**Catatan Pembelajaran Java**

**Penggunaan Library Java**

|  |  |
| --- | --- |
| Menggunakan seluruh library pada “util” | import java.util.\*; |
| Penggunaan String | import java.lang.String; |
| Mendapatkan input | import java.util.Scanner; |
| Penggunaan Array | Import java.util.Arrays; |
|  |  |
|  |  |

**Mengambil input / user input**

1. Panggil library scanner/ untuk input
2. Buat **Objek** scanner yang terhubung dengan kelas dari scanner

*Scanner vaiabel = new Scanner(System,in);*

1. Buat variabel yang akan menyimpan input dari user

*Variabel\_Input = variabel.next();* => next() digunakan untuk tipe data string,

nextInt() digunakan untuk tipe data Integer,

untuk tipe data lain juga berbeda.

**Arrays**

Pemanggilan untuk menggunakan fungsi arrays :

java.util.Arrays; => libary array di Java

Sifat array :

1. Jika kita mendeklarasikan array baru dan mengisi nilainya sama dengan array yang lama maka yang dilakukan adalah **menempatkan array baru pada alamat yang sama** dengan array lama. Contoh :

Int[] array1= {1,2,3,4,5};

Int[] array2 = new int[5];

array2 = array1; //dengan begini isi array2 sama dngan array1, yang mana alamat

array 2 disamakan dengan alamat memori array1

tetapi jika kita **mengubah salah satu isi** di array2 maka array1 **juga ikut berubah**.

Hal ini terjadi karena isi array yang diubah pada array2 merupakan array yang sama dengan isi array pada array 1 karena **alamat yang digunakan sama**.

1. Untuk mengambil isi dari array lama dapat menggunakan metode **for loop** sehingga array baru **tidak di tempatkan di alamat memori yang sama** dengan array yang lama.

Selain *for loop* dapat juga menggunakan perintah aray ***Arrays.copyOf(nama array)*.**

|  |  |
| --- | --- |
| Arrays.toString(*nama array*); | Mengubah array menjadi string, betuk string :  [1,2,3,4,5] |
| Arrays.deepToString(*nama array*); | Mengubah array 2 dimensi menjadi sebuah string, bentuk string: [[1,2], [3, 4]] |
| Int[] arrayB = Arrays.copyOf(*arrayA*, 5)  “5 => panjang isi array yang ingin di copy” | Membuat isi arrayB menjadi sama dengan isi pada arrayA, tetapi yang disalin adalah isi nya bukan menyamakan alamatnya. Sehingga jika arrayB berubah maka arrayA tidak ikut berubah, begitupun sebaliknya. |
| arrayB = Arrays.copyOfRange(arrayA, 2, 5)  “2, 5 => array yang di copy merupakan array pada index ke-2 sampe ke-5” | Membuat isi array b menjadi sama dengan isi pada arrayA menggunakan range tertentu |
| Arrays.fill(arrayB, 2);  “2 => angka yang akan diisikan ke dalam array” | Mengisi semua tempat apada array dengan angka yang ditentukan |
| Arrays.equals(arrayA,arrayB) | Melakukan pengecekkan isi dari array. Akan mengembalikan nilai *true* atau *false* |
| Arrays.compare(araryA, arrayB) | Membandingkan isi dari kedua arary. Mengembalikan nilai 1 jika array pada parameter pertama yang lebih besar, nilai -1 jika parameter kedua yang lebih besar, dan mengembalikan 0 jika nilainya sama. |
| Arrays.mismatch(arrayA, arrayB) | Melihat index dari isi array yang berbeda. |
| Arrays.sort(arrayA) | Melakukan sorting / pengurutan isi array dari nilai terendah ke nilai yang tertinggi. |
| arayA.length | Menghitung panjang array |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**String**

Library => *import java.lang.String;*

Sifat string :

1. Ketika variabel string baru yang isi nya disamakan dengan string lama kemudian isinya diubah maka string tersebut akan menggunakan alamat **memori yang baru**.

( **Hal ini berbeda dengan sifat Array ).**

1. Variabel yang diisi dengan **string saat dideklarasikan** akan masuk ke dalam **alamat memori literal**. Jika variabel string yang isinya sama dilakukan pengecekkan menggunakan IF maka hasilnya akan **true**.

*If( stringA == stringB)* => akan menghasilkan nilai True.

1. Tetapi jika varabel diisi dengan menggunakan string yang **diinputkan oleh user** ketika program sudah berjalan, variabel tersebut akan diletakkan di alamat **memori** **heap.**

Artinya memori yang digunakan **sama** dengan memori untuk menyimpan array, yang mana jika dilakukan pengecekan isi yang sama menggunakan **IF** maka menghasilkan nilai **false.** Untuk melakukan pengecekannya dapat menggunakan perintah **equals**, bisa dilihat pada tabel di bawah.

|  |  |
| --- | --- |
| *Nama\_string*.charAt(5);  “5 => mengambil index ke 5” | Mengambil karakter dari string pada index tertentu |
| *Nama\_string*.substring(10, 15);  “10, 15 => range index yang diambil 10-15” | Mengambil karakter dari string pada index range index tertentu lalu mengembalikan dalam bentuk string. |
| *namaString*.toUpperCase(); | Mengubah isi string menjadi hurup kapital semua / Uppercase |
| *namaString*.toLowerCase(); | Mengubah isi string menjadi hurup kecil semua / Lowercase |
| *namaString*.equals(*namaString2)*; | Untuk mengecek apakah isi string pertama sama dengan string kedua |
| *namaString*.compareTo(*namaString2);* | Untuk mengetahui perbandingan isi string menurut abjad dan pengecekan dilakukan dari karakter pertama yang **berbeda**. Fungsi ini akan mengembalikan nilai integer. Nilai ini adalah jarak antara karakter **String1 dari String2**. Misal **string1 = Alam** dan **string2 = Cerah**, maka fungsi akan mengembalikan nilai **-2.** Jika variabel yang dicek dibalikan  **String2 compare string1** , maka akan mengembalikan nilai **2.** |