Algoritmo adottato per l'ordinamento casuale degli operatori in elenco o catalogo

Il software permette di aggiornare il numero d'ordine degli operatori economici in maniera casuale.
La funzione può essere eseguita sull'intero elenco operatori o solo sui "nuovi iscritti" (qualora si intenda effettuare un ordinamento casuale in fasi successive, solo per gli operatori aggiunti nel tempo all'elenco).
In particolare vengono svolte le operazioni di seguito descritte.

- viene definita la lista L degli operatori abilitati "non ordinata" (ovvero nell'ordine esistente).
- viene definito un vettore V di lunghezza N, dove N corrisponde al numero di operatori economici da ordinare.
- viene effettuato un ciclo di operazioni per N volte; per ogni ciclo:
  - viene estratto casualmente dal vettore V uno dei valori per i quali non è già stato assegnato il numero d'ordine nella lista L;
  - l'estrazione viene effettuata utilizzando la funzione "Java Random"; tale funzione assicura che:
    - ogni numero estratto ha la stessa probabilità di uscire ogni volta che viene lanciata la funzione sullo stesso intervallo di numeri
    - non c'è mai dipendenza fra un numero che esce e il successivo
  - viene aggiornato il numero d'ordine dell'operatore nella lista L in base al valore casuale estratto
  - viene ridefinito il vettore V togliendo il valore estratto

Esempio:

Lista operatori "non ordinata":

Indice	Ordine esistente	Operatore economico
0	1	Impresa Antares
1	2	Impresa Bilow
2	3	Impresa Castelli
3	4	Impresa Dondi
4	5	Impresa Effegi

Definizione del vettore:

Indice	Valore
0	1
1	2
2	3
3	4
4	5

Ciclo da 0 a 4:

Ciclo 0
Estrazione casuale di un numero da 0 a 4: 3
Seleziono il numero dal vettore dove l'indice è uguale all'estratto, quindi indice 3, valore 4
Seleziono l'operatore dalla lista con indice del ciclo = 0, quindi Antares
Assegno il 4° posto (valore 4, dell'indice 3 estratto) all'operatore Antares
Tolgo dal vettore il valore corrispondente all'indice 3, rigenerando il vettore:

Indice	Valore
0	1
1	2
2	3
3	5

Ciclo 1
Estrazione casuale di un numero da 0 a 3: 3
Seleziono il numero dal vettore dove l'indice è uguale all'estratto, quindi indice 3, valore 5
Seleziono l'operatore dalla lista con indice del ciclo = 1, quindi Bilow
Assegno il 5° posto (valore 5, dell'indice 3 estratto) all'operatore Bilow

Indice	Valore
0	1
1	2
2	3

Ciclo 2
Estrazione casuale di un numero da 0 a 2: 0
Seleziono il numero dal vettore dove l'indice è uguale all'estratto, quindi indice 0, valore 1
Seleziono l'operatore dalla lista con indice del ciclo = 2, quindi Castelli
Assegno il 1° posto (valore 1, dell'indice 0 estratto) all'operatore Castelli

Indice	Valore
0	2
1	3

Ciclo 3
Estrazione casuale di un numero da 0 a 1: 1
Seleziono il numero dal vettore dove l'indice è uguale all'estratto, quindi indice 1, valore 3
Seleziono l'operatore dalla lista con indice del ciclo = 3, quindi Dondi
Assegno il 3° posto (valore 3, dell'indice 1 estratto) all'operatore Dondi
Tolgo dal vettore il valore corrispondente all'indice 1, rigenerando il vettore:

Indice	Valore
0	2

Ciclo 4
Estrazione casuale di un numero da 0 a 0: 0
Seleziono il numero dal vettore dove l'indice è uguale all'estratto, quindi indice 0, valore 2
Seleziono l'operatore dalla lista con indice del ciclo = 4, quindi Effegi
Assegno il 2° posto (valore 2, dell'indice 0 estratto) all'operatore Effegi

Indice	Valore
0	0

Fine ciclo

Risultato Lista Operatori "ordinata casualmente":

Indice	Ordine esistente	Operatore economico
2	1	Impresa Castelli
4	2	Impresa Effegi
3	3	Impresa Dondi
0	4	Impresa Antares
1	5	Impresa Bilow

Per maggiore trasparenza, viene riportato un estratto dal codice del software, commentato passo-passo.

```
// Si crea un vettore (di nome "valori") di dimensione N
// con indice da 0 a N-1
// dove N è pari al numero di operatori economici da ordinare;
for (int i=0;i<numeroOperatori;i++)
valori.add(i, new Long(i + 1 + numordplPartenza));
// Si istanzia la classe Java Random
Random r = new Random();
// Si effettua un ciclo da 0 a N-1
for(int i = 0;i<numeroOperatori;i++) {
// Si estrae casualmente un indice da 0 a M,
// dove M è pari alla dimensione (size) del vettore nel ciclo;
// l'indice estratto viene assegnato alla variabile "rand".
int rand= r.nextInt(valori.size());
// Si seleziona progressivamente l'operatore
// dalla lista non ordinata.
Vector operatore = (Vector) listaOperatori.get(i);
String dittao = (String)((JdbcParametro) operatore.get(0)).getValue();
// Si aggiorna in database l'ordine dell'operatore selezionato
// con il valore del vettore abbinato all'indice estratto sopra
// (vedi variabile "rand" e vettore "valori")
this.sqlManager.update(update, new Object[] { valori.get(rand),codgar,
ngara,dittao});
// Si rimuove dal vettore "valori" il numero già estratto.
valori.remove(rand);
}
```

Specifica pubblica della funzione (classe) Java Random.

java.util

Class Random

java.lang.Object
  └ **java.util.Random**

All Implemented Interfaces:
Serializable

Direct Known Subclasses:
SecureRandom

```
public class Random
extends Object
implements Serializable
```

An instance of this class is used to generate a stream of pseudorandom numbers. The class uses a 48-bit seed, which is modified using a linear congruential formula. (See Donald Knuth, *The Art of Computer Programming, Volume 2*, Section 3.2.1.)

If two instances of `Random` are created with the same seed, and the same sequence of method calls is made for each, they will generate and return identical sequences of numbers. In order to guarantee this property, particular algorithms are specified for the class `Random`. Java implementations must use all the algorithms shown here for the class `Random`, for the sake of absolute portability of Java code. However, subclasses of class `Random` are permitted to use other algorithms, so long as they adhere to the general contracts for all the methods.

The algorithms implemented by class `Random` use a `protected` utility method that on each invocation can supply up to 32 pseudorandomly generated bits.