

Note ed esercizi aggiuntivi

13. Di nuovo eccezioni, Stream di I/O (inizio)

Esempio. Esempio di codice che può sollevare svariati problemi in esecuzione (individuareli tutti!)

```
class Prova {
    public static void main(String[] a) {
        int somma = 0;
        for (String s: a)
            somma = somma + s.length();
        System.out.println(somma);
        System.out.println(somma / a.length);
        Integer n = new Integer(a[1]);
        System.out.println(10 / n);
    }
}
```

Lo stesso codice con un blocco try-catch per intercettare le singole eccezioni.

```
class Prova {
    public static void main(String[] a) {
        int somma = 0;
        for (String s: a)
            somma = somma + s.length();
        System.out.println(somma);
        try {
            System.out.println(somma / a.length);
            Integer n = new Integer(a[1]);
            System.out.println(10 / n);
        } catch (ArithmeticException e) {
            if (a.length > 0)
                System.out.println("Il secondo argomento deve essere un numero diverso da 0");
            else
                System.out.println("Sono richiesti almeno due argomenti");
        } catch (NumberFormatException e) {
            System.out.println("Il secondo argomento deve essere un numero");
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Sono richiesti almeno due argomenti");
        }
    }
}
```

Esempio. Visualizza sul monitor il contenuto di un file di caratteri.

```
import java.io.FileReader;
import java.io.IOException;

class VisualizzaFile {

    public static void main(String[] args) throws IOException {
        //costruzione dello stream di caratteri
        FileReader frd = new FileReader(args[0]);

        int i;
        //lettura e visualizzazione
        while ((i = frd.read()) != -1)
            System.out.print((char)i);

        //chiusura dello stream
        frd.close();
    }
}
```

Note

Il costruttore di `FileReader` può sollevare una `FileNotFoundException` (sottoclasse di `IOException`). Il metodo `read` e il metodo `close` possono sollevare una `IOException`. Poiché si tratta di *eccezioni controllate*, `main` deve intercettarle con un costrutto `try-catch` oppure delegarle esplicitamente al chiamante. Qui si è scelta la seconda alternativa: la delega esplicita al chiamante è espressa scrivendo `throws IOException` alla fine dell'istestazione di `main`.