**Iterator** adalah sebuah perulangan pada objek untuk melakukan looping melalui koleksi. Perulangan tersebut dapat disimpan dalam package java.util. dirancang untuk dengan mudah mengubah koleksi yang dilaluinya. Pada umumnya digunakan untuk mengakses ArrayList dan method-method lainnya. Iterator harus digunakan setiap kali kita ingin menghitung elemen dalam semua kerangka Koleksi yang diimplementasikan antarmuka seperti Set, List, Queue, Deque dan juga di semua kelas yang diimplementasikan dari antarmuka Map.

Contohnya.

// Java program to demonstrate Iterator

import java.util.ArrayList;

import java.util.Iterator;

public class Test

{

    public static void main(String[] args)

    {

        ArrayList<Integer> al = new ArrayList<Integer>();

        for (int i = 0; i < 10; i++)

            al.add(i);

        System.out.println(al);

        // at beginning itr(cursor) will point to

        // index just before the first element in al

        Iterator itr = al.iterator();

        // checking the next element availabilty

        while (itr.hasNext())

        {

            //  moving cursor to next element

            int i = (Integer)itr.next();

            // getting even elements one by one

            System.out.print(i + " ");

            // Removing odd elements

            if (i % 2 != 0)

               itr.remove();

        }

        System.out.println();

        System.out.println(al);

    }

}

**Exception** adalah fungsi yang digunakan untuk mengetahui error dalam program tersebut. Kesalahan (errors) yang terjadi saat runtime, menyebabkan gangguan pada alur eksekusi program. Terdapat beberapa tipe error yang dapat muncul. Sebagai contoh adalah error pembagian 0, mengakses elemen di luar jangkauan sebuah array, input yang tidak benar dan membuka file yang tidak ada.

Exception (try, catch,finally,throw)

Try digunakan untuk mencoba menjalankan program untuk mencari kesalahan di dalam program tersebut.

Catch digunakan untuk menangkap kesalahan atau bug yang terjadi pada suatu program dan harus dipasangkan dengan try.

Finally digunakan untuk menunjukan block program tersebut akan selalu dieksekusi meskipun adanya kesalahan yang muncul atau pun tidak ada.

Throw digunakan untuk melemparkan suatu bug secara manual.

**Comparator**

Antarmuka ini memiliki dua metode yang sama dan membandingkan. Kelas yang objeknya akan diurutkan tidak perlu mengimplementasikan antarmuka komparator ini. Beberapa kelas ketiga dapat mengimplementasikan antarmuka ini untuk mengurutkan. Dalam hal ini ada beberapa bentuk penyortiran, kita dapat menulis penyortiran yang berbeda berdasarkan atribut objek yang berbeda untuk diurutkan, misalnya Kita dapat menyortir pemain berdasarkan usia, dengan memberi peringkat, dan dengan namewithout mengubah kelas Player kita, untuk melakukan itu kita menggunakan lainnya kelas untuk Perbandingan Usia dan kelas lainnya untuk Perbandingan Peringkat dan kelas lainnya untuk Perbandingan Nama.