**File e Flussi**



In Java input e output sono definiti in termini di flussi (stream): Sequenze ordinate di dati.

* Per dati binari, usare la classe **InputStream, OutputStream.**
* Per caratteri, usare la classe **Reader, Writer.**

Costruttori che ricevono come parametro il nome di un file/directory o un oggetto File possono lanciare una ***FileNotFoundException*** (sottoclasse di ***IOException***).

**Flussi di Oggetti:**

Consentono di operare su interi oggetti:

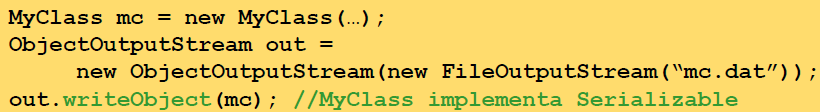
* Per scrivere un oggetto non dobbiamo prima decomporlo
* Per leggere un oggetto non dobbiamo leggere i dati separatamente e poi ricomporre l’oggetto

Flussi in *scrittura*: Classe ***ObjectOutputStream***

Flussi in *lettura*: Classe ***ObjectInputStream***

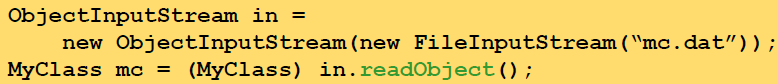
L’oggetto da inserire nel flusso deve essere serializzabile altrimenti viene sollevata la **NotSerializableException**

* Appartenere a una classe che implementa l’interfaccia **Serializable**
* **Serializable** non ha metodi



Legge un ***Object*** da file e restituisce un riferimento a tale ***Object***

* L’output necessita di un cast
* Può lanciare un’eccezione controllata di tipo ***ClassNotFoundException***



**Esempio:**

import java.io.\*;

public class ObjectInputStreamExample {

public static class Person implements Serializable {

public String name = null;

public int age = 0;

}

public static void main(String[] args) throws IOException, ClassNotFoundException {

ObjectOutputStream objectOutputStream = new ObjectOutputStream(new FileOutputStream("person.txt"));

Person person = new Person();

person.name = "Jakob Jenkov";

person.age = 40;

objectOutputStream.writeObject(person);

objectOutputStream.close();

ObjectInputStream objectInputStream = new ObjectInputStream(new FileInputStream("person.txt"));

Person personRead = (Person) objectInputStream.readObject();

objectInputStream.close();

System.out.println(personRead.name);

System.out.println(personRead.age);

}

}