Achmad Syarief Hidayatullah

H13116004

PENGANTAR PEMROGRAMAN

Resume

BAB I

Pengantar Tentang Java

APAKAH ITU JAVA ?

**Java** adalah [bahasa pemrograman](https://id.wikipedia.org/wiki/Bahasa_pemrograman) yang dapat dijalankan di berbagai [komputer](https://id.wikipedia.org/wiki/Komputer) termasuk [telepon genggam](https://id.wikipedia.org/wiki/Telepon_genggam). Bahasa ini banyak mengadopsi sintaksis yang terdapat pada [C](https://id.wikipedia.org/wiki/C_(bahasa_pemrograman)) dan [C++](https://id.wikipedia.org/wiki/C%2B%2B) namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam [p-code](https://id.wikipedia.org/wiki/P-code) (*bytecode*) dan dapat dijalankan pada berbagai [Mesin Virtual Java (JVM)](https://id.wikipedia.org/wiki/Mesin_virtual_java). Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik (*general purpose*), dan secara khusus didisain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform [sistem operasi](https://id.wikipedia.org/wiki/Sistem_operasi) yang berbeda, java dikenal pula dengan slogannya, "*Tulis sekali, jalankan di mana pun*" (write one, run everywhere). Saat ini java merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi

Sebelum kita belajar lebih banyak pengantar tentang Java, kita lihat Dulu Sejarahnya.

SEJARAH JAVA

pada tahun 1991, Java bermula dari proyek penelitian **Sun Microsystem**dengan nama sandi Green. Waktu itu, Terdapat prediksi bahwa mikroprosesor akan digunakan luas pada peralatan-peralatan elektronik. Oleh karena itu, Maka dibutuhkanlah bahasa pemrograman  yang dapat berjalan di semua tipe mikroprosesor. Terciptalah sebuah bahasa pemrograman baru. Oleh James Gosling, program ini diberi nama OAK, sesuai dengan nama pohon yang ada di kantor Sun Microsystem. Selang beberapa waktu kemudian, ternyata nama OAK sudah digunakan untuk bahasa pemrograman. Akhirnya Sun Microsystem memilih nama JAVA yang berasal dari nama biji kopi yang dijual di kedai kopi dan kopi itu ialah kopi jawa, dan secara formal pada tahun 1995 Sun Microsystem mengumumkan kehadiran bahasa Java.

PERKEMBANGAN TEKNOLOGI JAVA

* Java Card : teknologi Java yang digunakan pada peralatan elektronik yang bisa dibilang memiliki memori terbatas.
* J2ME : Java 2 Platform, Micro Edition, merupakan teknologi Java yang digunakan untuk perangkat mobile. Seperti hp, smartphone dsb
* J2SE : Java 2 Platform, Standard Edition, merupakan teknologi Java yang digunakan untuk aplikasi desktop. Contohnya PC atau LAptop
* J2EE : Java 2 Platform, Enterprise Edition, merupakan teknologi Java yang digunakan untuk penerapan teknologi java pada komputer server.

Setiap edisi Java terdiri atas 2 komponen utama:

* Java API (Application Programming Interfaces), terdiri atas kumpulan library yang digunakan untuk keperluan pemrograman.
* JRE (Java Run Time Environment), berfungsi untuk membuat aplikasi java dapat dijalankan.

produk pertama proyek Green adalah star 7 (\*7), sebuah kendali jarak jauh yang sangat cerdas. Dikarenakan pasar masih belum tertarik dengan produk consumer cerdas maka proyek Green harus menemukan pasar lain dari teknologi yang diciptakan. Pada saat yang sama, implementasi WWW dan internet sedang mengalami perkembangan pesat. Di lain pihak, anggota dari proyek Green juga menyadari bahwa Java dapat digunakan pada pemrograman internet, sehingga penerapan selanjutnya mengarah menjadi teknologi yang berperan di web.

JAVA ADALAH….

* Sebuah bahasa pemrograman berorientasi objek yang dapat dijalankan pada berbagai *platform.*
* Sebuah *Development Environment* atau penyedia banyak *tools*:*compiler, interpreter,*penyusun dokumentasi, paket kelas dll .
* Sebuah Aplikasi serba guna yang dapat dijalankan pada seluruh mesin yang memiliki *Java Runtime Environment*(JRE).
* Sebuah *Deployment Environmen*antara lain ada 2 komponen: JRE (ada pada paket J2SDK mengandung kelas” untuk semua paket teknologi Java yang meliputi kelas dasar dari Java, komponen GUI dll) dan Web Browser (hampir seluruh Web Browser komersial menyediakan *interpreter*dan *runtime environment*dari teknologi Java).

KENAPA BELAJAR JAVA ?

Mengapa Kita Belajar Java ? Bisa diketahui bahwa Java memiliki karakteristik berikut :

1)      Sederhana

Bahasa pemrograman Java menggunakan Sintaks mirip dengan C++ namun sintaks pada Java telah banyak diperbaiki terutama menghilangkan penggunaan pointer yang rumit dan multiple inheritance. Java juga menggunakan automatic memory allocation dan memory garbage collection.

2)      Berorientasi objek (Object Oriented)

Java menggunakan pemrograman berorientasi objek yang membuat program dapat dibuat secara modular dan dapat dipergunakan kembali. Pemrograman berorientasi objek memodelkan dunia nyata ke dalam program dan melakukan interaksi antar objek-objek tersebut.

3)      Terdistribusi (Distributed)

Java dibuat untuk membuat aplikasi terdistribuasi secara mudah dengan adanya libraries networking yang terintegrasi pada Java.

4)      Interpreted

Program Java dijalankan menggunakan interpreter yaitu Java Virtual Machine (JVM). Hal ini menyebabkan source code Java yang telah dikompilasi menjadi Java bytecodes dapat dijalankan platform yang berbeda-beda.

5)      Robust

Java mempunyai reliabilitas yang tinggi. Compiler pada Java mempunyai kemampuan mendeteksi eror secara lebih teliti dibandingkan bahasa pemrograman lain. Java mempunyai runtime-Exception handling untuk membantu mengatasi eror pada pemrograman.

6)      Secure

Sebagai bahasa pemrograman untuk aplikasi internet dan terdistribusi, Java memiliki beberapa mekanisme keamanan untuk menjaga aplikasi tidak digunakan untuk merusak system computer yang menjalankan aplikasi tersebut.

7)      Architecture neutral

Program Java merupakan platform independent. Program cukup mempunyai satu buah versi yang dapat dijalankan pada platform berbeda dengan Java Virtual Machine.

8)      Portable

Source code maupun program Java dapat dengan mudah dibawa ke platform yang berbeda-beda tanpa harus dikompilasi ulang.

9)      Performance

Performance pada Java sering dikatakan kurang tinggi. Namun performance Java dapat ditingkatkan menggunakan kompilasi Java lain seperti buatan Inprise, Microsoft ataupun Symantec yang menggunakan Just In Time Compilers (JIT).

10)   Multithreaded

Java mempunyai kemampuan untuk membuat suatu program yang dapat melakukan beberapa perkerjaan secara sekaligus dan simultan.

11)   Dinamis

Java didesain untuk dapat dijalankan pada lingkungan yang dinamis. Perubaan pada suatu class dengan menambahkan properties ataupun method dapat dilakukan tanpa mengganggu program yang menggunakan class tersebut.

KENAPA JAVA BERSIFAT MULTIPLATFORM ?

Mengapa java bersifat multiplatform ? itu dikarenakan Java dilengkapi dengan JVM (Java Virtual Machine) yakni suatu perangkat ‘penerjemah’ bahasa pemrograman sehingga bahasa Java dapat dijalankan di beberapa OS yang berbeda (Cross platform). Di samping itu juga terdapat JRE (Java Runtime environment) untuk menerjemah bahasa di Sistem Operasi yang kita gunakan.

APA ITU IDE ?

IDE (Integrated Development Environment) merupakan sebuah software aplikasi yang memberikan fasilitas kepada programmer komputer ketika membuat program. Biasanya sebuah IDE terdiri dari source code, editor, build automation tools, dan debugger.

Sebuah IDE, atau secara bebas dapat diterjemahkan sebagai Lingkungan Pengembangan Terpadu, (Integrated Development Environment) setidaknya memiliki fasilitas:

1. Editor, yaitu fasilitas untuk menuliskan kode bahasa pemprograman.
2. Compiler, yaitu fasilitas untuk mengecek sintaks dari kode sumber kemudian mengubah dalam bentuk biner yang sesuai dengan bahasa mesin.
3. Linker, yaitu fasilitas untuk menyatukan data binari yang dihasilkan compiler sehingga data-data binari tersebut menjadi satu kesatuan dan menjadi suatu program komputer yang siap dieksekusi.
4. Debuger, yaitu fasilitas untuk mengetes jalannya program, untuk mencari bug/kesalahan yang terdapat dalam program.

IDE juga membantu dalam memeberikan saran dalam penulisan bahasa pemprograman serta memberitahukan jika terdapat error dalam penulisan bahasa pemprograman.

Eclipse IDE adalah salah satu contoh IDE, dimana sifatnya multiplatform (Bisa digunakan pada berbagai sistem operasi, misalnya Windows, Linux, maupun Mac OS), multilanguage (Mendukung berbagai bahasa pemprograman, misalnya C/C++, Cobol, Python, Perl, PHP, dan lain sebagainya) dan multi-role (Selain sebagai IDE untuk pengembangan aplikasi, Eclipse pun bisa digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, test perangkat lunak, pengembangan web, dan lain sebagainya).

TIPE TIPE DATA

* 1. Tipe Data Sederhana
     1. Integer (Bilangan Bulat)

Tipe data yang termasuk tipe data integer 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 | -127 s.d. 128 |
| *Short* | 16 | -32767 s.d. 32768 |
| *Int* | 32 | -2147483647 s.d. 2147483648 |
| *Long* | 64 | -9223372036854775807  s.d.  9223372036854775808 |

1. 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.
2. Short  
   Pada umumnya diaplikasikan pada komputer-komputer 16-bit, yang saat ini semakin jarang keberadaanya.
3. Int  
   Tipe ini merupakan tipe 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.
4. 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.*
   * 1. 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 |

* 1. 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.
  2. 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.
     1. 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* |

* + 1. 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*.
  1. 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*).

* + - 1. 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.
  1. 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.

APA ITU CASTING ?

Casting adalah perubahan tipe data ke tipe data lainnya dengan memperhatikan ukuran kapasitas di setiap tipe data. Beberapa casting disebut konversi. Proses konversi disarankan dari ukuran tipe data kecil ke besar. Alasannya karena jika konversi dilakukan dari ukuran besar ke kecil, ada kemungkinan akan menghilangkan beberapa data karena kapasitas yang tidak mencukupi. Jika di konversi contohnya adalah dari integer ke float, integer ke long, float ke double, dan seterusnya.maka dalam casting yang terjadi adalah dari integer ke string, string ke float dan masih banyak lagi sesuai ketentuan yang ada di java.

SALAH SATU CONTOH PROGRAM SEDERHANA JAVA ITU ADALAH DIBAWAH INI :

**package** interactive;

**import** java.util.Scanner;

**public** **class** interactive {

**public** **static** **void** main(String[] args) {

Scanner input = **new** Scanner(System.***in***);

String name = "John", heshe ="";

**int** age = 35, retirement = 70, daystill=0;

**double** gpa = 4.1;

**boolean** isFemale;

System.***out***.print("Masukkan nama Anda : "); //diisi nama

name = input.nextLine();

System.***out***.print("Masukkan Umur anda : "); // umur untuk menentukan berapa tahun hingga pensiun

age = input.nextInt();

System.***out***.print("Masukkan Indeks anda : ");

gpa = input.nextDouble();

System.***out***.print("Apakah kamu perempuan ? (True/False) : ");

isFemale = input.nextBoolean();

heshe = isFemale ? "Dia(p)" : "Dia(l)";

daystill = retirement - age;

System.***out***.printf("%s berumur %d tahun (%d tahun menuju pensiun). %s mempunyai indeks %4.3f.\n", name, age, daystill, heshe, gpa);

}

}

BAB II

ALUR KONTROL

MENGAPA PROGRAM MEMERLUKAN ALUR KONTROL ?

Sebuah program dikatakan baik apabila alur dari program tersebut tersusun dengan baik berdasarkan proses-proses yang diinginkan. Seperti halnya dengan sebuah drama, program juga membutuhkan alur cerita yang menjadikan program tersebut dapat dimengerti apa tujuannya. Untuk itu kita membutuhkan pengontrol alur dari sebuah program sehingga perogram tersebut dapat dikendalikan. Secara garis besar ada 4 macam pengontrol alur program yaitu pengkondisian, looping, exception handling, dan pencabangan. Namun disini kita akan menjelaskan 2 saja yaitu pengkondisian dan perulangan

* **PENGKONDISIAN**  
  Pengkondisian adalah pengontrol program dengan menggunakan operator bersyarat dengan kondisi yang telah ditentukan. Pada pengontrol alur program dengan pengkondisian ini terdapat dua macam cara yaitu dengan menggunakan if – else atau dengan menggunakan switch – break. Kedua jenis pengkondisian ini memiliki fungsi dan penulisan yang berbeda.  
  Pengkondisain dengan IF – ELSE  
  Pengontrol alur program ini akan mengatur setiap statement yang akan dikerjakan berdasarkan kondisi yang disyaratkan. Jika syarat yang ditentukan bernilai true maka statement pada blok true akan dijalankan tetapi apabilai bernilai false maka statement pada blok false yang dijalankan. Pengkondisian dengan if – else adalah pengrontrol alur program dengan menggunakan operasi boolean. Penulisan pengontrol alur program dengan menggunakan pengkondisian if – else adalah sebagai berikut :

**if** (<operasi **boolean**>){

….. // blok statement jika bernilai true

} **else** {

….. // blok statement jika bernilai false

}  
Contoh penggunaan pengkondian dengan if – else :  
**int** angka = 10;

**if** (angka >= 20){

String hasil = “Angkanya lebih dari 20”; // Jika bernilai True

} **else** {

String hasil = “Angkanya kurang dari 20”; // Jika bernilai false

}

Dari contoh di atas statemant yang dijalankan adalah blok setelah if (blok true) yaitu String hasil = “Angkanya lebih dari 20”;. Dengan demikian dapat dijelaskan bahwa apabila variabel angka bernilai lebih besar sama dengan 20 maka kondisinya adalah true dan memenuhi persyaratan yang telah ditentukan maka statement yang akan dijalankan adalah statement yang ada di blok true yaitu String hasil = “Angkanya lebih dari 20”; sedangkan apabila nilai variabel angka tidak memenuhi persyaratan yang disyaratkan (berkondisi false) maka blok statement yang dijalankan adalah blok setelah else yaitu String hasli = “Angkanya kurang dari 20”;.  
Selain penggunaan if – else, Anda juga dapat menggunakan if – else – if untuk mengontrol program. Penggunaan if – else – if ini apabila persyaratan yang ditentukan lebih dari satu. Penulisan pengontrol alur program dengan menggunakan pengkondisian if – else – if adalah sebagai berikut :

**if** (<operasi **boolean** 1>){

….. // blok statement jika operasi boolean 1 bernilai true

} **else** **if** (<operasi **boolean** 2>){

….. // blok statement jika operasi boolean 1 bernilai false tetapi operasi boolean 2 bernilai true

} **else** {

….. // blok statement jika operasi boolean 1 dan operasi boolean 2 bernilai false

}

***Pengkondisian dengan SWITCH – CASE***  
Pengontrol alur program ini akan mengatur setiap statement yang akan dikerjakan berdasarkan pilihan yang disyaratkan. Dengan kata lain, program akan menjalankan statement sesuai dengan nilai yang didapat. Pengkondisian dengan switch – case adalah pengontrol alur program dengan menggunakan operasi integer (byte, short, dan int). Penulisan pengontrol alur program dengan menggunakan pengkondisian swicth – case adalah sebagai berikut :

**switch** (<ekpresi integer>){

**case** <nilai1> : <blok statement>; **break**;

**case** <nilai1> : <blok statement>; **break**;

**case** <nilai1> : <blok statement>; **break**;

**case** <nilai1> : <blok statement>; **break**;

**default** : <blok statement>; **break**;

}

Contoh penggunakan pengkondisian dengan switch – case  
 **switch** (nilai){

**case** 1 : System.out.println(“Nilai = 1); **break**;

**case** 2 : System.out.println(“Nilai = 2); **break**;

**case** 3 : System.out.println(“Nilai = 3); **break**;

**case** 4 : System.out.println(“Nilai = 4); **break**;

**default** : System.out.println(“Nilai selain 1, 2, 3, dan 4); **break**;

}

Pada contoh di atas, apabila variabel nilai = 1 maka statement yang dijalankan adalah statement yang terdapat pada case 1 yaitu System.out.println(“Nilai = 1);.  
Dari contoh-contoh di atas, dapat ditarik kesimpulan bahwa perbedaan antara pengkondisian dengan if – else dengan pengkondisian dengan switch – case adalah pada jenis data primitif yang digunakan. Pada pengkondisian if – else menggunakan operasi dengan data primitif adalah boolean, sedangkan pengkondisian switch – case menggunakan operasi dengan data primitif berupa integer (selain long).

* **LOOPING (PERULANGAN)**  
  Pengontrol alur program ini akan mengontrol program atau dijalankan sampai memenuhi nilai yang ditentukan. Dengan kata lain looping adalah menjalankan perintah yang sama berulang-ulang dan akan berhenti apabila syarat yang ditentukan telah dipenuhi.  
  Alur program ini biasanya digunakan untuk menuliskan beberapa proses yang sama dengan tidak perlu lagi menuliskannya berulang kali. Misalnya kita ingin menuliskan kata “Serbaguna” sebanyak 100 kali maka kita tidak perlu menuliskannya satu persatu, cukup dengan menggunakan alur program looping.  
  Ada beberapa cara melakukan pengontrol alur program looping yaitu dengan menggunakan for, while, dan do-while.  
  ***for***  
  Perulangan (Looping) dengan for yaitu melakukan perulangan sampai syarat yang telah ditentukan terpenuhi. Looping for seringkali digunakan untuk iterasi sederhana dimana kita dapat mengulang blok pernyataan beberapa kali kemudian berhenti. Perulangan for ini akan selalu berjalan selama kondisi syarat bernilai true dan akan berhenti apabila kondisi syarat bernilai false. Penulisan pengontrol alur program dengan menggunakan looping for adalah sebagai berikut :

**for**(inisialisasi; test ; step){

blok pernyataan;

}

Contoh penggunaan looping for adalah sebagai berikut :

**for**(**int** x = 0; x < 10 ; x++){

System.out.println(“Perulangan ke : “+x);

}

Dari contoh diatas, blok pernyataan akan dijalankan selama nilai x lebih kecil dari 10. Dengan kata lain apabila nilai test adalah true maka blok pernyataan akan selalu dijalankan, dan apabila nilai test adalah false maka proses looping akan dihentikan.  
while  
Perulangan (looping) while digunakan untuk mengulang pernyataan atau blok pernyataan selama kondisi syarat dipenuhi (true). Apabila nilai kondisi syarat tidak dipenuhi (false) maka proses perulangan akan berhenti. Penulisan pengontrol alur program dengan menggunakan looping while adalah sebagai berikut :

**while**(syarat){

blok pernyataan;

}

Contoh penggunaan looping while adalah sebagai berikut :

**int** x = 1;

**while** (x<10){

System.out.println(“Perulangan dengan **while** ke : “ + x);

x++;

}

Dari contoh di atas, blok pernyataan akan dijalankan selama nilai x lebih kecil dari 10, dan akan berhenti apabila nilai z lebih besar atau sama dengan 10.  
do-while  
Perulangan do-while memiliki urutan yang terbalik dibandingan dengan while dan memiliki perbedaan utama yaitu pada do-while pernyataan dijalankan terlebih dahulu kemudian melakukan pemeriksaan kondisi syarat.  
Penulisan pengontrol alur program dengan menggunakan looping do-while adalah sebagai berikut :

**do** {

blok pernyataan;

} **while** (syarat);

Contoh penggunaan looping do-while adalah sebagai berikut :

**int** x = 1;

**do** {

System.out.println(“Perulangan dengan **do**-**while** ke : “ + x);

x++;

} **while** (x<10);

**BAB III**

**DASAR INPUT OUTPUT**

1. DASAR TEORI

Pada program-program yang membutuhkan data-data eksternal, maka diperlukan suatu proses input dan ouput (I/O), dimana pada Java dukungan proses I/O ini sudah disediakan dalam paket java.io. Di dalam paket tersebut tersimpan banyak kelas dan interface siap pakai yang akan memudahkan programmer dalam pengambilan dan penyimpanan informasi dari/ke media lain (misalnya file).

Program Java melakukan proses I/O melalui stream, yaitu sebuah abstraksi yang dapat memberikan atau mendapatkan informasi. Stream dapat dihubungkan dengan peralatan fisik yang terdapat dalam sistem I/O Java, seperti keyboard, file, layar console, soket jaringan, dan lainnya. Walaupun dihubungkan dengan peralatan fisik yang berbeda, cara kerja stream selalu sama, sehingga kode program yang ditulis juga sama untuk masing-masing peralatan fisik. Misalnya, untuk melakukan penulisan sebuah teks ke layar console maupun ke dalam file, maka dapat digunakan kelas dan method yang sama.

Stream ada dua jenis, yaitu stream byte dan stream karakter. Stream byte digunakan untuk memberikan atau menyimpan informasi data dalam bentuk byte, misalnya untuk menulis dan membaca file biner. Sedangkan stream karakter pada proses I/O yang melibatkan data-data berbentuk karakter, misalnya proses baca/tulis ke suatu file teks, dengan menggunakan karakter Unicode.

Pendefinisian stream dilakukan dengan menggunakan empat kelas abstrak, yaitu InputStream dan OutputStream, sebagai superclass untuk kelas-kelas dalam kategori stream byte, dan kelas abstrak Reader dan Writer untuk kategori stream karakter. Melalui proses pewarisan (inheritance), semua kelas yang diturunkan dari

InputStream maupun Reader akan memiliki method read(), yang digunakan dalam proses pembacaan data. Adapun untuk proses penulisan data digunakan method write() dalam semua kelas yang diturunkan dari OutputStream maupun Writer. Daftar beberapa kelas dalam paket java.io yang termasuk dalam kategori stream byte ditunjukkan pada Tabel 1., sedangkan kelas-kelas dalam kategori stream karakter ditunjukkan pada Tabel 2.

Tabel 1. Kelas-kelas stream byte

|  |  |
| --- | --- |
| Nama Kelas | Keterangan |
| BufferedInputStream | Stream input yang telah terbuffer |
| BufferedOutputStream | Stream output yang telah terbuffer |
| ByteArrayInputStream | Stream input yang membaca dari array byte |
| ByteArrayOutputStream | Stream input yang menulis ke array byte |
| DataInputStream | Stream input yang berisi method-method untuk |
|  | membaca tipe data standar |
| DataOutputStream | Stream output yang berisi method-method untuk |
|  | menuliskan tipe data standar |
| FileInputStream | Stream input yang membaca dari sebuah file |
| FileOutputStream | Stream output yang menulis ke sebuah file |
| FilterInputStream | Mengimplementasikan InputStream |
| FilterOutputStream | Mengimplementasikan OutputStream |
| InputStream | Kelas abstrak yang menjelaskan stream input |
| OutputStream | Kelas abstrak yang menjelaskan stream output |
| PipedInputStream | Penyalur input |
| PipedOutputStream | Penyalur output |
| PrintStream | Stream output yang berisi method println() dan print() |

Semua program Java otomatis akan mengimpor paket java.lang yang mendefinisikan sebuah kelas bernama System yang berkaitan dengan lingkungan runtime. Kelas System juga mendefinisikan tiga variabel stream, yaitu in, out, dan err, dimana data-datanya dideklarasikan sebagai public atau static. Dengan demikian variabel-variabel tersebut dapat digunakan tanpa harus membentuk objek dari kelas

System terlebih dahulu. System.out adalah stream output standar dengan default layar console. System.in adalah objek dari tipe PrintStream, defaultnya berupa keyboard dan mengacu pada stream input standar. Sedangkan pada stream error standar terdapat System.err yang secara default berupa layar console. System.out dan System.err adalah objek dari tipe PrintStream dan mempunyai cara kerja yang sama, yaitu mengeluarkan pesan kesalahan ke layar console

|  |  |  |
| --- | --- | --- |
|  | Tabel 2. Kelas-kelas stream karakter | |
|  |  |  |
| Nama Kelas |  | Keterangan |
| BufferedReader |  | Stream karekter input yang telah terbuffer |
| BufferedWriter |  | Stream karakter output yang telah terbuffer |
| CharArrayReader |  | Stream input yang membaca dari array karakter |
| CharArrayWriter |  | Stream input yang menulis ke array karakter |
| FileReader |  | Stream input yang membaca dari file |
| FileWriter |  | Stream output yang menulis ke file |
| FilterReader |  | Reader yang terfilter |
| FilterWriter |  | Writer yang terfilter |
| InputStreamReader |  | Stream input yang menerjemahkan byte ke karakter |
| LineNumberReader |  | Stream input yang menghitung jumlah baris |
| OutputStreamWriter |  | Stream input yang menerjemahkan karakter ke byte |
| PipedReader |  | Penyalur input |
| PipedWriter |  | Penyalur output |
| PrintWriter |  | Stream output yang berisi method println() dan print() |
| Reader |  | Kelas abstrak yang menjelaskan stream karakter input |
| StringReader |  | Stream input yang membaca dari sebuah string |
| StringWriter |  | Stream output yang menulis ke sebuah string |
| Writer |  | Stream output |

* Node Stream

Terdapat tiga tipe dasar node, yaitu:

* File
* Memori (misalnya objek array atau String)
* Pipe (suatu kanal dari satu proses)

Tipe dari node stream dapat dibagi lagi sesuai dengan Tabel 3.

Tabel 3. Tipe Node Stream

|  |  |  |
| --- | --- | --- |
| Tipe | Stream Karakter | Stream Byte |
| File | FileReader | FileInputStream |
|  | FileWriter | FileOutputStream |
| Memory: | CharArrayReader | ByteArrayInputStream |
| array | CharArrayWriter | ByteArrayOutputStream |
| Memory: | StringReader | N/A |
| String | StringWriter |  |
| Pipe | PipeReader | PipedInputStream |
|  | PipeWriter | PipedOutputStream |

* Melakukan Input

Dalam Java, input console dilakukan melalui pembacaan terhadap stream System.in. Untuk mendapatkan karakter-karakter yang dimasukkan melalui keyboard ke dalam layer console, diperlukan membungkus System.in di dalam objek BufferedReader. Hal ini dilakukan untuk membentuk stream karakter karena System.in sebenarnya merupakan stream bute. Adapun bentuk constructor dari BufferedReader sebagai berikut,

BufferedReader(Reader inputReader) inputReader adalah stream yang akan dihubungkan dengan instance atau objek dari kelas BufferedReader yang dibuat. Karena Reader merupakan kelas abstrak, maka perlu dicari kelas turunannya yang berupa kelas konkrit. Salah satunya adalah kelas InputStreamReader, yang dapat mengonversi byte ke karakter. Sedangkan agar objek dari InputStreamReader dapat dihubungkan dengan System.in, perlu digunakan bentuk constructor seperti berikut,

*InputStreamReader(InputStream inputStream)*

Dalam hal ini, inputStream dapat diisi dengan System.in. Sehingga untuk membuat objek BufferedReader yang dapat terhubung dengan keyboard, perlu digunakan

kode berikut:

*BufferedReader br = new BufferedReader (new InputStreamReader (System.in));*

Atau bisa juga ditulis:

*InputStreamReader isr = new InputStreamReader(System.in);*

*BufferedReader br = new BufferedReader(isr);*

Pada tahap ini objek br sudah siap digunakan untuk melakukan proses input, yaitu dengan melakukan pemanggilan terhadap method read() maupun readline().

* Membaca Input Data Karakter

Untuk membaca input berupa karakter, digunakan method read() yang terdapat pada kelas BufferedReader, dengan pendeklarasian sebagai berikut:

*int read() throws IOException*

* Membaca Input Data String

Untuk melakukan input berupa string harus digunakan method readLine(), bukan read(), dengan deklarasi method sebagai berikut:

*String readLine() throws IOException*

Proses input data string di dalam Java juga berlaku untuk karakter spasi.

* Membaca Input Data Numerik

Untuk input berupa data numerik, maka caranya sama dengan melakukan input terhadap data string. Selanjutnya string hasil input tersebut dikonversi ke tipe numerik dengan memanggil method parseInt() yang terdapat pada kelas Integer untuk bilangan bulat, atau parseDouble() yang terdapat pada kelas Double untuk bilangan riil. Parameter yang dilewatkan ke dalam method tersebut harus bertipe string. Sebaiknya proses konversi berada di dalam blok try-catch untuk menghindari masukan string yang tidak dapat dikonversi ke bilangan bulat, misalnya string “abc”.

* Menampilkan Output

Untuk menampilkan output ke layar console, dapat dengan mudah dilakukan melalui method print() maupun println(). Untuk media output lainnya (misalnya: file), method yang digunakan untuk melakukan proses ini adalah write().

Selain menggunakan System.out untuk menulis ke layar console, Java juga menyediakan kelas PrintWriter untuk keperluan yang sama. PrintWriter adalah salah satu kelas stream yang berbasis pada data karakter (character-based). Bentuk constructor dari kelas PrintWriter adalah sebagai berikut:

*PrintWriter(OutputStream outputStream, Boolean flushOnNewLine)*

Dimana outputStream adalah objek dari OutputStream dan flushOnNewLine adalah parameter Boolean yang menyatakan apakah stream output akan dibuang atau tidak setiap kali method println() dipanggil.

Seperti halnya System.out, PrintWriter juga mendukung method print() dan println() untuk semua tipe data, termasuk yang berjenis objek; cara kerjanya pun sama seperti pada saat menggunakan System.out. Apabila argumen yang dilewatkan berupa objek, maka PrintWriter akan memanggil method toString() dari objek bersangkutan, kemudian menampilkan hasilnya. Cara pembentukan objek PrintWriter untuk menampilkan output adalah sebagai berikut:

*PrintWriter pw = new PrintWriter(System.out, true)*