Java mempunyai 11 macam tipe data yang terdiri atas tipe data sederhana dan referensi / komposit. Tipe sederhana meliputi byte, short, int, long, char, float, double dan boolean yang terbagi menjadi 3 tipe. Sedangkan tipe data referensi meliputi class,array dan interface.

[](http://1.bp.blogspot.com/-myphRTN76P4/U1KGzHqfJEI/AAAAAAAAArU/GoRtxJSnDdM/s1600/java_tech.jpg)

**A. Tipe Data Sederhana**

1. **Integer (Bilangan Bulat)**

Tipe data yang masuk menjadi bagian ini adalah byte, short, int dan long. Semua tipe data ini bersifat       *Signed,* yaitu bisa mempresentasikan nilai positif dan negatif. Tidak seperti tipe data lainnya, Java tidak mendukung tipe data *unsigned* yang hanya bisa mempresentasikan nilai postif. Untuk jelasnya akan dijelaskan oleh tabel dan penjelasan di bawah ini :

|  |  |  |
| --- | --- | --- |
| **Tipe Data** | **Ukuran (bit)** | **Range** |
| *Byte* | 8 | -128 s.d. 127 |
| *Short* | 16 | -32768 s.d. 32767 |
| *Int* | 32 | -2147483648 s.d. 2147483647 |
| *Long* | 64 | -9223372036854775808  s.d. 9223372036854775807 |

**·     Byte**

Type *byte* umumnya digunakan pada saat kita bekerja dengan sebuah data *stream* dari suatu file maupun jaringan, yaitu untuk kepeluan proses membaca/menulis. Selain itu, tipe ini juga digunakan saat bekerja dengan data biner yang tidak kompatibel dengan tipe-tipe lain yang didefiniskan di dalam Java. Contoh :

class ContohByte {public static void main(String [] args){byte a;a=127;System.out.println(a);}}

**·     Short**

Pada umumnya diaplikasikan pada komputer-komputer 16-bit, yang saat ini semakin jarang keberadaanya. Contoh :

class ContohShort {public static void main(String[]args){short umurWafiy;short umurAdit;short selisih;umurWafiy=23;umurAdit=13;selisih=umurWafiy-umurAdit;System.out.println(“Selisih umur mereka adalah “ + selisih + ” tahun”);

**·     Int**

Tipe ini merupakantipe yang paling banyak dipakai dalam merepresentasikan angka dalam Java, dikarenakan dianggap paling efisien dibandingkan dengan tipe-tipe integer lainnya. Tipe *Int* banyak digunakan untuk indeks dalam struktur pengulangan maupun dalam konstruksi sebuah *array.*Selain itu, secara teori setiap ekspresi yang melibatkan tipe integer *byte, short, int, long)* semuanya akan dipromosikan ke *int* terlebih dahulu sebelum dilakukan proses perhitungan. Contoh :

class HitungGaji{public static void main(String[]args){int gaji;int lamaKerja;int besarGajigaji=5000000;lamaKerja=4;besarGaji=gaji\*lamaKerja;System.out.println(besarGaji);}}

**·     Long**

Tipe ini digunakan untuk kasus-kasus tertentu yang nilainya berada di luar rentang tipe *int,* karna tipe ini punya range paling tinggi dibanding *Intege*r lainnya. Dengan kata lain, tipe *long* terpaksa digunakan jika data memiliki range diluar range *int.* Contoh:

public class MassaPlanet{public static void main (String[]args){long volum=1864824217374668;long massaJenis=77886;long massa=volum\*massaJenis;System.out.println(massa);}}

**2.   Floating-Point (Bilangan Pecahan)**

Tipe *floating-point* digunakan untuk merepresentasikan nilai-nilai yang mengandung pecahan atau angka decimal di belakang koma, seperti 3.1416,5.25, dan sebagainya. Bilangan semacam ini disebut sebagai bilangan riil. Dalam Java tipe ini dibedakan menjadi dua jenis, yaitu *float*, dan *double*. Untuk jelasnya akan dijelaskan oleh tabel dan penjelasan di bawah ini :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tipe** | **Ukuran** | | **Range** | **Presisi (jumlah digit)** |
| **bytes** | **bit** |
| float | 4 | 32 | +/- 3.4 x 1038 | 6-7 |
| double | 8 | 64 | +/- 1.8 x 10308 | 15 |

**·     Float**

Tipe ini digunakan untuk menandakan nilai–nilai yang mengandung presisi atau ketelitan tunggal (*single-precision*) yang menggunakan ruang penyimpanan 32-bit. Presisi tunggal biasanya lebih cepat untuk processor-processor tertentu dan memakan ruang penyimpanan setengah kali lebih sedikit dibandingkan presisi ganda (*double precision*). Permasalahan yang timbul dari pemakaian tipe *float* untuk nilai-nilai yang terlalu kecil atau justru terlalu besar, karena nilai yang dihasilkan akan menjadi tidak akurat. Contoh penggunaan variabel :

float suhu;

**·     Double**

Tipe ini mengandung tingkat ketelitian ganda atau presisi ganda (*double precision*) dan menggunakan ruang penyimpanan 64-bit untuk menyimpan nilai. Tipe *double* tentu lebih cepat untuk melakukan perhitungan-perhitungan matematis daripad tipe *float*. Untuk perhitungan yang bersifat bilangan riil dan menghasilkan hasil yang lebih akurat, maka lebih baik menggunakan tipe *double*. Contoh :

class KelilingLingkaran {public static void main (String[] args) {double pi = 3.1416;double r = 2.12;double keliling;keliling = 2\*pi\*r;System.out.println(“Keliling Lingkaran = ”+ keliling);}}

**3.   Char**

Tipe data char merupakan tipe untuk menyatakan sebuah karakter.  Java menggunakan karakter *Unicode* untuk merepresentasikan semua karakter yang ada . *Unicode* ialah sekumpulan  karakter yang terdapat pada semua bahasa, seperti bahasa Latin, Arab, Yunani dan lain-lainnya. Karena bahasa Java dirancang untuk dapat diterapkan di berbagai macam *platform*, maka Java menggunakan karakter *Unicode* yang membutuhkan ukuran 16-bit. Untuk karakter-karakter yang tidak dapat diketikkan secara langsung melalui keyboard, java menyediakan beberapa *escape sequence* (pasangan karakter yang dianggap sebagai karakter tunggal). *Escape sequence*tidak dianggap sebagai *String*, melainkan tetap sebagai tipe karakter khusus. Di bawah ini akan dijelaskan beberapa contoh tentang escape sequence.

|  |  |
| --- | --- |
| ***Escape Sequence*** | **Keterangan** |
| \ddd | Karakter octal (ddd) |
| \uxxxx | Karakter Unicode heksadecimal (xxxx) |
| \’ | Petik tunggal |
| \’’ | Petik ganda |
| \\ | *Backslash* |
| \r | *Carriage return* |
| \n | Baris baru (*line feed*) |
| \f | *Form feed* |
| \t | *Tab* |
| \b | *Backspace* |

Contoh :

class ContohKarakter {public static void main (String[] args) {char ch = 65;// 65 merupakan kode untuk karakter A;System.out.println(“ch1=”+ch);ch++;   //*increment*(penaikan nilai sebesar 1)System.out.println(“ch2 = ”+ ch);}}

**4.   Boolean**

Tipe *boolean* adalah tipe data yang digunakan untuk menampung nilai logika, yaitu nilai yang hanya memiliki dua buah kemungkinan (benar atau salah). Tipe ini ditandai dengan kata kunci *Boolean*. Dalam bahasa Java, nilai benar dipresentasikan dengan kata kunci *true* dan nilai salah dengan kata kunci *false*. Contoh :

class ContohBolean {public static void main (String[] args) {boolean a = true;if (a) {System.out.println(“Perintah dilaksanakan ”);}//negasi dari aIf (!a) {System.out.println(“Perintah tidak dilaksanakan ”);}}}

**B. Tipe Data Referensi**

**1.    Class**

Kelas dapat didefiniskan sebagai cetak biru (*blueprint*) atau prototipe/kerangka yang mendefiniskan variabel-variabel (data) dan method-method (perilaku) umum dari sebuah objek. Dengan kata lain kelas adalah sebuah kesatuan yang terintegrasi antara method dan data yang mengacu pada suatu objek.

Dalam dunia permrograman, sebenarnya kelas tidak jauh berbeda dengan tipe data sederhana. Perbedaannya, tipe data sederhana digunakan untuk mendeklarasikan variabel ‘normal’, sedangkan kelas digunakan untuk mendeklarasikan sebuah variabel yang berupa objek. Variabel yang berupa objek ini sering disebut dengan referensi objek (*object reference*).

Pada saat kita membuat sebuah kelas baru. Sekali didefiniskan, maka tipe data baru ini dapat digunakan untuk membuat suatu objek dari tipe tersebut. Dengan kata lain, kelas adalah pola (template) untuk pembuatan objek, dan objek adalah wujud nyata (instance) dari sebuah kelas. Contoh :

public Class Mahasiswa{public String nama;public int nrp;Mahasiswa(String a, int b){nama =a;nrp= b;}public void cetak (){System.out.println(“Nama : “+nama+” nrp : “+nrp);}}

Setelah kita membuat sebuah kelas, untuk menggunakannya maka kita harus membuat sebuah instance dari kelas tersebut. Berikut cara membuat objek dari kelas :

class Demo {public static void main(String[]args){Mahasiswa mhs;mhs = new Mahasiswa(“Rezki”,5211100048)}}

**2.   Array**

Tipe data ini memiliki kemampuan untuk menggunakan satu variabel yang dapat menyimpan sebuah data list dan kemudian memanipulasinya dengan lebih efektif.

Sebuah array akan menyimpan beberapa item data yang memiliki tipe data sama didalam sebuah blok memori yang berdekatan yang kemudian dibagai menjadi beberapa slot.

**3.   Interface**

Interface merupakan sekumpulan method yang hanya memuat deklarasi dan struktur method, tanpa detail implementasinya. Sedangkan detail dari method tersebut berada pada class yang mengimplementasikan interface tersebut. Interface digunakan bila Anda ingin mengaplikasikan suatu method yang spesifik,  yang tidak diperoleh dari proses inheritance yang lebih terbatas. Tipe data yang boleh pada interface hanya tipe data konstan.