Esercitazione N4: Utilizzo degli stream di input, gestione delle eccezioni e generazione di numeri casuali

Testo: "Si desidera realizzare un sistema che richiede all'utente di indovinare un numero segreto generato casualmente e ad ogni tentativo sbagliato notifichi se il numero da indovinare è più piccolo o più grande del tentativo effettuato."

La **generazione di numeri casuali** (pseudo-casuali) in java si può ottenere in due modi diversi. Nel primo caso si utilizza il metodo statico **Math.random()** che ritorna un valore **double** compreso tra **0.0** e **1.0**. Vediamo un semplice esempio:

```
double v = Math.random();
```

Questo è un modo molto veloce per ottenere un valore **random**. Da questo valore poi è possibile ottenere un valore intero effettuando moltiplicazioni ed operazioni di **cast**. Se ad esempio desideriamo ottenere un intero compreso tra 0 e 10 basta moltiplicare il valore casuale generato - che è compreso tra 0.0 e 1.0 - per 10 ed effettuare il cast del risultato:

```
int n = (int)(Math.random()*10);
```

Il metodo Math.random() non fa altro che utilizzare un'istanza della classe **Random** del package *java.util* invocando il metodo **nextDouble()**. Ma vediamo nel dettaglio questa classe che implementa diverse funzionalità.

La classe Random

La classe java.util.Random ci mette a disposizione diversi metodi per la generazione di numeri o meglio valori casuali, poichè è possibile anche generare valori booleani. Per ogni tipo primitivo è disponibile un metodo che genera un valore **random** in base al tipo. Data un'istanza della classe Random:

```
Random random = new Random();
```

Possiamo ottenere:

- un **valore booleano**:

```
boolean b = random.nextBoolean();
```

- un **intero**:

```
int k = random.nextInt();
```

- un **intero** compreso tra 0 e **n** (n positivo):

```
int n = 10;
int k = random.nextInt(n);//Valori compresi tra 0 e 10
```

Se invece si desidera ottenere un intero casuale compreso tra **j** e **n** (ad esempio tra 3 e 10) basta aggiungere una somma:

```
int j = 3;
int n = 10-j;
int k = random.nextInt(n)+j;//Valori compresi tra 3 e 10
```

- un valore **float** compreso sempre tra 0.0 e 1.0:

```
float f = random.nextFloat();
```

- un valore **long**:

```
long l = random.nextLong();
```

- un valore **double** che non è altro che il metodo visto sopra poichè la classe Math utilizza questo metodo:

```
doubke d = random.nextDoube();
```

- il prossimo valore **double** di una distribuzione **Gaussiana**:

```
double d = random.nextGaussian();
```

Adesso che abbiamo una ampia visione di come generare numeri casuali vediamo come porebbe essre risolto il problea specificato nel testo:

```
// Preannunciamo al compilatore che useremo alcune classi di libreria
import java.util.Random;            // generatore di numeri pseudocasuali
import java.io.InputStreamReader;   // lettura a caratteri da flusso di input a byte
import java.io.BufferedReader;      // lettura a righe di testo da flusso di input a caratteri

// Classe principale (e unica) del programma
public class Indovina
{
    // Metodo di innesco standard del programma e unico metodo della classe
    public static void main(String [] args)
    {
        // predisponi un contatore di tentativi 'n' con valore iniziale pari a 0
        int n=0;

        // Scegli un numero casuale e ricordalo nella variabile 'x'
        Random r=new java.util.Random();
        int x=r.nextInt(100)+1;
        // (condizione iniziale: numero non ancora indovinato)
        boolean indovinato=false;
        // (predisponi un flusso in lettura per righe di testo e collegalo alla tastiera)
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        // Ripeti quanto segue fintantoche il numero non viene indovinato
        // (v. condizione 'while' a fine ciclo)
```

```
        do
        {
            // (predisponi la cattura e gestione di eventuali errori verificatisi
            // nella decodifica del tentativo immesso)
            try
            {
                // Ricevi dall'esterno il tentativo e ricordalo come 't'
                System.out.println("fai un Tentativo per indovinare un numero compreso tra 1 e 100? ");
                String tentativo=br.readLine(); // legge una stringa da tastiera
                int t=Integer.parseInt(tentativo);

                if(t>x) // Se t e' maggiore di x...
                {
                    System.out.println("Il numero segreto e' minore");
                    n=n+1; // incrementa contatore di tentativi
                }
                else if(t<x) // Se t e' minore di x...
                {
                    System.out.println("Il numero segreto e' maggiore");
                    n=n+1; // incrementa contatore di tentativi
                }
                else // Per esclusione: t uguale a x - numero indovinato !
                {
                    System.out.println("Hai indovinato!");
                    System.out.println("Numero di tentativi impiegati:"+n);
                    indovinato=true; // segnala che si possono interrompere le ripetizioni
                }
            }
            catch(Exception e) // Qui vengono gestiti gli errori catturati nel blocco try
            {
                System.out.println("Tentativo non valido - ripetere, prego");
            }
        } while(indovinato==false); // Il ciclo terminera' solo quando 'indovinato' varra' true
    } // fine metodo main
} // fine classe Indovina
```

Provare il funzionamento del codice sopradescritto con Eclipse e modificarlo aggiungendo un numero di tentativi prefissato e alla fine aggiungere la possibilità di indovinare un altro numero se l'utente lo desidera.