COLLECTION FRAMEWORK

PRESENTED BY KELOMPOK 2

PRESENTATION

OUR TEAMS

01 CITRA AULIA
2111521022

02 FIKRAN SHADIQ EL YAFIT
2111521024

03 GILANG KHARISMA
2111522002

04 ARIQ ABDURRAHMAN HAKIM
2111522006

05 MUHAMMAD SATRIA GEMILANG LUBIS
2111522008

06 ATHIFA RIFDA ANDRA
2111522010

07 ANNISA HASIFAH CANTIKA
2111522012

08 SUCI RAHMADHANI
2111522014

09 ALVINO ALBAS
2111522016

10 PUTRA ILHAM
2111522018

11 SYAHNIA PUTRI HENDRY
2111522020

12 BRIANA FIRSTA
2111522024

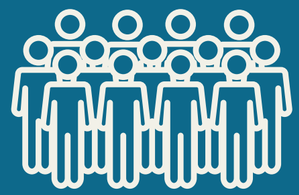
13 M. YAMIN

IDENTIFY

TOPIC OF DISCUSSION

HASHSET

LINKEDHASHSET



TREESET

HASHSET

TOPIC

HASHSET

DEFINITION

Kelas HashSet mengimplementasikan Interface Set, dengan menggunakan Hash table yang merupakan instance HashMap.
HashSet adalah sebuah koleksi elemen yang mana setiap elemen dalam HashSet bernilai unik dan elemen yang sama akan selalu memiliki nilai yang sama pula

ALGORITHM

Index number = sum ASCII codes Mod Size of Array

TOPIC

HASHSET

CONTOH:

Misalnya ada sebuah array sebagai berikut:

Bea	Tim	Leo	Sam	Mia	Zoe	Jan	Lou	Max	Ada	Ted
0	1	2	3	4	5	6	7	8	9	10

Akan di cari element "Ada" pada array tersebut

Nilai ASCII "Ada"

- A = 65
- d = 100
- a = 97

Ada = (65 + 100 + 97) = 262

Ukuran array = **11**

Maka,
262 mod 11 = **9**

Sehingga element "Ada" berada pada urutan array ke **9**

TOPIC

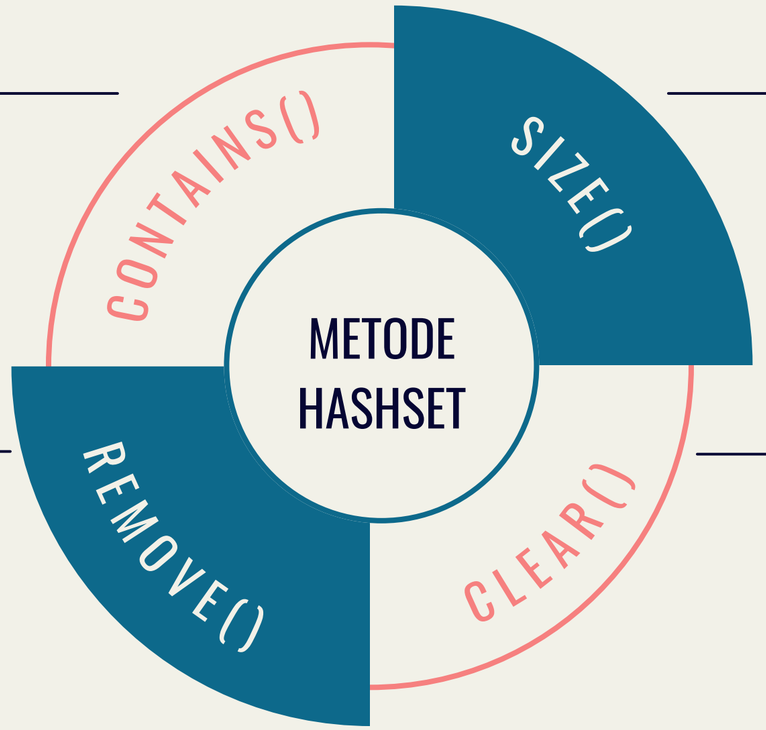
HASHSET

Metode Contains digunakan untuk memeriksa apakah sebuah elemen ada pada sebuah HashSet.

Contoh:
cars.contains("Mazda");

Metode remove() digunakan untuk menghapus elemen yang ada pada class.

Contoh:
cars.remove("Volvo");



Metode ini akan menghitung jumlah elemen yang ada pada HashSet

Contoh:
cars.size();

Metode clear digunakan untuk menghapus semua elemen pada class.

Contoh:
cars.clear();

TOPIC

KONSTRUKTOR HASHSET

HashSet

Untuk membuat objek HashSet ksoong dengan kapasitas awal default 16 dan faktor beban default 0,75

Code:

HashSet<E> hs = HashSet baru<E>();

HashSet (Collection)

Untuk mengkonversi koleksi elemen isi dari sebuah objek ke objek HashSet

Code:

HashSet<E> hs = HashSet baru<E>(Koleksi C);

HashSet (int initialCapacity, float loadFactor)

Membangun objek HashSet kosong dimana initialCapacity dan Load Factor ditentukan pada saat pembuatan objek.

Code:

HashSet<E> hs = new HashSet<E>(int kapasitas awal, float faktor muat);

HashSet (int InitialCapacity)

Pembuatan objek HashSet kosong dengan kapasitas awal ditentukan saat pembuatan objek dan Load Factor tetap 0,75.

Code:

HashSet<E> hs = new HashSet<E>(int initialCapacity);

PROGRAM HASHSET

hashset.java

C: > Users > user > Downloads > HASHSET > HASHSET > hashset.java > hashset > main(String[])

```java
import java.util.ArrayList;
import java.util.HashSet;


public class hashset{
    public static void main(String[] args){
        ArrayList<Integer> data_array = new ArrayList<>();
        ArrayList<String> data_array2 = new ArrayList<>();
        HashSet<Integer> data_hashSet = new HashSet<>();
        HashSet<String> data_hashSet2 = new HashSet<>();
        //Memasukan Nilai Default
        //< ArrayList > dengan urutan  1 3 2 5 4
        data_array.add(1);
        data_array.add(3);
        data_array.add(2);
        data_array.add(5);
        data_array.add(4);
        data_array2.add("Budi");
        data_array2.add("anton");

        //< HashSet > dengan urutan  1 3 2 5 4
        data_hashSet.add(1);
        data_hashSet.add(3);
        data_hashSet.add(2);
        data_hashSet.add(5);
        data_hashSet.add(4);
        data_hashSet2.add("Budi");
        data_hashSet2.add("Anton");


        //Memasukan Nilai Duplukat/Yang Sama Dengan Nilai Sebelumnya
         //< ArrayList >
        data_array.add(5);
        data_array.add(4);
        data_array.add(3);
        data_array2.add("Budi");
        data_array2.add("anton");

```

PROGRAM HASHSET

hashset.java

C: > Users > user > Downloads > HASHSET > HASHSET > hashset.java > hashset

```
38
39          //< HashSet >
40          data_hashSet.add(5);
41          data_hashSet.add(4);
42          data_hashSet.add(3);
43          data_hashSet2.add("Budi");
44          data_hashSet2.add("Anton");
45
46          //Menampilkan Daftar Nilai
47          System.out.println("Array List :");
48          System.out.println("Data Array data_array :" + data_array);
49          System.out.println("Data Array data_array :" + data_array2);
50          System.out.println("\nHashSet :");
51          System.out.println("Data Array data_HashSet :" + data_hashSet);
52          System.out.println("Data Array data_HashSet :" + data_hashSet2);
53          System.out.println(" ");
54
55          //method menghapus di HashSet menggunakan method Remove()
56
57          data_hashSet.remove(4);
58          data_hashSet.remove(3);
59
60          System.out.println("data HashSet sesudah remove : "+ data_hashSet);
61          System.out.println(" ");
62
63          // Menghapus semua data pada Hash set
64          data_hashSet.clear();
65          data_hashSet2.clear();
66          System.out.println("Menghapus semua data :");
67          System.out.println("Data Array data_HashSet :" + data_hashSet);
68          System.out.println("Data Array data_HashSet :" + data_hashSet2);
69
70      }
71
72  }
```

LINKEDHASHSET

TOPIC

LINKEDHASHSET

DEFINITION

Linked Hash Set adalah versi terurut dari Hash Set yang mempertahankan daftar tertaut ganda di semua elemen.

Ketika urutan iterasi diperlukan untuk dipertahankan, kelas ini digunakan. Saat melakukan iterasi melalui HashSet, urutannya tidak dapat diprediksi, sedangkan Linked Hash Set memungkinkan kita melakukan iterasi melalui elemen sesuai urutan penyisipannya.

Saat beredar melalui LinkedHashSet menggunakan iterator, elemen akan dikembalikan sesuai urutan penyisipannya.

TOPIC

LINKEDHASHSET

Konstruktor di kelas LinkedHashSet memiliki bentuk yang sama dengan konstruktor di kelas Hashset.

- Kelas LinkedHashSet memperluas kelas HashSet dan mengimplementasikan antarmuka Set.
- Kelas LinkedHashSet tidak mendefinisikan metode eksklusifnya sendiri.

TOPIC

LINKEDHASHSET

PERBEDAAN DENGAN HASHSET

- Urutan penyisipan elemen dipertahankan selama pembuatan LinkedHashSet.
- Urutan penyisipan elemen dipertahankan selama pembuatan LinkedHashSet.
- Duplikat tidak diperbolehkan di LinkedHashSet.

PROGRAM LINKEDHASHSET

link_hashSet.java

C: > Users > user > Downloads > Linked_Hashset > Linked_Hashset > link_hashSet.java > link_hashSet > main(String[])

```java
import java.util.LinkedHashSet;

public class link_hashSet {
    public static void main(String[] args) {

        LinkedHashSet<String> menu = new LinkedHashSet<String>();
        menu.add("Mie Pedas");
        menu.add("Ayam Geprek");
        menu.add("Pempek");
        menu.add("Kebab");
        menu.add("Jasuke");
        menu.add("Lemon Tea");

        System.out.println("Menu Awal Expo\t\t\t:" + menu);
        System.out.println("Banyak menu\t\t\t:" + menu.size());
        System.out.println("Menghapus Lemon Tea dari menu\t:" + menu.remove("Lemon Tea"));
        System.out.println("Menghapus Green Tea dari menu\t:" + menu.remove("Green Tea"));
        System.out.println("Menu tanpa minuman\t\t:" + menu);
        System.out.println("Menambah menu Kebab kembali\t:" + menu.add("Kebab")); // kelebihan linkedlist tidak bisa
                                                                                  // duplikat data
        System.out.println("Melihat tersedianya Kebab di menu:" + menu.contains("Kebab"));
        System.out.println("Banyak menu akhir\t\t:" + menu.size());
        System.out.println("Menampilkan menu secara vertikal:");
        for (String strLHS : menu) {
            System.out.println(strLHS);
        }

    }
}

```

TREESET

TOPIC

TREESET

- TREESET MERUPAKAN CLASS YANG SERING DIGUNAKAN UNTUK MENGEKSTRAK ELEMEN DARI COLLECTION DALAM URUTAN TERTENTU.
- TREESET MENGIMPLEMENTASIKAN ANTARMUKA SORTEDSET SEHINGGA NILAI DUPLIKAT TIDAK DIPERBOLEHKAN. OBJEK DALAM TREESET DISIMPAN DALAM URUTAN YANG DIURUTKAN DAN MENAIK.
- TREESET TIDAK BISA MENYIMPAN SEMUA OBJEK, KARENA OBJEK YANG DISIMPAN HARUS BISA DIURUTKAN.
- TREESET MENYEDIAKAN IMPLEMENTASI DARI INTERFACE SET YANG MENGGUNAKAN POHON UNTUK PENYIMPANAN. OBJEK DISIMPAN DALAM DISORTIR, URUTAN MENAIK.
- AKSES DAN PENGAMBILAN WAKTU YANG CUKUP CEPAT, MENYIMPAN SEJUMLAH BESAR INFORMASI DIURUTKAN YANG HARUS DITEMUKAN DENGAN CEPAT.
- TREESET BERFUNGSI UNTUK MENAMPILKAN DATA YANG BERSIFAT UNIK DENGAN URUTAN YANG TERATUR DARI KECIL KE BESAR ATAU SEBALIKNYA.

TOPIC

KONSTRUKTOR TREESET

- TreeSet ()
- TreeSet (Collection c)
- TreeSet (Comparator comp)
- TreeSet (SortedSet ss)

PROGRAM TREESET

treeSet.java

C: > Users > user > Downloads > treeSet > treeSet > treeSet > src > treeSet.java > ...

```java
import java.util.*;
/**
 * treeSet
 */
public class treeSet {
    public static void main(String[] args) {
        /* Creating a Set interface with
        reference to TreeSet class
        Deklarasi objek bertipe String */
        TreeSet<String> ts = new TreeSet<>();

        // Element ditambahkan menggunakan method add()
        ts.add("SI");
        ts.add("Adalah");
        ts.add("Sistem Informasi");

        // Print semua element pada object
        System.out.println(ts);
        String check = "Sistem Informasi";

        // Memperiksa apakah String check diatas ada pada TreeSet atau tidak
        System.out.println("Contains " + check + " " + ts.contains(check));

        // Print element pertama pada TreeSet
        System.out.println("First Value " + ts.first());

        // Print element terakhir pada TreeSet
        System.out.println("Last Value " + ts.last());

        String val = "SI";

        /* Mencari nilai apakah nilai lebih besar
        atau lebih kecil dari String diatas*/
        System.out.println("Higher " + ts.higher(val));
        System.out.println("Lower " + ts.lower(val));
    }


}
```

PERSAMAAN HASHSET, LINKEDHASHSET & TREESET

TOPIC

PERSAMAAN

DUPLIKAT	Mereka tidak diizinkan untuk menyimpan duplikat.
KEAMANAN THREAD	Tidak aman untuk thread, jika menggunakannya di lingkungan multi-threading di mana setidaknya satu Thread diubah. Set perlu menyinkronkannya secara eksternal.
FAIL-FAST INTERATOR	Iterator dikembalikan oleh TreeSet, LinkedHashSet dan HashSet adalah Iterator yang gagal-cepat.

PERBEDAAN HASHSET, LINKEDHASHSET & TREESET

TOPIC

PERBEDAAAN

	HashSet	LinkedHashSet	TreeSet
Bekerja secara Internal	Menggunakan HashMap secara internal untuk menyimpan elemen-elemennya	Menggunakan LinkedHashMap secara internal untuk menyimpan elemen-elemennya	Menggunakan TreeMap secara internal untuk menyimpan elemenelemennya.
Urutan Elemen	Tidak mempertahankan urutan elemen apa pun	Mempertahankan urutan penyisipan elemen	Memesan elemen sesuai dengan Comparator yang disediakan
Performa	Lebih baik dari dua lagi	Berada di tengah-tengah	Lebih rendah dari dua lagi

TOPIC

PERBEDAAAN

	HashSet	LinkedHashSet	TreeSet
Operasi Penyisipan, Penghapusan, dan Pengambilan	Memberikan kinerja urutan O(1) untuk setiap operasi tersebut	Memberikan kinerja urutan O(1) untuk setiap operasi tersebut	Memberikan kinerja urutan O(log(n)) untuk setiap operasi tersebut
Membandingkan Elemen	Menggunakan metode equals() dan hashCode(), dan untuk menghapus kemungkinan duplikat.	Menggunakan metode equals() dan hashCode()	mMenggunakan metode compare() atau compareTo(), dan untuk menghapus kemungkinan duplikat.

TOPIC

PERBEDAAAN

	HashSet	LinkedHashSet	TreeSet
Null Elements	Memungkinkan maksimum satu elemen null	Memungkinkan maksimum satu elemen null	Tidak mengizinkan satu elemen null
Pekerjaan Memori	Membutuhkan lebih sedikit memori daripada LinkedHashSet dan TreeSet	Membutuhkan lebih banyak memori daripada HashSet karena juga memelihara LinkedList	Membutuhkan lebih banyak memori daripada HashSet karena juga memelihara Comparator
Kapan Digunakan	Saat tidak ingin mempertahankan elemen	Saat ingin mempertahankan urutan penyisipan elemen	Saat ingin mengurutkan elemen menurut beberapa Comparator


THANK YOU!!