Università di Ferrara
Dipartimento di Ingegneria

Esercitazioni di

FONDAMENTI DI INFORMATICA MODULO B

ESERCITAZIONE 4 JAVA

Tutor

Arnaud Nguembang Fadja: ngmrnd@unife.it
Damiano Azzolini: damiano.azzolini@student.unife.it

Esercizio 1 – Lettura da File Binario - A

Leggere dal file **IN.dat** una serie di numeri interi.

Per ogni intero letto, se il valore appartiene al range [-5;5], lo si scriva su un secondo file binario.

Altrimenti sollevare una eccezione la quale stampa a video un messaggio di errore.

Esercizio 1 – Lettura da File Binario - B

Se si utilizza la classe **FileInputStream**, si ha a disposizione solamente il metodo **read()** (controllare le API sul Desktop per capire come si comporta).

Attenzione: leggere un **byte** alla volta (non un intero) restituendone il valore come intero.

In realtà per leggere un intero bisogna leggere 4 byte (usando, per esempio, **read(byte[])**) e convertire il vettore di 4 byte nel corrispettivo valore int.

Esercizio 1 – Lettura da File Binario - C

Una seconda possibilità è usare il metodo **readInt()** della classe **DataInputStream** (da collegare ad un oggetto **FileInputStream**).

Il problema di questo secondo metodo è che, per segnalare la fine del file, solleva l'eccezione **EOFException** che deve essere gestita.

CONSIGLIO: usare il secondo metodo

Esercizio 2 – Lettura da Riga di Comando - A

Leggere da riga di comando una serie di interi e/o caratteri (args[]).

Nel caso in cui args[i] sia un **intero**, scrivere il suo valore sul file “interi.bin”

Nel caso in cui args[i] sia un **char**, scrivere il suo valore sul file “caratteri.bin”

Controllare infine che i file creati siano corretti, leggendone il contenuto.

Esercizio 2 – Lettura da Riga di Comando - B

La conversione da **String** a numero si può fare con il metodo statico **parseInt(String)** della classe **Integer**, che solleva l'eccezione **NumberFormatException** nel caso la conversione non possa essere fatta.

UTILIZZARE STREAM DI MANIPOLAZIONE

Esercizio 3 – Lettura da Riga di Comando - A

Scrivere un metodo **main** che dovrà leggere da tastiera delle parole/frasi, che verranno poi inserite in un file di testo.

L'inserimento si interromperà quando verrà digitata la parola **quit**.

La lettura dovrà essere fatta mediante utilizzo della classe **InputStreamReader** che verrà collegata al **System.in** (tastiera).

La scrittura su file dovrà essere fatta mediante l'utilizzo della classe **PrintStream** che verrà collegata al file.

Esercizio 3 – Lettura da Riga di Comando - B

Rieseguire il ciclo di input delle parole/frasi inserendole in un secondo file di testo.

L'inserimento si interromperà quando verrà digitata la parola **quit**.

La lettura dovrà essere fatta mediante utilizzo della classe **BufferedReader**.

ATTEZIONE: A cosa andrà collegato l'istanza dell'oggetto **BufferedReader**? (Consultare le API presenti sul Desktop).

La scrittura su file dovrà essere fatta mediante l'utilizzo della classe **PrintStream** che verrà collegata al file.

Esercizio 4 – Lettura da Riga di Comando - A

Definire la classe **MyBufferedReader** che estende la classe **BufferedReader**, implementando il metodo

```
public String readNRows(int nRows)
```

Il metodo **readNRows** legge dallo stream **nRows** righe e restituisce una stringa formata dalle righe lette concatenate l'una dopo l'altra, se **nRows** righe possono essere lette.

Se non sono presenti abbastanza righe nello stream il metodo lancia l'eccezione **RowsNotFoundException**.

Definire la classe **RowsNotFoundException** che estende la classe **Exception**.

Esercizio 4 – Lettura da Riga di Comando - B

Definire la classe **MyFileWriter** che estende la classe **FileWriter**, che contiene la variabile

```
private int row;
```

Implementare i costruttori in modo da azzerare la variabile **row** e ridefinire i metodi:

```
public void write(String str)
```

```
public void close()
```

Esercizio 4 – Lettura da Riga di Comando - C

Il metodo **write(String)** dovrà scrivere prima di **str** il numero di riga attuale seguita da una tabulazione, il carattere due punti **':'** e uno spazio.

```
> 1    : pippo
> 2    : pluto
> 3    : paperino
```

Il metodo **close()** dovrà scrivere nel file il numero totale di righe scritte e infine chiudere il file.

Esercizio 4 – Lettura da Riga di Comando - D

Il metodo main dovrà, usando le classi già **implementate**, leggere da un file di testo (input.txt) tutte le righe e scriverle in un secondo file.

Una volta eseguita la copia del file, chiudere e riaprire lo stream di lettura e leggere una riga del file di input scrivendola nel file di output usando il metodo **readRows(int)**.

TIPS: Provare a inserire un valore di righe tale da scatenare l'eccezione **RowsNotFoundException.**