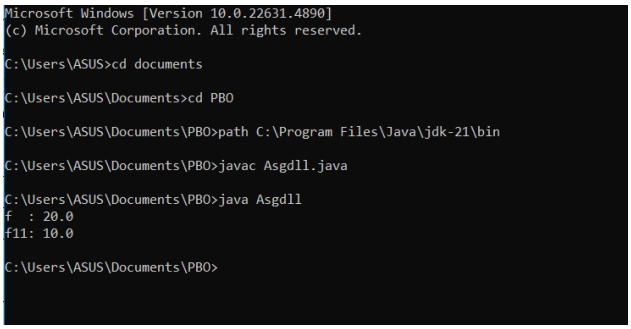
Tugas 1

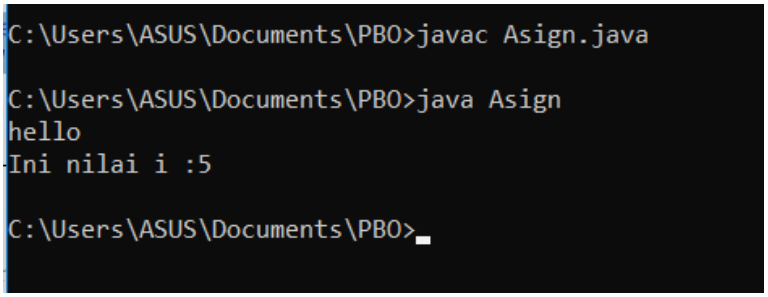
1. Asgdll



Program ini mendemonstrasikan perbedaan tipe data floating point di Java:

- Deklarasi variabel f dengan tipe data float dan nilai awal 20.0f (suffix 'f' menunjukkan literal float)
- Deklarasi variabel fll dengan tipe data double tanpa nilai awal
- Assignment nilai 10.0f ke variabel fll (double bisa menerima nilai float karena memiliki presisi lebih tinggi)
- Menampilkan kedua nilai dalam satu pernyataan println dengan \n sebagai karakter newline
- Output akan menampilkan "f : 20.0" dan "f11: 10.0" pada baris terpisah

2. Asign



Program ini menunjukkan dasar-dasar deklarasi dan penggunaan variabel:

- Deklarasi variabel i dengan tipe int tanpa inisialisasi
- Penggunaan System.out.print() untuk mencetak teks "hello" diikuti karakter newline (\n)
- Assignment nilai 5 ke variabel i
- Penggunaan System.out.println() untuk mencetak string "Ini nilai i :" diikuti nilai variabel i
- Perbedaan antara print() dan println() adalah println() otomatis menambahkan baris baru setelah output

3. ASGNi

```
C:\Users\ASUS\Documents\PBO>javac ASIGNi.java

C:\Users\ASUS\Documents\PBO>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:\Users\ASUS\Documents\PBO>
```

Program ini mendemonstrasikan berbagai tipe data primitif di Java:

- short ks = 1: Deklarasi dan inisialisasi variabel tipe short (16-bit integer)
- int ki = 1: Variabel tipe int (32-bit integer)
- long kl = 10000: Variabel tipe long (64-bit integer)
- char c = 65: Inisialisasi karakter dengan nilai ASCII (65 adalah 'A')
- char c1 = 'Z': Inisialisasi karakter langsung dengan literal karakter
- double x = 50.2f: Inisialisasi double (64-bit floating point) dengan nilai float
- float y = 50.2f: Inisialisasi float (32-bit floating point)

Program mencetak nilai semua variabel, menunjukkan:
- Karakter dapat diinisialisasi dengan nilai integer (kode ASCII)
- Karakter dapat ditampilkan sebagai karakter asli
- Teks dapat diformat dengan '\t' (tab)
- Perbedaan representasi dari tipe-tipe numerik

4. BacaData

```
C:\Users\ASUS\Documents\PBO>javac BacaData.java

C:\Users\ASUS\Documents\PBO>java BacaData
Contoh membaca dan menulis, ketik nilai integer:
120
Nilai yang dibaca : 120
C:\Users\ASUS\Documents\PBO>
```

Program ini menunjukkan penggunaan class Scanner untuk membaca input:

- Import java.util.Scanner untuk menggunakan class Scanner
- Deklarasi variabel a (tipe int) dan masukan (tipe Scanner)
- Mencetak prompt untuk meminta input dari pengguna
- Inisialisasi objek Scanner masukan yang membaca dari System.in (input standar)

- Memanggil masukan.nextInt() untuk membaca nilai integer dari input
- Menyimpan nilai yang dibaca ke variabel a
- Mencetak nilai yang telah dibaca
- Perhatikan comment mengenai apa yang terjadi jika pengguna memasukkan nilai non-integer (akan menyebabkan InputMismatchException)

5. Bacakar

```
C:\Users\ASUS\Documents\PBO>javac Bacakar.java

C:\Users\ASUS\Documents\PBO>java Bacakar
hello
baca 1 karakter : F
baca 1 bilangan : 120
F
120
bye

C:\Users\ASUS\Documents\PBO>
```

Program ini mendemonstrasikan penggunaan BufferedReader untuk membaca input:

- Import class yang diperlukan: BufferedReader, IOException, dan InputStreamReader
- Method main mendeklarasikan throws IOException untuk menangani exception yang mungkin terjadi saat membaca input
- Deklarasi variabel cc (char) dan bil (int)
- Menunjukkan dua cara untuk membuat BufferedReader:
    - Dengan membuat InputStreamReader terlebih dahulu, lalu membuat BufferedReader dari InputStreamReader
    - Langsung dalam satu baris (inline construction)
- Membaca karakter dengan memanggil readLine() (membaca seluruh baris) dan mengambil karakter pertama dengan charAt(0)
- Membaca integer dengan memanggil readLine() dan mengkonversinya ke integer dengan Integer.parseInt()
- Mencetak hasil pembacaan dan pesan penutup
- Program ini juga menunjukkan penggunaan karakter newline (\n) untuk pemformatan output

6. Casting 1

```
C:\xampp\htdocs\PBO\Tugas_1>java Casting1
5.0
6.0
2
3.200000047683716
53
53.0
53.0
3
3.14

C:\xampp\htdocs\PBO\Tugas_1>
```

program ini menunjukkan casting (konversi) tipe data primitif:

- Deklarasi variabel dengan berbagai tipe data: int, float, char, dan double
- Mendemonstrasikan explicit casting dengan sintaks (tipe_tujuan)variabel:
    - (float)a: Mengkonversi int (5) menjadi float (5.0)
    - (double)b: Mengkonversi int (6) menjadi double (6.0)
    - (int)d: Mengkonversi float (2.0) menjadi int (2), dengan truncation (membuang bagian desimal)
    - (double)e: Mengkonversi float (3.2) menjadi double (3.2) dengan presisi lebih tinggi
    - (int)g: Mengkonversi char ('5') menjadi nilai ASCII-nya (53)
    - (float)g dan (double)g: Mengkonversi char ke float dan double berdasarkan nilai ASCII
    - (int)k: Mengkonversi double (3.14) menjadi int (3) dengan truncation
    - (float)k: Mengkonversi double (3.14) menjadi float (3.14) dengan kemungkinan kehilangan presisi
- Program ini menunjukkan bagaimana data dapat kehilangan presisi atau bahkan berubah drastis saat dikonversi antar tipe yang berbeda

7. Casting 2

```
C:\Users\ASUS\Documents\PBO>javac Casting2.java

C:\Users\ASUS\Documents\PBO>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:\Users\ASUS\Documents\PBO>
```

Program ini mendemonstrasikan casting menggunakan class wrapper (Integer, Double, Float, String):

- Deklarasi dan inisialisasi variabel berbagai tipe primitif dan String
- Konversi String ke tipe primitif:
    - Integer.parseInt(n): Mengkonversi String "67" menjadi int 67
    - Double.parseDouble(m): Mengkonversi String "45" menjadi double 45.0
    - Float.parseFloat(l): Mengkonversi String "100" menjadi float 100.0
- Konversi tipe primitif ke String:
    - String.valueOf(b): Mengkonversi int 9 menjadi String "9"
    - String.valueOf(g): Mengkonversi char '5' menjadi String "5"
    - String.valueOf(e): Mengkonversi float 3.2 menjadi String "3.2"
- Konversi kompleks dengan method chaining:
    - Double.valueOf(a).intValue(): Mengkonversi int ke Double wrapper kemudian mengambil nilai int-nya
    - Integer.valueOf(b).doubleValue(): Mengkonversi int ke Integer wrapper kemudian mengambil nilai double-nya
- Program menunjukkan perbedaan antara casting primitif dan konversi menggunakan class wrapper

8. Ekspresi

```
C:\Users\ASUS\Documents\PBO>javac Ekspresi.java

C:\Users\ASUS\Documents\PBO>java Ekspresi
x = 1
y = 2
hasil ekspresi = (x<y)?x:y = 1
C:\Users\ASUS\Documents\PBO>
```

Program ini mendemonstrasikan operator kondisional (ternary):

- Deklarasi dan inisialisasi variabel x = 1 dan y = 2
- Menampilkan nilai x dan y
- Menggunakan operator ternary (kondisi) ? nilai_jika_benar : nilai_jika_salah:
    - Kondisi: x < y (1 < 2) bernilai true
    - Karena kondisi true, nilai yang dikembalikan adalah nilai pertama (x), yaitu 1
    - Output akhir: "hasil ekspresi = (x<y)?x:y = 1"
- Perhatikan penggunaan tanda kurung ((x < y) ? x : y) untuk memperjelas bahwa seluruh ekspresi kondisional dianggap sebagai satu kesatuan
- Program menunjukkan cara singkat untuk memilih nilai berdasarkan kondisi tanpa perlu menggunakan struktur if-else

9. Ekspresi 1

```
C:\Users\ASUS\Documents\PBO>javac Ekspresi1.java

C:\Users\ASUS\Documents\PBO>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:\Users\ASUS\Documents\PBO>
```

Program ini mengilustrasikan pembagian antara tipe data yang berbeda:

- Deklarasi variabel int (x = 1, y = 2) dan float (fx, fy) tanpa inisialisasi awal
- Menampilkan hasil pembagian integer (x/y), yang menghasilkan 0 karena pembagian antara integer dibulatkan ke bawah (1/2 = 0)
- Mengkonversi nilai int ke float:
    - fx = x dan fy = y (implicit casting)
    - Menampilkan fx/fy yang menghasilkan nilai float 0.5
- Menggunakan explicit casting: (float)x/(float)y, yang juga menghasilkan 0.5
- Mengganti nilai x = 10 dan y = 3, kemudian mencetak pembagian integer x/y yang menghasilkan 3 (pembulatan ke bawah dari 10/3)
- Program ini menunjukkan bahwa tipe data hasil operasi ditentukan oleh tipe data operand, dan explicit casting dapat mengubah perilaku operasi

10. Hello

```
C:\Users\ASUS\Documents\PBO>javac Hello.java

C:\Users\ASUS\Documents\PBO>java Hello
Hello
Hello World
Welcome

C:\Users\ASUS\Documents\PBO>
```

Program ini menunjukkan dasar-dasar output di Java:

- System.out.print("Hello"): Mencetak teks "Hello" tanpa baris baru
- System.out.print("\nHello "): Mencetak newline dan teks "Hello " (dengan spasi di akhir)
- System.out.println("World"): Mencetak teks "World" diikuti baris baru
- System.out.println("Welcome"): Mencetak teks "Welcome" diikuti baris baru
- Program menunjukkan perbedaan antara print() dan println(), serta penggunaan karakter newline ('\n')

11. Incr

```
C:\Users\ASUS\Documents\PBO>javac Incr.java

C:\Users\ASUS\Documents\PBO>java Incr
Nilai i : 5
Nilai j : 3

C:\Users\ASUS\Documents\PBO>
```

Program ini mendemonstrasikan operator increment (++):

1. Deklarasi variabel i dan j tanpa inisialisasi
    - Inisialisasi i = 3
    - Operasi j = i++:

2.Nilai i (3) diassign ke j

3.Kemudian i ditambah 1 menjadi 4 (post-increment)

- Penulisan ++i dalam println:

4.i ditambah 1 menjadi 5 (pre-increment)

5.Kemudian nilai i (5) digunakan dalam ekspresi

- Output: "Nilai i : 5" dan "Nilai j : 3"
- Program menunjukkan perbedaan antara:
- Post-increment (i++): Nilai lama digunakan dalam ekspresi, kemudian variabel ditambah 1

Pre-increment (++i): Variabel ditambah 1 terlebih dahulu, kemudian nilai baru digunakan dalam ekspresi

12. Oper1

```
C:\Users\ASUS\Documents\PBO>javac Oper1.java

C:\Users\ASUS\Documents\PBO>java Oper1
n = 10
x = 1
y = 2
n & 8 = 8
x & ~ 8 = 1
y << 2 = 8
y >> 3 = 0

C:\Users\ASUS\Documents\PBO>
```

Program ini mendemonstrasikan operator bitwise:

- Deklarasi variabel n=10 (1010 dalam biner), x=1 (1 dalam biner), dan y=2 (10 dalam biner)
- Operasi bitwise AND (&):
  - n & 8 = 1010 & 1000 = 1000 (desimal 8) - karena hanya bit ketiga yang sama-sama 1
  - x & ~8 = 1 & ~1000 = 1 & 0111 = 0001 (desimal 1) - operator ~ melakukan negasi bit
- Operasi shift:
  - y << 2 = 10 << 2 = 1000 (desimal 8) - menggeser bit 2 posisi ke kiri
  - y >> 3 = 10 >> 3 = 0000 (desimal 0) - menggeser bit 3 posisi ke kanan
- Program menunjukkan bagaimana manipulasi bit bekerja dalam bahasa Java, yang penting untuk operasi level rendah dan optimisasi

13. Oper2

```
C:\Users\ASUS\Documents\PBO>javac Oper2.java

C:\Users\ASUS\Documents\PBO>java Oper2
i = 3
j = ♦
i & j = 0
i | j = 7
i ^ j = 7
81.0
 ~i = -4

C:\Users\ASUS\Documents\PBO>
```

program ini mendemonstrasikan operasi bitwise pada tipe data char:

Deklarasi variabel char i dan j

- Inisialisasi i = 3 dan j = 4 (nilai numerik, bukan karakter)
- Casting (int) i untuk menampilkan nilai numerik dari i
- Operasi bitwise:
  - i & j = 00000011 & 00000100 = 00000000 (0) - AND: 1 jika kedua bit 1
  - i | j = 00000011 | 00000100 = 00000111 (7) - OR: 1 jika salah satu atau kedua bit 1

    - i ^ j = 00000011 ^ 00000100 = 00000111 (7) - XOR: 1 jika tepat satu bit 1
    - ~i = ~00000011 = 11111100 (sebagai signed int: -4) - NOT: membalik semua bit
- Penggunaan Math.pow(i, j) untuk pemangkatan (3^4 = 81.0)
- Program menunjukkan bahwa operator ^ dalam Java adalah XOR, bukan pangkat (untuk pangkat harus menggunakan Math.pow)

14. Oper3

```
C:\Users\ASUS\Documents\PBO>javac Oper3.java

C:\Users\ASUS\Documents\PBO>java Oper3
true
false
true
true
true

C:\Users\ASUS\Documents\PBO>
```

Program ini mendemonstrasikan operator logika:

- Penggunaan && (short-circuit AND):
    - true && true bernilai true, mencetak "true"
    - Evaluasi berhenti jika operand pertama false
- Penggunaan & (logical AND):
    - true & false bernilai false, mencetak "false"
    - Selalu mengevaluasi kedua operand
- Kondisional sederhana: if (true) selalu true
- Penggunaan || (short-circuit OR):
    - true || true bernilai true, mencetak "true"
    - Evaluasi berhenti jika operand pertama true
- Penggunaan | (logical OR):
    - true | false bernilai true, mencetak "true"
    - Selalu mengevaluasi kedua operand
- Program menunjukkan perbedaan antara operator short-circuit (&&, ||) dan operator logika biasa (&, |)

15. Oper4

```
C:\Users\ASUS\Documents\PBO>javac Oper4.java

C:\Users\ASUS\Documents\PBO>java Oper4
Nilai e = 10
Nilai k = 0
Nilai k = 4
C:\Users\ASUS\Documents\PBO>
```

Program ini mendemonstrasikan operator ternary dan efek samping:

- Deklarasi variabel integer i=0, j=0, dan karakter c=8, d=10
- Operator ternary pertama: (((int)c > (int)d) ? c: d)

    - Membandingkan nilai int dari c (8) dan d (10)
    - Karena 8 tidak > 10, maka nilai d (10) dipilih
    - Hasil disimpan di variabel e
- Operator ternary kedua: ((i>j) ? i: j)
    - Membandingkan i (0) dan j (0)
    - Karena 0 tidak > 0, maka nilai j (0) dipilih
    - Hasil disimpan di variabel k
- Mengubah nilai i=2 dan j=3
- Operator ternary dengan efek samping: ((i++>j++) ? i: j)
    - Membandingkan i (2) dan j (3) sebelum increment
    - Karena 2 tidak > 3, maka nilai j (4, setelah increment) dipilih
    - i dan j keduanya di-increment (menjadi 3 dan 4) akibat operasi i++ dan j++
    - Hasil (4) disimpan di variabel k
- Output: e=10, k=0, dan k=4
  Program menunjukkan bagaimana efek samping (side effect) dari operator increment dapat mempengaruhi hasil operasi ternary

16. Oprator

```
C:\Users\ASUS\Documents\PBO>javac Oprator.java

C:\Users\ASUS\Documents\PBO>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

C:\Users\ASUS\Documents\PBO>
```

program ini merupakan rangkuman komprehensif berbagai operator dalam Java:

- Deklarasi variabel boolean, int, dan float
- Demonstrasi operasi boolean:
  - AND (&&): true && false = false
  - OR (||): true || false = true
  - NOT (!): !true = false
  - XOR (^): true ^ false = true
- Demonstrasi operasi numerik integer:
  - Penjumlahan (+): 5 + 2 = 7
  - Pengurangan (-): 5 - 2 = 3
  - Pembagian (/): 5 / 2 = 2 (pembulatan ke bawah)
  - Perkalian (*): 5 * 2 = 10
  - Modulo (%): 5 % 2 = 1 (sisa pembagian)
- Demonstrasi operasi numerik floating point:
  - Operasi aritmatika dengan float menghasilkan hasil dengan presisi decimal
- Demonstrasi operasi relasional:
  - Equal (==): membandingkan kesamaan nilai
  - Not equal (!=): kebalikan dari equal
  - Less than (<), greater than (>), less than or equal (<=), greater than or equal (>=)
- Program menunjukkan semua jenis operator dasar dalam Java dan bagaimana mereka bekerja dengan berbagai tipe data