***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_collection\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

**Nama : Muhamad Wahyu Saputra**

**Kelas : XII RPL B**

**Absen : 34**

**Resume Materi**

1. ***Collection framework***

*Framework pada java untuk menampung dan mengelola sekumpulan data / object  
Pada bahasa C++ dikenal dengan nama Standart Template Library (STL)*

**Perbedaan Collection dan Array:**

Kekurangan **array** adalah :

1. *Kapasitas yang tidak bisa diubah*
2. *Waktu untuk menambah dan menghapus data lambat*
3. *Tidak memiliki fungsi pengolah data*

Kelebihan **collection** adalahuntuk memperbaiki kekurangan tersebut.

Tujuan dari **collection framework**:

1. *Menambah data*
2. *Menghapus data*
3. *Mengubah data*
4. *Mencari data*
5. *Mengurutkan data*
6. **Collection**

Collection merupakan suatu struktur data/kontainer yang memuat sekumpulan object-object dan digunakan untuk menyimpan,dan memanipulasi data. Suatu arsitektur untuk mewakili dan memanipulasi Collection, terdiri dari:

* **Interfaces:**

*merupakan tipe data abstract yang mewakili Collection,yang membuat collection dapat dimanipulasi secara terpisah dari detail representasinya.*

* **Implementation:**

*merupakan implementasi konkret dari Collection Interface. Intinya merupakan struktur data yang dapat digunakan kembali.*

* **Algorithms:**

*merupakan metode yang melakukan tugas komputasi, seperti searching dan sorting.*

1. **INTERFACE COLLECTION**

**Collection** menerima parameter berupa tipe data yang akan disimpan menggunakan <> seperti berikut **Collection<E>**

Berikut adalah fungsi dasar yang dimiliki oleh **Collection:**

* ***add: menambah data***
* ***remove: menghapus data***
* ***contains: mencari data pada collection***
* ***size: mendapatkan jumlah data yang tersimpan***
* ***clear: mengosongkan collection***
* ***toArray: mengubah collection menjadi array***
* **Collection framework secara umum :**

1. **Set**

*Struktur data yang menampung elemen-elemen yang unik (tidak boleh ada elemen kembar). Disini set memiliki aturan bahwa data/object di dalamnya tidak boleh ada yang sama (unik).Set berguna untuk mengelola data yang tidak mungkin sama, misalnya: “Nomor telepon, nomor KTP, alamat email”.*

*Set dapat diimplementasikan menggunakan:*

1. ***HashSet****: tidak memperhatikan posisi data*
2. ***LinkedHashSet****: data disimpan berdasarkan urutan masukannya*
3. ***TreeSet:*** *data disimpan berdasarkan nilai terkecil ke terbesar*

*Untuk operasi tambah dan hapus data, HashSet dan LinkedHashSet memiliki kompleksitas yang sama yaitu O(1).Namun dari sisi performa HashSet lebih baik. Sementara TreeSet memiliki kompleksitas O(log(n)).*

* 1. **Hashset**

*Penyimpanan elemen diletakkan secara acak (tidak terurut). Maka output yang keluar adalah acak.*

* 1. **LinkedHashset**

*Elemen-elemen didalam LinkedHashSet berurutan seperti saat disisipkan.* *Maka outputnya adalah terurut.*

* 1. **Treeset**

*Treeset merupakan implementasi dari interface SortedSet. Maka outputnya akan keluar dari nilai terkecil, atau urutan alfabet paling pertama.*

1. **List**

* *List menyimpan data secara sekuensial seperti array, sehingga pengaksesannya dapat menggunakan sistem indexing.*
* *List dapat menyimpan elemen-elemen yang duplikat/kembar, dan mengijinkan user untuk menentukan di mana elemen disimpan.*
* *List berguna untuk mengelola data yang perlu memperhatikan posisi data*
* *List dapat diimplementasikan menggunakan:*
* ***ArrayList:*** *menggunakan array.*
* ***Vector:*** *ArrayList yang tersinkronisasi.*
* ***LinkedList:*** *menggunakan double linkedlist.*

*Untuk operasi ambil data,* ***ArrayList/Vector*** *memiliki performa yang paling baik.Sedangkan untuk operasi tambah dan hapus data,* ***LinkedList*** *memiliki performa yang paling baik.*

* 1. **Arraylist**

*Menyimpan elemen-elemen di dalam suatu array, dimana array tersebut diciptakan secara dinamis. Penggunaannya ketika diperlukan akses secara acak melalui index tanpa penyisipan atau penghapusan elemen-elemen kecuali pada ujung list*

* 1. **LinkedList**

*Menyimpan elemen-elemen di dalam suatu LinkedList . Penggunaannya ketika diperlukan penyisipan atau penghapusan elemen-elemen di mana saja di dalam list.*

* 1. **VECTOR**

*vector memiliki metode tersinkronisasi untuk mengakses dan memodifikasi vektor. Sinkronisasi tersebut dapat mencegah korupsi data ketika suatu vektor diakses dan dimodifikasi dua thread atau lebih secara bersamaan.*

1. **Queue (Interface)**

*Queue menggunakan prinsip first in first out (FIFO), data yang ditambahkan paling dulu akan diambil paling awal. Queue berguna untuk mengelola data yang menggunakan prinsip FIFO atau antrian, misalnya:*

* *Antrian rumah sakit.*
* *Antrian bank.*
* *Antrian customer service.*

*Queue dapat diimplementasikan menggunakan:*

* ***LinkedList****: antrian FIFO standar.*
* ***PriorityQueue****: antrian yang urutannya berdasarkan kriteria tertentu (menggunakan interface Comparable).*

1. **Map**

*Merupakan container yang menyimpan elemen bersama dengan kuncinya(index). Map menggunakan sistem indexing (disebut dengan key), dimana index tidak harus angka, bisa juga berupa text atau object.kunci harus unik/tidak boleh kembar dan bisa berupa sembarang object. Misalnya :*

* *Penduduk (key: nomor KTP)*
* *Siswa (key: nomor induk)*
* *Buku (key: ISBN)*

*Map dapat diimplementasikan menggunakan:*

* ***HashMap****: tidak memperhatikan posisi data*
* ***LinkedHashMap****: Data disimpan berdasarkan urutan masukannya*
* ***TreeMap:*** *data disimpan berdasarkan nilai terkecil ke terbesar*

*Untuk operasi tambah dan hapus data, HashMap dan LinkedHashMap memiliki kompleksitas yang sama yaitu O(1). Namun dari sisi performa HashMap lebih baik. Sementara TreeMap memiliki kompleksitas O(log(n)).*

1. **INTERFACE ITERATOR**

*Fasilitas pada Java API yang dapat digunakan untuk melakukan iterasi komponen-komponen dalam Koleksi.Ada tiga method yang sering digunakan dalam Iterator:*

* ***hasNext()*** *: Menentukan apakah masih ada sisa koleksi.*
* ***next()****: Mengembalikan elemen object pada koleksi.jika sudah tidak ada elemen lagi namun berusaha diambil akan muncul pesan : “NoSuchElementException”.*
* ***remove() :*** *Menghapus elemen yang terakhir kali diakses oleh lterator.*

1. **Mengambil data pada collection**

* *Mengambil data pada collection disarankan untuk menggunakan iterator*
* *Alternatif lain adalah dengan mengubah collection (selain map) menjadi array dengan method toArray()*
* *Namun beberapa turunan dari collection juga memiliki method khusus untuk mengambil data.*

=========================================================================================()=========================================()=================================================--------------================================

---------------------------------------------(----------------------------)---------------------------------------------------------------------------------------------------------------------------------------------------------------------------