evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Číslo projektu	CZ.1 .07/1.1 .36/02.0066
Autor	Ladislav Kašpárek
Předmět	Programování a vývoj aplikací
Téma	Generování permutací
Metodický pokyn	výkladový text s ukázkami

Generování permutací

Problém

Vstup: Máte zadáno kladné celé číslo N.
Úkol: Máte najít algoritmus, který najde libovolnou permutaci čísla N.

Co je to permutace?

Permutace čísla N je posloupnost čísel od 1 do N, taková že každé z čísel v rozsahu se v posloupnosti vyskytuje právě jednou.

```
Perm = Permutations[{1, 2, 3, 4}];
Perm[[RandomInteger[{1, Length[Perm]}]]]
```

```
{1, 3, 2, 4}
```

Hrubá síla:

Základní myšlenka :

Vygeneruji náhodné číslo v rozsahu 1 až N a prohledám již vygenerované pole od nulté pozici po aktuální, zda se tam toto číslo již nenachází. Pokud ne, pak číslo umístím na aktuální pozici a posunu se na další kde postup opakuji. Pokud vygenerované číslo již v poli je, pak generuji znovu.

Kód:

```
public void bruteForce() {
    Random rd = new Random();
    for (int i = 0; i < this.array.length; i++) {
        int tmp = 0;
        do {
            tmp = rd.nextInt(this.array.length) + 1;
        } while (isInArray(tmp));
        this.array[i] = tmp;
    }
```

```
}

private boolean isInArray(int value) {
    for (int i = 0; i < array.length; i++) {
        if (array[i] == value) {
            return true;
        }
    }
    return false;
}
```

Vlastnosti:

Jednoprůchodový algoritmus.
Ze začátku velmi rychlé (snadno se trefují volné prvky).
Ke konci velmi pomalý (mnoho zbytečných průběhů, než se náhodně trefí několik posledních neobsazených čísel.
Vyhledávání v každém kroku velmi zdržuje a navíc se doba prohledávání neustále prodlužuje (prohledáváme stále větší pole).

Pomocí pole výskytů:

Základní myšlenka :

Kromě pole, kam generujeme náhodná čísla, máme ještě pole kam zaznamenáváme, zda dané číslo již bylo použito (true/false). Postup je stejný, ale místo “zdlouhavého” hledání máme možnost jedním přístupem do pole výskytů zkontrolovat, zda už prvek byl použit, nebo ne.

Kód:

```
public void memoryForce() {
    Random rd = new Random();
    boolean isIn[] = new boolean[this.array.length];
    for (int i = 0; i < isIn.length; i++) {
        isIn[i] = false;
    }
    int tmp = 0;
    for (int i = 0; i < this.array.length; i++) {
        do {
            tmp = rd.nextInt(this.array.length);
        } while (isIn[tmp]);
        isIn[tmp] = true;
        this.array[i] = tmp + 1;
    }
}
```

Vlastnosti:

Jednoprůchodový algoritmus.
Ze začátku velmi rychlé (snadno se trefují volné prvky).
Ke konci velmi pomalý (mnoho zbytečných průběhů, než se náhodně trefí několik posledních neobsazených čísel.
Oproti předchozímu odstraněno “pomalé vyhledávání”.

Dvouprůchodový algoritmus:

Základní myšlenka :

V prvním průchodu naplníme pole rostoucí posloupností (stačí velmi jednoduchá na i-tou pozici umístíme hodnotu i). Poté pole projdeme druhým průchodem a k i-té pozici vygenerujeme náhodnou pozici a prvky na těchto pozicích prohodíme.

Kód:

```
public void twoPases() {
    Random rd = new Random();
    for (int i = 0; i < this.array.length; i++) {
        array[i] = i + 1;
    }
    for (int i = 0; i < this.array.length; i++) {
        int tmp = rd.nextInt(this.array.length);
        int tmp2 = this.array[i];
        this.array[i] = this.array[tmp];
        this.array[tmp] = tmp2;
    }
}
```

Vlastnosti:

Dvouprůchodový algoritmus.
Stejná rychlost v průběhu celého algoritmu (žádný “zbytečný průběh”)
Nejrychlejší z předvedených algoritmů.

Příklady

Vyzkoušejte si vygenerovat libovolnou permutaci 10000 metodou:
- hrubá síla
- pole výskytů
- dvou-průchodový algoritmus

Změřte průměrný čas trvání běhu.
Zkuste vygenerovat všechny možné permutace čísla N. Navrhněte algoritmus, zkontrolujte pomocí WM.

Zdroje

http://reference.wolfram.com/mathematica/ref/Permutations.html
http://reference.wolfram.com/mathematica/ref/RandomInteger.html
http://reference.wolfram.com/mathematica/ref/Length.html