas PBO\tugas 1>javac Asgdll.java

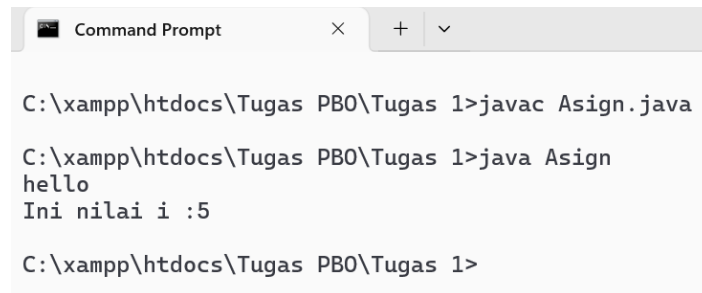
C:\xampp\htdocs\tugas PBO\tugas 1>java Asgdll
f : 20.0
f11: 10.0

C:\xampp\htdocs\tugas PBO\tugas 1>
```

**Ket :** Javac itu berarti java compiler, kemudian diberi ekstensi “.java” agar kompiler java bisa membaca file. “Javac Asgdll.java” itu perintah untuk mengkompilasi Asgdll.java agar menjadi Asgdll.class, nahh diubah ke extension .class agar bisa dibaca oleh komputer melalui Java Virtual Machine(JVM). Perintah Java Asgdll untuk menjalankan java yang telah dikompilasi. Setelah di kompilasi maka keluar lah output:

Deklarasi variabel float f = 20.0f; Dimana f itu float, kemudian double f11; itu berarti f11 bertipe double. dan f11=10.0f; berarti nilai 10.0f itu diubah ke variable f11 yang Dimana itu bertipe double. Kemudian sistem println output mencetak nilai variable f dan f11 ke konsol.

## 2. Output Program Assign



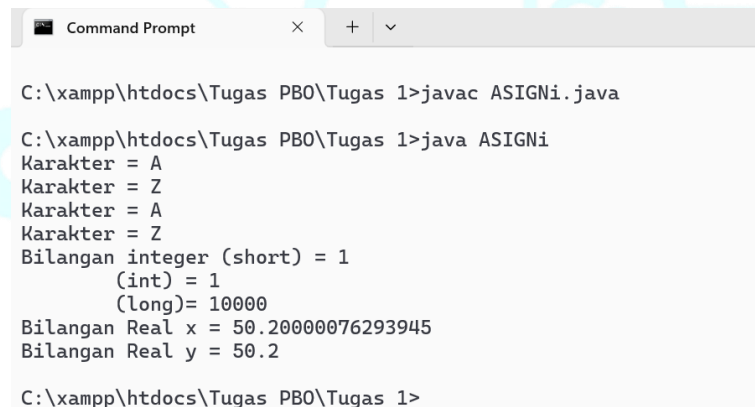
```
C:\xampp\htdocs\Tugas PBO\Tugas 1>javac Assign.java

C:\xampp\htdocs\Tugas PBO\Tugas 1>java Assign
hello
Ini nilai i :5

C:\xampp\htdocs\Tugas PBO\Tugas 1>
```

**Ket :** Pada code tertulis `int i`; yang berarti variable `i` tipenya integer. Kemudian pada code `System.out.print("hello\n");` yang berarti mencetak string **“hello”** ke konsol. Pada `i=5`; berarti kita mengisi nilai 5 kedalam variable `i`. Setelah itu kita akan mencetak lagi `System.out.println("Ini nilai i : " + i);` yang Dimana nilai `i` nya tadi bernilai 5, maka itulah yang akan di tampilkan ke konsol.

## 3. Output Program ASIGNi



```
C:\xampp\htdocs\Tugas PBO\Tugas 1>javac ASIGNi.java

C:\xampp\htdocs\Tugas PBO\Tugas 1>java ASIGNi
Karakter = A
Karakter = Z
Karakter = A
Karakter = Z
Bilangan integer (short) = 1
(int) = 1
(long)= 10000
Bilangan Real x = 50.20000076293945
Bilangan Real y = 50.2

C:\xampp\htdocs\Tugas PBO\Tugas 1>
```

**Ket :** Pada code yang tertulis, `short ks = 1`; `short` itu dia untuk bil.bulat yang kecil dan nilai `ks` itu 1. `int ki = 1`; tipe data `int` untuk bilangan bulat pada umumnya dan nilai `ki` nya itu 1. `long kl = 10000`; digunakan tipe data `long` untuk bilangan bulat yang lebih besar missal dari integer. `char c = 65`; tipe data `char` yaitu karakter `c` menyimpan nilai 65 dan setiap karakter ternyata memiliki kode numerik yang sesuai kode ASCII, 65 adalah kode 'A'. `char c1 = 'Z'`; ada juga cara lain menginisialisasi `char` dengan langsung memberi karakter misalnya 'Z'. `double x = 50.2f`; pakai tipe data `double` itu karena nilainya desimal dan juga perhitungan akurasi nya tinggi. `float y = 50.2f`; `float` ini tipe data untuk bilangan riil dengan presisi yang rendah. Kemudian setelah penjelasan kode-kode tadi maka sistem akan memprint perintah sesuai output dan nilai yang dituliskan kemudian mengkonversi data, seperti misalnya tadi ada `System.out.println("Karakter = " + c);` maka dia akan menghasilkan karakter 'A', karena sudah dikonversi langsung tadi dari 65 menjadi A, begitu seterusnya kebawah secara berurut.

#### 4. Output Program BacaData

```
Command Prompt
C:\xampp\htdocs\Tugas PBO\Tugas 1>javac BacaData.java
C:\xampp\htdocs\Tugas PBO\Tugas 1>java BacaData
Contoh membaca dan menulis, ketik nilai integer: 5
8
Nilai yang dibaca : 8
C:\xampp\htdocs\Tugas PBO\Tugas 1>
```

**Ket :** `int a;` deklarasi variable `a` dengan tipe integer untuk simpan inputan kita. `Scanner masukan;` deklarasi variable masukan dengan tipe `Scanner` untuk baca inputan.

`System.out.print ("Contoh membaca dan menulis, ketik nilai integer: \n");` dengan ini maka yang akan tampil di konsol sesuai kalimat ini, kita disuruh memasukkan nilai integer. `masukan = new Scanner(System.in);` maka kita membuat objek scanner baru. `a = masukan.nextInt();` /\* coba ketik : `masukan.nextInt();` ; Apa akibatnya ?\*/ maka sistem akan membaca nilai integer yg kita masukkan dan disimpan ke variable `a`, terus pertanyaannya apa akibat dari perintha di atas? Yaitu, jika kita menulis `masukan.nextInt();` tanpa `a =` , maka nilai yg kita masukkan akan tetap bisa dibaca tapi tidak bisa disimpan atau nilai nya hilang atau tidak tersimpan, tapi program tetap berjalan hanya saja nilai yg kita masukkan tidak akan dipakai dalam program.

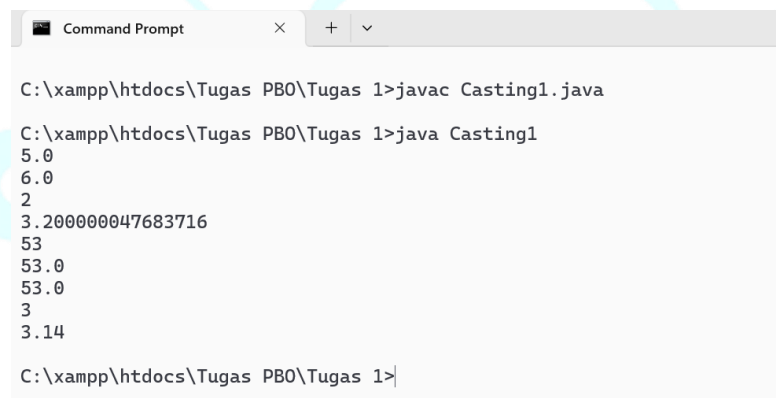
#### 5. Output Program Bacakar

```
Command Prompt
C:\xampp\htdocs\Tugas PBO\Tugas 1>javac Bacakar.java
C:\xampp\htdocs\Tugas PBO\Tugas 1>java Bacakar
hello
baca 1 karakter : a
baca 1 bilangan : 9
a
9
bye
C:\xampp\htdocs\Tugas PBO\Tugas 1>
```

**Ket :** `import java.io.BufferedReader;` Mengimpor kelas `BufferedReader` untuk membaca karakter secara efisien (baris per baris). `import java.io.IOException;` Mengimpor `IOException` untuk menangani kesalahan input/output. `import java.io.InputStreamReader;` digunakan untuk mengubah input byte menjadi karakter. Kemudian kita masuk dalam deklarasi variable mulai dari variable dengan

tipe data char dan bil dengan tipe data integer , kemudian InputStreamReader isr = new InputStreamReader(System.in); perintah membuat objek baru supaya membaca inputan, BufferedReader dataIn = new BufferedReader(isr); agar membaca input baris per baris, BufferedReader dataIn = new BufferedReader(new InputStreamReader(System.in)); membuat objek BufferedReader lainnya dengan cara yang beda. System.out.print("hello\n"); print “hello” pada konsol. System.out.print("baca 1 karakter : "); dan cc = dataIn.readLine().charAt(0); kita input karakter kemudian sistem mebaca nya dan menyimpannya di cc. System.out.print("baca 1 bilangan : "); dan bil = Integer.parseInt(dataIn.readLine()); sama seperti sebelumnya kita input bilangan dan dibaca sistem kemudian dikonversi jadi integer dan disimpan di bil. System.out.print(cc + "\n" + bil + "\n"); akan ditampilkan nilai cc dan bil pada konsol, missal tadi saya masukkan karakter “u” dan bilangan “1” pada cc dan bil . System.out.print("bye \n"); kemudian dicetak pula “bye”.

## 6. Output Program Casting1



```

C:\xampp\htdocs\Tugas PB0\Tugas 1>javac Casting1.java

C:\xampp\htdocs\Tugas PB0\Tugas 1>java Casting1
5.0
6.0
2
3.200000047683716
53
53.0
53.0
3
3.14

C:\xampp\htdocs\Tugas PB0\Tugas 1>|

```

**Ket :** Pada kode ini int a = 5, b = 6; float d = 2.f, e = 3.2f; char g = '5'; double k = 3.14; ada tipe data integer, float, char, dan double dengan masing” variabelnya yang sudah terisi nilai masing-masing. Kemudian kita print dengan perintah yang sudah ada. Maka keluarlah output seperti diatas yang mana variable yg sebelumnya punya tipe data integer bisa berubah menjadi float dan sebagainya karena sesuai perintah yang tertera pada output yang diinginkan. Dengan ini berarti program bisa mengubah tipe data atau casting dalam tipe data primitif, dann dengan adanya perubahan tipe data ini maka ada juga potensi perubahan data contoh menjadi desimal atau nilai presisi yang berubah seperti tadi contoh ada 3.2 maka bisa nilai presisisnya berubah menjadi 3.200000047683716.

## 7. Output Program Casting2

```
Command Prompt
C:\xampp\htdocs\Tugas PB0\Tugas 1>javac Casting2.java
C:\xampp\htdocs\Tugas PB0\Tugas 1>java Casting2
a : 67
k : 45.0
d : 100.0
n : 9
m : 5
l : 3.2
k : 67.0
c : 9.0
l : 3.2
C:\xampp\htdocs\Tugas PB0\Tugas 1>
```

**Ket :** Pada kode `int a = 8, b = 9; float d = 2.f, e = 3.2f; char g = '5'; double k = 3.14; String n = "67", m = "45", l = "100";` ada beberapa tipe data yang dimana variabelnya sudah terisi oleh nilai masing-masing variabel itu. Kemudian kita konversi data mulai dari string ke numerik, numerik ke string sampai numerik yang menggunakan wrapper classes, dengan ini maka program menggunakan metode **`parseInt()`, `parseDouble()`, `parseFloat()`, dan `valueOf()`** untuk konversi tipe data itu semua. Maka tampilah output seperti pada gambar.

## 8. Output Program Ekspresi

```
Command Prompt
C:\xampp\htdocs\Tugas PB0\Tugas 1>javac Ekspresi.java
C:\xampp\htdocs\Tugas PB0\Tugas 1>java Ekspresi
x = 1
y = 2
hasil ekspresi = (x<y)?x:y = 1
C:\xampp\htdocs\Tugas PB0\Tugas 1>
```

**Ket :** Pada program ini kita menggunakan operator kondisional. Deklarasi variabel `x` dan `y` dengan tipe data integer dimana keduanya sudah diberi nilai masing-masing. Kemudian program mencetak nilai `x` ke konsol begitu juga `y` sesuai dengan perintah yang ada di kode program. Setelah itu, pada perintah output ini `System.out.print("hasil ekspresi = (x<y)?x:y = " + ((x < y) ? x : y));`, jika `x < y` benar, maka ekspresi kembalikan nilai `x`. Dan jika `x < y` salah, maka ekspresi kembalikan nilai `y`. Jadi karena `x < y` itu benar, maka ekspresi mengembalikan nilai `x = 1`.

## 9. Output Program Ekspresi1

```
Command Prompt
C:\xampp\htdocs\Tugas PBO\Tugas 1>javac Ekspresi1.java
C:\xampp\htdocs\Tugas PBO\Tugas 1>java Ekspresi1
x/y (format integer) = 0
x/y (format float) = 0
x/y (format integer) = 0.5
x/y (format float) = 0.5
float(x)/float(y) (format integer) = 0.5
float(x)/float(y) (format float) = 0.5
x/y (format integer) = 3
x/y (format float) = 3
C:\xampp\htdocs\Tugas PBO\Tugas 1>
```

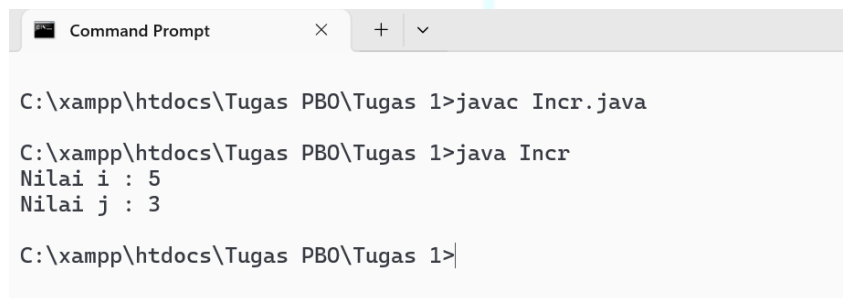
**Ket :** Pada Program ini menunjukkan perbedaan antara pembagian integer dan pembagian float Deklarasi variable x dan y dengan tipe data integer Dimana keduanya sudah diberi nilai masing”, Dan Deklarasi variable fx dan fy dengan tipe data float Dimana keduanya sudah diberi nilai masing” . System.out.print("x/y (format integer) = " + x / y); bagi x dan y atau 1 bagi 2 dan cetak sebagai int = 0. System.out.print("\nx/y (format float) = " + x / y); sama saja dan hasilnya masih tetap 0 karena tipe datanya masih sama” int. jadi supaya hasilnya tidak 0 makanya diubah jadi fx dan fy . System.out.print("\nx/y (format integer) = " + fx / fy); dan System.out.print("\nx/y (format float) = " + fx / fy); , dibagi fx dengan fy atau 1.0 bagi 2.0 yang hasilnya itu adalah 0.5. Kemudian sekarang kita melakukan casting(mengubah tipe data) sebelum pembagian, tadi kan x dan y itu adalah int maka itu kita casting menjadi float dengan kode program seperti ini System.out.print("\nfloat(x)/float(y) (format integer) = " + (float) x / (float) y); System.out.print("\nfloat(x)/float(y) (format float) = " + (float) x / (float) y); dan hasilnya masih 0.5 . Setelah itu, kita melakukan pembagian baru dengan nilai x = 10; y = 3; kemudian kita bagi dengan perintah output System.out.print("\nx/y (format integer) = " + x / y); System.out.print("\nx/y (format float) = " + x / y); dengan nilai x/y atau 10/3 hasilnya 3 karena integer makanya jadi bil bulat. Dan tampillah output seperti diatas .

## 10. Output Program Hello

```
Command Prompt
C:\xampp\htdocs\Tugas PBO\Tugas 1>javac Hello.java
C:\xampp\htdocs\Tugas PBO\Tugas 1>java Hello
Hello
Hello World
Welcome
C:\xampp\htdocs\Tugas PBO\Tugas 1>
```

**Ket :** Pada program ini membedakan antara `System.out.print()` dan `System.out.println()`. “Hello” itu Output dari `System.out.print("Hello");` , “Hello World” itu Output dari `System.out.print("\nHello ");` dan `System.out.println("World");`. Karena `\n` di awal `System.out.print("\nHello ");` maka baris baru dibuat sebelum kata "Hello". Kemudian, “Welcome” dari Output `System.out.println("Welcome");` . Bedanya itu hanya `print()` cetak string di konsol tidak tmbah baris baru tapi kalau `println()` tambah baris baru di akhir.

## 11. Output Program Incr



```
Command Prompt
C:\xampp\htdocs\Tugas PB0\Tugas 1>javac Incr.java
C:\xampp\htdocs\Tugas PB0\Tugas 1>java Incr
Nilai i : 5
Nilai j : 3
C:\xampp\htdocs\Tugas PB0\Tugas 1>
```

**Ket :** Pada Program ini menunjukkan efek dari operator increment (`++`). Pada kode program diatas variable `i` dan `j` memiliki tipe data integer, Dimana `i` memiliki nilai 3 kemudian `j = i++`; yg berarti nilai `i` diberi sementara ke `j`. Nah, `i` jadi meningkat menjadi 4. Kemudian pada `System.out.println("Nilai i : " + (++i) + "\nNilai j : " + j)`; yang Dimana `++i = i` meningkat menjadi 5 sebelum nilainya dipakai. Jadi `j` tadi sudah bernilai 3 dan `i` bernilai 5 . `i : 5; j : 3;` . Perbedaananya `post-increment(i++)` : Nilai variabel digunakan terlebih dahulu, kemudian variabel ditingkatkan. Sementara `Pre-increment (++i)`: Variabel ditingkatkan terlebih dahulu, kemudian nilainya digunakan.



## 12. Output Program Oper1

```
Command Prompt

C:\xampp\htdocs\Tugas PBO\Tugas 1>javac Oper1.java

C:\xampp\htdocs\Tugas PBO\Tugas 1>java Oper1
n = 10
x = 1
y = 2
n & 8 = 8
x & ~ 8 = 1
y << 2 = 8
y >> 3 = 0

C:\xampp\htdocs\Tugas PBO\Tugas 1>
```

**Ket :** Pada program ini membahas pemakaian beberapa operator terhadap bit :

int n = 10 (10 dalam biner adalah 1010), int x = 1; (1 adalah 1), int y = 2 (2 di biner adalah 10). Kemudian program minta sistem untuk memprint 'n,x,y' beserta nilainya dengan perintah output masing". Tampilah output n = 10, x = 1, y = 2.

Dan selanjutnya pada perintah System.out.println("n & 8 = " + (n & 8)), yang Dimana n & 8, nahh n tadi bernilai (10) dalam biner = 1010, (8) dalam biner = 1000, operator (&) atau (AND) ini menghasilkan 1 kalau kedua bit yang sesuai itu adalah 1.

1010&1000 = 1000 ( yang berarti 8 dalam desimal ). Maka System.out.println("n & 8 = " + (n & 8)); tampilah output n & 8 = 8 .

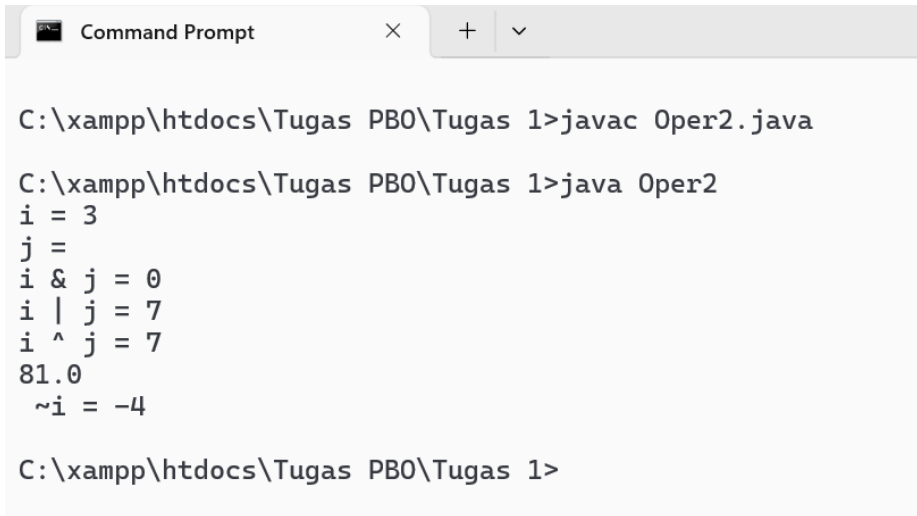
Pada x & ~8 . (8) biner = 1000, (~8) atau (NOT 8) berarti membalikkan semua bit nya menjadi = 0111 (7 dalam desimal), (x) itu yang berarti (1) dalam biner = 0001. Jadi kesimpulannya 0001&0111 = 0001 (1 dalam desimal) , tampilah x & ~ 8 = 1.

Pada y << 2 , y(2) dalam biner = 10, (<< 2) : Left shift 2 artinya geser bit ke kiri sebanyak 2 kali, kemudian ditambah 0 disebelah kanan. Jadi 10 << 2 = 1000 (8 dalam desimal) . Tampilah output

y << 2 = 8

Pada y >> 3 , sama juga seperti tadi y(2) dalam biner = 10, (>> 3) right shift 3 berarti geser juga bit tapi ke kanan sebanyak 3 kali, terus tambah juga 0 di kiri, Jadi 10 >> 3 = 0000 (hasil nya 0 ) . Tampilah output y >> 3 = 0.

### 13. Output Program Oper2



```
Command Prompt

C:\xampp\htdocs\Tugas PBO\Tugas 1>javac Oper2.java

C:\xampp\htdocs\Tugas PBO\Tugas 1>java Oper2
i = 3
j =
i & j = 0
i | j = 7
i ^ j = 7
81.0
~i = -4

C:\xampp\htdocs\Tugas PBO\Tugas 1>
```

**Ket :** Pada program ini kita membahas pemakaian beberapa operator terhadap RELATIONAL DAN bit, pertama-tama kita deklarasikan dulu variabelnya jadi (i) dan (j) tipe datanya (char), kemudian kita analisis nilainya i = 3; ( 3 dalam biner 00000011), j = 4; (4 dalam biner 00000100).

Pada `System.out.println("i = " + (int) i);` berarti dicetak nilai sebagai integer, yaitu 3.

Pada `System.out.println("j = " + j);` berarti cetak nilai j sebagai karakter, karena variabel j di deklarasikan sebagai char.

Pada `System.out.println("i & j = " + (i & j));` kita melakukan operasi AND bitwise antara i dan j. 3 (00000011) & 4 (00000100) = 0 (00000000).

Pada `System.out.println("i | j = " + (i | j));` kita melakukan operasi OR bitwise antara i dan j. 3 (00000011) | 4 (00000100) = 7 (00000111).

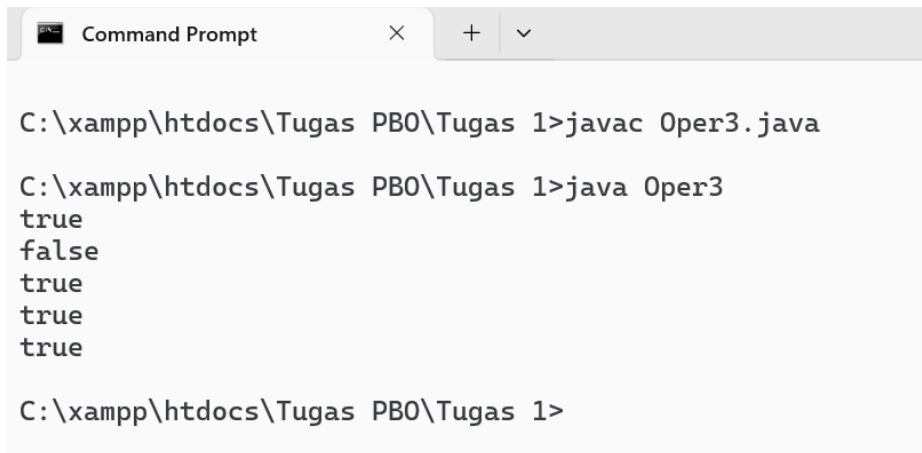
Pada `System.out.println("i ^ j = " + (i ^ j));` kita melakukan operasi XOR bitwise antara i dan j. 3 (00000011) ^ 4 (00000100) = 7 (00000111). Karena “^” bukan pangkat di java tapi XOR.

Pada `System.out.println(Math.pow(i, j));` dihitung i pangkat j menggunakan `Math.pow()`, yaitu 3 pangkat 4 = 81.0 . barulah `Math.pow()` digunakan untuk menghitung perpangkatan.

Pada `System.out.println("~i = " + ~i);` dilakukan operasi NOT bitwise pada i. NOT 3 (00000011) = -4 (11111100 dalam biner).

Setelah semua nya sudah maka keluarlah output seperti pada gambar.

## 14. Output Program Oper3



```
Command Prompt

C:\xampp\htdocs\Tugas PBO\Tugas 1>javac Oper3.java

C:\xampp\htdocs\Tugas PBO\Tugas 1>java Oper3
true
false
true
true
true

C:\xampp\htdocs\Tugas PBO\Tugas 1>
```

**Ket :** Pada program ini membahas Operator dalam konteks if statement.

- `if (true && true) { System.out.println(true && true); }`

`true && true` adalah operasi AND logis. Karena keduanya true, hasilnya true. Dan pada output yang diminta juga true true jadi hasilnya True.

- `if (true & true) { System.out.println(true & false); }`

Sebenarnya seharusnya **true & true** adalah operasi AND bitwise. Karena keduanya true, hasilnya true, Tapi karena yang diminta pada output itu true & false, jadi outputnya False, walaupun statementnya itu true.

- `if (true) { System.out.println(true); }`

Disini kondisi true selalu benar, jadi blok if akan dieksekusi, maka akan ditampilkan True.

- `if (true || true) { System.out.println(true); }`

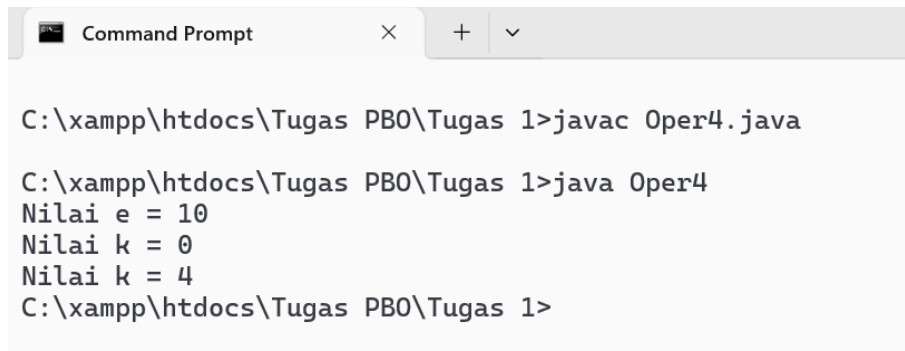
`true || true` adalah operasi OR logis. Karena setidaknya satu dari keduanya true, hasilnya True.

- `if (true | false) { System.out.println(true | false); }`

**true | false** adalah operasi OR bitwise. Karena setidaknya satu dari keduanya true, hasilnya True. Sama saja tadi seperti || adalah OR logis (short-circuit), sedangkan | adalah OR bitwise.

Maka tampilah semua output seperti Digambar.

## 15. Output Program Oper4



```
Command Prompt

C:\xampp\htdocs\Tugas PB0\Tugas 1>javac Oper4.java

C:\xampp\htdocs\Tugas PB0\Tugas 1>java Oper4
Nilai e = 10
Nilai k = 0
Nilai k = 4
C:\xampp\htdocs\Tugas PB0\Tugas 1>
```

**Ket :** Pada program ini kita membahas Operator ternary, Pertama-tama kita deklarasikan dulu variabelnya, *i* dengan nilai 0 dan tipe data integer, begitu juga dengan *j*. *c* dengan nilai 8 dan tipe data char, *d* dengan nilai 10 dan tipe data char juga.

- `int e = (((int) c > (int) d) ? c : d);` Operator ternary. Kita ambil dulu **(int) c** > **(int) d**, kemudian kita bandingkan 8 dan 10. Karena 8 tidak lebih besar dari 10, jadi hasilnya False. Makanya, **e** diberi nilai **d**, yaitu **10**.
- `int k = ((i > j) ? i : j);` Operator ternary. Kita ambil **i > j** dan kita bandingkan lagi 0 dan 0. Karena 0 tidak lebih besar dari 0, hasilnya false. Makanya, **k** diberi nilai **j**, yaitu **0**.

Pada output pertama kali `System.out.print("Nilai e = " + e);` dan `System.out.print("\nNilai k = " + k);` akan menampilkan nilai dari **e = 10** dan **k = 0**.

Kemudian pada `i = 2; j = 3;`, berarti kita mengubah nilai *i* jadi 2 dan *j* jadi 3.

Pada `k = ((i++ > j++) ? i : j);` Operator ternary dengan post-increment. Diambil **i++ > j++** kemudian membandingkan 2 dan 3. Karena 2 tidak lebih besar dari 3, jadi hasilnya False. Makanya **k** akan diberi nilai **j**.

Post-increment **i++** dan **j++** terjadi setelah perbandingan. Jadi, perbandingan menggunakan nilai *i* dan *j* itu sebelum increment. Setelah itu, **i** menjadi **3** dan **j** menjadi **4**. Jadi **k** diberi nilai **j** yang sudah di increment yaitu **4**.

`System.out.print("\nNilai k = " + k);` tampilah **Nilai k = 4**. Setelah semua nya sudah maka keluarlah output seperti pada gambar. Setelah semua nya sudah maka keluarlah output seperti pada gambar.

## 16. Output Program Oprator

```
Command Prompt

C:\xampp\htdocs\Tugas PBO\Tugas 1>javac Oprator.java

C:\xampp\htdocs\Tugas PBO\Tugas 1>java Oprator
Bool1 && Bool2 = false
Bool1 || Bool2 = true
!Bool1 = false
Bool1 ^ Bool2 = true

Nilai i = 5, j = 2
i + j = 7
i - j = 3
i / j = 2
i * j = 10
i % j = 1

Nilai i = 5
Nilai x = 5.0, y = 5.0
x + y = 10.0
x - y = 0.0
x / y = 1.0
x * y = 25.0

Operasi Relasional Integer:
i == j : false
i != j : true
i < j : false
i > j : true
i <= j : false
i >= j : true

Operasi Relasional Float:
x != y : false
x < y : false
x > y : false
x <= y : true
x >= y : true

Nilai i = 5
```

**Ket :** Pada program ini akan dijelaskan tentang **Contoh pengoperasian variabel bertipe dasar**, yang pertama itu masukkan tipe data dan nama variable sesuai kebutuhan. Setelah itu kita jalankan algoritmanya.

- Operasi Boolean;
  - Bool1 = true; Bool2 = false;

- **TF = Bool1 && Bool2;** yang berarti (true AND false) dan hasilnya **false**. Makanya pada perintah `System.out.println("Bool1 AND Bool2: " + TF);` menampilkan "Bool1 AND Bool2: false".
- **TF = Bool1 || Bool2;** yang berarti (true OR false) dan hasilnya **true**. Makanya pada perintah `System.out.println("Bool1 OR Bool2: " + TF);` menampilkan "Bool1 OR Bool2: true".
- **TF = !Bool1;** yang berarti (NOT true) dan hasilnya **false**. Makanya pada perintah `System.out.println("NOT Bool1: " + TF);` menampilkan "NOT Bool1: false".
- **TF = Bool1 ^ Bool2;** yang berarti (true XOR false) dan hasilnya **true**. Makanya pada perintah `System.out.println("Bool1 XOR Bool2: " + TF);` menampilkan "Bool1 XOR Bool2: true".
- Operasi numerik(integer);
  - **i = 5; j = 2;**
  - **hsl = i + j;** yang berarti (5 + 2) dan hasilnya **7**. Makanya pada perintah `System.out.println("i + j: " + hsl);` menampilkan "i + j: 7".
  - **hsl = i - j;** yang berarti (5 - 2) dan hasilnya **3**. Makanya pada perintah `System.out.println("i - j: " + hsl);` menampilkan "i - j: 3".
  - **hsl = i \* j;** yang berarti (5 \* 2) dan hasilnya **10**. Makanya pada perintah `System.out.println("i * j: " + hsl);` menampilkan "i \* j: 10".
  - **hsl = i / j;** yang berarti (5 / 2) dan hasilnya **2(integer division)**. Makanya pada perintah `System.out.println("i / j: " + hsl);` menampilkan "i / j: 2".
  - **hsl = i % j;** yang berarti (5 % 2) dan hasilnya **1**. Makanya pada perintah `System.out.println("i % j: " + hsl);` menampilkan "i % j: 1".
- Operasi numerik(float);
  - **x = 5; y = 5;**
  - **res = x + y;** yang berarti (5.0 + 5.0) dan hasilnya **10.0**. Makanya pada perintah `System.out.println("x + y: " + res);` menampilkan "x + y: 10.0".
  - **res = x - y;** yang berarti (5.0 - 5.0) dan hasilnya **0.0**. Makanya pada perintah `System.out.println("x - y: " + res);` menampilkan "x - y: 0.0".
  - **res = x / y;** yang berarti (5.0 / 5.0) dan hasilnya **1.0**. Makanya pada perintah `System.out.println("x / y: " + res);` menampilkan "x / y: 1.0".

- **res = x \* y;** yang berarti (5.0 \* 5.0) dan hasilnya **25.0** . Makanya pada perintah `System.out.println("x * y: " + res);` menampilkan “x \* y: 25.0”.
- Operasi relasional numerik (integer);
  - **TF = (i == j);** yang berarti (5 == 2) dan hasilnya **false** . Makanya pada perintah `System.out.println("i == j: " + TF);` menampilkan “i == j: false”.
  - **TF = (i != j);** yang berarti (5 != 2) dan hasilnya **true** . Makanya pada perintah `System.out.println("i != j: " + TF);` menampilkan “i != j: true”.
  - **TF = (i < j);** yang berarti (5 < 2) dan hasilnya **false** . Makanya pada perintah `System.out.println("i < j: " + TF);` menampilkan “i < j: false”.
  - **TF = (i > j);** yang berarti (5 > 2) dan hasilnya **true** . Makanya pada perintah `System.out.println("i > j: " + TF);` menampilkan “i > j: true”.
  - **TF = (i <= j);** yang berarti (5 <= 2) dan hasilnya **false** . Makanya pada perintah `System.out.println("i <= j: " + TF);` menampilkan “i <= j: false”.
  - **TF = (i >= j);** yang berarti (5 >= 2) dan hasilnya **true** . Makanya pada perintah `System.out.println("i >= j: " + TF);` menampilkan “i >= j: true”.
- Operasi relasional numerik (float);
  - **TF = (x != y);** yang berarti (5.0 != 5.0) dan hasilnya **false** . Makanya pada perintah `System.out.println ("x != y: " + TF);` menampilkan “x != y: false”.
  - **TF = (x < y);** yang berarti (5.0 < 5.0) dan hasilnya **false** . Makanya pada perintah `System.out.println ("x < y: " + TF);` menampilkan “x < y: false”.
  - **TF = (x > y);** yang berarti (5.0 > 5.0) dan hasilnya **false** . Makanya pada perintah `System.out.println ("x > y: " + TF);` menampilkan “x > y: false”.
  - **TF = (x <= y);** yang berarti (5.0 <= 5.0) dan hasilnya **true** . Makanya pada perintah `System.out.println ("x <= y: " + TF);` menampilkan “x <= y: true”.
  - **TF = (x >= y);** yang berarti (5.0 >= 5.0) dan hasilnya **true** . Makanya pada perintah `System.out.println ("x >= y: " + TF);` menampilkan “x >= y: true”. Setelah semua nya sudah maka keluarlah output seperti pada gambar.

