

MODUL 2

“COLLECTION PADA JAVA”



Oleh
SAFIRA MAYA SHOVIE, S.Pd

JAVA COLLECTION FRAMEWORK

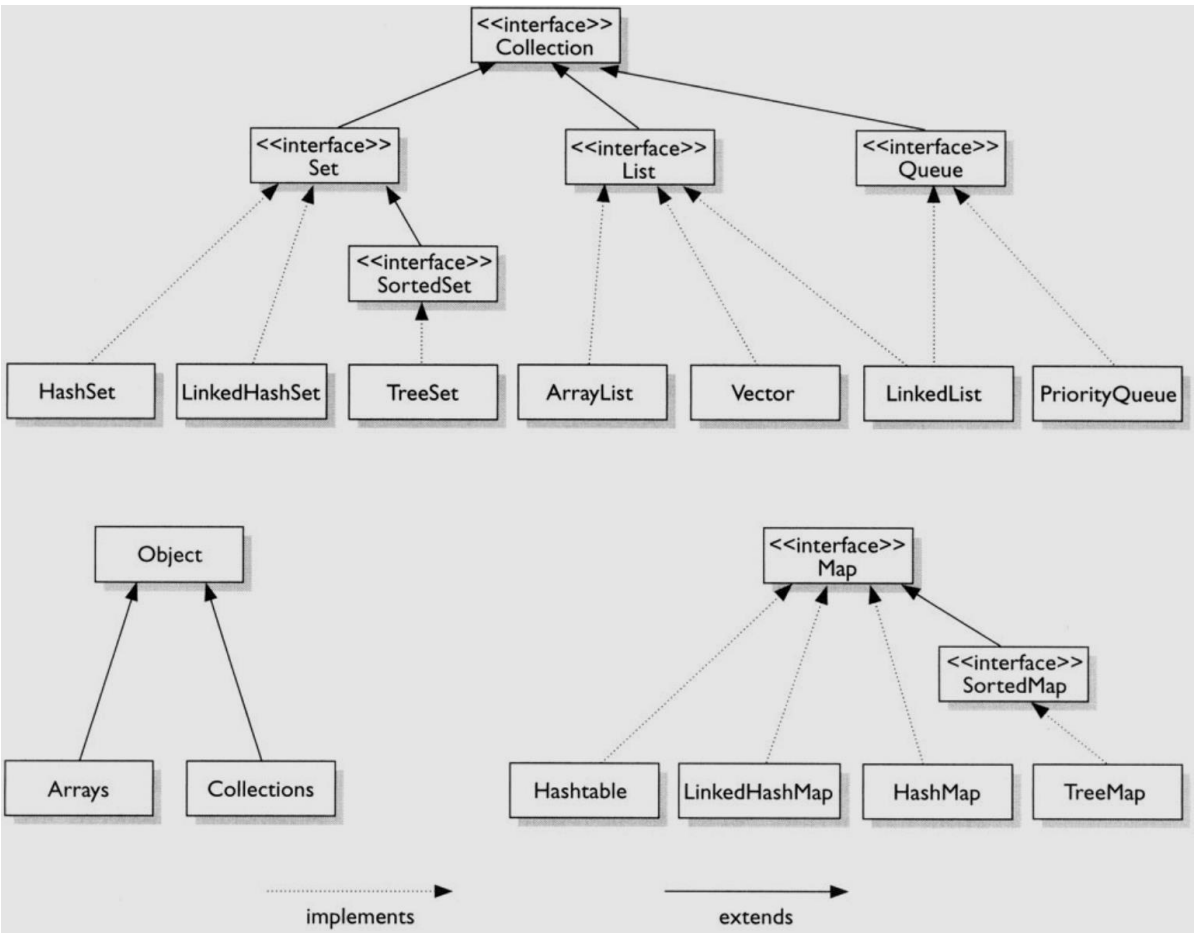
A. TUJUAN PEMBELAJARAN

- 1. Memahami cara penyimpanan obyek menggunakan Collection.
- 2. Mengetahui pengelompokan dari Collection.
- 3. Mengetahui perbedaan dari interface Set, List dan Map.
- 4. Mengetahui penggunaan class-class dari interface Set, List dan Map.
- 5. Mengetahui cara penggunaan Iterasi dan Enumeration.

B. DASAR TEORI

Collection adalah suatu obyek yang bisa digunakan untuk menyimpan sekumpulan obyek. Obyek yang ada dalam Collection disebut elemen. Collection menyimpan elemen yang bertipe Object, sehingga berbagai tipe obyek bisa disimpan dalam Collection. Class class mengenai Collection tergabung dalam Java Collection Framework. Class-class Collection diletakkan dalam package java.util dan mempunyai dua interface utama yaitu Collection dan Map. Mulai java 1.5 (juga dikenal sebagai J2SE 5), semua class yang termasuk Java Collection Framework adalah class generics. Untuk kompatibilitas dengan versi java sebelumnya, penggunaan generics tidak diharuskan, namun sangat disarankan.

Collection terbagi menjadi 3 kelompok yaitu Set, List dan Map. Berikut ini adalah struktur hierarki interface dan class yang termasuk dalam kelompok collection ini.



Gambar 3.1 Struktur Collection di Java

Java Collections Framework terbagi menjadi tiga kelompok:

1. Set

Set mengikuti model himpunan, dimana obyek/anggota yang tersimpan dalam Set harus unik. Urutan maupun letak dari anggota tidaklah penting, hanya keberadaan anggota saja yang penting. Class-class yang mengimplementasikan interface Set adalah `HashSet`. Interface `SortedSet` merupakan subInterface dari interface Set. Untuk mengurutkan Set, kita dapat menggunakan class yang mengimplementasikan interface `SortedSet` yaitu class `TreeSet`.

2. List

List digunakan untuk menyimpan sekumpulan obyek berdasarkan urutan masuk (ordered) dan menerima duplikat. Cara penyimpanannya seperti array, oleh sebab itu memiliki posisi awal dan posisi akhir, menyisipkan obyek pada posisi tertentu, mengakses dan menghapus isi list, dimana semua proses ini selalu didasarkan pada urutannya. Class-class yang mengimplementasikan interface List adalah `Vector`, `Stack`, `LinkedList` dan `ArrayList`. Terdapat interface `Queue` yang cara penyimpanan seperti List, interface ini menyimpan obyek menggunakan metode FIFO (First In First Out) yaitu obyek yang masuk pertama keluar pertama. Class-class yang mengimplementasikan interface `Queue` adalah `PriorityQueue` dan `LinkedList`. Data yang tersimpan pada obyek `PriorityQueue` akan diurutkan, data tersebut harus mengimplementasikan obyek `Comparable` atau `Comparator`.

3. Map

Perbedaan mendasar map dengan collection yang lain, untuk menyimpan obyek pada Map, perlu sepasang obyek, yaitu key yang bersifat unik dan nilai yang disimpan. Untuk mengakses nilai tersebut maka kita perlu mengetahui key dari nilai tersebut. Map juga dikenal sebagai dictionary/kamus. Pada saat menggunakan kamus, perlu suatu kata yang digunakan untuk pencarian. Class-class yang mengimplementasikan Map adalah `Hashtable`, `HashMap`, `LinkedHashMap`. Untuk mengurutkan Map menggunakan interface `SortedMap`, class yang mengimplementasikan interface tersebut adalah `TreeMap`.

C. LATIHAN

1. SET

a. HashSet

```
1 package collection;
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         Set<String> set = new HashSet<String>();
6         set.add("081233");
7         set.add("082221");
8         set.add("083556");
9         set.add("081334");
10        set.add("088854");
11        set.add("083577");
12
13        System.out.println(set);
14    }
15 }
16
17
```

Output - Collection (run) %

run:
[082221, 081233, 083556, 081334, 088854, 083577]
BUILD SUCCESSFUL (total time: 4 seconds)

Berdasarkan program dan output diatas menggunakan *HashSet* maka hasilnya adalah acak.

b. LinkedHashSet

```
1 package collection;
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         Set<String> set = new LinkedHashSet<String>();
6         set.add("081233");
7         set.add("082221");
8         set.add("083556");
9         set.add("081334");
10        set.add("088854");
11        set.add("083577");
12
13        System.out.println(set);
14    }
15 }
16
```

Output - Collection (run) %

run:
[081233, 082221, 083556, 081334, 088854, 083577]
BUILD SUCCESSFUL (total time: 0 seconds)

Berdasarkan program dan output diatas menggunakan *LinkedHashSet* maka hasilnya adalah urut sesuai inputan.

c. TreeSet

```
1 package collection;
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         Set<String> set = new TreeSet<String>();
6         set.add("081233");
7         set.add("082221");
8         set.add("083556");
9         set.add("081334");
10        set.add("088854");
11        set.add("083577");
12
13        System.out.println(set);
14    }
15 }
16 }
```

collection.Tester > main >

Output - Collection (run) %

run:
[081233, 081334, 082221, 083556, 083577, 088854]
BUILD SUCCESSFUL (total time: 0 seconds)

Berdasarkan program dan output diatas menggunakan *TreeSet* yaitu ascending / nilai dari kecil ke besar dan jika dari huruf yaitu dari a ke z.

2. LIST

a. ArrayList

Syntax Program :

```
1 package collection;
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         List<String> list = new ArrayList<String>();
6         list.add("Pemrograman Berorientasi Objek");
7         list.add("Pemrograman Dasar");
8         list.add("Sistem Komputer");
9         list.add("Jaringan Dasar");
10        list.add("Android Studio");
11        list.add(1, "Pemrograman Web");
12        System.out.println(list);
13    }
14 }
15 }
```

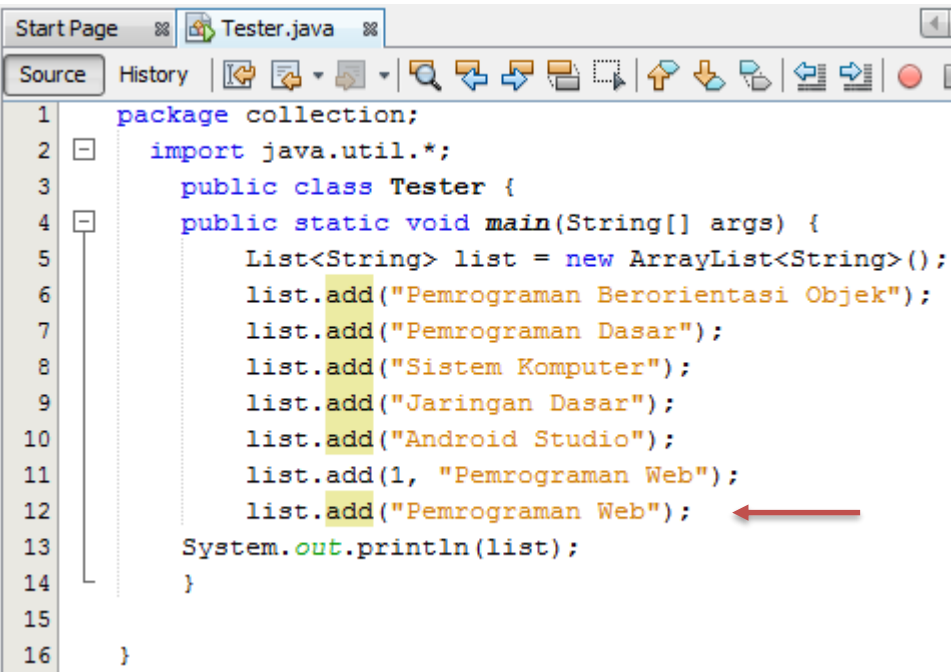
Output Program :

Output - Collection (run) %

run:
[Pemrograman Berorientasi Objek, Pemrograman Web, Pemrograman Dasar, Sistem Komputer, Jaringan Dasar, Android Studio]
BUILD SUCCESSFUL (total time: 0 seconds)

Pada output tersebut setelah “Pemrograman Berorientasi Objek yaitu Pemrograman Web, mengapa demikian ? karena pada syntax diatas menunjukkan bahwa output dipanggil berdasarkan indeks, dimana Pemrograman Web berada pada indeks 1.

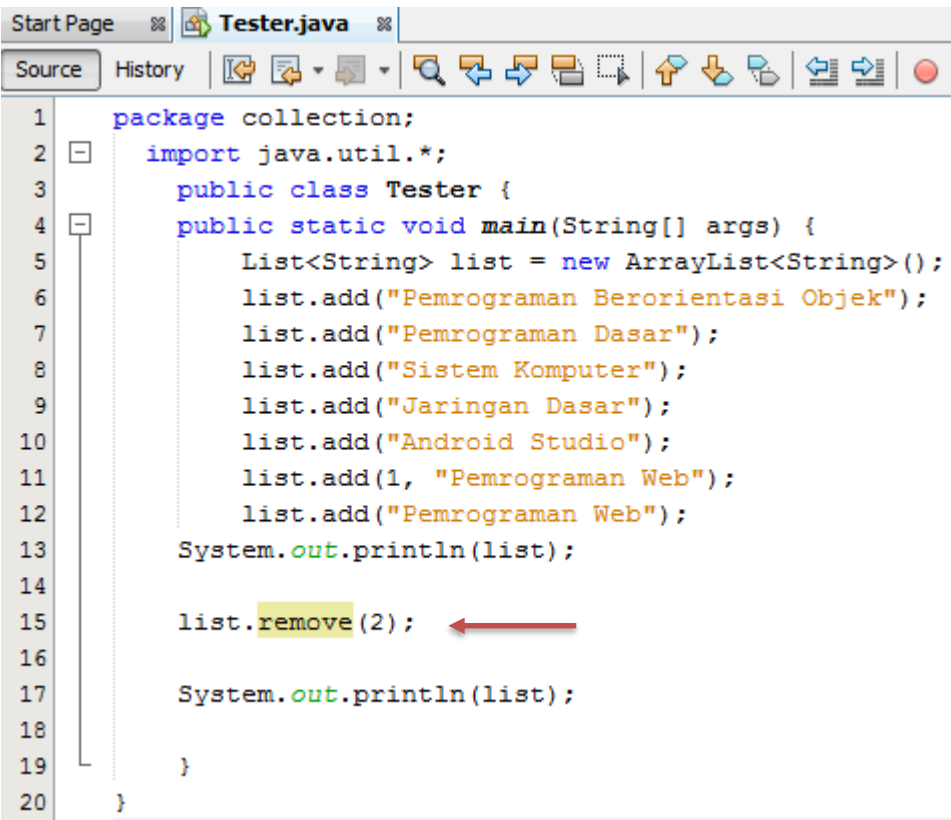
Pada ArrayList juga dapat menambah data duplikat, dan jika ingin menghapus data menggunakan perintah remove (index)



```
1 package collection;
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         List<String> list = new ArrayList<String>();
6         list.add("Pemrograman Berorientasi Objek");
7         list.add("Pemrograman Dasar");
8         list.add("Sistem Komputer");
9         list.add("Jaringan Dasar");
10        list.add("Android Studio");
11        list.add(1, "Pemrograman Web");
12        list.add("Pemrograman Web");
13        System.out.println(list);
14    }
15 }
16 }
```

Maka outputnya adalah :

run:
[Pemrograman Berorientasi Objek, Pemrograman Web, Pemrograman Dasar, Sistem Komputer, Jaringan Dasar, Android Studio, Pemrograman Web]
BUILD SUCCESSFUL (total time: 4 seconds)



```
1 package collection;
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         List<String> list = new ArrayList<String>();
6         list.add("Pemrograman Berorientasi Objek");
7         list.add("Pemrograman Dasar");
8         list.add("Sistem Komputer");
9         list.add("Jaringan Dasar");
10        list.add("Android Studio");
11        list.add(1, "Pemrograman Web");
12        list.add("Pemrograman Web");
13        System.out.println(list);
14
15        list.remove(2);
16
17        System.out.println(list);
18    }
19 }
20 }
```

Maka outputnya adalah :

```
run:
[Pemrograman Berorientasi Objek, Pemrograman Web, Pemrograman Dasar, Sistem Komputer, Jaringan Dasar, Android Studio, Pemrograman Web]
[Pemrograman Berorientasi Objek, Pemrograman Web, Sistem Komputer, Jaringan Dasar, Android Studio, Pemrograman Web]
BUILD SUCCESSFUL (total time: 0 seconds)
```

b. LinkedList menggunakan Queue

```
1 package collection;
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         Queue<String> queue = new LinkedList<String>();
6         queue.add("Pemrograman Berorientasi Objek");
7         queue.add("Pemrograman Dasar");
8         queue.add("Sistem Komputer");
9         queue.add("Jaringan Dasar");
10        queue.add("Android Studio");
11        queue.add("Pemrograman Web");
12
13        System.out.println(queue);
14    }
15 }
16
17 }
```

Output program diatas adalah berurutan sesuai inputan.

Jika ingin menghapus data menggunakan perintah remove (index) dan mengambil data menggunakan perintah (element dan peek)

```
1 package collection;
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         Queue<String> queue = new LinkedList<String>();
6         queue.add("Pemrograman Berorientasi Objek");
7         queue.add("Pemrograman Dasar");
8         queue.add("Sistem Komputer");
9         queue.add("Jaringan Dasar");
10        queue.add("Android Studio");
11        queue.add("Pemrograman Web");
12
13        System.out.println(queue);
14
15        queue.remove("Sistem Komputer");
16
17        System.out.println(queue);
18    }
19 }
20
21 }
```

Output Program :

```
run:
[Pemrograman Berorientasi Objek, Pemrograman Dasar, Sistem Komputer, Jaringan Dasar, Android Studio, Pemrograman Web]
[Pemrograman Berorientasi Objek, Pemrograman Dasar, Jaringan Dasar, Android Studio, Pemrograman Web]
BUILD SUCCESSFUL (total time: 0 seconds)
```

```

1 package collection;
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         Queue<String> queue = new LinkedList<String>();
6         queue.add("Pemrograman Berorientasi Objek");
7         queue.add("Pemrograman Dasar");
8         queue.add("Sistem Komputer");
9         queue.add("Jaringan Dasar");
10        queue.add("Android Studio");
11        queue.add("Pemrograman Web");
12
13        System.out.println(queue);
14
15        queue.remove("Sistem Komputer");
16
17        System.out.println(queue);
18
19        System.out.println(queue.element());
20    }
21 }
22
23

```

Output Program :

```

run:
[Pemrograman Berorientasi Objek, Pemrograman Dasar, Sistem Komputer, Jaringan Dasar, Android Studio, Pemrograman Web]
[Pemrograman Berorientasi Objek, Pemrograman Dasar, Jaringan Dasar, Android Studio, Pemrograman Web]
Pemrograman Berorientasi Objek

```

c. PriorityQueue

```

1 package collection;
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         Queue<String> queue = new PriorityQueue<String>();
6         queue.add("Pemrograman Berorientasi Objek");
7         queue.add("Pemrograman Dasar");
8         queue.add("Sistem Komputer");
9         queue.add("Jaringan Dasar");
10        queue.add("Android Studio");
11        queue.add("Pemrograman Web");
12
13        System.out.println(queue);
14    }
15 }
16
17

```

Output program :

```

run:
[Android Studio, Jaringan Dasar, Pemrograman Web, Pemrograman Dasar, Pemrograman Berorientasi Objek, Sistem Komputer]
BUILD SUCCESSFUL (total time: 4 seconds)

```

Ket : Tergantung kebutuhannya seperti apa dan bisa diatur, dimana terdapat kriterianya yang dapat diatur di interface comparable.

3. MAP

a. HashMap

```
1 package collection;
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         Map<String, String> map = new HashMap<String, String>();
6         map.put("0001", "Sofia");
7         map.put("0654", "Haikal");
8         map.put("4003", "Raisya");
9         map.put("0123", "Panji");
10        map.put("43792", "Ivana");
11        map.put("74501", "Ridwan");
12
13        System.out.println(map);
14    }
15 }
16
17
```

Output - Collection (run) %

```
run:
{0654=Haikal, 0123=Panji, 43792=Ivana, 74501=Ridwan, 0001=Sofia, 4003=Raisya}
BUILD SUCCESSFUL (total time: 0 seconds)
```

Berdasarkan program dan output diatas menggunakan HashMap, hasilnya adalah acak.

b. LinkedHashMap

```
1 package collection;
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         Map<String, String> map = new LinkedHashMap<String, String>();
6         map.put("0001", "Sofia");
7         map.put("0654", "Haikal");
8         map.put("4003", "Raisya");
9         map.put("0123", "Panji");
10        map.put("43792", "Ivana");
11        map.put("74501", "Ridwan");
12
13        System.out.println(map);
14    }
15 }
16
17
```

collection.Tester >

Output - Collection (run) %

```
run:
{0001=Sofia, 0654=Haikal, 4003=Raisya, 0123=Panji, 43792=Ivana, 74501=Ridwan}
BUILD SUCCESSFUL (total time: 0 seconds)
```

Berdasarkan program dan output diatas menggunakan LinkedHashMap, hasilnya adalah berurutan sesuai dengan inputan.

c. **TreeMap**

```
1 package collection;
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         Map<String, String> map = new TreeMap<String, String>();
6         map.put("0001", "Sofia");
7         map.put("0654", "Haikal");
8         map.put("4003", "Raisya");
9         map.put("0123", "Panji");
10        map.put("43792", "Ivana");
11        map.put("74501", "Ridwan");
12
13        System.out.println(map);
14    }
15 }
16
17 }
```

collection.Tester > main >

Output - Collection (run) %

run:
{0001=Sofia, 0123=Panji, 0654=Haikal, 4003=Raisya, 43792=Ivana, 74501=Ridwan}
BUILD SUCCESSFUL (total time: 0 seconds)

Berdasarkan program dan output diatas menggunakan *TreeMap* yaitu ascending / nilai dari kecil ke besar dan jika dari huruf yaitu dari a ke z.

Jika ingin menghapus data, yaitu dengan memasukkan key-nya seperti berikut :

```
2 import java.util.*;
3 public class Tester {
4     public static void main(String[] args) {
5         Map<String, String> map = new TreeMap<String, String>();
6         map.put("0001", "Sofia");
7         map.put("0654", "Haikal");
8         map.put("4003", "Raisya");
9         map.put("0123", "Panji");
10        map.put("43792", "Ivana");
11        map.put("74501", "Ridwan");
12
13        System.out.println(map);
14
15        map.remove("4003");
16
17        System.out.println(map);
18    }
19 }
20 }
```

Output - Collection (run) %

run:
{0001=Sofia, 0123=Panji, 0654=Haikal, 4003=Raisya, 43792=Ivana, 74501=Ridwan}
{0001=Sofia, 0123=Panji, 0654=Haikal, 43792=Ivana, 74501=Ridwan}
BUILD SUCCESSFUL (total time: 0 seconds)

Untuk memanggil data menggunakan method “get” sebagai berikut

```
4 public static void main(String[] args) {
5     Map<String, String> map = new TreeMap<String, String>();
6     map.put("0001", "Sofia");
7     map.put("0654", "Haikal");
8     map.put("4003", "Raisya");
9     map.put("0123", "Panji");
10    map.put("43792", "Ivana");
11    map.put("74501", "Ridwan");
12
13    System.out.println(map);
14
15    map.remove("4003");
16
17    System.out.println(map);
18
19    System.out.println(map.get("0123"));
20
21 }
22 }
```

Output - Collection (run) ✖

```
run:
{0001=Sofia, 0123=Panji, 0654=Haikal, 4003=Raisya, 43792=Ivana, 74501=Ridwan}
{0001=Sofia, 0123=Panji, 0654=Haikal, 43792=Ivana, 74501=Ridwan}
Panji
BUILD SUCCESSFUL (total time: 0 seconds)
```

D. TUGAS RUMAH

1. Buatlah Program dengan menggunakan **Collection Set** (*HashSet, LinkedHashSet, TreeSet*) dan berikan penjelasan pada setiap program!
2. Buatlah Program dengan menggunakan **Collection List** (*ArrayList, LinkedList, Queue, PriorityQueue*) dan berikan penjelasan pada setiap program!
3. Buatlah Program dengan menggunakan **Collection Map** (*HashMap, LinkedHashMap, TreeMap*) dan berikan penjelasan pada setiap program!

Catatan :

1. Setiap program antara Set, List, Queue, Map harus berbeda.
2. Kerjakan latihan dan tugas yang terdapat di modul di atas.
3. Deadline pengumpulan tugas, 3 hari setelah materi dibagikan. Jika telat dalam pengumpulan, tugas tidak diterima.
4. Tugas dikumpulkan melalui link berikut :
<https://drive.google.com/drive/folders/1J1yaVoE0IQIbbwIWw-TqvLRDW9FkTUKX?usp=sharing>

SELAMAT MENGERJAKAN ^_^